Oracle VM VirtualBox®

Manuel de l'utilisateur

Version 4.2.12_RPMFusion

http://www.virtualbox.org

Contents

1 Premières étapes

Bienvenu sur Oracle VM VirtualBox!

VirtualBox est une application de virtualisation de machines inter-plateformes. Qu'est-ce que cela veut dire? Premièrement, il s'installe sur vos ordinateurs existants basés sur Intel ou AMD, qu'ils exécutent les systèmes d'exploitation Windows, Mac, Linux ou Solaris. Deuxièmement, il augmente les capacités de votre ordinateur existant de telle sorte qu'il puisse exécuter plusieurs systèmes d'exploitation à la fois (au sein de plusieurs machines virtuelles). Donc, par exemple, vous pouvez lancer Windows et Linux sur votre Mac, lancer Windows Server 2008 sur votre serveur Linux, lancer Linux sur votre PC Windows, etc., tout en laissant ouvertes vos applications existantes. Vous pouvez installer et exécuter autant de machines virtuelles que vous le voulez; les seules limites pratiques sont l'espace disque et la mémoire.

VirtualBox est tout autant simple que puissant. Il peut se lancer depuis n'importe où, sur de petits systèmes embarqués ou des machines de bureau jusqu'au déploiement tout autant que sur des centres de données de développement et mêmes dans des environnements Cloud dit en nuages.

L'aperçu suivant vous montre comment VirtualBox, installé sur une machine Linux, lance Windows 7 dans une fenêtre de machine virtuelle:

Dans ce manuel de l'utilisateur, nous allons simplement commencer par une rapide introduction à la virtualisation et sur la façon de lancer votre première machine virtuelle avec l'interface graphique VirtualBox facile à utiliser. Les chapitres suivants donneront beaucoup plus de détails concernant des outils et des fonctionnalités plus puissants, mais heureusement, il n'est pas nécessaire de lire tout le manuel de l'utilisateur avant de pouvoir utiliser VirtualBox.

Vous pouvez trouver un résumé des possibilités de VirtualBox au chapitre 1.3, *Aperçu des fonctionnalités*, page 12. Pour les utilisateurs qui ont déjà VirtualBox et qui veulent seulement savoir ce qu'il y a de nouveau dans cette version, il y a une liste détaillée au chapitre 15, *Change log*, page 204.

1.1 Pourquoi la virtualisation est-elle utile?

Les techniques et les fonctionnalités fournies par VirtualBox sont utiles pour plusieurs scénarii:

- **Support du système d'exploitation.** Avec VirtualBox, on peut lancer un logiciel écrit pour un système d'exploitation sur un autre (par exemple, un logiciel Windows sur Linux ou un Mac) sans avoir à redémarrer pour l'utiliser. Comme vous pouvez configurer les types de matériel qui devraient être présentés à chaque machine virtuelle, vous pouvez même installer un vieux système d'exploitation comme DOS ou OS/2 dans une machine virtuelle si le matériel de votre ordinateur n'est plus supporté par ce système d'exploitation.

- **Tester et réparer un dommage.** Une fois installée, on peut considérer une machine virtuelle et ses disques virtuels comme un container que vous pouvez de façon arbitraire figer, réveiller, copier, sauvegarder et transporter entre les hôtes.

  Au surplus, avec l'utilisation d'une autre caractéristique de VirtualBox appelée instantané, vous pouvez sauvegarder un état particulier de la machine virtuelle et revenir vers à cet état si nécessaire. De cette façon, vous pouvez expérimenter librement en divers environnements de travail. Si quelque chose ne va pas (comme un logiciel qui ne se comporte pas bien après avoir été installé ou un virus qui a infecté la machine invité), vous pouvez facilement revenir à un dépôt précédent et éviter de devoir sauvegarder et restaurer souvent.

  Vous pouvez créer n'importe quel nombre de dépôts, ce qui vous permet de voyager en arrière et dans le temps des machines virtuelles. Vous pouvez effacer des dépôts alors qu'une VM est en fonction pour récupérer de l'espace disque.

- **Consolidation de l'infrastructure..** La virtualisation peut réduire significativement les coûts de matériel et d'électricité. Aujourd'hui, les serveurs tournent se lancent typiquement avec des charges système très faibles et sont rarement utilisés dans leur plein potentiel. Beaucoup de potentiel du matériel et d'électricité est ainsi gaspillé. Alors, au lieu de lancer plusieurs ordinateurs physiques qui ne sont que partiellement utilisés, vous pouvez englober beaucoup de machines virtuelles au sein de quelques hôtes puissants et équilibrer les charges entre eux.

  Avec VirtualBox, vous pouvez même lancer des machines virtuelles en tant que purs serveurs pour le VirtualBox Remote Desktop Protocol (VRDP) (protocole de bureau distant de VirtualBox), avec un support complet du client USB. Cela permet de consolider les machines de bureau dans une entreprise sur à peine quelques serveurs RDP, tandis que les clients finals ne doivent qu'être capables d'afficher des données VRDP.

- **Installations plus faciles de logiciels.** Les machines virtuelles peuvent être utilisées par des vendeurs de logiciels pour livrer porter des configurations globales entières de logiciels. Par exemple, la solution d'installer un serveur de messagerie complet sur une machine réelle peut être une tâche ennuyeuse. Avec la virtualisation, il devient possible de livrer une solution logicielle globale entière, consistant éventuellement en plusieurs composants différents, dans une machine virtuelle, ce qui s'appelle alors le plus souvent l'appliance. Installer et lancer un serveur de messagerie devient aussi facile que d'importer un tel service (appliance) dans VirtualBox.

1.2 Un peu de terminologie

Quand on parle de virtualisation (et aussi pour comprendre les chapitres suivants de cette documentation), il est utile de se familiariser un peu avec une terminologie fondamentale, surtout les termes suivants:

Système d'exploitation hôte (host OS): le système d'exploitation de l'ordinateur physique sur lequel VirtualBox a été installé. Il existe des versions de VirtualBox pour les hôtes Windows, Mac OS X, Linux et Solaris; pour plus de détails, merci de consulter le chapitre 1.4, *Systèmes d'exploitation hôtes supportés*, page 14. Tandis que les diverses versions de VirtualBox sont

généralement traitées ensemble dans ce document, il peut y avoir des différences propres aux plateformes sur lesquelles nous reviendrons aux endroits adéquats.

Système d'exploitation invité (guest OS): le système d'exploitation qui est exécuté à l'intérieur de la machine virtuelle. En théorie, VirtualBox peut exécuter n'importe quel système d'exploitation x86 (DOS, Windows, OS/2, FreeBSD, OpenBSD), mais pour obtenir du code invité les performances les plus proches d'une installation native sur votre machine, nous avons dû effectuer un grand nombre d'optimisations spécifiques à certains systèmes d'exploitation. Donc si votre système d'exploitation *peut* s'exécuter comme un invité, nous supportons et optimisons officiellement quelques uns qui sont sélectionnés (ce qui comprend néanmoins la plupart de ceux courants).

Voir chapitre 3.1, *Systèmes d'exploitation invités supportés*, page 43 pour des détails.

Machine virtuelle (VM). Lorsqu'elle est exécutée, une VM est l'environnement spécial que VirtualBox crée pour votre système d'exploitation invité. Donc, en d'autres termes, vous lancez votre système d'exploitation invité dans une VM. Normalement, une VM apparaîtra comme une fenêtre sur le bureau de votre ordinateur, mais selon le front-ends de VirtualBox que vous utilisez, il peut être affiché en mode plein écran ou à distance en utilisant le VirtualBox Remote Desktop Protocol (VRDP).

Parfois nous utilisons aussi le terme machine virtuelle d'une façon plus abstraite: en interne, VirtualBox considère comme une VM un ensemble de paramètres qui déterminent son comportement. Ils comprennent tant les réglages matériels (combien de mémoire doit avoir la VM, quels disques durs VirtualBox doit virtualiser parmi les containers de fichiers, quel CDs sont montés etc.) que des informations d'état (si la VM est actuellement en fonction, sauvegardée, ses instantanés etc.).

Ces paramètres se reflètent aussi bien dans l'interface graphique que dans le programme `VBoxManage` en ligne de commande; voir le chapitre chapitre 8, *VBoxManage*, page 110. En d'autres termes, une VM est aussi ce que vous pouvez voir dans la boîte de dialogue de ses paramètres.

Additions invité. Avec les Additions invité, nous faisons référence à des paquets de logiciel spéciaux qui sont fournis avec VirtualBox. Même s'ils font partie de VirtualBox, ils sont faits pour être installés à l'intérieur d'une VM pour améliorer la performance de l'OS invité et pour ajouter des fonctionnalités supplémentaires. Ceci est décrit en détail au chapitre chapitre 4, *Additions invité*, page 58.

1.3 Aperçu des fonctionnalités

Voici un bref résumé des principales fonctionnalités de VirtualBox:

- **Portabilité.** VirtualBox s'exécute sur un grand nombre de systèmes d'exploitation hôtes 32 bits et 64 bits (de nouveau, voir le chapitre 1.4, *Systèmes d'exploitation hôtes supportés*, page 14 pour des détails). VirtualBox est ce qu'on appelle un hyperviseur hébergé (parfois désigné comme un hyperviseur de type 2). Alors qu'un hyperviseur bare-metal ou de type 1 s'exécuterait directement sur du matériel, VirtualBox exige qu'un système d'exploitation soit installé. Il peut ainsi s'exécuter avec aux côtés d'applications existantes sur cet hôte.

  Dans une large mesure, VirtualBox est, au plan fonctionnel, identique sur toutes les plateformes hôtes et les mêmes formats de fichiers et d'images sont utilisés. Cela vous permet de lancer des machines virtuelles créées sur un hôte sur un autre hôte au système d'exploitation hôte différent; par exemple vous pouvez créer une machine virtuelle sur Windows puis la lancer sous Linux.

  En outre, les machines virtuelles peuvent être importées et exportées facilement en utilisant l'Open Virtualization Format (OVF, voir chapitre 1.12, *Importer et exporter des machines*

virtuelles, page 29), une technologie standard créée à cette fin. Vous pouvez même importer des OVFs qui ont été créés avec un logiciel de virtualisation différent.

- **Pas besoin de virtualiser du matériel.** Dans la plupart des scénarii, VirtualBox n'exige pas que les fonctionnalités du processeur soit construites dans le nouveau matériel comme Intel VT-x ou AMD-V. Contrairement à bien d'autres solutions de virtualisation, vous pouvez ainsi utiliser VirtualBox même sur du matériel ancien où ces fonctionnalités ne sont pas présentes. Vous pouvez trouver plus de détails au chapitre chapitre 10.3, *Hardware vs. software virtualization*, page 175.

- **Additions invité: dossiers partagés, fenêtres intégrées, virtualisation 3D.** Les additions invité de VirtualBox sont des paquets de logiciels qui peuvent être installés *à l'intérieur* des systèmes invité supportés pour améliorer leur performance et pour offrir une meilleure intégration supplémentaire et une communication avec le système hôte. Après avoir installé les additions invité, une machine virtuelle supportera l'ajustement automatique des résolutions graphiques, les fenêtres intégrées seamless, l'accélération 3D graphique et davantage. Les additions invité sont décrits en détail au chapitre chapitre 4, *Additions invité*, page 58.

  En particulier, les additions invité offrent les répertoires partagés qui vous permettent d'accéder à des fichiers à partir du système hôte à partir de l'intérieur de la machine invitée. Les dossiers partagés sont décrits au chapitre chapitre 4.3, *Dossiers partagés*, page 68.

- **Excellent support matériel.** Entre autres, VirtualBox supporte:
  - **Multitâches invité (SMP).** VirtualBox peut présenter jusqu'à 32 processeurs virtuels à une machine virtuelle, indépendamment du nombre de cœurs présents effectivement sur le processeur de votre hôte.
  - **Support du périphérique USB 2.0.** VirtualBox implémente un contrôleur USB virtuel et vous permet de connecter à vos machines virtuelles des périphériques USB de votre choix sans devoir installer de pilotes spécifiques aux périphériques sur l'hôte. Le support USB n'est pas limité à certaines catégories de périphériques. Pour des détails, voir le chapitre 3.10.1, *Paramètres USB*, page 54.
  - **Compatibilité matérielle.** VirtualBox virtualise une large gamme de périphériques, dont beaucoup de périphériques qui sont en général fournis par d'autres plateformes de virtualisation. Cela inclut les contrôleurs de disque IDE, SCSI et SATA, plusieurs cartes réseau et cartes sons virtuelles, ports série et parallèle virtuels et Input/Output Advanced Programmable Interrupt Controller (I/O APIC) (contrôleur d'interruption programmable avancé entrée/sortie), que l'on trouve dans beaucoup de systèmes PC modernes. Cela facilite le clonage d'images de PC à partir de machines réelles et l'importation de machines virtuelles tierces dans VirtualBox.
  - **Support ACPI complet.** L'Advanced Configuration and Power Interface (ACPI) (interface de configuration et d'énergie avancée) est pleinement supportée par VirtualBox. Avec son **support unique de statut d'énergie ACPI**, VirtualBox peut même signaler à des systèmes d'exploitation invités gérant l'ACPI le statut de l'énergie de l'hôte. Pour les systèmes portables fonctionnant sur batterie, l'invité peut ainsi activer l'économie d'énergie et notifier l'utilisateur de la charge restante (par exemple en modes plein écran).
  - **Résolutions de plusieurs écrans..** Les machines virtuelles de VirtualBox supportent les résolutions d'écran autant importantes qu'avec un écran physique, leur permettant d'être étendus à un grand nombre d'écrans attachés au système hôte.
  - **Support iSCSI inclu.** Cette fonctionnalité unique vous permet de connecter une machine virtuelle directement à un serveur de stockage iSCSI sans passer par le système hôte. La VM accède la cible iSCSI directement sans la charge intermédiaire requise pour des disques durs de virtualisation dans des fichiers de container. Pour des détails, voir chapitre chapitre 5.11, *Serveurs iSCSI*, page 90.

  - **Amorçage par réseau PXE.** Les cartes réseau virtuelles intégrées de VirtualBox supportent complètement l'amorçage à distance à travers le Preboot Execution Environment (PXE (environnement d'exécution préamorçage).

- **Multigénération de prises branchées.** VirtualBox peut sauvegarder des instantanés de votre choix de l'état de la machine virtuelle. Vous pouvez revenir en arrière au moment voulu et rétablir la machine virtuelle dans n'importe quel des dépôts et démarrer une configuration de VM alternative à partir de là, en créant de fait une arborescence de dépôts. Pour des détails, voir le chapitre 1.8, *Instantanés*, page 24. Vous pouvez supprimer un instantané alors que la VM tourne.

- **Architecture propre; modularité sans précédent.** VirtualBox a un aspect extrêmement modulaire avec des interfaces de programmation internes bien définies et une séparation claire du code client du code serveur. Cela facilite son contrôle à partir de plusieurs interfaces en une seule fois: par exemple, vous pouvez démarrer une VM simplement en cliquant sur un bouton dans l'interface graphique de VirtualBox puis contrôler cette machine à partir de la ligne de commande ou même à distance. Voir le chapitre chapitre 1.13, *Interfaces alternatives*, page 30.

  Du fait de son architecture modulaire, VirtualBox peut montrer toutes ses fonctionnalités et son caractère configurable à travers un **kit de développement de logiciel (SDK)** complet, ce qui permet d'intégrer tous les aspects de VirtualBox à d'autres systèmes de logiciels. Merci de voir le chapitre chapitre 11, *VirtualBox programming interfaces*, page 181 pour des détails.

- **Affichage de machine distante.** Vous pouvez lancer n'importe quelle machine virtuelle dans un programme VirtualBox spécial qui agit comme un serveur pour le protocole de bureau distant de VirtualBox ((VRDP), une extension rétro-compatible en fond du protocole de bureau distant. Avec cette fonctionnalité unique, VirtualBox fournit un accès à distance à haute performance à n'importe quelle machine virtuelle.

  Le support VRDP de VirtualBox ne se base pas sur le serveur RDP qui est construit sur Microsoft Windows. Un serveur VRDP personnalisé a plutôt été directement construit dans la couche (layer) de virtualisation. Il en résulte qu'il fonctionne avec n'importe quel système d'exploitation (même en mode texte) et n'exige pas le support d'une application dans l'autre machine virtuelle.

  Le support VRDP est décrit en détail au chapitre chapitre 7.1, *Remote display (VRDP support)*, page 101.

  En plus de cette fonctionnalité particulière, VirtualBox vous offre d'autres fonctionnalités uniques:

  - **Authentification RDP extensible.** VirtualBox supporte déjà Winlogon sur Windows et PAM sur Linux l'authentification RDP. En outre, il inclut un SDK facile à utiliser qui vous permet de créer des interfaces de votre choix pour d'autres méthodes d'authentification; voir le chapitre 7.1.5, *RDP authentication*, page 105 pour des détails.
  - **L'USB à travers RDP.** À travers le support de canaux virtuels RDP virtuel de canaux USB, VirtualBox vous permet également de connecter localement des périphériques USB de votre choix sur une machine virtuelle qui est exécutée à distance sur un serveur RDP VirtualBox; voir le chapitre 7.1.4, *Remote USB*, page 104 pour les détails.

1.4 Systèmes d'exploitation hôtes supportés

Actuellement, VirtualBox se lance sur les systèmes d'exploitation hôtes suivants:

- Hôtes **Windows**:
  - Windows XP, tous les packs service (32 bits)
  - Windows Server 2003 (32 bits)
  - Windows Vista (32 bits et 64 bits[1]).
  - Windows Server 2008 (32 bits et 64 bits)
  - Windows 7 (32 bits et 64 bits)
- Hôtes **Mac OS X**:[2]
  - 10.5 (Leopard, 32 bits)
  - 10.6 (Snow Leopard, 32 bits et 64 bits)

  Il faut du matériel Intel; merci de voir le chapitre 14, *Known limitations*, page 201 also.
- Hôtes **Linux** (32 bits et 64 bits[3]). Cela comprend entre autres:
  - Ubuntu 6.06 (“Dapper Drake”), 6.10 (“Edgy Eft”), 7.04 (“Feisty Fawn”), 7.10 (“Gutsy Gibbon”), 8.04 (“Hardy Heron”), 8.10 (“Intrepid Ibex”), 9.04 (“Jaunty Jackalope”), 9.10 (“Karmic Koala”), 10.04 (“Lucid Lynx”).
  - Debian GNU/Linux 3.1 (“sarge”), 4.0 (“etch”) and 5.0 (“lenny”)
  - Oracle Enterprise Linux 4 et 5
  - Redhat Enterprise Linux 4 et 5
  - Fedora Core 4 à 12
  - Gentoo Linux
  - SUSE Linux 9 et 10, openSUSE 10.3, 11.0, 11.1, 11.2
  - Mandriva 2007.1, 2008.0, 2009.1, 2010.0

  Il devrait être possible d’utiliser VirtualBox sur la plupart des systèmes basés sur un noyau Linux 2.6 en utilisant soit l’installeur VirtualBox, soit en effectuant une installation manuelle; voir chapitre 2.3, *Installation sur des hôtes Linux*, page 34.

  Remarquez que à partir de VirtualBox 2.1, les systèmes d’exploitation basés sur Linux 2.4 ne sont plus supportés.
- Les hôtes **Solaris** (32 bits et 64 bits[4]) sont supportés avec les restrictions listées au chapitre 14, *Known limitations*, page 201:
  - OpenSolaris (2008.05 et supérieur, construction “Nevada” et supérieur)
  - Solaris 10 (u5 et supérieur)

1.5 Installer et lancer VirtualBox

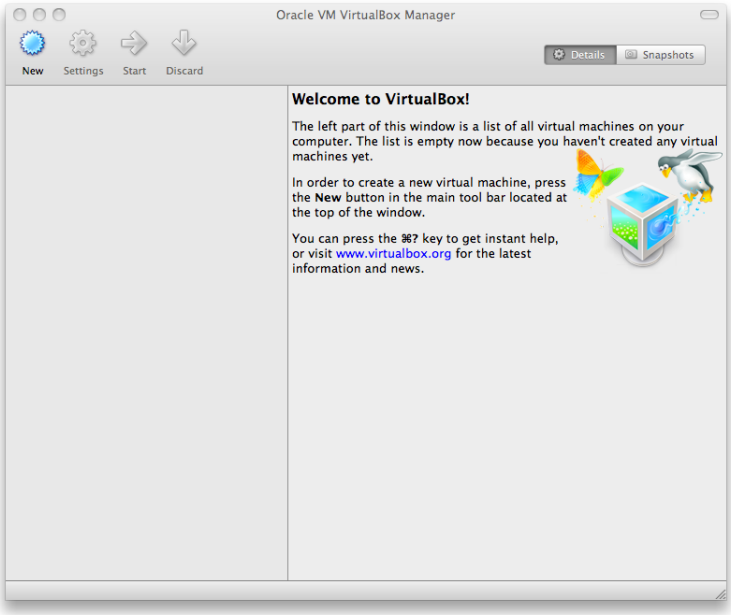
VirtualBox est fourni sous différents paquetages avec beaucoup de paquets différents et son installation dépend de votre plateforme hôte. Si vous avez déjà installé un logiciel, l’installation devrait être transparente puisque selon la plateforme, VirtualBox utilise la méthode d’installation la plus courante et la plus facile à utiliser. Si vous rencontrez un problème ou si vous avez des exigences particulières, merci de vous référer au chapitre 2, *Détails d’installation*, page 32 pour des détails sur les diverses méthodes d’installation.

Après l’installation, vous pouvez démarrer VirtualBox comme suit:

[1] Le support de Windows 64 bits a été ajouté avec VirtualBox 1.5.
[2] Le support préliminaire de Mac OS X (au stade beta) a été ajouté avec VirtualBox 1.4, le support complet avec la 1.6. Le support de Mac OS X 10.4 (Tiger) a été supprimé avec VirtualBox 3.1.
[3] Le support de Linux 64 bits a été ajouté à VirtualBox 1.4.
[4] Le support d’OpenSolaris a été ajouté avec VirtualBox 1.6.

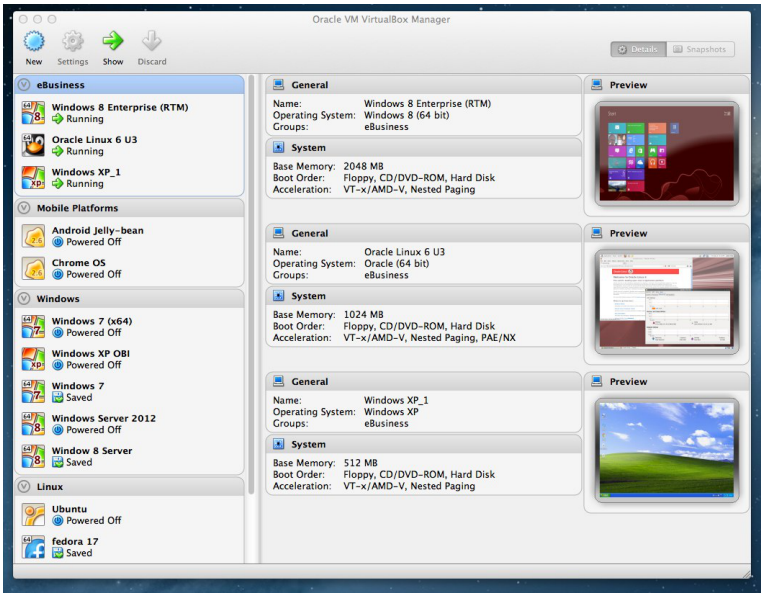
- Sur un hôte Windows, dans le menu Programmes habituel, cliquez sur l'icône dans le groupe VirtualBox. Sur Vista ou Windows 7, vous pouvez aussi taper VirtualBox dans la boîte de recherche du menu Démarrer.
- Sur un hôte Mac OS, dans le Finder, double-cliquez sur l'icône VirtualBox dans le dossier Applications. (Vous pouvez déplacer cet icône sur votre bureau Dock.)
- Sur un hôte Linux ou Solaris, selon votre environnement de bureau, un icône VirtualBox peut avoir été placé soit dans le groupe Système soit dans celui Outils système de votre menu Applications. Vous pouvez aussi taper `VirtualBox` dans un terminal.

Quand vous lancez VirtualBox pour la première fois, une fenêtre comme celle qui suit devrait apparaître:



Sur la gauche, vous pouvez voir un panneau qui listera plus tard toutes vos machines virtuelles. Comme vous n'en avez pas créées, la liste est vide. Une groupe de boutons au-dessus d'elle vous permet de créer de nouvelles VMs et de travailler sur les VMs existantes, une fois que vous en avez. Le panneau à droite affiche les propriétés de la machine virtuelle actuellement sélectionnée s'il y en a. De même, comme vous n'avez pas encore créé de machine, le panneau affiche un message de bienvenue.

Pour vous donner une idée de ce à quoi VirtualBox peut ressembler plus tard, après avoir créé beaucoup de machines, voici un autre exemple:



1.6 Créer votre première machine virtuelle

Cliquez sur le bouton Nouveau en haut de la fenêtre VirtualBox. Un assistant va apparaître pour vous guider à travers le paramétrage d'une nouvelle machine virtuelle (VM):

Dans les pages qui suivent, l'assistant vous demandera le minimum d'informations nécessaires pour créer une VM, en particulier:

1. Un **nom** pour votre VM et le **type de système d'exploitation** (OS) que vous voulez installer.

   Le nom est celui que vous verrez plus tard dans la fenêtre principale de VirtualBox et sous lequel vos paramètres vont se stocker. Il est purement informatif, mais une fois que vous avez créé quelques VMs, vous apprécierez d'avoir donné des noms de VMs parlants. Ma VM n'est ainsi probablement pas aussi utile que Windows XP SP2.

   Pour le Type de système d'exploitation, sélectionnez le système d'exploitation que vous voulez installer plus tard. En fonction de votre sélection, VirtualBox activera ou désactivera

certains paramètres que votre système d'exploitation invité peut exiger. C'est particulièrement important pour les invités 64 bits (voir le chapitre 3.1.2, *Invités 64 bits*, page 45). Il est ainsi recommandé de toujours mettre la bonne valeur.

2. La **taille de la mémoire (RAM)** de la machine virtuelle pour elle-même. Chaque fois qu'une machine virtuelle démarre, VirtualBox allouera cette mémoire à partir de votre machine hôte et la présentera au système d'exploitation, qui verra cette mémoire comme la RAM installée de l'ordinateur (virtuel).

   Note: Faites attention en choisissant ce paramètre. La mémoire que vous donnez à la VM ne sera pas disponible pour votre OS hôte tant que la VM sera en fonction, donc ne spécifiez pas plus que ce dont vous pouvez disposer. Par exemple, si votre machine hôte a 1 Gio de RAM et si vous entrez 512 Mio comme quantité de RAM pour une machine virtuelle en particulier, pendant que la VM sera en fonction, vous n'aurez plus que 512 Mio pour tous les autres logiciels sur votre hôte. Si vous lancez deux VMs en même temps, plus de mémoire sera alloué pour la seconde VM (qui peut même ne pas être en mesure de démarrer si cette mémoire n'est pas disponible). D'un autre côté, vous devriez spécifier autant de mémoire que l'OS invité (et vos applications) exigeront pour s'exécuter correctement.

   Un invité Windows XP exigera au moins quelques centaines de Mio de RAM pour s'exécuter correctement, et Windows Vista va même refuser de s'installer avec moins de 512 Mio. Bien sûr, si vous voulez lancer des applications gourmandes en graphique dans votre VM, vous pouvez avoir besoin d'encore plus de RAM.

   Alors, la règle de conduite est que si vous avez 1 Gio de RAM ou plus dans votre ordinateur hôte, il est en général sûr d'allouer 512 Mio à chaque VM. Mais dans tous les cas, assurez-vous toujours de libérer au moins 256 à 512 Mio de RAM sur votre système d'exploitation hôte. Sans cela vous pourriez obliger votre système à utiliser la mémoire d'échange de votre disque dur de façon excessive, ce qui ferait planter votre système.

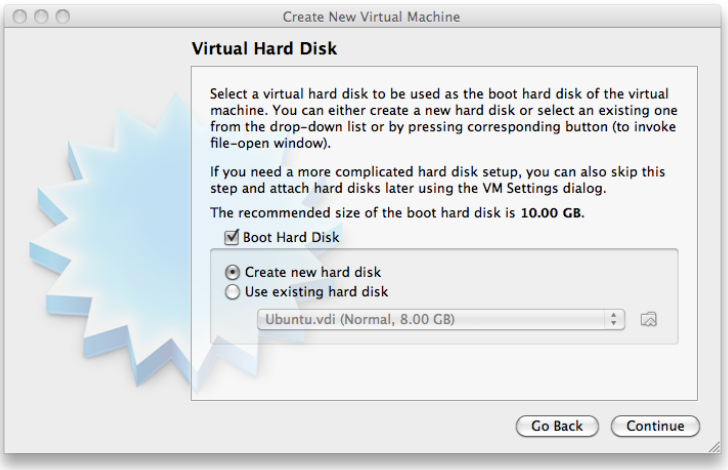
   Note: VirtualBox limite la quantité de RAM invitée à 1500 Mio sur les hôtes Windows 32 bits et à 2560 Mio sur les hôtes Linux et Solaris 32 bits, à cause des limites de l'espace d'adressage. Ces restrictions ne s'appliquent pas aux hôtes 64 bits.

   Comme tous les paramètres, vous pouvez modifier ce paramètre plus tard, après avoir créé la VM.

3. Ensuite, vous devez spécifier un **disque dur virtuel** pour votre VM.

   Il y a plusieurs façons, potentiellement complexes, de fournir un espace de disque dur à une VM (voir le chapitre 5, *Stockage virtuel*, page 78 pour des détails), mais la plus courante est d'utiliser un gros fichier image sur votre disque dur réel dont VirtualBox présente le contenu à votre VM comme s'il s'agissait d'un disque dur complet.

   L'assistant vous montre la fenêtre suivante:



L'assistant vous permet de créer un fichier image ou d'en utiliser un existant. Remarquez aussi que les images de disque peuvent être distinctes d'une VM particulière, donc même si vous effacez une VM, vous pouvez garder l'image, ou la copier vers un autre hôte et lui créer une nouvelle VM dessus.

Dans l'assistant, vous avez les options suivantes:

- Si vous avez préalablement créé des disques durs qui n'ont pas été rattachés à d'autres machines virtuelles, vous pouvez les sélectionner à partir de la liste déroulante dans la fenêtre de l'assistant.
- Sinon, pour créer un nouveau disque dur virtuel, appuyez sur le bouton **"Nouveau"**.
- Enfin, pour des opérations plus compliquées avec les disques virtuels, le bouton **"Existant..."** lancera le gestionnaire de médias virtuels, qui est décrit de façon plus détaillée au chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82.

Très probablement, si vous utilisez VirtualBox pour la première fois, vous voudrez créer une nouvelle image de disque. Pour cela, appuyez sur le bouton Nouveau.

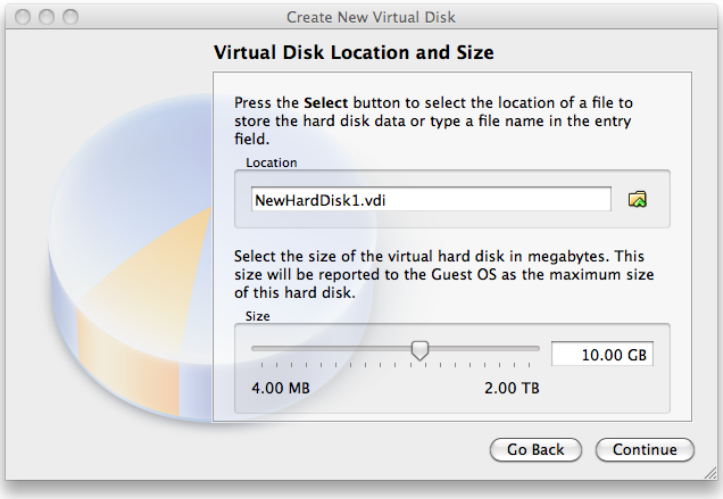
Cela ouvre une nouvelle fenêtre, l'**"assistant de création de nouveaux disques virtuels".**

VirtualBox supporte deux types de fichiers images:

- Un **fichier extensible de façon dynamique** ne grossira que lorsque l'invité stockera réellement les données sur son disque dur virtuel. Il sera donc au début petit sur le disque dur de l'hôte et ne grossira jusqu'à la taille spécifiée seulement que quand on lui fournira des données.
- Un **fichier à taille fixe** occupera immédiatement le fichier spécifié, même si une fraction seulement de l'espace disque dur virtuel est utilisée en réalité. Tout en occupant beaucoup plus d'espace, un fichier à taille fixe nécessite moins d'accès et s'avère donc légèrement plus rapide qu'un fichier extensible de façon dynamique.

Pour des détails sur les différences, merci de vous reporter au chapitre 5.2, *Fichiers images de disque (VDI, VMDK, VHD, HDD)*, page 81.

Pour empêcher que votre disque dur physique ne soit plein, VirtualBox limite la taille d'une image de fichier. Elle doit toutefois être assez grande pour accueillir le contenu de votre système d'exploitation et les applications que vous souhaitez installer – pour un invité Windows ou Linux récent, vous aurez probablement besoin de plusieurs gigaoctets pour toute utilisation sérieuse:



Après avoir sélectionné ou créé votre fichier image, appuyez à nouveau sur **"Suivant"**, pour aller à la page suivante.

4. Après avoir cliqué sur **"Terminer"**, votre nouvelle machine virtuelle sera créée. Vous la verrez alors dans la liste du côté gauche de la fenêtre principale, avec le nom que vous avez entré.

1.7 Lancer votre machine virtuelle

Vous allez maintenant voir votre nouvelle machine dans la liste des machines virtuelles, sur la gauche de la fenêtre principale de VirtualBox. Pour démarrer la machine virtuelle, double-cliquez simplement dessus, ou sélectionnez-la puis appuyez sur le bouton Lancer en haut.

Ceci ouvre une nouvelle fenêtre et la machine virtuelle sélectionnée va démarrer. Tout ce que vous devriez voir normalement sur le moniteur du système virtuel est montré dans la fenêtre, comme vous pouvez le voir dans l'image au chapitre 1.2, *Un peu de terminologie*, page 11.

Puisque c'est la première fois que exécutez cette VM, un autre assistant apparaîtra pour vous aider à sélectionner un média d'installation. Puisque la VM est créée vide, elle devrait se comporter comme un ordinateur réel sans système d'exploitation installé: elle ne fera rien et affichera un message d'erreur selon lequel elle ne peut pas démarrer de système d'exploitation.

C'est pourquoi l' assistant premier démarrage vous aide à sélectionner un média de système d'exploitation à partir duquel on peut installer un système d'exploitation. Dans la plupart des cas, soit il s'agira d'un CD ou d'un DVD réel (VirtualBox peut alors configurer la machine virtuelle pour utiliser le lecteur de votre hôte), soit vous pourriez avoir l'image ISO d'un CD ou d'un DVD sous la main, que VirtualBox peut alors présenter à la machine virtuelle.

Dans les deux cas, après avoir fait vos choix dans l'assistant, vous pourrez installer votre système d'exploitation.

En général, vous pouvez vraiment utiliser votre machine virtuelle comme vous utiliseriez un ordinateur réel. Quelques points méritent cependant d'être mentionnés.

1.7.1 Support du clavier et de la souris dans les machines virtuelles

1.7.1.1 1.7.1.1 Capturer et utiliser le clavier et la souris

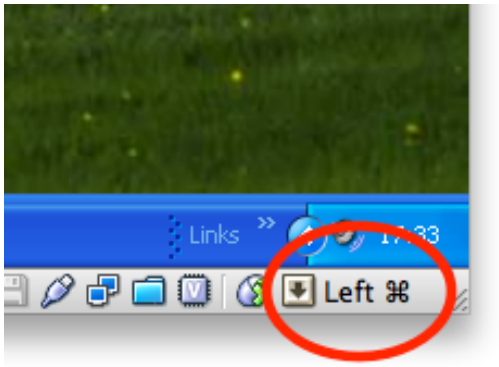
À partir de la version 3.2. VirtualBox fournit un périphérique USB virtuel de tablette aux nouvelles machines virtuelles, à travers laquelle les événements de la souris sont communiqués au système d'exploitation invité. Il s'en suit que si vous lancez un système d'exploitation invité très

récent qui peut gérer de tels périphériques, le support de la souris peut être assuré directement sans que la souris ne soit capturée comme décrit ci-dessous; voir le chapitre chapitre 3.4.1, *Onglet Carte mère*, page 47 pour plus d'informations.

Sinon, si la machine virtuelle ne voit que les périphériques de souris et de clavier PS/2 standards, étant donné que le système d'exploitation de la machine virtuelle ne sait pas qu'il ne fonctionne pas sur un ordinateur réel, il s'attend à bénéficier d'un contrôle exclusif sur votre clavier et votre souris. Mais ce n'est pas le cas puisque, sauf si lancez la VM en mode plein-écran, votre VM a besoin de partager le clavier et la souris avec d'autres applications et, éventuellement, avec d'autres VMs sur votre hôte.

Il s'en suit que, d'emblée après avoir installé un système d'exploitation invité et avant que vous installiez les additions invité (nous vous expliquerons cela dans une minute), seule une des deux – votre VM ou le reste de votre ordinateur – peut « posséder » le clavier et la souris. Vous verrez un second pointeur de souris qui sera toujours confiné aux limites de la fenêtre de la VM. De façon basique, vous activez la VM en cliquant à l'intérieur.

Pour redonner le clavier et de la souris à votre système d'exploitation hôte, VirtualBox réserve une touche spéciale de votre clavier pour cela: la **"touche hôte".** Par défaut, il s'agit de la touche Contrôle droite sur votre clavier ; sur un hôte Mac, la touche hôte par défaut est la touche Commande gauche. Vous pouvez modifier ce paramètre par défaut dans les paramètres globaux de VirtualBox. Dans tous les cas, le réglage actuel de la touche hôte est toujours affiché en bas à droite de la fenêtre de votre VM, au cas où vous l'auriez oubliée:



En détail, tout ceci se transcrit comme suit:

- Votre **clavier** appartient à la VM si la fenêtre de la VM sur votre bureau hôte contient le focus (et aussi, si vous avez plusieurs fenêtres ouvertes dans votre système d'exploitation invité, la fenêtre qui contient le focus dans la VM). Ceci signifie que si vous voulez taper quelque chose à l'intérieur de votre VM, cliquez d'abord sur la barre de titre de votre fenêtre de VM.

  Pour rendre le clavier, appuyez sur la touche hôte (comme expliqué ci-dessus, en général la touche Contrôle droite).

  Remarquez que tandis que la VM dispose du clavier, certaines séquences de touches (comme par exemple Alt-Tab) ne seront plus vues par l'hôte, mais iront à la place vers l'invité. Après que vous ayez appuyé sur la touche hôte pour réactiver le clavier de l'hôte, tous les appuis sur une touche retournent à l'hôte, si bien que les séquences comme Alt-Tab n'atteindront plus l'invité.

- Votre **souris** n'appartient à la VM qu'après que vous ayez cliqué dans la fenêtre de la VM. Le pointeur de la souris hôte va disparaître et la souris va piloter le pointeur de l'invité au lieu de votre pointeur de souris normal.

  Remarquez que l'appartenance de la souris est indépendante de celle du clavier: même après que vous ayez cliqué sur la barre de titre pour pouvoir taper quelque chose à

l'intérieur de la fenêtre de la VM, votre souris n'appartient pas nécessairement encore à la VM.

Pour que la VM rende disponible la souris, appuyez aussi sur la touche Hôte.

Comme ce comportement peut être gênant, VirtualBox fournit un ensemble d'outils et de pilotes de périphérique pour les systèmes invités appelé Additions invité VirtualBox qui rendent les opérations du clavier et de la souris avec la VM beaucoup plus transparentes. En particulier, les additions vont vous débarrasser du deuxième pointeur de souris invité et faire fonctionner le pointeur de votre souris hôte directement dans l'invité.

Ceci sera décrit plus tard au chapitre 4, *Additions invité*, page 58.

1.7.1.2 Entrer des caractères spéciaux

Les systèmes d'exploitation s'attendent à ce que certaines combinaisons de touches démarrent certaines procédures. Il se peut que certaines de ces combinaisons de touches soient difficiles à entrer dans une machine virtuelle, vu qu'il y a trois candidats pour recevoir l'entrée clavier: le système d'exploitation hôte, VirtualBox ou le système d'exploitation invité. Qui des trois reçoit les appuis de touches dépend d'un certain nombre de facteurs, parmi lesquels la touche elle-même.

- Les systèmes d'exploitation hôte réservent certaines combinaisons de touches pour eux. Par exemple, il est impossible d'entrer la combinaison **Ctrl+Alt+Delete** si vous voulez redémarrer le système d'exploitation invité dans votre machine virtuelle, car cette combinaison de touches est en général rattachée à l'OS hôte (Windows comme Linux l'interceptent), et si vous faites cette combinaison de touches, elle redémarrera votre *hôte*.

  En outre, sur Linux et Solaris, qui utilisent le système X Window, la combinaison de touches **Ctrl+Alt+Backspace** réinitialise normalement le serveur X (pour redémarrer toute l'interface graphique au cas où elle plante). Comme le serveur X intercepte cette combinaison, l'appui sur celle-ci va en général redémarrer votre interface graphique hôte (et tuer tous les programmes en fonction, y compris VirtualBox, dans l'opération).

  Troisièmement, sur les hôtes Linux qui supportent les terminaux virtuels, la combinaison de touches **Ctrl+Alt+Fx** (où Fx est une des touches de fonction de F1 à F12) permet normalement de basculer entre les terminaux virtuels. Comme avec Ctrl+Alt+Supp, ces combinaisons sont interceptées par le système d'exploitation hôte et donc, basculent toujours entre les terminaux de l'*hôte*.

  Si vous voulez plutôt envoyer ces combinaisons de touches au système d'exploitation invité dans la machine virtuelle, vous devrez utiliser une des méthodes suivantes:

  - Utilisez les icônes dans le menu Machine de la fenêtre de la machine virtuelle. Vous y trouverez Insertion Ctrl+Alt+Supp et Ctrl+Alt+Effacement; toutefois ces dernières n'auront d'effet qu'avec des invités Linux ou Solaris.
  - Appuyez sur des combinaisons de touches spéciales avec la touche Hôte (normalement la touche Contrôle droite), alors VirtualBox traduira pour la machine virtuelle:
    - **Touche hôte + Supp** pour envoyer Ctrl+Alt+Del (pour redémarrer l'invité);
    - **Touche hôte + Effacement** pour envoyer Ctrl+Alt+Backspace (pour redémarrer l'interface graphique d'un invité Linux ou Solaris);
    - **Touche hôte + F1** (ou d'autres touches de fonction) pour simuler Ctrl+Alt+F1 (ou d'autres touches de fonction, à savoir pour basculer entre des terminaux virtuels dans un invité Linux).

- Pour d'autres combinaisons de touches comme Alt-Tab (pour basculer entre des fenêtres ouvertes), VirtualBox vous permet de configurer si ces combinaisons affecteront l'hôte ou l'invité, si une machine virtuelle contient actuellement le focus. C'est un réglage global pour toutes les machines virtuelles et vous pouvez le trouver sous Fichier -> Préférences -> Entrée -> Auto-capture du clavier.

1.7.2 Changer de média amovible

Pendant qu'une machine virtuelle est en fonction, vous pouvez changer de média amovible dans le menu Périphériques de la fenêtre de la VM. Vous pouvez y sélectionner en détail ce que VirtualBox présente à votre VM comme un CD, un DVD, ou une disquette.

Les paramètres sont les mêmes que ceux qui seraient disponibles pour la VM dans la boîte de dialogue Paramètres de la fenêtre principale de VirtualBox, mais puisque cette boîte de dialogue est désactivée lorsque la VM est dans l'état En fonction ou sauvegardée, ce menu supplémentaire vous évite de devoir arrêter et redémarrer la VM chaque fois que vous voulez changer de média.

Donc, dans le menu « Périphériques », VirtualBox vous permet d'attacher le lecteur hôte à l'invité ou de sélectionner l'image d'une disquette ou d'un DVD en utilisant le gestionnaire d'images de disque, tout comme il est décrit au chapitre 1.9, *Configuration de la machine virtuelle*, page 27.

1.7.3 Resizing the machine's window

You can resize the virtual machine's window when it is running. In that case, one of three things will happen:

1. If you have **"scale mode"** enabled, then the virtual machine's screen will be scaled to the size of the window. This can be useful if you have many machines running and want to have a look at one of them while it is running in the background. Alternatively, it might be useful to enlarge a window if the VM's output screen is very small, for example because you are running an old operating system in it.

   To enable scale mode, press the **host key + C**, or select "Scale mode" from the "Machine" menu in the VM window. To leave scale mode, press the host key + C again.

   The aspect ratio of the guest screen is preserved when resizing the window. To ignore the aspect ratio, press Shift during the resize operation.

   Please see chapitre 14, *Known limitations*, page 201 for additional remarks.

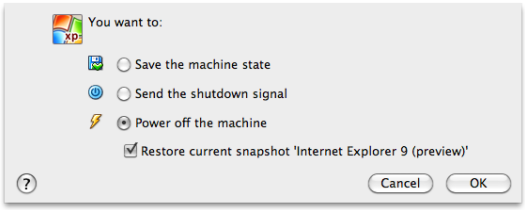
2. If you have the Guest Additions installed and they support automatic **resizing**, the Guest Additions will automatically adjust the screen resolution of the guest operating system. For example, if you are running a Windows guest with a resolution of 1024x768 pixels and you then resize the VM window to make it 100 pixels wider, the Guest Additions will change the Windows display resolution to 1124x768.

   Please see chapitre 4, *Additions invité*, page 58 for more information about the Guest Additions.

3. Otherwise, if the window is bigger than the VM's screen, the screen will be centered. If it is smaller, then scroll bars will be added to the machine window.

1.7.4 Sauvegarder l'état de la machine

Quand vous cliquez sur le bouton Fermer de la fenêtre de votre machine virtuelle (en haut à droite de la fenêtre, comme vous fermeriez n'importe quelle autre fenêtre sur votre système) (ou quand vous appuyez à la fois sur la touche Hôte et Q), VirtualBox vous demande si vous voulez Sauvegarder ou couper la VM.



La différence entre ces trois options est fondamentale. Elles signifient:

- **Sauvegarder l'état de la machine:** Avec cette option, VirtualBox « gèle » la machine virtuelle en sauvegardant complètement son état sur votre disque local. Quand vous restaurerez plus tard la VM (en cliquant à nouveau sur le bouton Démarrer de la fenêtre principale de VirtualBox), vous trouverez que la VM continue exactement là où vous l'aviez quittée. Tous vos programmes seront encore ouverts, et votre ordinateur reprend son travail.

  Sauvegarder l'état de la machine virtuelle est ainsi dans certains cas équivalent à la suspension d'un ordinateur portable (en fermant son écran par exemple).

- **Envoyer le signal de fin.** Ceci enverra le signal de fin ACPI à la machine virtuelle, ce qui a le même effet que si vous aviez appuyé sur le bouton marche/arrêt d'un ordinateur réel. Selon la modernité du système d'exploitation en fonction dans la VM, ceci devrait faire appel au bon mécanisme d'arrêt dans la VM.

- **Couper la machine:** Avec cette option, VirtualBox s'arrête d'exécuter la machine virtuelle, mais *sans* sauvegarder son état.

  Cela revient à retirer la prise électrique d'un ordinateur réel sans l'arrêter correctement. Si vous redémarrez la machine virtuelle après l'avoir coupée, votre système d'exploitation devra redémarrer complètement et il se peut qu'il se lance dans une longue vérification de ses disques systèmes (virtuels).

  Vous ne devriez donc normalement pas le faire, puisque cela peut potentiellement entraîner une perte de données ou un état incohérent du système invité sur le disque.

  Par exception, si votre machine virtuelle a un dépôt (voir le chapitre suivant), vous pouvez utiliser cette option pour restaurer rapidement le dépôt actuel de la machine virtuelle. Seulement dans ce cas, couper la machine n'est pas risqué.

Le bouton **"Désactiver"** de la fenêtre principale de VirtualBox désactive l'état de sauvegarde de la machine virtuelle. Ceci a le même effet que de l'éteindre, et cela produira les mêmes avertissements.

1.8 Instantanés

Avec les instantanés, vous pouvez sauvegarder un état en particulier d'une machine virtuel pour l'utiliser plus tard. À tout moment, vous pourrez plus tard revenir à l'état, même si vous avez énormément changé la VM entre-temps.

Vous pouvez voir les instantanés d'une machine virtuelle en sélectionnant une machine à partir de la liste à gauche dans la fenêtre principale de VirtualBox, puis en sélectionnant l'onglet instantanés à droite. Au départ, jusqu'à ce que vous enregistriez le instantané d'une machine, cette liste est vide sauf pour l'objet « État actuel » qui représente le point Présent dans le temps de la vie de la machine virtuelle.

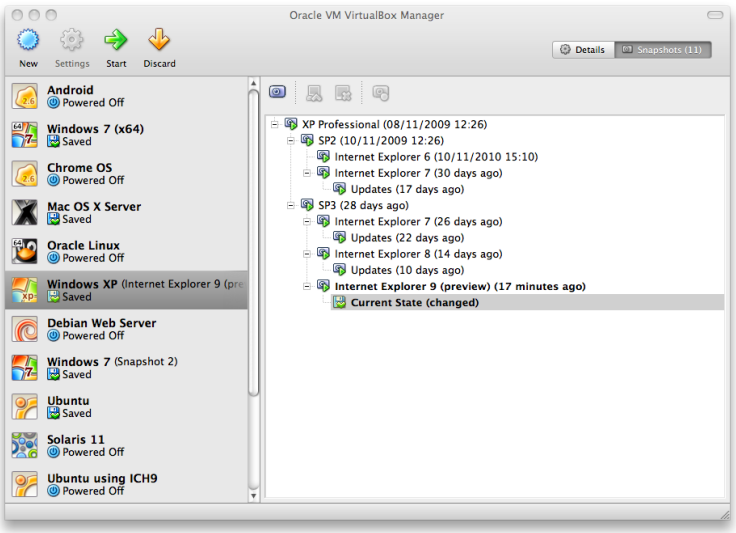
Trois opérations sont liées aux instantanés:

1. Vous pouvez **faire un instantané**.
   - Si votre VM est actuellement en fonction, sélectionnez Prendre un instantané depuis le menu déroulant Machine de la fenêtre de la VM.
   - Si votre VM est actuellement dans l'état Sauvegardée ou Éteinte, (comme affiché à côté de la VM dans la fenêtre principale de VirtualBox), cliquez sur l'onglet instantanés en haut à droite, puis
     - Soit sur l'icône représentant un petit appareil photo (pour Prendre un instantané), soit

– faites un clic droit sur l'icône État actuel dans la liste et sélectionnez Prendre un instantané depuis le menu.

Dans tous les cas, une fenêtre s'affichera vous demandant un nom d'instantané. Ce nom a un but exclusivement de référencement pour vous aider à vous souvenir de l'état du instantané. Par exemple, un nom utile serait Installation toute neuve à partir de rien, aucun lecteur externe. Vous pouvez aussi ajouter un texte plus long dans le champ Description si vous le voulez.

Votre nouvel instantané apparaîtra alors dans la liste des instantanés sous l'onglet instantanés. En-dessous, vous verrez une icône appelée État actuel, signifiant que l'état actuel de votre VM est une variante basée sur le instantané que vous avez pris tout à l'heure. Si vous prenez plus tard un autre instantané, vous verrez qu'ils seront affichés en sections, et que chaque instantané en sous-sections est un dérivé du premier:



VirtualBox vous permet de prendre un nombre illimité de instantanés – la seule limite étant la taille de vos disques. Gardez à l'esprit que chaque instantané stocke l'état de la machine virtuelle et, ainsi, prend de l'espace disque.

2. Vous pouvez **restaurer un instantané** en faisant un clic droit sur un instantané que vous avez pris dans la liste des instantanés. En restaurant un instantané, vous revenez (ou voyagez) dans le temps: l'état actuel de la machine est perdu, et la machine est restaurée exactement dans le même état où elle se trouvait lorsque vous avez pris l'instantané.[5]

Note: La restauration d'un instantané affectera les disques durs virtuels qui sont connectés à votre VM, vu que l'état complet du disque dur virtuel sera rétabli. Cela signifie aussi que tous les fichiers qui ont été créés depuis l'instantané et toutes les autres modifications de fichiers *seront perdus*. Afin d'empêcher une telle perte de données tout en utilisant la fonctionnalité d'instantané, il est possible d'ajouter un deuxième disque dur en mode write-through en utilisant l'interface `VBoxManage` et de l'utiliser pour stocker vos données. Comme les disques durs write-through *ne sont pas* inclus dans les instantanés, il restent inchangés quand une machine est rétablie. Voir le chapitre 5.4, *Modes spéciaux d'écriture d'images*, page 83 pour des détails.

[5] La terminologie et la fonctionnalité de restauration des instantanés a changé avec VirtualBox 3.1. Avant cette version, il était seulement possible de revenir en arrière sur le dernier instantané que vous avez pris – pas ceux précédents, et l'opération s'appelait Désactiver l'état actuel et non Restaurer le dernier instantané. La limitation a été surmontée avec la version 3.1. Il est maintenant possible de restaurer *n'importe quel* instantané, de revenir en arrière et en avant dans le temps.

En restaurant un instantané précédent et en prenant des instantanés à partir de celui-ci, il est même possible de créer une espèce de réalité alternative et de basculer entre ces différents historiques de la machine virtuelle. Il peut en résulter un véritable arbre de instantanés de machine virtuelle, comme montrée dans la capture d'écran ci-dessous.

3. Vous pouvez aussi **effacer un instantané**, ce qui n'affectera pas l'état de la machine virtuelle mais seulement les fichiers du disque que VirtualBox utilisait pour stocker les données du instantané, libérant ainsi de l'espace disque. Pour effacer un instantané, faites un clic droit dessus dans l'arborescence des instantanés et sélectionnez Effacer. Depuis VirtualBox 3.2, vous pouvez effacer des instantanés même lorsqu'une machine est en fonction.

> **Note:** Si prendre et restaurer des instantanés sont des opérations très rapides, effacer un instantané peut prendre un temps considérable étant donné qu'il se peut qu'il faille copier une grande quantité de données entre plusieurs fichiers images de disque. Il se peut que les fichiers de disques temporaires nécessitent beaucoup d'espace disque pendant que l'opération est en cours.

Certaines opérations ne sont pas faisables lorsqu'une VM est en fonction et vous obtiendrez un message spécifique selon lequel vous devez effectuer la suppression du instantané quand la VM sera éteinte.

Considérez un instantané comme un moment dans le temps que vous avez préservé. Plus formellement, un instantané consiste en trois choses:

- Il contient une copie complète des paramètres de la VM, de sorte que quand vous restaurez un instantané, les paramètres de la VM sont également restaurés. (Par exemple, si vous avez changé la configuration du disque dur, ce changement est écrasé quand vous restaurez l'instantané.)

- L'état de tous les disques virtuels attachés à la machine est préservé. Retourner à un instantané signifie que toutes les modifications, bit par bit, qui ont été effectuées sur les disques de la machine seront entièrement annulées.

  (Plus exactement, cela n'est vrai que pour les disques durs virtuels en mode normal. Comme précisé ci-dessus, vous pouvez configurer les disques pour qu'ils se comportent différemment avec les instantanés; voir le chapitre 5.4, *Modes spéciaux d'écriture d'images*, page 83. De façon encore plus formelle et plus exact au plan technique, ce n'est pas le disque virtuel lui-même qui est restauré quand on restaure un instantané. Quand on prend un instantané, VirtualBox crée plutôt des images de différenciation qui ne contiennent que les modifications depuis que l'instantané a été pris, et lorsqu'on restaure l'instantané, VirtualBox envoie cette image de différenciation, revenant ainsi à l'état précédent. Non seulement cela est plus rapide mais cela utilise moins d'espace disque. Pour les détails, qui peuvent être complexes, merci de vous reporter au chapitre chapitre 5.5, *Images de différenciation*, page 85.)

- Enfin, si vous avez pris un instantané alors que la machine était en fonction, l'état de la mémoire de la machine est également sauvegardé dans l'instantané (comme la mémoire peut être sauvegardée quand vous fermez la fenêtre de la VM) de sorte que quand vous restaurerez l'instantané, l'exécution reprend très exactement à l'endroit où l'instantané a été pris.

1.9 Configuration de la machine virtuelle

Quand vous sélectionnez une machine virtuelle à partir de la liste dans la fenêtre principale de VirtualBox, vous verrez un résumé des paramètres de cette machine à droite sur la fenêtre, sous l'onglet Détails.

Cliquer sur le bouton Paramètres de la barre d'outil en haut de la fenêtre principale de VirtualBox ouvre une fenêtre détaillée où vous pouvez configurer un grand nombre de propriétés de la VM actuellement sélectionnée. Mais soyez prudent: même s'il est possible de modifier tous les paramètres de la VM après avoir installé un système d'exploitation invité, certaines modifications pourraient empêcher un système d'exploitation invité de fonctionner correctement si cela est fait après l'installation.

> **Note:** Le bouton Paramètres est désactivé quand une VM est dans l'état En fonction ou sauvegardé. Ceci simplement car la boîte de dialogue de paramètres vous permet de modifier des aspects fondamentaux de l'ordinateur virtuel créé pour votre système d'exploitation, et il se peut que ce système d'exploitation gère mal le fait que, par exemple, on lui enlève la moitié de sa mémoire sous les pieds. Il s'en suit que si le bouton Paramètres est désactivé, coupez d'abord la VM actuelle.

VirtualBox fournit une pléthore de paramètres que vous pouvez modifier pour une machine virtuelle. Les divers paramètres que vous pouvez modifier dans la fenêtre Paramètres sont décrits en détail au chapitre 3, *Configurer des machines virtuelles*, page 43. Et davantage de paramètres sont disponibles avec l'interface en ligne de commande; voir le chapitre 8, *VBoxManage*, page 110.

Pour l'instant, si vous venez de créer une VM vide, vous serez probablement très intéressé par les paramètres présentés dans la section CD/DVD-ROM si vous voulez rendre disponible un CD ou un DVD la première fois que vous la lancerez, afin d'installer votre système d'exploitation invité.

Pour cela, vous avez deux options:

- Si vous avez un média CD ou DVD physique final à partir duquel vous voulez installer votre système d'exploitation invité (comme dans le cas d'un CD ou d'un DVD d'installation de Windows), mettez le média dans le lecteur de CD ou de DVD de votre hôte.

  Puis dans la boîte de dialogue des paramètres, allez à la section CD/DVD-ROM et sélectionnez Lecteur hôte avec la bonne lettre de lecteur (ou, s'il s'agit d'un hôte Linux, le fichier de périphérique). Cela permettra à votre VM d'accéder au média de votre lecteur hôte et vous pourrez effectuer l'installation à partir de là.

- Si vous avez téléchargé un média d'installation sur Internet sous la forme d'un fichier image ISO (le plus vraisemblablement dans le cas d'une distribution Linux), en principe vous graveriez ce fichier sur un CD ou un DVD vide puis procéderiez comme il vient d'être décrit. Mais avec VirtualBox, vous pouvez sauter cette étape et monter le fichier ISO directement. VirtualBox présentera alors ce fichier comme un lecteur CD ou un DVD-ROM à la machine virtuelle, exactement comme avec des images de disque dur virtuel.

  Dans ce cas, dans la boîte de dialogue des paramètres, aller à la section CD/DVD-ROM et sélectionnez Fichier image ISO. Ceci ouvre le gestionnaire de médias virtuels où vous effectuez les étapes suivantes:

  1. Appuyez sur le bouton Ajouter pour ajouter votre fichier ISO à la liste des images enregistrées. Ceci présentera une boîte de dialogue de fichier ordinaire qui vous permet de chercher votre fichier ISO sur votre machine hôte.
  2. De retour à la fenêtre du gestionnaire, sélectionnez le fichier ISO que vous venez d'ajouter et appuyez sur le bouton Sélectionner. Ceci sélectionne le fichier ISO pour votre VM.

Le gestionnaire de médias virtuels est décrit en détail au chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82.

1.10 Effacer des machines virtuelles

Pour supprimer une machine virtuelle dont vous n'avez plus besoin, faites un clic droit dessus dans la liste des machines virtuelles de la fenêtre principale et sélectionnez Supprimer depuis le menu contextuel qui apparaît. Tous les paramètres de cette machine seront perdus.

L'option Supprimer du menu est désactivée lorsqu'une machine est dans l'état Sauvegardé. Pour effacer une telle machine, désactivez d'abord l'état de sauvegardé en appuyant sur le bouton Désactiver.

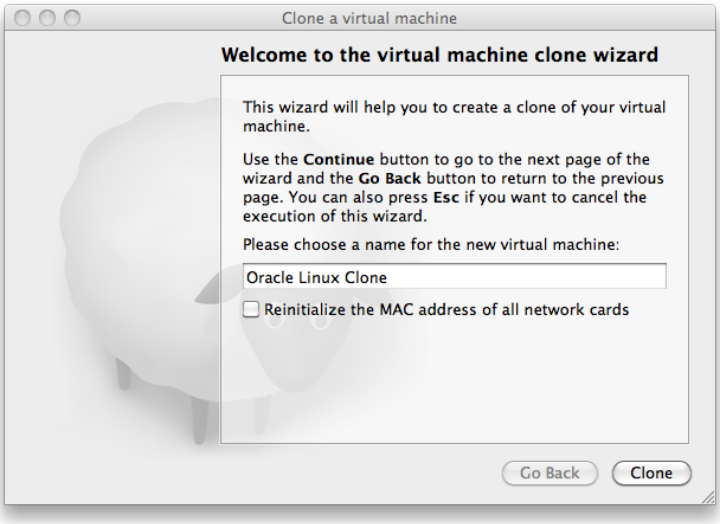
Néanmoins, toutes les images de disque dur attachées à la machine seront conservées; vous pouvez les effacer séparément en utilisant le gestionnaire de médias virtuels; voir le chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82.

Vous ne pouvez supprimer une machine qui a des instantanés ou qui est en état de sauvegardée, donc vous devez d'abord les désactiver.

1.11 Cloning virtual machines

To experiment with a VM configuration, test different guest OS levels or to simply backup a VM, VirtualBox can create a full copy of an existing VM.[6]

A wizard will guide you through the clone process:



This wizard can be invoked from the context menu of the Manager's VM list (select "Clone") or the "Snapshots" view of the selected VM. You have the choice to create a exact copy of the current state without any snapshots or with all snapshots included.

The clone operation itself can be a lengthy operation depending on the size and count of the attached disk images. Also keep in mind that every snapshot has differencing disk images attached, which need to be cloned as well.

The "Clone" menu item is disabled while a machine is running.

For how to clone a VM at the command line, please see chapitre 8.8, *VBoxManage clonevm*, page 127.

[6] Cloning support was introduced with VirtualBox 4.1.

1.12 Importer et exporter des machines virtuelles

À partir de la version 2.2, VirtualBox peut importer et exporter une machine virtuelle au format standard Open Virtualization Format (OVF).

OVF est un standard de plateforme croisée supporté par beaucoup de produits de virtualisation qui permettent de créer des machines virtuelles toutes faites que vous pouvez ensuite importer dans un virtualiseur tel que VirtualBox. Contrairement aux autres produits de virtualisation, VirtualBox supporte maintenant l'OVF avec une interface graphique facile à utiliser ou en utilisant la ligne de commande. Cela permet d'empaqueter ce qu'on appelle des **des outils virtuels**: images de disque avec des paramètres de configuration qui peuvent être distribués facilement. De cette façon, n'importe qui peut proposer des paquets de logiciel prêts à l'emploi (systèmes d'exploitation avec les applications) qui n'exigent pas de configuration ou d'installation autre que l'importation dans VirtualBox.

> **Note:** Le standard OVF est complexe, et son support dans VirtualBox est une processus en perpétuelle évolution. En particulier, il n'est pas garanti que VirtualBox supporte toutes les applications créées par d'autres logiciels de virtualisation. Pour une liste des limites connues, merci de voir le chapitre 14, *Known limitations*, page 201.

Un paquet au format OVF consistera en général en plusieurs fichiers:

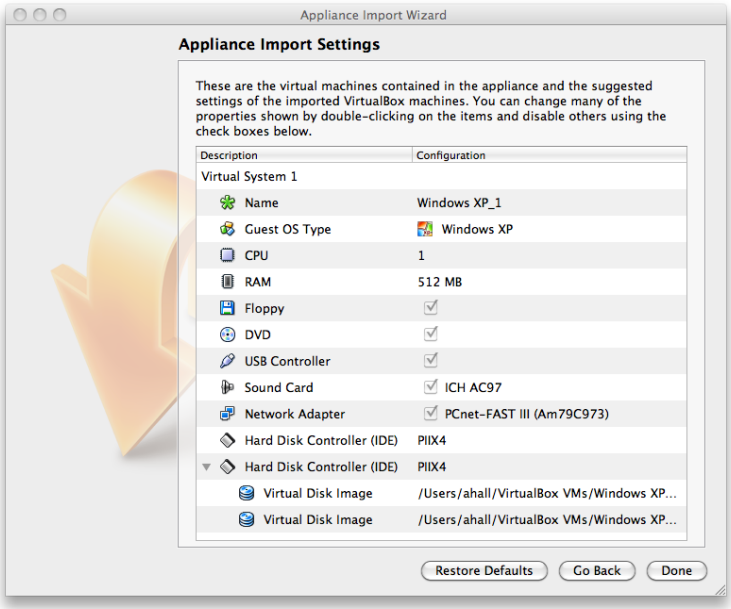
1. une ou plusieurs images de disque, en général au format VMDK largement répandu (voir le chapitre 5.2, *Fichiers images de disque (VDI, VMDK, VHD, HDD)*, page 81) et
2. un fichier de description texte en langage XML avec une extension `.ovf`.

Ces fichiers doivent se trouver dans le même répertoire pour que VirtualBox puisse les importer.

Une version future de VirtualBox supportera aussi les paquets qui comprennent un fichier XML OVF et les images de disque enpaquetées ensemble dans une seule archive.

Pour **importer** une paquet au format OVF, sélectionnez Fichier -> Importer une application depuis la fenêtre principale de l'interface graphique de VirtualBox. Ouvrez alors une boîte de dialogue de fichier et cherchez le fichier texte OVF avec l'extension de fichier `.ovf`.

Si VirtualBox interprète le fichier, une boîte de dialogue similaire à celle qui suit apparaîtra :



Celle-ci présente les machines virtuelles décrites dans le fichier OVF et vous permet de modifier les paramètres de la machine virtuelle en double-cliquant sur les options de description. Une fois que vous cliquez sur Importer, VirtualBox copie les images et crée les machines virtuelles locales avec les paramètres décrits dans la boîte de dialogue. Celles-ci apparaîtront alors dans la liste des machines virtuelles.

Remarquez que vu que les images de disque ont tendance à être grosses, et que les images VMDK fournies avec les applications virtuelles sont en général emballées dans un format compressé qui ne peut pas être utilisé directement par les machines virtuelles, les images devront être d'abord déballées et copiées, ce qui peut prendre quelques minutes.

Pour savoir comment importer une image en ligne de commande, merci de voir le chapitre 8.9, *VBoxManage import*, page 127.

Inversement, pour exporter des machines virtuelles que vous avez dans VirtualBox, sélectionnez les machines et Fichier -> Exporter une application. Une fenêtre de dialogue différente apparaît et vous permet de combiner plusieurs machines virtuelles vers une application OVF. Puis, vous sélectionnez l'emplacement de la cible où les fichiers OVF et VMDK devraient être stockées, et le processus de conversion commence. Ceci peut également prendre du temps.

Pour savoir comment exporter une image en ligne de commande, merci de voir le chapitre 8.10, *VBoxManage export*, page 128.

Note: OVF ne peut pas décrire les instantanés qui ont été pris pour une machine virtuelle. Il en résulte que quand vous exportez une machine virtuelle ayant des instantanés, seul l'état actuel de la machine sera exporté et les images de disque de l'export auront un état « aplati » identique à l'état actuel de la machine virtuelle.

1.13 Interfaces alternatives

Comme indiqué brièvement au chapitre 1.3, *Aperçu des fonctionnalités*, page 12, VirtualBox a un système de présentation interne flexible qui vous permet d'utiliser différentes interfaces pour contrôler les mêmes machines virtuelles. Pour l'illustrer, vous pouvez, par exemple, lancer une machine virtuelle avec l'interface graphique facile à utiliser de VirtualBox puis l'arrêter depuis la ligne de commande. Avec le support de VirtualBox du Remote Desktop Protocol (VRDP) (protocole de bureau à distance), vous pouvez même lancer des machines virtuelles à distance sur un serveur headless et rediriger tous les sorties graphiques à travers le réseau.

En détail, les interfaces suivantes sont incluses dans le paquet VirtualBox standard:

1. `VirtualBox` est notre graphical user interface (GUI) (interface graphique), à la description de laquelle se consacre la plupart de ce manuel de l'utilisateur, en particulier au chapitre chapitre 3, *Configurer des machines virtuelles*, page 43. Bien qu'étant la plus facile de nos interfaces à utiliser, elle ne couvre pas (encore) toutes les fonctionnalités fournies par VirtualBox. Cela reste encore la meilleure façon de connaître VirtualBox au début.

2. `VBoxManage` VBoxManage est notre interface en ligne de commande pour un contrôle automatisé et très détaillé de tous les aspects de VirtualBox. Il est décrit au chapitre chapitre 8, *VBoxManage*, page 110.

3. `VBoxSDL` est une interface alternative, simple et graphique avec des fonctionnalités volontairement limitées, destiné uniquement à afficher des machines virtuelles contrôlées dans le détail par `VBoxManage`. C'est intéressant pour les environnements d'entreprise où l'affichage de toutes les fioritures d'une interface graphique n'est pas souhaitable. `VBoxSDL` est décrit au chapitre 9.1, *VBoxSDL, the simplified VM displayer*, page 148.

4. Enfin, `VBoxHeadless` est une autre interface qui ne produit aucune sortie visible sur l'hôte, mais qui agit simplement comme un serveur VRDP. Maintenant, même si les autres

interfaces graphiques (VirtualBox et VBoxSDL) ont aussi d'intégré le support de VRDP et peuvent agir comme un serveur VRDP, cette interface particulière n'exige aucun support graphique. C'est utile, par exemple, si vous voulez héberger vos machines virtuelles sur un serveur Linux headless n'ayant pas de X Window system installé. Pour des détails, voir le chapitre 7.1.2, *VBoxHeadless, the remote desktop server*, page 102.

Si les interfaces ci-dessus ne satisfont toujours pas vos besoins particuliers, il est relativement indolore de créer une interface au moteur complexe de virtualisation qui est le cur de VirtualBox, étant donné que le cur de VirtualBox présente clairement toutes ses fonctionnalités dans une API claire; merci de vous référer au chapitre 11, *VirtualBox programming interfaces*, page 181.

2 Détails d'installation

Étant donné que l'installation de VirtualBox varie selon votre système d'exploitation hôte, nous fournissons des instructions d'installation dans quatre chapitres distincts, respectivement pour Windows, Mac OS X, Linux et Solaris.

2.1 Installation sur des hôtes Windows

2.1.1 Prérequis

Pour les différentes versions de Windows que nous supportons comme systèmes d'exploitation hôtes, merci de vous référer au chapitre 1.4, *Systèmes d'exploitation hôtes supportés*, page 14.

En outre, l'installateur de Windows 1.1 ou supérieur doit être présent sur votre système. Cela devrait être le cas si vous avez installé toutes les dernières mises à jour.

2.1.2 Effectuer l'installation

Vous pouvez démarrer l'installation de VirtualBox

- soit en double-cliquant sur son fichier exécutable (contenant les architectures 32 bits et 64 bits)
- ou en entrant

  ```
  VirtualBox.exe -extract
  ```

  sur la ligne de commande. Ceci va extraire les deux installateurs dans un répertoire temporaire dans lequel vous trouverez ensuite les fichiers .MSI habituels. Puis, vous pouvez faire un

  ```
  msiexec /i VirtualBox-<version>-MultiArch_<x86|amd64>.msi
  ```

  pour effectuer l'installation.

Dans tous les cas, ceci affichera la boîte de dialogue de bienvenue de l'installation et vous permettra de choisir où installer VirtualBox et quels composants installer. Outre l'application VirtualBox, les composants suivants sont disponibles:

Support USB Ce paquet contient des pilotes spéciaux pour votre hôte Windows dont VirtualBox a besoin pour supporter complètement les périphériques USB à l'intérieur de vos machines virtuelles.

Réseau Ce paquet contient des pilotes supplémentaires de réseau pour votre hôte Windows dont VirtualBox a besoin pour supporter Host Interface Networking (réseau interface hôte) (pour rendre les cartes réseau virtuelles accessibles depuis les autres machines de votre réseau physique).

Support Python Ce paquet contient le support de scripts Python pour l'API de VirtualBox (voir le chapitre chapitre 11, *VirtualBox programming interfaces*, page 181). Pour que cette fonctionnalité soit installée, une installation de Python opérationnelle sur le système est nécessaire.

En fonction de votre configuration de Windows, il se peut que vous voyez des avertissements sur des pilotes non signés ou quelque chose de ce genre. Merci de sélectionner Continuer sur ces avertissements car sans cela, VirtualBox pourrait ne pas fonctionner correctement après l'installation.

L'installateur va créer le groupe VirtualBox dans le répertoire Démarrer Les programmes qui vous permet de lancer l'application et d'accéder à sa documentation.

Avec des paramètres standards, VirtualBox sera installé pour tous les utilisateurs du système local. Au cas où vous ne le voulez pas, vous devez invoquer l'installateur en l'extrayant d'abord en utilisant

```
VirtualBox.exe -extract
```

puis faites comme suit:

```
VirtualBox.exe -msiparams ALLUSERS=2
```

ou

```
msiexec /i VirtualBox-<version>-MultiArch_<x86|amd64>.msi ALLUSERS=2
```

sur les fichiers .MSI extraits. Ceci ne va installer VirtualBox que pour l'utilisateur actuel.

Pour ne pas installer certaines fonctionnalités de VirtualBox, vous pouvez spécifier aussi un paramètre `ADDLOCAL` pour nommer explicitement les fonctionnalités à installer. Les fonctionnalités suivantes sont disponibles:

VBoxApplication Binaires principaux de VirtualBox.

> **Note:** Cette fonctionnalité ne peut jamais être absente puisqu'elle contient l'ensemble de fichiers minimum pour que l'installation de VirtualBox fonctionne!

VBoxUSB Support USB.

VBoxNetwork Tout le support réseau; inclut les fonctionnalités VBoxNetworkFlt et VBoxNetworkAdp (voir ci-dessous).

VBoxNetworkFlt Support réseau bridgé.

VBoxNetworkAdp Support réseau Host-only.

VBoxPython Support Python.

Pour n'installer que le support USB avec les binaires principaux, faites un:

```
VirtualBox.exe -msiparams ADDLOCAL=VBoxApplication,VBoxUSB
```

ou un

```
msiexec /i VirtualBox-<version>-MultiArch_<x86|amd64>.msi ADDLOCAL=VBoxApplication,VBoxUSB
```

2.1.3 Désinstallation

Comme nous utilisons l'installateur Windows, vous pouvez désinstaller VirtualBox en sécurité n'importe quand en choisissant l'entrée du programme dans la fonction Ajout/suppression de programmes dans le panneau de configuration Windows.

2.1.4 Installation sans efforts

Vous pouvez faire des installations sans effort en utilisant le support MSI standard.

2.2 Installation sur des hôtes Mac OS X

2.2.1 Effectuer l'installation

Pour les hôtes Mac OS X, VirtualBox est emballé dans une fichier image de disque (dmg). Effectuez les étapes suivantes:

1. Double-cliquez sur ce fichier pour monter son contenu.
2. Une fenêtre va s'ouvrir vous disant de double-cliquer sur le fichier installateur `VirtualBox.mpkg` affiché dans la fenêtre.
3. Ceci démarrera l'installateur, ce qui vous permettra de sélectionner où installer VirtualBox.

Après l'installation, vous pouvez trouver une icône VirtualBox dans le menu Applications dans le Finder.

2.2.2 Désinstallation

Pour désinstaller VirtualBox, ouvrez à nouveau le fichier image (dmg) et double-cliquez sur l'icône Désinstaller contenue dedans.

2.2.3 Installation sans efforts

Pour effectuer une installation non interactive de VirtualBox, vous pouvez utiliser la version en ligne de commande de l'application installateur.

Montez le fichier image de disque (dmg) comme décrit dans l'installation normale. Puis, ouvrez une session de terminal et exécutez:

```
sudo installer -pkg /Volumes/VirtualBox/VirtualBox.mpkg \
      -target /Volumes/Macintosh\ HD
```

2.3 Installation sur des hôtes Linux

2.3.1 Prérequis

Pour les diverses versions de Linux que nous supportons en tant que systèmes d'exploitation hôtes, merci de vous référer au chapitre 1.4, *Systèmes d'exploitation hôtes supportés*, page 14.

Vous aurez besoin d'installer les paquets suivants sur votre système Linux avant de commencer l'installation (certains systèmes feront cela pour vous automatiquement quand vous installerez VirtualBox):

- Qt 4.4.0 ou supérieur;
- SDL 1.2.7 ou supérieur (cette bibliothèque graphique est en général appelée `libsdl` ou quelque chose du genre).

Note: Pour être précis, ces paquets ne sont nécessaires que si vous voulez lancer les interfaces graphiques de VirtualBox. En particulier, `VirtualBox`, notre interface graphique principale, exige à la fois Qt et SDL; `VBoxSDL`, notre interface graphique simplifiée, n'exige que SDL. À l'inverse, si vous ne voulez qu'exécuter le serveur VRDP headless fourni avec VirtualBox, ni Qt ni SDL ne sont nécessaires.

2.3.2 Le module noyau de VirtualBox

VirtualBox utilise un module noyau spécial pour effectuer l'allocation de la mémoire physique et pour obtenir le contrôle du processeur pour l'exécution du système invité. Sans ce module noyau, vous pourrez travailler avec des machines virtuelles dans l'interface de configuration mais vous ne pourrez lancer aucune machine virtuelle.

Le module noyau de VirtualBox est automatiquement installé sur votre système quand vous installez VirtualBox. Pour le maintenir avec les futures mises à jour du noyau, pour les distributions Linux qui le fournissent – la plupart des distributions actuelles –, nous recommandons d'installer le Dynamic Kernel Module Support (DKMS) (support de module noyau dynamique)[1]. Cet environnement aide à construire des modules de noyau et à gérer les mises à jour du noyau.

Si DKMS n'est pas déjà installé, exécutez une des étapes suivantes:

- Sur un système Ubuntu:

  ```
  sudo apt-get install dkms
  ```

- Sur un système Fedora:

  ```
  yum install dkms
  ```

- Sur un système Mandriva ou Mageia:

  ```
  urpmi dkms
  ```

Si DKMS est disponible et installé, le module noyau de VirtualBox devrait toujours fonctionner automatiquement et il sera automatiquement reconstruit si votre noyau hôte est mis à jour.

Sinon, il n'y a que deux situations où vous devrez vous inquiéter du module noyau:

1. L'installation échoue dès le départ. Cela signifie probablement que votre système Linux n'est pas préparé à construire des modules noyau externes.

   La plupart des distributions Linux peuvent être paramétrées simplement en installant les bons paquets – normalement il s'agira du compilateur GNU (GCC), GNU Make (make) et des paquets contenant les fichiers d'en-tête de votre noyau - et en vous assurant que toutes les mises à jour du système sont installées et que le système exécute le noyau le plus récent inclus dans la distribution. *Les numéros de version des paquets de fichiers d'en-tête doivent être les mêmes que ceux du noyau que vous utilisez.*

   - Avec les versions de Debian et d'Ubuntu, vous devez installer la bonne version de `linux-headers` et, s'il existe, du paquet `linux-kbuild`. Les versions actuelles d'Ubuntu devraient avoir les bons paquets installés par défaut.
   - Dans des versions de Debian et d'Ubuntu encore plus anciennes, vous devez installer la bonne version du paquet `kernel-headers`.
   - Sur les systèmes Fedora et Redhat, le paquet s'appelle `kernel-devel`.
   - Sur SUSE et openSUSE Linux, vous devez installer la bonne version des paquets `kernel-source` et `kernel-syms`.
   - Sinon, si vous avez construit votre propre noyau, `/usr/src/linux` devrait pointer vers les sources de votre noyau. Si vous n'avez pas supprimé les fichiers créés pendant le processus de construction, alors votre système sera déjà paramétré correctement.

2. Le noyau de votre hôte Linux a été mis à jour. Dans ce cas le module noyau aura besoin d'être réinstallé en exécutant (en tant qu'administrateur):

   ```
   /etc/init.d/vboxdrv setup
   ```

[1] Voir http://en.wikipedia.org/wiki/Dynamic_Kernel_Module_Support pour une introduction.

2.3.3 Support de l'USB et du réseau avancé

Afin d'utiliser le support USB de VirtualBox, le compte utilisateur sous lequel vous entendez lancer VirtualBox doit avoir l'accès en lecture et en écriture au système de fichiers USB (`usbfs`).

En outre, l'accès à `/dev/net/tun` sera requis si vous voulez utiliser le Host Interface Networking (réseau d'interface hôte), ce qui est décrit en détail au chapitre 6.4, *Réseau bridgé*, page 95.

2.3.4 Effectuer l'installation

VirtualBox est disponible dans un grand nombre de formats de paquets de base pour de nombreuses distributions Linux (voir le chapitre 1.4, *Systèmes d'exploitation hôtes supportés*, page 14 pour les détails). En outre, il y a un installateur générique alternatif (.run) qui devrait fonctionner sur la plupart des distributions Linux.

2.3.4.1 Installer VirtualBox à partir d'un paquet Debian/Ubuntu

Tout d'abord, téléchargez le paquet adapté à votre distribution. Les exemples suivants supposent que vous installez sur un système 32 bits Ubuntu Karmic. Utilisez `dpkg` pour installer le paquet Debian:

```
sudo dpkg -i VirtualBox-3.2_4.2.12_RPMFusion_Ubuntu_karmic_i386.deb
```

On vous demandera d'accepter la licence VirtualBox Personal Use and Evaluation License (licence d'évaluation et d'utilisation personnelle de VirtualBox). Sauf si vous répondez Yes ici, l'installation sera arrêtée.

Le groupe `vboxusers` sera créé pendant l'installation. Notez qu'un utilisateur qui va exécuter VirtualBox doit être membre de ce groupe. Un utilisateur peut devenir membre du groupe `vboxusers` à travers la gestion des utilisateurs/groupes en graphique ou en ligne de commande avec

```
sudo usermod -a -G vboxusers nom_utilisateur
```

Remarquez aussi qu'ajouter un utilisateur actif à ce groupe exigera que l'utilisateur se déconnecte puis revienne. Cela devrait se faire à la main après une installation réussie du paquet.

L'installateur cherchera aussi un module noyau VirtualBox qui convient à votre noyau. Le paquet comprend des modules précompilés pour la plupart des configurations du noyau. Si aucun module noyau convenable n'est trouvé, le script d'installation essaie de construire un module lui-même. Si le processus de construction ne réussit pas, on vous montrera un avertissement et le paquet sera laissé non configuré. Merci de jeter un il sur `/var/log/vbox-install.log` pour trouver pourquoi la compilation a échoué. Il se peut que vous deviez installer les en-têtes du noyau Linux adéquat (voir le chapitre 2.3.2, *Le module noyau de VirtualBox*, page 35). Après avoir corrigé les problèmes, faites

```
sudo /etc/init.d/vboxdrv setup
```

Ceci démarrera une nouvelle tentative de construire le module.

Si un module noyau convenable est trouvé dans le paquet ou si le module a été construit avec succès, le script d'installation va tenter de charger le module. Si ceci échoue, merci de voir le chapitre chapitre 12.6.1, *Linux kernel module refuses to load*, page 193 pour de plus amples informations.

Une fois que VirtualBox a été installé avec succès et configuré, vous pouvez le démarrer en sélectionnant VirtualBox dans votre menu de démarrage ou depuis la ligne de commande (voir le chapitre 2.3.5, *Démarrer VirtualBox sur Linux*, page 40).

2.3.4.2 Utiliser l'installateur alternatif (VirtualBox.run)

L'installateur alternatif effectue les étapes suivantes:

- Il déballe les fichiers de l'application vers un répertoire cible de votre choix. Par défaut,

  `/opt/VirtualBox/`

  sera utilisé.
- Il construit le module noyau VirtualBox (`vboxdrv`) et l'installe.
- Il crée `/etc/init.d/vboxdrv`, un script de démarrage pour démarrer le module noyau de VirtualBox.
- Il crée un nouveau groupe système appelé `vboxusers`.
- Il crée des liens symboliques vers `VirtualBox`, `VBoxSDL`, `VBoxVRDP`, `VBoxHeadless` et `VBoxManage` dans `/usr/bin`.
- Il crée `/etc/udev/60-vboxdrv.rules`, un fichier de description pour udev, s'il est présent ce qui rend le module accessible à n'importe qui dans le groupe `vboxusers`.
- Il écrit le répertoire d'installation dans `/etc/vbox/vbox.cfg`.

L'installateur doit être exécuté en tant qu'administrateur avec `install` ou `uninstall` en premier paramètre. Si vous ne voulez pas que l'installateur vous demande si vous acceptez ou non l'accord de licence (par exemple pour effectuer une installation sans efforts), vous pouvez ajouter le paramètre `license_accepted_unconditionally`. Enfin, si vous voulez utiliser un autre répertoire que celui d'installation par défaut, ajoutez le chemin désiré en paramètre supplémentaire.

```
sudo ./VirtualBox.run install /opt/VirtualBox
```

Ou si vous n'avez pas de commande sudo disponible, lancez plutôt ce qui suit en tant qu'administrateur:

```
./VirtualBox.run install /opt/VirtualBox
```

Après quoi vous devez mettre tout utilisateur qui devrait pouvoir utiliser VirtualBox dans le groupe `vboxusers`, soit avec les outils graphiques de gestion d'utilisateur soit en lançant la commande suivante en tant qu'administrateur:

```
sudo usermod -a -G vboxusers username
```

Note: La commande `usermod` de certaines distributions Linux anciennes ne supporte pas l'option `-a` (qui ajoute l'utilisateur au groupe donné sans modifier sa qualité de membre d'autres groupes). Dans ce cas, cherchez les appartenances de groupe avec la commande `groups` et ajoutez tous ces groupes dans une liste séparée par des virgules à la ligne de commande après l'option `-G`, comme ceci par exemple: `usermod -G group1,group2,vboxusers nom_utilisateur`.

Si un utilisateur sur votre système devrait pouvoir accéder aux périphériques USB de l'hôte, depuis vos invités VirtualBox, vous devriez aussi les ajouter au groupe utilisateur utilisé par votre distribution pour l'accès USB (comme `usb` ou `usbusers`.

2.3.4.3 Effectuer une installation manuelle

Si pour une raison quelconque vous ne pouvez pas utiliser le script shell d'installation décrit précédemment, vous pouvez aussi effectuer une installation manuelle. Invoquez l'installateur comme ceci:

```
./VirtualBox.run --keep --noexec
```

Cela va déballer tous les fichiers nécessaires à l'installation dans le répertoire `install` sous le répertoire cohérent. Les fichiers de l'application VirtualBox sont contenus dans `VirtualBox.tar.bz2` que vous pouvez déballer dans n'importe quel répertoire sur votre système. Par exemple:

```
sudo mkdir /opt/VirtualBox
sudo tar jxf ./install/VirtualBox.tar.bz2 -C /opt/VirtualBox
```

ou en tant qu'administrateur:

```
mkdir /opt/VirtualBox
tar jxf ./install/VirtualBox.tar.bz2 -C /opt/VirtualBox
```

Les sources du module noyau de VirtualBox sont fournies dans le répertoire `src`. Pour construire le module, allez dans ce répertoire et exécutez

```
make
```

Si tout se construit correctement, exécutez la commande suivante pour installer le module dans le répertoire de modules aqéquats:

```
sudo make install
```

Au cas où vous n'avez pas sudo, allez du compte utilisateur vers celui administrateur et effectuez

```
make install
```

Le module noyau de VirtualBox a besoin d'un nud de périphérique pour fonctionner. La commande make ci-dessus vous dira comment créer le nud de périphérique en fonction de votre système Linux. La procédure est légèrement différente pour une installation Linux classique avec un répertoire `/dev`, un système avec un répertoire `devfs`, maintenant obsolète, et un système Linux moderne avec `udev`.

Sur certaines distributions Linux, vous pourriez rencontrer des difficultés pour construire le module. Vous devrez analyser les messages d'erreur du système de construction pour diagnostiquer la cause des problèmes. De façon générale, assurez-vous d'utiliser les bonnes sources du noyau Linux pour le processus de construction.

Remarquez que l'utilisateur qui va lancer VirtualBox a besoin des droits de lecture et d'écriture sur le nud de périphérique du module noyau `/dev/vboxdrv`. Vous pouvez soit définir un groupe `vboxusers` en entrant

```
groupadd vboxusers
chgrp vboxusers /dev/vboxdrv
chmod 660 /dev/vboxdrv
```

soit, alternativement, simplement donner à tous les utilisateurs l'accès (non sécurisé, non recommandé!)

```
chmod 666 /dev/vboxdrv
```

Vous devriez aussi ajouter les utilisateurs qui seront autorisés à utiliser les périphériques USB de l'hôte des invités de VirtualBox au groupe d'utilisateurs USB adéquat pour votre distribution. Ce groupe s'appelle souvent `usb` ou `usbusers`.

Ensuite, vous devrez installer le script de démarrage système du module noyau:

```
cp /opt/VirtualBox/vboxdrv.sh /etc/init.d/vboxdrv
```

(en supposant que vous avez installé VirtualBox dans le répertoire `/opt/VirtualBox`) et activer le script de démarrage en utilisant la méthode adaptée à votre distribution. Vous devriez créer le fichier de configuration de VirtualBox:

```
mkdir /etc/vbox
echo INSTALL_DIR=/opt/VirtualBox > /etc/vbox/vbox.cfg
```

et, par souci de pratique, créer les liens symboliques suivants:

```
ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VirtualBox
ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxSVC
ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxManage
ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxHeadless
ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxSDL
```

2.3.4.4 Mettre à jour et désinstaller VirtualBox

Avant de mettre à jour ou de désinstaller VirtualBox, vous devez arrêter toutes les machines virtuelles qui sont actuellement en fonction et quitter les applications VirtualBox ou VboxSVC. Pour mettre à jour VirtualBox, lancez simplement l'installateur de la version de mise à jour. Pour désinstaller VirtualBox, appelez l'installateur comme ceci:

```
sudo ./VirtualBox.run uninstall
```

ou en tant qu'administrateur

```
./VirtualBox.run uninstall
```

. À partir de la version 2.2.2, vous pouvez désinstaller le paquet .run en appelant

```
/opt/VirtualBox/uninstall.sh
```

Pour désinstaller VirtualBox à la main, effectuez simplement les étapes d'installation manuelle dans l'ordre inverse.

2.3.4.5 Installation automatique des paquets Debian

Les paquets Debian exigeront quelques retours d'utilisateurs lorsqu'ils seront installés pour la première fois. Le système debconf est utilisé pour effectuer cette tâche. Pour éviter toute action de l'utilisateur pendant l'installation, vous pouvez définir des valeurs par défaut. Un fichier `vboxconf` peut contenir les réglages debconf suivants:

```
virtualbox virtualbox/module-compilation-allowed boolean true
virtualbox virtualbox/delete-old-modules boolean true
```

La première ligne permet de compiler le module noyau vboxdrv si aucun module n'a été trouvé pour le noyau actuel. La deuxième ligne permet au paquet d'effacer n'importe quel ancien module noyau vboxdrv compilé par les installations précédentes.

Ces réglages par défaut peuvent être appliqués avec

```
debconf-set-selections vboxconf
```

avant l'installation du paquet Debian VirtualBox.

Il y a au surplus des options de configuration courantes que vous pouvez régler avant l'installation, décrites au chapitre 2.3.4.7, *Options d'installation automatique*, page 40.

2.3.4.6 Installation automatique de paquets .rpm

Le format .rpm n'offre pas de système de configuration comparable au système debconf. Voir le chapitre 2.3.4.7, *Options d'installation automatique*, page 40 pour savoir comment régler des options d'installation courantes fournies par VirtualBox.

2.3.4.7 Options d'installation automatique

Pour configurer le processus d'installation de nos paquets .rpm et .deb, un fichier `/etc/default/virtualbox` est interprété. La génération automatique de la règle udev peut être empêcher avec le paramétrage suivant:

```
INSTALL_NO_UDEV=1
```

La création du groupe vboxusers peut être empêchée par

```
INSTALL_NO_GROUP=1
```

Si la ligne

```
INSTALL_NO_VBOXDRV=1
```

est spécifiée, l'installateur du paquet n'essaiera pas de construire le module noyau `vboxdrv` si aucun module correspond au noyau actuel n'a été trouvé.

2.3.5 Démarrer VirtualBox sur Linux

La manière la plus facile de démarrer un programme VirtualBox est de lancer le programme de votre choix (`VirtualBox`, `VBoxManage`, `VBoxSDL` ou `VBoxHeadless`), à partir d'un terminal. Ce sont des liens symboliques vers `VBox.sh` qui démarrent le programme requis pour vous.

Les instructions détaillées suivantes ne devraient vous intéresser que si vous souhaitez lancer VirtualBox sans l'installer au préalable. Vous devriez commencer par compiler le module noyau `vboxdrv` (voir ci-dessus) et l'insérer dans le noyau Linux. VirtualBox consiste en un service démon (`VBoxSVC`) et plusieurs programmes d'application. Le démon est automatiquement démarré si nécessaire. Toutes les applications VirtualBox vont communiquer avec le démon par les sockets du domaine local Unix. Il peut y avoir plusieurs instances de démon sous différents comptes utilisateur et les applications ne peuvent communiquer qu'avec le démon qui fonctionne sous le compte de l'utilisateur en tant qu'application. Le socket du domaine local réside dans un sous-répertoire de votre répertoire système pour les fichiers temporaires appelé `.vbox-<nom_utilisateur>-ipc`. En cas de problème de communication ou de démarrage du serveur, vous pouvez essayer de supprimer ce répertoire.

Toutes les applications de VirtualBox (`VirtualBox`, `VBoxSDL`, `VBoxManage` et `VBoxHeadless`) exigent que le répertoire VirtualBox soit dans le chemin de la bibliothèque:

```
LD_LIBRARY_PATH=. ./VBoxManage showvminfo "Windows XP"
```

2.4 Installation sur des hôtes Solaris

Pour les diverses solutions de Solaris que nous supportons comme systèmes d'exploitation hôtes, merci de vous référer au chapitre 1.4, *Systèmes d'exploitation hôtes supportés*, page 14.

Si vous avez une session de VirtualBox préalablement installée sur votre hôte Solaris, merci de la désinstaller avant d'installer une nouvelle session. Reportez-vous au chapitre 2.4.3, *Désinstallation*, page 41 pour les instructions de désinstallation.

2.4.1 Effectuer l'installation

VirtualBox est disponible comme paquet Solaris standard. Téléchargez le paquet VirtualBox SunOS qui comprend à la fois les versions 32 bits et 64 bits de VirtualBox. *Vous devez effectuer l'installation en tant qu'administrateur et depuis la zone globale* car l'installateur de VirtualBox charge des pilotes du noyau, ce qui ne peut pas se faire depuis des zones non globales. Pour vérifier dans quelle zone vous vous trouvez, exécutez la commande `zonename`. Exécutez les commandes suivantes:

```
gunzip -cd VirtualBox-4.2.12_RPMFusion-SunOS.tar.gz | tar xvf -
```

À partir de VirtualBox 3.1, le paquet du noyau VirtualBox n'est plus séparé et a été intégré au paquet principal. Installez le paquet VirtualBox en utilisant:

```
pkgadd -d VirtualBox-4.2.12_RPMFusion-SunOS.pkg
```

> **Note:** Si vous utilisez les Zones Solaris, pour n'installer que dans la zone actuelle et dans aucune autre zone, utilisez `pkgadd -G`. Pour plus d'informations reportez-vous au manuel de `pkgadd`; voir aussi le chapitre 2.4.5, *Configurer une zone pour faire fonctionner VirtualBox*, page 42.

L'installateur vous demandera alors d'entrer le paquet que vous souhaitez installer. Choisissez 1 ou tout et poursuivez. Ensuite, l'installateur vous demandera si vous voulez autoriser le script post installation à se lancer. Choisissez y et poursuivez, vu qu'il est essentiel d'exécuter ce script qui installe le module noyau VirtualBox. Suite à cette confirmation, l'installateur va installer VirtualBox et exécuter le script d'initialisation postinstall.

Une fois que le script postinstall a été exécuté, votre installation est à présent terminée. Vous pouvez maintenant effacer en toute sécurité de votre système le paquet décompressé et les fichiers d'`auto-réponse`. VirtualBox devrait être installé dans `/opt/VirtualBox`.

2.4.2 Démarrer VirtualBox sur Solaris

La façon la plus facile de démarrer un programme VirtualBox est de lancer le programme de votre choix (`VirtualBox`, `VBoxManage`, `VBoxSDL` ou `VBoxHeadless`) à partir du terminal. Ce sont des liens symboliques vers `VBox.sh` qui démarre le programme requis pour vous.

Vous pouvez alternativement invoquer les programmes requis à partir de `/opt/VirtualBox`. L'utilisation des liens fournis est plus facile puisque vous n'êtes pas obligé de taper le chemin complet.

Vous pouvez configurer certains éléments du GUI Qt de `VirtualBox` tels que les polices et les couleurs en exécutant `VBoxQtconfig` depuis le terminal.

2.4.3 Désinstallation

La désinstallation de VirtualBox sur Solaris exige les droits administrateur. Pour effectuer la désinstallation, démarrez une session de terminal administrateur et exécutez:

```
pkgrm SUNWvbox
```

Après confirmation, ceci supprimera VirtualBox de votre système.

Si vous désinstallez la version 3.0 de VirtualBox ou inférieure, vous devez supprimer le paquet d'interface noyau de VirtualBox, exécutez donc:

```
pkgrm SUNWvboxkern
```

2.4.4 Installation sans efforts

Pour effectuer une installation non interactive de VirtualBox, nous avons fourni un fichier de réponse nommé `autoresponse` que l'installateur utilisera pour entrer les réponses aux questions plutôt que de vous les demander.

Extrayez le paquet .tar.gz comme décrit dans l'installation normale. Puis ouvrez une session de terminal administrateur et exécutez:

```
pkgadd -d VirtualBox-4.2.12_RPMFusion-SunOS-x86 -n -a autoresponse SUNWvbox
```

Pour effectuer une désinstallation non interactive, ouvrez une session de terminal administrateur et exécutez:

```
pkgrm -n -a /opt/VirtualBox/autoresponse SUNWvbox
```

2.4.5 Configurer une zone pour faire fonctionner VirtualBox

À partir de VirtualBox 1.6, il est possible de lancer VirtualBox depuis des zones Solaris. Pour une introduction aux zones Solaris, merci de vous référer à http://www.sun.com/bigadmin/features/articles/solaris_zones.jsp.

En supposant que VirtualBox a déjà été installé dans votre zone, vous devez donner à la zone l'accès au noeud de périphérique de VirtualBox. Ceci se fait en effectuant les étapes suivantes. Démarrez un terminal administrateur et exécutez:

```
zonecfg -z vboxzone
```

À l'intérieur de l'invite `zonecfg`, ajoutez la ressource de `périphérique` et les propriétés correspondantes à la zone. Voici comment vous pouvez faire cela:

```
zonecfg:vboxzone>add device
zonecfg:vboxzone:device>set match=/dev/vboxdrv
zonecfg:vboxzone:device>end
zonecfg:vboxzone>verify
zonecfg:vboxzone>exit
```

Si vous exécutez VirtualBox 2.2.0 ou supérieur sur des hôtes OpenSolaris ou Nevada, vous devriez ajouter aussi un périphérique pour `/dev/vboxusbmon`, comme montré ci-dessus. Cela ne s'applique pas aux hôtes Solaris 10 du fait de l'absence du support USB.

Remplacez vboxzone par le nom de la zone 4ans la quelle vous souhaitez faire fonctionner VirtualBox. Ensuite, redémarrez la zone en utilisant `zoneadm` et vous devriez pouvoir lancer VirtualBox depuis l'intérieur de la zone configurée.

3 Configurer des machines virtuelles

Alors que le chapitre 1, *Premières étapes*, page 10 vous a donné une brève introduction à VirtualBox et à la façon de faire fonctionner votre première machine virtuelle, le chapitre suivant décrit en détail comment configurer les machines virtuelles.

Vous disposez d'une latitude considérable pour décider quel matériel virtuel sera fourni à l'invité. Le matériel virtuel peut être utilisé pour communiquer avec le système hôte ou avec d'autres invités. Par exemple, si vous fournissez à VirtualBox l'image d'un CD-ROM dans un fichier ISO, VirtualBox peut présenter cette image à un système invité comme s'il s'agissait d'un CD-ROM physique. De la même façon, vous pouvez donner à un système invité l'accès à un réseau réel par sa carte réseau virtuelle et, si vous le voulez, donner au système hôte, à d'autres invités ou à d'autres ordinateurs sur Internet un accès au système invité.

3.1 Systèmes d'exploitation invités supportés

Comme VirtualBox est conçu pour fournir un environnement de virtualisation générique pour les systèmes x86, il peut exécuter des systèmes d'exploitation de toute sorte, même ceux qui ne sont pas officiellement supportés par Oracle Corporation. Cependant, notre objectif est d'optimiser les performances du produit pour une liste sélectionnée de systèmes invités:

Windows NT 4.0 Toutes les versions/éditions et les packs service sont complètement supportés; néanmoins, il y a quelques problèmes avec les packs services plus anciens. Nous recommandons d'installer le pack service 6a. Les additions invité sont disponibles avec un ensemble de fonctionnalités limité.

Windows 2000 / XP / Server 2003 / Vista / Server 2008 / Windows 7 Toutes les versions/éditions et les packs service sont complètement supportées (y compris les versions 64 bits, sous les conditions listées ci-dessous). Les additions invité sont disponibles.

DOS / Windows 3.x / 95 / 98 / ME Des tests limités ont été effectués. L'utilisation des mécanismes d'installation non régulier n'est pas recommandée. Aucune addition invité disponible.

Linux 2.4 Support limité.

Linux 2.6 Toutes les versions/éditions sont complètement supportées (32 bits et 64 bits). Les additions invité sont disponibles.

Nous vous recommandons fort d'utiliser une version du noyau Linux 2.6.13 ou supérieure pour de meilleures performances.

> **Note:** Certaines versions du noyau Linux ont des bogues qui les empêchent de s'exécuter dans un environnement virtuel; merci de voir le chapitre 12.4.3, *Buggy Linux 2.6 kernel versions*, page 190 pour les détails.

Solaris 10, OpenSolaris Complètement supporté (32 bits et 64 bits). Les additions invité sont disponibles.

FreeBSD Exige qu'on active la virtualisation du matériel. Support limité. Les additions invité ne sont pas encore disponibles.

OpenBSD Exige d'activer la virtualisation du matériel. Les versions 3.7 et supérieures sont supportées. Les additions invité ne sont pas encore disponibles.

OS/2 Warp 4.5 Exige qu'on active la virtualisation du matériel. Nous ne supportons officiellement que MCP2; les autres versions d'OS/2 peuvent ne pas fonctionner. Les additions invité sont disponibles avec un ensemble de fonctionnalités limité.[1]

Mac OS X Server VirtualBox 3.2 a ajouté un support expérimental pour les invités Mac OS X Server, mais il est fourni avec des restrictions. Merci de voir la section suivante ainsi que le chapitre 14, *Known limitations*, page 201.

3.1.1 Invités Mac OS X Server

À partir de la version 3.2, VirtualBox comporte un support expérimental pour les invités Mac OS X Server. Ceci vous permet d'installer et d'exécuter des versions non modifiées de Mac OS X Server sur du matériel hôte supporté.

Alors que les solutions concurrentes apportent des modifications aux DVDs d'installation de Mac OS X Server (comme un chargeur d'amorçage différent et des fichiers remplacés), VirtualBox est le premier produit qui fournit l'architecture PC moderne à laquelle s'attend OS X sans impliquer de bidouilles.

Vous devriez garder en tête un certain nombre de **questions importantes** avant d'essayer d'installer un invité Mac OS X Server:

1. Mac OS X est un logiciel propriétaire, sous licence, et contient des restrictions à la fois de droits et techniques qui limitent son utilisation à certains matériels et certains scénarii. Il est important que vous compreniez et que vous respectiez ces restrictions. Seul Mac OS X Server est conçu pour être utilisé dans un environnement virtuel et, ainsi, VirtualBox ne supporte pas de clients Mac OS X comme invités.

   Il en résulte que, avant d'essayer d'installer Mac OS X Server dans une machine virtuelle, assurez-vous de comprendre les restrictions de droits de la version de Mac OS X que vous voulez utiliser. Pour la plupart des versions de Mac OS X Server, Apple interdit leur installation sur du matériel non Apple.

   Ces restrictions de droits sont implémentées au plan technique: Mac OS X Server vérifie s'il fonctionne sur du matériel Apple et la plupart des DVDs fournis avec le matériel Apple vérifient même le modèle exact. Ces restrictions ne sont *pas* contournées par VirtualBox et continuent à s'appliquer.

2. Seuls les **processeurs** connus et testés par Apple sont supportés. Il en résulte que si le processeur Intel est plus récent que la construction de Mac OS X Server, il tombera très vraisemblablement en erreur lors du démarrage avec un message Unsupported CPU. Il est généralement mieux d'utiliser le DVD Mac OS X Server fourni avec votre matériel Apple.

3. L'installateur de Mac OS X Server s'attend à ce que le disque dur soit **partitionné**, donc lorsqu'il n'offre pas de sélection, vous devez lancer l'outil de disque (Disk Utility) depuis le menu Outils et partitionner le disque dur. Puis, fermez l'outil de disque et recommencez l'installation.

4. En outre, comme le support pour Mac OS X Server dans VirtualBox est encore expérimental, merci de vous reporter également au chapitre 14, *Known limitations*, page 201.

[1] Voir chapitre 14, *Known limitations*, page 201.

3.1.2 Invités 64 bits

À partir de VirtualBox 2.0, VirtualBox supporte les systèmes d'exploitation invités 64 bits. À partir de la version 2.1, vous pouvez même lancer des invités 64 bits sur un système d'exploitation hôte 32 bits. Les pré-requis matériels sont identiques dans les deux cas.

En particulier, les invités 64 bits sont supportés dans les conditions suivantes:

1. Vous avez besoin d'un processeur 64 bits avec le support de la virtualisation matérielle (voir le chapitre 10.3, *Hardware vs. software virtualization*, page 175).

2. Vous devez activer la virtualisation matérielle pour la VM particulière pour laquelle vous voulez le support 64 bits; la virtualisation logicielle n'est pas supportée pour les VMs 64 bits.

3. Si vous voulez utiliser le support de l'invité 64 bits sur un système d'exploitation hôte 32 bits, vous devez aussi sélectionner un système d'exploitation 64 bits pour la VM en question. Puisque le support du 64 bits sur des hôtes 32 bits implique des traitements supplémentaires, VirtualBox n'active ce support que sur demande explicite.

   Sur les hôtes 64 bits, le support de l'invité 64 bits est toujours activé, donc vous pouvez simplement installer un système d'exploitation 64 bits dans l'invité.

Avertissement: Sur n'importe quel hôte, vous devriez activer l'**I/O APIC** pour les machines virtuelles où vous envisagez d'utiliser le mode 64 bits. Cela est surtout vrai pour les VMs Windows 64 bits. Voir le chapitre 3.3.2, *Onglet Avancé*, page 46. En outre, pour les invités Windows 64 bits, vous devriez vous assurer que la VM utilise le **périphérique réseau Intel**, puisqu'il n'y a pas de support de pilote 64 bits pour la carte AMD PCNet ; voir le chapitre 6.1, *Matériel de réseau virtuel*, page 91.

Si vous utilisez l'assistant Créer une VM de l'interface graphique de VirtualBox (voir le chapitre 1.6, *Créer votre première machine virtuelle*, page 17), VirtualBox utilisera automatiquement les bons paramètres pour chaque type de système d'exploitation 64 bits sélectionné.

3.2 Matériel émulé

VirtualBox virtualise presque tout le matériel de l'hôte. Selon la configuration de la VM, l'invité verra le matériel virtuel suivant:

- **Périphériques d'entrée.** Par défaut, VirtualBox émule un clavier et une souris PS/2 standards. Ces périphériques sont supportés par presque tous les systèmes d'exploitation passés et actuels. En outre, VirtualBox peut fournir des périphériques d'entrée USB.

- **Vidço.** les périphériques vidéos de VirtualBox (parfois désigné comme le périphérique VGA) ne sont pas basés, contrairement à presque tous les périphériques émulés, sur une base physique. C'est un périphérique simple et synthétique qui se montre compatible avec le VGA standard et avec plusieurs registres étendus utilisés par VESA BIOS Extensions (VBE).

- **Stockage.** VirtualBox émule actuellement l'interface standard ATA des chipsets Intel PIIX3/PIIX4, l'interface SATA (AHCI), et deux adaptateurs SCSI (LSI Logic et BusLogic); voir le chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78 pour des détails. Bien que fournir un d'eux suffirait à VirtualBox en lui-même, cette variété d'adaptateurs de stockage est requise pour des questions de compatibilité avec d'autres hyperviseurs. Windows est particulièrement pointilleux avec ses périphériques d'amorçage et la migration de VMs entre hyperviseurs est très difficile voire impossible si les contrôleurs de stockage sont différents.

- **Réseau.** Voir chapitre 6.1, *Matériel de réseau virtuel*, page 91.
- **USB.** VirtualBox émule deux deux contrôleurs hôtes USB, EHCI et OHCI. Les deux contrôleurs sont nécessaires car OHCI ne gère que les périphériques USB à vitesse pleine ou faible (low- et full-speed) (tant USB 1.x que 2.0), alors qu'EHCI ne gère que les périphériques à haute vitesse (seulement USB 2.0). Les contrôleurs USB émulés ne communiquent pas directement avec les périphériques de l'hôte, mais plutôt avec la couche VUSB (voir ci-dessous), qui émule le protocole USB et permet l'utilisation de périphériques USB distants (avec VRDP).
- **Son.** Deux périphériques audio sont émulés, un contrôleur AC97 avec codec, et une classique SoundBlaster 16 (seulement audio digital).

3.3 Paramètres généraux

Dans la fenêtre Paramètres, sous Général, vous pouvez configurer la plupart des aspects fondamentaux de la machine virtuelle tels que la mémoire et le matériel essentiel. Il y a trois onglets: Paramètres simples, Avancé et Description.

3.3.1 Onglet Simple

Sous l'onglet Simple de la catégorie des paramètres Général, vous pouvez trouver ces paramètres:

Nom Le nom sous lequel sera montrée la VM dans la liste des VMs dans la fenêtre principale. Sous ce nom, VirtualBox enregistre aussi les fichiers de configuration de la VM. En changeant le nom, VirtualBox renomme aussi ces fichiers. Il en résulte que vous ne pouvez utiliser que les caractères autorisés par les noms de fichiers de votre système d'exploitation.

Remarquez qu'en interne, VirtualBox utilise des identifiants uniques (UUIDs) pour identifier les machines virtuelles. Vous pouvez les afficher avec `VBoxManage`.

Système d'exploitation / Version Le type de système d'exploitation invité qui est (ou qui sera) installé dans la VM. C'est le même paramètre que celui spécifié dans l'assistant Nouvelle machine virtuelle, comme décrit au chapitre chapitre 1.6, *Créer votre première machine virtuelle*, page 17 ci-dessus.

3.3.2 Onglet Avancé

Dossier des instantanés Par défaut, VirtualBox enregistre les données d'instantané avec vos autres données de configuration de VirtualBox; voir le chapitre 10.1, *Where VirtualBox stores its files*, page 171. Avec ce paramètre, vous pouvez spécifier un autre répertoire pour chaque VM.

Presse-papier partagé Si vous avez installé les additions invité dans la machine virtuelle, vous pouvez sélectionner ici si le presse-papier du système d'exploitation invité devrait être partagé avec celui de votre hôte. Si vous sélectionnez Bidirectionnel, alors VirtualBox s'assurera toujours que les deux presse-papiers contiennent les mêmes données. Si vous sélectionnez Hôte vers invité ou invité vers hôte, alors VirtualBox ne fera que copier les données dans un sens.

Média amovible: enregistrer les changements pendant l'exécution Si ceci est coché, VirtualBox sauvegardera l'état des médias montés entre plusieurs exécutions de la machine virtuelle.

Barre d'outils compacte En mode plein-écran ou intégré, VirtualBox peut afficher une petite barre d'outils qui contient certains des menus qui sont normalement disponibles à partir de la barre de menu de la machine virtuelle. Cette barre d'outils se réduit à une petite ligne grise sauf si vous déplacez la souris dessus. Avec la barre d'outils, vous pouvez quitter le mode plein-écran ou le mode intégré, contrôler l'exécution de la machine ou activer certains périphériques. Si vous ne voulez pas voir la barre d'outils, désactivez ce paramètre.

3.3.3 Onglet Description

Ici vous pouvez entrer n'importe quelle description pour votre machine virtuelle, si vous le voulez. Elle n'a pas d'effet sur les fonctionnalités de la machine mais il se peut que vous trouviez utile cet espace pour noter des choses comme la configuration d'une machine virtuelle et le logiciel qui y a été installé.

3.4 Paramètres système

La catégorie Système regroupe divers paramètres liés au matériel de base présenté à la machine virtuelle.

> **Note:** Comme le mécanisme d'activation de Microsoft Windows est sensible aux changements de matériel, si vous changez des paramètres pour les invités Windows, certains de ces changements peuvent entraîner la demande d'une autre activation par Microsoft.

3.4.1 Onglet Carte mère

Sur l'onglet Carte mère, vous pouvez modifier le matériel virtuel qui serait normalement présent sur la carte mère d'un vrai ordinateur.

Mémoire de base Ceci règle la taille de la RAM allouée et donnée à la VM quand elle est en fonction. La quantité de mémoire spécifiée sera demandée au système d'exploitation hôte, donc elle doit être disponible ou libérée sur l'hôte quand vous essayez de démarrer la VM et elle ne sera pas disponible pour l'hôte tant que la VM sera en fonction. C'est le même paramètre que celui qui a été spécifié dans l'assistant Nouvelle machine virtuelle comme décrit dans les grandes lignes sous le chapitre chapitre 1.6, *Créer votre première machine virtuelle*, page 17 ci-dessus.

En général, il est possible de modifier la taille de la mémoire après l'installation du système d'exploitation invité (pourvu que vous ne réduisiez pas la mémoire à une quantité où le système d'exploitation ne démarrerait plus).

Ordre d'amorçage Ce paramètre détermine l'ordre des divers périphériques d'amorçage virtuels à partir desquels le système d'exploitation démarre. Comme le réglage du BIOS d'un vrai PC, VirtualBox peut dire à un OS invité de démarrer à partir d'une disquette virtuelle, le lecteur de CD/DVD virtuel, le disque dur virtuel (chacun d'eux selon ce qui est défini par les paramètres de la VM), le réseau, ou aucun d'eux.

Si vous sélectionnez Réseau, la VM cherchera à démarrer depuis le réseau via le mécanisme PXE. Cela doit être configuré en détail en ligne de commande; merci de voir le chapitre 8.7, *VBoxManage modifyvm*, page 120.

Activer I/O APIC Advanced Programmable Interrupt Controllers (APICs) (les contrôleurs d'interruption programmables avancés) sont une fonctionnalité récente du matériel x86 qui a remplacé l'ancien Programmable Interrupt Controllers (PICs) (contrôleurs d'interruption programmable) ces dernières années. Avec un APIC I/O, les systèmes d'exploitation peuvent utiliser plus de 16 requêtes d'interruption (interrupt requests) (IRQs) et donc éviter le partage d'IRQs pour une meilleure fiabilité.

Note: L'activation de l'APIC I/O est nécessaire pour les systèmes d'exploitation invités 64 bits, surtout Windows Vista; elle est aussi requise si vous voulez utiliser plus d'un CPU virtuel dans une machine virtuelle.

Cependant, le support logiciel de l'APICs I/O a été peu fiable pour certains systèmes d'exploitation autres que Windows. En outre, l'utilisation d'une APIC I/O augmente légèrement la surcharge de la virtualisation et donc, ralentit quelque peu l'OS invité.

Avertissement: Tous les systèmes d'exploitation Windows à partir de Windows 2000 installent des noyaux différents en fonction de l'activation ou non de l'APIC I/O. Comme avec ACPI, L'APIC I/O *ne doit donc pas être désactivé* après l'installation d'un OS invité Windows. L'activer après l'installation n'aura cependant aucun effet.

Activer EFI Ceci active l'Extensible Firmware Interface (EFI) qui remplace le BIOS de base, ce qui peut être utile pour certains cas d'utilisation avancée. Merci de vous référer au chapitre 3.12, *Firmware alternatif (EFI)*, page 56 pour des détails.

Horloge matérielle en temps UTC Si vous le cochez, VirtualBox donnera à l'invité l'heure système au format UTC plutôt qu'en temps local (de l'hôte). Ceci change la façon dont l'horloge de temps réel virtuelle (real-time clock, RTC) opère, et cela peut être utile pour des systèmes d'exploitation invités de type Unix, qui en général s'attend à ce que l'heure matérielle soit réglée sur UTC.

Activer les périphériques de pointages absolus S'il est activé, VirtualBox indique à la machine virtuelle qu'un périphérique de tablette USB est présent et il communique les événements de la souris à la machine virtuelle à travers ce périphérique. S'il est désactivé, les événements de la souris sont communiqués à travers un périphérique virtuel de souris PS/2 traditionnel.

Utiliser une tablette USB virtuelle présente l'avantage que les mouvements sont sighifiés en coordonnées absolues (et non en tant que changement de position relative), ce qui permet à VirtualBox de traaduire les événements de la souris de la fenêtre de la VM en événement de tablette sans devoir capturer la souris dans l'invité comme décrit au chapitre 1.7.1.1, *1.7.1.1 Capturer et utiliser le clavier et la souris*, page 20. Cela rend l'utilisation de la VM moins pénible même si on n'installe pas les additions invité.[2]

En outre, vous pouvez désactiver la **Advanced Configuration and Power Interface (ACPI, Configuration Avancée et l'Interface Alimentation)** que VirtualBox présente au système d'exploitation invité par défaut. ACPI est le standard industriel actuel pour permettre aux systèmes d'exploitation de reconnaître le matériel, de configurer les cartes mères et autres périphériques et gérer l'alimentation. Comme tous les PCs modernes contiennent cette fonctionnalité et comme Windows et Linux la supportent depuis des années, elle est également activée par défaut dans VirtualBox. Vous pouvez la désactiver en ligne de commande; voir le chapitre 8.7, *VBoxManage modifyvm*, page 120.

[2] La tablette USB virtuelle a été ajoutée à VirtualBox 3.2. Selon le système d'exploitation invité sélectionné, elle est maintenant activée par défaut pour les nouvelles machines virtuelles.

Avertissement: Tous les systèmes d'exploitation Windows à partir de Windows 2000 installent des noyaux différents selon l'activation ou non de l'ACPI, donc ACPI ne doit pas être désactivé après l'installation d'un OS invité Windows. Son activation après l'installation n'aura cependant aucun effet.

3.4.2 Onglet processeur

Sur l'onglet Processeur, vous pouvez déclarer combien de cur de processeurs virtuels devrait voir le système d'exploitation invité. À partir de la version 3.0, VirtualBox supporte le multiprocessing symétrique (symmetrical multiprocessing) (SMP) et peut présenter jusqu'à 32 curs de processeur virtuel à chaque machine virtuelle.

Néanmoins, vous ne devriez pas configurer des machines virtuelles pour utiliser plus de curs de processeur que vous n'en avez de disponibles physiquement.

En outre, le paramètre **"Activer PAE/NX"** détermine si les fonctionnalités PAE et NX du processeur hôte seront présentées à la machine virtuelle. PAE signifie Physical Address Extension (extension d'adresses physiques). Normalement, si elle est activée et supportée par le système d'exploitation, alors même un processeur x86 32 bits peut accéder à plus de 4 Gio de RAM. Cela est rendu possible en ajoutant 4 bits aux adresses mémoires, afin qu'avec 36 bits, jusqu'à 64 Gio puissent avoir une adresse. Certains systèmes d'exploitation (tel qu'Ubuntu Server) exige le support PAE du processeur et ne peuvent pas se lancer dans une machine virtuelle sans lui.

Avec des machines virtuelles qui exécutent des systèmes d'exploitation serveur modernes, VirtualBox supporte aussi le branchement à chaud de processeur. Pour des détails à ce sujet, merci de vous reporter au chapitre 9.5, *CPU hot-plugging*, page 154.

3.4.3 Onglet Accélération

Sur cet onglet, vous pouvez déterminer si VirtualBox devrait utiliser les extensions de la virtualisation du matériel que le processeur de votre hôte peut supporter et comment. C'est le cas avec la plupart des processeurs fabriqués après 2006.

Vous pouvez sélectionner, individuellement pour chaque machine virtuelle, si VirtualBox devrait utiliser la virtualisation logicielle ou matérielle.[3]

Dans la plupart des cas, les paramètres par défaut iront très bien; VirtualBox aura sélectionné des paramètres par défaut adéquats au système d'exploitation que vous avez sélectionné quand vous avez créé la machine virtuelle. Cependant, dans certaines situations, il se peut que vous vouliez modifier ces réglages pré-configurés par défaut.

Il se peut que les utilisateurs avancés soient intéressés par des détails techniques sur la virtualisation logicielle vs. La virtualisation matérielle; voir le chapitre 10.3, *Hardware vs. software virtualization*, page 175.

Si le processeur de votre hôte supporte les fonctionnalités de pagination empilée (nested chez AMD-V) ou d'EPT (Intel VT-x), vous pouvez alors espérer une augmentation conséquente des performances en activant la pagination empilée en plus de la virtualisation matérielle. La pagination empilée est encore désactivée par défaut même pour les nouvelles machines virtuelles, mais vous pouvez l'activer individuellement pour chaque machine virtuelle. Pour des détails techniques, voir le chapitre 10.6, *Nested paging and VPIDs*, page 179.

[3] Avant VirtualBox version 2.2, la virtualisation logicielle était le réglage par défaut; à partir de la version 2.2, VirtualBox activera par défaut la virtualisation matérielle sur les nouvelles machines virtuelles que vous créez. (Les machines virtuelles existantes ne sont pas automatiquement modifiées pour des questions de compatibilité et vous pouvez bien sûr modifier le réglage par défaut de chaque machine virtuelle.)

3.5 Paramètres d'affichage

Taille de la mémoire graphique Ceci règle la taille de la mémoire fournie par la carte graphique virtuelle disponible pour l'invité, en Mio. Comme avec la mémoire principale, la quantité spécifiée sera allouée à partir de la mémoire résidente de l'hôte. Basée sur la quantité de mémoire graphique, résolutions plus élevées et profondeurs de couleurs peuvent être disponibles.

Compteur de moniteurs Avec ce paramètre, VirtualBox peut offrir plus d'un moniteur virtuel à une machine virtuelle. Si un système d'exploitation invité (tel que Windows) supporte les moniteurs multiples attachés, VirtualBox peut simuler que plusieurs moniteurs virtuels sont présents.[4] Jusqu'à 8 moniteurs virtuels de ce type sont supportés.

La sortie des divers moniteurs sera affichée sur l'hôte dans plusieurs fenêtres de VM qui se lancent côte à côte.

Néanmoins, en mode plein écran et intégré, elles utiliseront les moniteurs physiques disponibles attachés à l'hôte. Il en résulte que pour que les modes plein écran et intégré fonctionnent avec plusieurs moniteurs, vous aurez besoin d'au moins autant de moniteurs physiques que le nombre de moniteurs virtuels que vous aurez configurés, ou VirtualBox renverra une erreur. Vous pouvez configurer la relation entre les moniteurs invités et hôtes en utilisant le menu vues en appuyant sur la touche Hôte + Origine quand vous êtes en mode plein écran ou intégré.

Merci de voir aussi le chapitre 14, *Known limitations*, page 201.

Activer l'accélération 3D Si vous avez installé les additions invité dans une machine virtuelle, vous pouvez sélectionner ici si l'invité supporte la vidéo graphique accélérée. Merci de vous reporter au chapitre xref linkend=“guestadd-3d” /> pour des détails.

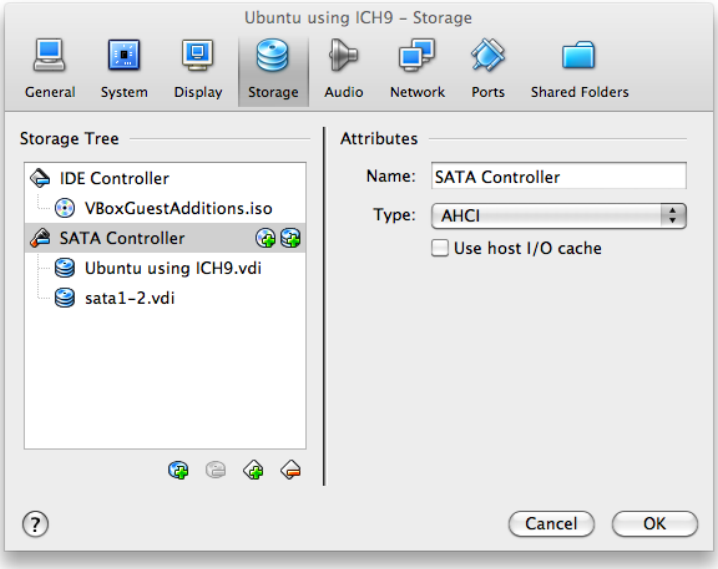
Activer l'accélçration graphique 2D Si vous avez installé les additions invité sur une machine virtuelle avec Microsoft Windows, vous pouvez sélectionner ici si l'invité devrait supporter les graphiques vidéos 2D accélérés. Merci de vous référer au chapitre chapitre 4.5.2, *Accélération vidéo 2D pour Windows*, page 73 pour les détails.

Affichage distant Sous l'onglet Affichage distant, vous pouvez activer le serveur VRDP construit dans VirtualBox pour vous permettre de vous connecter à la machine virtuelle à distance. Pour cela, vous pouvez utiliser n'importe quel visualiseur de RDP standard tels que `mstsc.exe` qui est fourni avec Microsoft Windows ou, sur les systèmes Linux, le programme standard libre `rdesktop`. Ces fonctionnalités sont décrites en détail au chapitre 7.1, *Remote display (VRDP support)*, page 101.

3.6 Paramètres de stockage

Dans la fenêtre des paramètres de la VM, la section « Stockage » vous permet de connecter des images virtuelles de disque dur, de CD/DVD et de disquette et des lecteurs à votre machine virtuelle:

[4] Le support de plusieurs moniteurs a été ajouté à VirtualBox 3.2.



Dans un vrai PC, ce que l'on appelle les « contrôleurs de stockage » connectent des lecteurs de disque physiques au reste de l'ordinateur. De la même façon, VirtualBox présente des contrôleurs de stockage virtuels à une machine virtuelle. Sous chaque contrôleur, les périphériques virtuels (disque dur, lecteurs CD/DVD ou de disquettes) sont montrés comme attachés au contrôleur.

Si vous avez utilisé l'assistant Créer une VM pour créer une machine, vous verrez normalement les périphériques suivants:

- Vous verrez un contrôleur IDE, sous lequel il y a deux périphériques:
  - un disque dur virtuel connecté au slot IDE appelé primaire maître; cela est représenté par les images de disque que vous avez créées avec la machine;
  - un lecteur de CD/DVD connecté au secondaire maître.
- En outre, il y a un contrôleur de disquette auquel est attaché un lecteur de disquette virtuel.

Vous pouvez modifier ces attachements de médias à votre guise. Par exemple, si vous souhaitez copier des fichiers depuis un autre disque virtuel que vous avez créé, vous pouvez connecter ce disque comme second disque dur. Vous pourriez aussi ajouter un second lecteur de CD/DVD virtuel où changer l'endroit auxquels ces éléments sont attachés.

En plus des contrôleurs IDE, VirtualBox peut aussi présenter un contrôleur SATA et un contrôleur SCSI à l'invité, ce qui vous donne respectivement 30 ou 16 emplacements supplémentaires pour y attacher des périphériques, Néanmoins cela peut exiger que vous exécutiez un système d'exploitation invité moderne. Voir le chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78 pour les détails.

Pour **,nouter un autre disque dur ou un autre lecteur de CD/DVD virtuel,** sélectionnez le contrôleur de stockage auquel cela devrait être ajouté (IDE, SATA ou SCSI) puis cliquez sur le bouton Ajouter un disque en-dessous de l'arborescence. Vous pouvez alors soit sélectionner Ajouter un périphérique CD/DVD ou Ajouter un disque dur. Alternativement, faites un clic droit sur le contrôleur de stockage puis sélectionnez-y une option du menu.

Sur la partie droite de la fenêtre, vous pouvez alors sélectionner l'endroit auquel le disque virtuel devrait être connecté sur le contrôleur et quel fichier image utiliser.

- Pour les disques durs virtuels, une liste déroulante apparaît à droite, listant toutes les images de disque dur que VirtualBox connaît actuellement. Si vous cliquez sur l'icône Ouvrir le gestionnaire de médias virtuels à droite, cela fera apparaître une fenêtre dans laquelle vous pouvez sélectionner ou créer une image de disque dur différente (voir le chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82 pour les détails).

- Pour les lecteurs de CD/DVD, il y a deux types options dans la liste déroulante.
  - Si vous sélectionnez Vide, alors VirtualBox présentera un lecteur de CD/DVD virtuel à l'invité avec aucun média inséré.
  - Si vous sélectionnez Lecteur hôte dans la liste, alors le périphérique physique de l'ordinateur hôte sera connecté à la VM, afin que le système d'exploitation invité puisse lire et écrire sur votre périphérique physique. C'est par exemple utile si vous voulez installer Windows à partir d'un vrai CD d'installation. Dans ce cas, sélectionnez votre lecteur hôte depuis la liste déroulante présentée.

    Note: Si vous voulez graver des CDs ou des DVDs en utilisant le lecteur de l'hôte, vous devez tout d'abord activer un paramètre spécial; voir le chapitre 5.10, *Écrire des CDs et des DVDs en utilisant le lecteur hôte*, page 90.

  - Les autres options de la liste comme les images de disque virtuelles seront des fichiers images sur votre hôte. Le format de fichier est ici le format ISO. La plupart du temps, vous sélectionnerez cette option quand vous installerez un système d'exploitation à partir d'un fichier ISO que vous avez obtenu sur Internet. Par exemple, la plupart des distributions Linux sont disponibles de cette matière.

Note: La chaîne d'identification du lecteur fournie à l'invité (qui, dans l'invité, serait affichée par les outils de configuration tels que gestionnaire de périphériques Windows) est toujours CD-ROM VBOX, indépendamment de la configuration actuelle du lecteur virtuel. Ceci empêche la détection de matériel d'être déclenchée dans le système d'exploitation invité chaque fois que la configuration est modifiée.

Remarquez que le contrôleur de disquette est spécial : vous ne pouvez pas lui ajouter de périphériques autres que les lecteurs de disquette. Les lecteurs de disquette virtuels, comme les lecteurs de CD/DVD, peuvent être connectés soit à un lecteur de disquette hôte (si vous en avez un) ou une image de disque qui, dans ce cas, doit être au format RAW.

Pour **supprimer un disque ou un lecteur virtuel,** sélectionnez-le et cliquez sur l'icône Supprimer en bas (ou faites un clic droit dessus et sélectionnez l'élément du menu).

Vous pouvez changer les médias amovibles (CD/DVDs et disquettes) alors que la machine invitée est en fonction. Puisque la boîte de dialogue Paramètres n'est pas disponible à ce moment, vous pouvez aussi accéder à ces paramètres depuis le menu Périphériques de la fenêtre de votre machine virtuelle.

Nous avons consacré un chapitre entier de ce manuel d'utilisateur au stockage virtuel: merci de voir le chapitre 5, *Stockage virtuel*, page 78 pour le moindre détail sur la configuration du stockage.

3.7 Paramètres audio

La section Audio dans la fenêtre de paramètres d'une machine virtuelle détermine si la VM verra une carte son connectée et si la sortie audio devrait être entendue sur le système hôte.

Si l'audio est activé pour un invité, vous pouvez choisir entre l'émulation d'un contrôleur Intel AC'97 ou une carte SoundBlaster 16. Dans tous les cas, vous pouvez sélectionner le pilote audio que VirtualBox utilisera sur l'hôte.

Sur un hôte Linux, selon la configuration de votre hôte, vous pouvez aussi choisir entre les sous systèmes OSS, ALSA ou PulseAudio. Sur les distributions Linux récentes, (Fedora 8 et supérieur, Ubuntu 8.04 et supérieur), vous devriez préférer le sous-système PulseAudio.

3.8 Paramètres réseau

La section réseau d'une fenêtre des paramètres d'une machine virtuelle vous permet de configurer la façon dont VirtualBox présente les cartes réseau virtuelles à votre VM et comment elles agissent.

Quand vous créez une machine virtuelle, VirtualBox active d'emblée par défaut une carte réseau virtuelle et sélectionne le mode Network Address Translation (NAT) pour elle. De cette façon, l'invité peut se connecter au monde extérieur en utilisant le réseau de l'hôte et le monde extérieur peut se connecter aux services de l'invité que vous choisissez de rendre visibles à l'extérieur de la machine virtuelle. Dans la plupart des cas, le réglage par défaut fonctionnera très bien pour vous.

Cependant, VirtualBox est extrêmement flexible quant à la façon de virtualiser le réseau. Il supporte jusqu'à huit cartes réseau par machine virtuelle, parmi lesquelles les quatre premières peuvent être configurées en détail dans l'interface graphique. Les huit cartes réseau peuvent être configurées en ligne de commande avec VBoxManage. Du coup, nous avons consacré un chapitre entier de ce manuel pour traiter de la configuration réseau; merci de voir le chapitre 6, *Réseau virtuel*, page 91.

3.9 Ports série

VirtualBox supporte pleinement les ports série dans une machine virtuelle d'une manière facile à utiliser.[5]

Dès le premier PC IBM, les ordinateurs personnels ont été équipé d'un ou de deux ports série (appelés aussi ports COM par le DOS et Windows). S'ils ne sont plus aussi importants qu'ils ne l'étaient il y a encore quelques années, (surtout depuis que les souris ne sont plus connectées au port série aujourd'hui), il y a encore des usages importants de ces derniers. Par exemple, on peut utiliser les ports série pour régler un réseau de base par un câble null-modem, au cas où l'Ethernet n'est pas disponible. Les ports série sont aussi indispensables pour les programmeurs système qui ont besoin de faire du débogage de noyau, vu que le logiciel de débogage du noyau interagit en général avec les développeurs par un port série. En d'autres termes, avec les ports série virtuels, les programmeurs peuvent faire du débogage de noyau sur une machine virtuelle au lieu d'avoir besoin qu'un vrai ordinateur ne s'y connecte.

Si vous activez un port série virtuel, le système d'exploitation invité le voit comme un port série de type 16450 standard. La réception et la transmission de données sont toutes deux supportées. La façon dont ce port série virtuel est alors connecté à l'hôte est configurable, et les détails dépendent de votre système d'exploitation hôte.

Vous pouvez utiliser soit l'interface graphique, soit l'outil `VBoxManage` en ligne de commande pour paramétrer les ports série virtuels. Pour ces derniers, merci de vous référer au chapitre 8.7, *VBoxManage modifyvm*, page 120;; dans cette section, cherchez les options `--uart` et `--uartmode`.

Dans tous les cas, vous pouvez configurer jusqu'à deux ports série simultanément. Pour chacun de ces périphériques, vous devrez déterminer

1. quel type de port série devrait voir la machine virtuelle en sélectionnant l'adresse de base E/S et l'interruption (IRQ). Pour ceux-ci, nous recommandons d'utiliser les valeurs traditionnelles[6], qui sont:
   a) COM1: I/O base 0x3F8, IRQ 4
   b) COM2: I/O base 0x2F8, IRQ 3
   c) COM3: I/O base 0x3E8, IRQ 4

[5] Le support du port série a été ajouté à VirtualBox 1.5.
[6] Voir, par exemple, http://en.wikipedia.org/wiki/COM_(hardware_interface).

d) COM4: I/O base 0x2E8, IRQ 3

2. Puis, vous devrez déterminer à quoi ce port virtuel sera connecté. Pour chaque port série virtuel, vous avez les options suivantes:

   - Vous pouvez décider que le port série sera déconnecté, ce qui signifie que l'invité le verra comme un matériel mais il se comportera comme si aucun câble n'y a été connecté.
   - Vous pouvez connecter le port série virtuel à un port série physique de votre hôte (sur un hôte Windows, cela sera un nom du type COM1; sur des hôtes Linux ou OpenSolaris, ce sera un nud de périphérique comme `/dev/ttyS0`). VirtualBox redirigera simplement toutes les données reçues et envoyées depuis et vers le port série virtuel sur ce port physique.
   - Vous pouvez dire à VirtualBox de connecter le port série virtuel à un tuyau logiciel sur l'hôte. Cela dépend de votre système d'exploitation:
     - Sur un hôte Windows, les données seront envoyées et reçues à travers un tuyau nommé. Vous pouvez utiliser un programme conducteur appelé VMware Serial Line Gateway, disponible en téléchargement sur `http://www.l4ka.org/tools/vmwaregateway.php`. Cet outil fournit un mode serveur fixe appelé tuyau sur `\\.\pipe\vmwaredebug` et connecte les connexions TCP entrant sur le port 567 avec le tuyau nommé.
     - Sur un hôte Mac, Linux ou OpenSolaris, un socket de domaine local est utilisé à la place. Sur Linux, plusieurs outils peuvent se connecter à un socket du domaine local ou en créer un en mode serveur. L'outil le plus flexible est `socat`, disponible comme partie intégrante de beaucoup de distributions.

     Dans ce cas, vous pouvez configurer si VirtualBox devrait créer lui-même le tuyau nommé (ou, sur des hôtes non-Windows, le socket du domaine local) ou si VirtualBox devrait supposer que le tuyau (ou le socket) existe déjà. Avec les options en ligne de commande de VBoxManage, cela s'appelle respectivement le mode Serveur ou Client.

Vous pouvez configurer jusqu'à deux ports série simultanément par machine virtuelle, mais vous pouvez utiliser n'importe quel numéro de port au-dessus de cette limite. Par exemple, vous pouvez configurer deux ports série capables de fonctionner avec COM2 et COM4 dans l'invité.

3.10 Support USB

3.10.1 Paramètres USB

La section USB dans la fenêtre de paramètres d'une machine virtuelle vous permet de configurer le support USB sophistiqué de VirtualBox.

VirtualBox peut permettre aux machines virtuelles d'accéder directement aux périphériques USB de votre hôte. Pour cela, VirtualBox présente au système d'exploitation invité un contrôleur USB virtuel. Dès que le système invité commence à utiliser un périphérique USB, celui-ci sera indisponible sur l'hôte.

Note:

1. Soyez prudent avec les périphériques USB actuellement utilisés sur l'hôte! Par exemple, si vous autorisez votre invité à se connecter à votre disque dur USB actuellement monté sur l'hôte, lorsque l'invité est activé, il sera déconnecté de l'hôte sans être éteint proprement. Cela peut causer une perte de données.

2. Les hôtes Solaris ont quelques limites connus concernant le support USB; merci de voir le chapitre 14, *Known limitations*, page 201.

En plus de permettre l'accès par l'invité à vos périphériques USB locaux, VirtualBox permet même à votre invité de se connecter à des périphériques USB distants en utilisant le protocole VRDP. Pour des détails à ce sujet, voir le chapitre 7.1.4, *Remote USB*, page 104.

Dans la boîte de dialogue des paramètres, vous pouvez d'abord configurer si l'USB est disponible dans l'invité et en plus aussi choisir d'activer le contrôleur USB 2.0 (EHCI) pour l'invité. Dans ce cas, vous pouvez déterminer en détails quels périphériques sont disponibles. Pour cela, vous devez créer ce que l'on appelle des « filtres » en spécifiant certaines propriétés du périphérique USB.

Un clic sur le bouton + à droite de la fenêtre Filtres de périphérique USB, crée un nouveau filtre. Vous pouvez donner un nom au filtre (pour le retrouver plus tard), et spécifier les critères du filtre. Plus vous spécifierez des critères précis, plus les périphériques seront sélectionnés avec précision. Par exemple, si vous ne spécifiez que l'ID du constructeur 046d, tous les périphériques fabriqués par Logitech seront disponibles pour l'hôte. À l'inverse, si vous remplissez tous les champs, le filtre ne s'appliquera qu'à un modèle de périphérique particulier d'un constructeur particulier, pas même aux autres périphériques du même type ayant un numéro de révision et de série différent.

Dans le détail, les critères suivants sont disponibles:

1. **ID du constructeur et du produit.** Avec l'USB, chaque constructeur de produits USB a un numéro d'identification unique pour le monde entier, l'ID du constructeur. De la même façon, chaque ligne de produits est associée à un numéro ID de produit. Les deux numéros sont écrits en général en hexadécimal (c'est-à-dire qu'ils sont composés des nombres 0-9 et des lettres A-F), et le signe deux-points sépare l'ID du constructeur et du produit. Par exemple, `046d:c016` s'applique au constructeur Logitech et au produit M-UV69a Optical Wheel Mouse.

   Vous pouvez aussi spécifier le nom du **"Fabricant"** et du **"Produit"** par leur nom.

   Pour lister tous les périphériques USB connectés à votre machine hôte avec leurs IDs de fabricant et de produit respectifs, vous pouvez utiliser la commande suivante (voir le chapitre 8, *VBoxManage*, page 110):

   ```
   VBoxManage list usbhost
   ```

   Sur Windows, vous pouvez aussi voir tous les périphériques USB attachés à votre système dans le gestionnaire de périphérique. Sur Linux, vous pouvez utiliser la commande `lsusb`.

2. **Numéro de série.** Alors que l'ID constructeur et produit sont déjà très spécifiques pour identifier des périphériques USB, si vous avez deux périphériques identiques de la même marque et de la même ligne de produit, vous aurez également besoin de leurs numéros de série pour les filtrer correctement.

3. **Distant.** Ce paramètre précise si le périphérique ne sera que local ou que distant (par VRDP), ou l'un ou l'autre.

Sur un hôte Windows, vous devrez débrancher et reconnecter le périphérique USB pour l'utiliser après lui avoir créé un filtre.

Par exemple, vous pourriez créer un nouveau filtre USB et préciser un ID vendeur de 046d (Logitech, Inc), un index de fabricant de 1, et non distant. Alors, tous les périphériques USB du système hôte fabriqués par Logitech, Inc ayant un index fabricant de 1 seront visible pour le système hôte.

Plusieurs filtres peuvent sélectionner un seul périphérique - par exemple, un filtre qui sélectionne tous les périphériques Logitech et un qui sélectionne une webcam en particulier.

Vous pouvez **désactiver** des filtres sans les effacer en cliquant sur la case à cocher à côté du nom du filtre.

3.10.2 Remarques d'implémentation pour les hôtes Windows et Linux

Sur les hôtes Windows, un pilote de périphérique en mode noyau fournit un support de proxy USB. Il implémente à la fois un moniteur USB, qui permet à VirtualBox de capturer les périphériques lorsqu'ils sont branchés, et un pilote de périphérique USB pour attribuer les périphériques USB à une machine virtuelle particulière. Contrairement aux versions inférieures à 1.4.0 de VirtualBox, les redémarrages du système ne sont plus nécessaires après avoir installé le pilote. De même, vous n'avez plus besoin de rebrancher les périphériques pour que VirtualBox les attribue.

Sur les hôtes Linux récents, VirtualBox accède aux périphériques USB par des fichiers spéciaux du système de fichiers. Lorsque VirtualBox est installé, ceux-ci sont rendus disponibles pour tous les utilisateurs dans le groupe système `vboxusers`. Afin de pouvoir accéder à l'USB à partir de systèmes invités, assurez-vous que vous êtes membre de ce groupe.

Sur les hôtes Linux plus anciens, on accède aux périphériques USB en utilisant le système de fichiers `usbfs`. Du coup, l'utilisateur qui exécute VirtualBox a besoin des droits en lecture et écriture sur le système de fichiers USB. La plupart des distributions fournissent un groupe (comme `usbusers`) auquel il faut ajouter l'utilisateur de VirtualBox. En outre, VirtualBox ne peut attribuer que un périphérique USB d'une machine virtuelle qui n'est pas attribué à un pilote USB de l'hôte Linux. L'entrée `Driver=` de `/proc/bus/usb/devices` vous montrera quels périphériques sont actuellement attribués. Merci de vous référer également au chapitre 12.6.7, *USB not working*, page 194 pour des détails sur `usbfs`.

3.11 Dossiers partagés

Les répertoires partagés vous permettent d'échanger facilement des données entre une machine virtuelle et votre hôte. Cette fonctionnalité exige que les additions invité soient installés sur une machine virtuelle et est décrite en détails au chapitre 4.3, *Dossiers partagés*, page 68.

3.12 Firmware alternatif (EFI)

À partir de la version 3.1, VirtualBox inclut le support expérimental de l'Extensible Firmware Interface (EFI) (interface extensible de firmware), qui est un nouveau standard industriel conçu pour remplacer éventuellement le BIOS de base en tant qu'interface primaire pour amorcer les ordinateurs, et certains services systèmes plus tard.

VirtualBox utilise par défaut le firmware BIOS pour les machines virtuelles. Pour utiliser l'EFI pour une machine virtuelle donnée, vous pouvez activer EFI dans la boîte de dialogue paramètres de la machine (voir le chapitre chapitre 3.4.1, *Onglet Carte mère*, page 47). Sinon, vous pouvez utiliser l'interface `VBoxManage` en ligne de commande comme ceci :

```
VBoxManage modifyvm "Nom VM" --firmware efi
```

Pour utiliser BIOS à nouveau, utilisez:

```
VBoxManage modifyvm "Nom VM" --firmware bios
```

Un des utilisateurs remarquables d'EFI est le Mac OS X d'Aple, mais les Linux récents (tels que Fedora 11)et les Windows récents (à partir de Vista) peuvent être démarrés en utilisant l'EFI.

Une autre utilisation possible d'EFI dans VirtualBox est le développement et le test des applications EFI, sans amorcer d'OS.

Notez que le support d'EFI par VirtualBox et expérimental et il sera adapté en fonction du développement d'EFI et au fur et à mesure de son expansion. Si les invités Mac OS X et Linux sont connus pour très bien fonctionner, les invités Windows sont actuellement incapables de démarrer en utilisant EFI.

3.12.1 Modes graphiques en EFI

EFI fournit deux interfaces graphiques différentes : GOP (Graphics Output Protocol) et UGA (Universal Graphics Adapter). Mac OS X utilise GOP, alors que Linux a tendance à utiliser UGA. VirtualBox fournit une option de configuration pour contrôler la taille du tampon de trame (framebuffer) pour les deux interfaces.

Pour contrôler GOP, utilisez la commande `VBoxManage` suivante:

```
VBoxManage setextradata "Nom VM" VBoxInternal2/EfiGopMode N
```

Où N peut être un parmi 0,1,2,3,4, se rapportant respectivement à la résolution d'écran 640x480, 800x600, 1024x768, 1280x1024, 1440x900.

Pour modifier la résolution UGA:

```
VBoxManage setextradata "nom VM" VBoxInternal2/UgaHorizontalResolution 1440
VBoxManage setextradata "nom VM" VBoxInternal2/UgaVerticalResolution    900
```

Le mode graphique de GOP et d'UGA ne peut être modifié que quand la VM est éteinte et il reste en l'état jusqu'à ce qu'il soit modifié.

4 Additions invité

Le chapitre précédent a traité de la façon de débuter avec VirtualBox et d'installer des systèmes d'exploitation dans une machine virtuelle. Pour une utilisation pousséed et interactive, les additions invité de VirtualBox vous faciliteront beaucoup la vie en vous fournissant une intégration poussée entre l'hôte et l'invité, et en améliorant les performances interactives des systèmes invités. Ce chapitre décrit en détail les additions invité.

4.1 Introduction

Comme indiqué au chapitre 1.2, *Un peu de terminologie*, page 11, les additions invité sont conçus pour être installés à l'intérieur d'une machine virtuelle après que le système d'exploitation invité a été installé. Ils consistent en des pilotes de périphériques et des applications systèmes qui optimisent le système d'exploitation invité pour de meilleures performances et un usage facilité. Merci de voir le chapitre 3.1, *Systèmes d'exploitation invités supportés*, page 43 pour des détails sur les systèmes d'exploitation invités pleinement supportés avec les additions invité par VirtualBox.

Les additions invité de VirtualBox de tous les systèmes d'exploitation invités sont fournies sous la forme d'un fichier image unique de CD-ROM appelé `VBoxGuestAdditions.iso`. Ce fichier image se situe dans le répertoire d'installation de VirtualBox. Pour installer les additions invité pour une VM particulière, vous montez ce fichier ISO dans votre VM en tant que CD-ROM virtuel et vous l'installez à partir de là.

Les additions invités offrent les fonctionnalités suivantes:

Intégration du pointeur de la souris Pour dépasser les limites du support de la souris décrites au chapitre 1.7.1.1, *1.7.1.1 Capturer et utiliser le clavier et la souris*, page 20, cela vous offre un support de souris intégrée. Vous n'aurez q'un pointeur de souris et l'appui sur la touche Hôte n'est plus nécessaire pour libérer la souris de sa capture par l'OS invité. Pour que cela fonctionne, un pilote de souris spécial est installé dans l'invité, qui communique avec le pilote de souris réel de votre hôte et il déplace le pointeur de la souris en fonction.

Meilleur support graphique Alors que la carte graphique virtuelle émulée par VirtualBox pour tous les systèmes invités fournit toutes les fonctionnalités de base, les pilotes graphiques personnalisés installés avec les additions invité vous offrent des modes vidéos plus avancés et non standards, ainsi que des performances graphiques accélérées.

En outre, avec les invités Windows et les récents Linux, Solaris et OpenSolaris, si les additions invité sont installés, vous pouvez redimensionner la fenêtre de la machine virtuelle et la résolution graphique de l'invité sera automatiquement ajustée (comme si vous aviez entré à la main une résolution de votre choix dans les paramètres d'affichage de l'hôte).

Enfin, si vous installez les additions invité, la 3D pour les applications invitées peut être accélérée ; voir le chapitre 4.5.1, *Accélération 3D matérielle (OpenGL et Direct3D 8/9)*, page 72.

Synchronisation du temps Une fois les additions invité installées, VirtualBox peut s'assurer que le système d'horloge de l'invité est mieux synchronisé avec celui de l'hôte.

Pour diverses raisons, le temps de l'invité peut s'écouler à une vitesse légèrement différente du temps de l'hôte. L'hôte pourrait recevoir des mises à jour par NTP et son propre temps pourrait ne pas s'écouler de façon linéaire. Une VM pourrait aussi être mise en pause, ce qui arrête le cours du temps dans l'invité pour une période plus ou moins longue. Quand

le temps horloge limite entre l'invité et l'hôte ne diffère que légèrement, le système de synchronisation tente d'ajuster graduellement et progressivement l'heure de l'invité par petites saccades pour rattraper ou perdre du temps. Quand la différence est trop importante (par exemple une VM mise en pause pendant des heures ou restaurée depuis un état sauvegardé), l'heure de l'invité est modifiée immédiatement, sans ajustements graduels.

Les additions invité resynchroniseront l'heure graduellement. Voir le chapitre 9.13.3, *Tuning the Guest Additions time synchronization parameters*, page 165 pour la façon de configurer les paramètres du mécanisme de synchronisation du temps.

Dossiers partagés Ceux-ci offrent un moyen facile d'échanger des fichiers entre l'hôte et l'invité. Tout comme les partages de réseau Windows ordinaires, vous pouvez dire à VirtualBox de traiter un répertoire hôte en particulier comme un dossier partagé et VirtualBox le rendra disponible pour le système d'exploitation invité en tant que partage réseau. Pour les détails, merci de vous référer au chapitre 4.3, *Dossiers partagés*, page 68.

Intégration de fenêtres Avec cette fonctionnalité, les fenêtres individuelles affichées sur le bureau de la machine virtuelle peuvent être gérées par le bureau de l'hôte, comme si l'application dans la fenêtre fonctionnait en fait sur l'hôte. Voir le chapitre 4.4, *Fenêtres intégrées*, page 71 pour les détails.

Presse-papier partagé Une fois les additions invité installées, le presse-papier du système d'exploitation invité peut éventuellement être partagés avec votre système d'exploitation hôte; voir le chapitre 3.3, *Paramètres généraux*, page 46.

Connexions automatiques (passage de certificats) Pour des détails, merci de voir le chapitre 9.2, *Automated guest logons*, page 150.

Chaque version de VirtualBox, même celles mineures, est fournie avec sa propre version des additions invité. Si les interfaces par lesquelles le coeur VirtualBox communique avec les additions invité sont laissées stables afin que les additions invité déjà installées dans une VM continuent à fonctionner lorsque VirtualBox est mis à jour sur l'hôte, il est recommandé, pour de meilleurs résultats, de maintenir les additions invité à la même version.

À partir de VirtualBox 3.1, les additions invité Windows et Linux vérifient donc automatiquement s'ils doivent être mis à jour. Si l'hôte exécute une version de VirtualBox plus récente que les additions invité, une notification avec toutes les instructions est affichée dans l'invité.

Pour désactiver la vérification des mises à jour des additions invité d'une machine virtuelle donnée, mettez la valeur de sa propriété d'invité `/VirtualBox/GuestAdd/CheckHostVersion` à `0`; voir le chapitre 4.6, *Propriétés invité*, page 73 pour les détails.

4.2 Installer et maintenir les additions invité

Les additions invité sont disponibles pour les machines virtuelles qui exécutent Windows, Linux, Solaris ou OS/2. Les sections suivantes décrivent en détail les spécificités de chaque variante.

4.2.1 Additions Invité pour Windows

Les additions invité Windows de VirtualBox sont conçues pour être installées sur une machine virtuelle exécutant un système d'exploitation Windows. Les versions suivantes des invités Windows sont supportées:

- Microsoft Windows NT 4.0 (tous les packs service)
- Microsoft Windows 2000 (tous les packs service)
- Microsoft Windows XP (tous les packs service)

- Microsoft Windows Server 2003 (tous les packs service)
- Microsoft Windows Server 2008
- Microsoft Windows Vista (toutes les éditions)
- Microsoft Windows 7 (toutes les éditions)

4.2.1.1 Installation

Dans le menu Périphériques de la barre de menu de la machine virtuelle, VirtualBox a un élément pratique de sous-menu appelé Installer les additions invité, qui monte le fichier ISO des additions invité dans votre machine virtuelle. Un invité Windows devrait alors lancer automatiquement l'installateur des additions invité qui installe les additions invité dans votre invité Windows.

Note: Pour que l'accélération Directe 3D fonctionne dans un invité Windows, vous devez installer les additions invité en mode sécurisé, voir le chapitre 14, *Known limitations*, page 201 pour les détails.

Si vous préférez monter les additions à la main, vous pouvez effectuer les étapes suivantes:

1. Démarrer la machine virtuelle dans laquelle vous avez installé Windows.
2. Sélectionner Monter un CD/DVD-ROM depuis le menu Périphériques de la barre de menu de la machine virtuelle, puis imag de CD/DVD-ROM. Cela affiche le gestionnaire de médias virtuels décrit au chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82.
3. Dans le gestionnaire de médias virtuels, appuyer sur le bouton Ajouter et chercher dans le système de fichiers le fichier `VBoxGuestAdditions.iso`:
   - Sur un hôte Windows, vous pouvez trouver ce fichier dans le répertoire d'installation de VirtualBox (généralement sous `C:\Program files\Oracle\VirtualBox`).
   - Sur les hôtes Mac OS X, vous pouvez trouver ce fichier dans le groupe application de VirtualBox. (Faire un clic droit sur l'icône VirtualBox de Finder et choisir *Montrer le contenu du paquet*. Il se trouve alors dans le dossier `Contents/MacOS`.)
   - Sur un hôte Linux, vous pouvez trouver ce fichier dans le dossier additions sous lequel vous avez installé VirtualBox (normalement `/opt/VirtualBox/`).
   - Sur les hôtes Solaris, vous pouvez trouver ce fichier dans le dossier `additions` sous lequel vous avez installé VirtualBox (normalement `/opt/VirtualBox`).
4. Retourner dans le gestionnaire de médias virtuels, sélectionner ce fichier ISO et appuyer sur le bouton Sélectionner. Cela montera le fichier ISO et le présentera à votre invité Windows comme un CD-ROM.

Sauf si vous avez désactivé la fonction de démarrage automatique sur votre invité Windows, Windows démarrera automatiquement alors le programme d'installation des additions invité de VirtualBox depuis l'ISO des additions. Si vous avez désactivé la fonction de démarrage automatique, choisissez `VBoxWindowsAdditions.exe` depuis le lecteur de CD/DVD à l'intérieur de l'invité pour démarrer l'installateur.

L'installateur ajoutera plusieurs pilotes de périphérique à la base de données des pilotes de Windows puis il invoquera l'assistant de détection du matériel.

Selon votre configuration, il pourrait afficher des avertissements selon lesquels les pilotes ne sont pas signés de façon digitale. Vous devez confirmer afin de poursuivre l'installation et d'installer correctement les additions.

Après l'installation, redémarrez votre système d'exploitation invité pour activer les additions.

4.2.1.2 Mettre à jour les additions invité Windows

Vous pouvez mettre à jour les additions invité Windows en relançant le programme d'installation comme décrit précédemment. Cela remplacera alors les pilotes des additions précédents par les versions mises à jour.

Sinon, vous pouvez aussi ouvrir le gestionnaire de périphériques et sélectionner Mettre à jour les périphériques... pour deux périphériques:

1. l'adaptateur vidéo de VirtualBox et
2. le périphérique système de VirtualBox.

Pour chacun d'eux, choisissez de fournir votre propre pilote et utilisez J'ai un disque pour orienter l'assistant vers le lecteur CD-ROM contenant les additions invité.

4.2.1.3 Installation sans effort

Afin d'obtenir des installations invité entièrement sans efforts, vous pouvez spécifier un paramètre en ligne de commande au lanceur de l'installation:

```
VBoxWindowsAdditions.exe /S
```

Cela installe automatiquement les bons fichiers et les pilotes pour la plateforme correspondante (32 ou 64 bits).

> **Note:** Comme les pilotes ne sont pas encore certifiés WHQL, il se pourrait que vous ayez encore des fenêtres pop-up d'installation de pilotes, selon la version de l'invité Windows.

Pour plus d'options concernant l'installation de l'invité sans efforts, consultez l'aide de la ligne de commande en utilisant la commande:

```
VBoxWindowsAdditions.exe /?
```

4.2.1.4 Extraction manuelle de fichiers

Si vous aimeriez installer les fichiers et les pilotes à la main, vous pouvez extraire les fichiers du paramétrage des additions invité Windows en tapant:

```
VBoxWindowsAdditions.exe /extract
```

Pour extraire explicitement les additions invité Windows pour une autre plateforme que celle actuellement en cours de fonctionnement (comme les fichiers 64 bits sur un système 32 bits), vous devez exécuter l'installateur de la plateforme appropriée (`VBoxWindowsAdditions-x86.exe` ou `VBoxWindowsAdditions-amd64.exe`) avec le paramètre `/extract`.

4.2.1.5 Le réseau et Windows Vista

Les versions précédentes de VirtualBox fournissaient par défaut une carte virtuelle AMD PCNet Ethernet à l'invité. Depuis que Microsoft n'inclut plus de pilote pour cette carte avec Windows (à partir de Windows Vista), si vous sélectionnez Windows Vista ou supérieur comme système d'exploitation invité de la machine virtuelle, VirtualBox présentera plutôt un contrôleur réseau virtuel Intel à l'invité (voir le chapitre 6.1, *Matériel de réseau virtuel*, page 91).

Cependant, si pour une raison quelconque vous avez une VM Windows Vista 32 bits configurée pour utiliser une carte AMD PCNet, vous n'aurez pas de réseau au départ dans l'invité.

Afin d'être pratique, VirtualBox est fourni avec un pilote 32 bits pour la carte AMD PCNet card, qui est fourni avec les additions invité Windows. Si vous l'installez sur un invité Vista 32 bits, le pilote sera automatiquement installé. Si pour une raison quelconque vous aimeriez installer le pilote à la main, vous pouvez extraire les fichiers requis du paramétrage des additions invité. Merci de consulter le chapitre 4.2.1.4, *Extraction manuelle de fichiers*, page 61 sur la façon d'effectuer cela. Vous trouverez alors les fichiers du pilote AMD PCNet dans le sous-répertoire `x86\Network\AMD\netamd.inf` du répertoire d'installation par défaut.

Sinon, modifiez les paramètres de la VM invité Vista pour utiliser une carte réseau Intel au lieu de la carte AMD PCNet par défaut ; voir le chapitre chapitre 3.8, *Paramètres réseau*, page 53.

Malheureusement, il n'y a pas de pilote 64 bits disponible pour la carte AMD PCNet. Donc pour les VMs Windows 64 bits, vous devriez toujours utiliser les périphériques réseau Intel.

4.2.2 Additions Invité pour Linux

Comme les additions invité Windows, les additions invité de VirtualBox pour Linux ont la forme d'un ensemble de pilotes de périphériques et d'applications système qui peuvent être installées dans le système d'exploitation invité.

Les distributions Linux suivantes sont officiellement supportées:

- Fedora à partir de Fedora Core 4;
- Redhat Enterprise Linux à partir de la version 3;
- SUSE et openSUSE Linux à partir de la version 9;
- Ubuntu à partir de la version 5.10.

De nombreuses autres distributions sont connues pour fonctionner avec les additions invité.

La version du noyau par défaut fournie par SUSE et openSUSE 10.2, Ubuntu 6.10 (toutes les versions) et Ubuntu 6.06 (édition serveur) contient un bogue qui peut l'amener à planter pendant le démarrage lorsqu'il est lancé dans une machine virtuelle. Les additions invité fonctionnent dans ces distributions.

Remarquez que certaines distributions Linux sont déjà fournies avec les additions invité de VirtualBox ou avec une partie d'entre eux. Vous pouvez conserver la version des additions invité de la distribution mais, souvent, ils ne sont pas à jour et leurs fonctionnalités sont limitées. Vous pouvez donc choisir d'installer les additions invité fournies avec VirtualBox, ce qui écrasera la version déjà installée. L'installateur des additions invité Linux de VirtualBox essaie de détecter une installation existante et il les remplace, mais selon la manière dont la distribution intègre les additions invité, ils peuvent exiger quelques interventions à la main. Il est hautement recommandé de prendre un instantané de la machine virtuelle avant d'écraser l'installation.

4.2.2.1 Installer les additions Invité Linux

Les additions invité de VirtualBox pour Linux sont fournies sur le même CD-ROM ISO que les additions pour Windows décrits ci-dessus. Ils sont aussi fournis avec un programme d'installation qui vous guide à travers le processus d'installation, bien que, du fait de différences importantes entre les distributions Linux, l'installation pourrait être légèrement plus complexe.

L'installation consiste en les étapes suivantes:

1. Avant d'installer les additions invités, vous devrez préparer votre système invité à construire les modules noyau externes. Cela fonctionne de la même façon que décrit au chapitre chapitre 2.3.2, *Le module noyau de VirtualBox*, page 35, sauf que cette étape doit se faire maintenant dans votre invité Linux au lieu d'un système Linux hôte, comme décrit ici.

   À nouveau, comme avec les hôtes Linux, nous vous recommandons d'utiliser DKMS pour les invités Linux. S'il n'est pas installé, utilisez cette commande pour les systèmes Ubuntu/Debian:

```
sudo apt-get install dkms
```

ou pour les systèmes Fedora:

```
yum install dkms
```

Assurez-vous d'installer DKMS *avant* d'installer les additions invité Linux.

2. Montez le fichier `VBoxGuestAdditions.iso` comme lecteur de CD-ROM virtuel de votre invité Linux, exactement de la même façon que décrit pour un invité Windows au chapitre 4.2.1.1, *Installation*, page 60.

3. Allez dans le répertoire où est monté votre lecteur de CD-ROM et exécutez en tant qu'administrateur:

```
sh ./VBoxLinuxAdditions-x86.run
```

Sur un invité Linux 64 bits, utilisez plutôt l'installeur `VBoxLinuxAdditions-amd64.run`.

Pour vous aider, les instructions pas-à-pas suivantes ont été vérifiées comme opérationnelles pour des copies fraîchement installées des distributions Linux les plus populaires. Après ces étapes préparatoires, vous pouvez exécuter l'installateur des additions invité de VirtualBox comme décrit ci-dessus.

Ubuntu 10.04 (Lucid Lynx)

1. Pour mettre à jour votre système vers la dernière version des paquets, ouvrez un terminal et, en tant qu'administrateur, exécutez

```
apt-get update
```

suivi de

```
apt-get upgrade
```

2. Installez DKMS en utilisant

```
apt-get install dkms
```

3. Redémarrez votre système invité afin d'activer les mises à jour puis procédez comme décrit ci-dessus.

Fedora 13 (Goddard)

1. Pour mettre à jour votre système vers la dernière version des paquets, ouvrez un terminal et, en tant qu'administrateur, exécutez

```
yum update
```

2. Installez DKMS et le compilateur GNU C en utilisant

```
yum install dkms
```

suivi de

```
yum install gcc
```

3. Redémarrez votre système invité afin d'activer les mises à jour puis procédez comme décrit ci-dessus.

openSUSE 11.2

1. Pour mettre à jour votre système vers la dernière version des paquets, ouvrez un terminal et, en tant qu'administrateur, exécutez

   ```
   zypper update
   ```

2. Installez l'outil make et le compilateur GNU C en utilisant

   ```
   zypper install make gcc
   ```

3. Redémarrez votre système invité afin d'activer les mises à jour.

4. Cherchez quel est le noyau que vous exécutez en utilisant

   ```
   uname -a
   ```

   Un exemple serait `2.6.31.12-0.2-default` qui se réfère au noyau par défaut. Puis, installez le bon paquet de développement du noyau. Dans l'exemple ci-dessus, il s'agirait de

   ```
   zypper install kernel-default-devel
   ```

5. Assurez-vous que votre noyau en fonction (uname -a) et les paquets du noyau que vous avez installés (`uname -a`) ont exactement le même numéro de version. Procédez comme décrit ci-dessus.

SuSE Linux Enterprise Desktop (SLED) 11

1. Pour mettre à jour votre système vers la dernière version des paquets, ouvrez un terminal et, en tant qu'administrateur, exécutez

   ```
   zypper update
   ```

2. Installez le compilateur GNU C en utilisant

   ```
   zypper install gcc
   ```

3. Redémarrez votre système invité afin d'activer les mises à jour.

4. Cherchez le noyau que vous exécutez en utilisant

   ```
   uname -a
   ```

   Un exemple serait `2.6.27.19-5.1-default` qui se réfère au noyau default. Puis installez le bon paquet de développement du noyau. Dans l'exemple ci-dessus, il s'agirait de

   ```
   zypper install kernel-syms kernel-source
   ```

5. Assurez-vous que votre noyau actuel (`uname -a`) et les paquets du noyau que vous avez installés (`rpm -qa kernel\*`) ont exactement le même numéro de version. Procédez comme décrit ci-dessus.

Mandrake 2010

1. Mandrake inclut les additions invité de VirtualBox qui seront remplacés si vous suivez ces étapes.

2. Pour mettre à jour votre système vers la dernière version des paquets, ouvrez un terminal et, en tant qu'administrateur, exécutez

   ```
   urpmi --auto-update
   ```

3. Redémarrez votre système invité afin d'activer les mises à jour.

4. Installez DKMS en utilisant

   ```
   urpmi dkms
   ```

   et assurez-vous de choisir le bon paquet kernel-devel lorsque cela vous est demandé (utilisez `uname -a` pour comparer).

CentOS 5.5, Red Hat Enterprise Linux 5.5 et Oracle Enterprise Linux 5.5

1. Ajoutez `divider=10` aux options de démarrage du noyau dans `/etc/grub.conf` pour réduire la charge d'occupation du processeur.
2. Pour mettre à jour votre système vers la dernière version des paquets, ouvrez un terminal et, en tant qu'administrateur, exécutez

   ```
   yum update
   ```

3. Installez le compilateur GNU C et les paquets de développement du noyau en utilisant

   ```
   yum install gcc
   ```

   suivi de

   ```
   yum install kernel-devel
   ```

4. Redémarrez votre système invité afin d'activer les mises à jour puis procédez comme décrit ci-dessus.
5. Remarquez que le support d'OpenGL n'est pas disponible, sauf si vous mettez à jour vers un noyau Linux récent.

   Si Oracle Enterprise Linux ne trruve pas les paquets requis, vous devez les installer soit à partir d'une autre source (comme un DVD) soit en utilisant le serveur public Yum d'Oracle qui se trouve sur `http://public-yum.oracle.com`.

Debian 5 (Lenny)

1. Pour mettre à jour votre système vers la dernière version des paquets, ouvrez un terminal et, en tant qu'administrateur, exécutez

   ```
   apt-get update
   ```

   suivi de

   ```
   apt-get upgrade
   ```

2. Installez l'outil make et le compilateur GNU C en utilisant

   ```
   apt-get install make gcc
   ```

3. Redémarrez votre système invité afin d'activer les mises à jour.
4. Déterminez la version exacte de votre noyau en utilisant `uname -a` et installez la bonne version du paquet linux-headers, en utilisant par exemple

   ```
   apt-get install linux-headers-2.6.26-2-686
   ```

5. Remarquez que le support d'OpenGL n'est pas disponible, sauf si vous mettez à jour vers un noyau Linux récent.

4.2.2.2 Paramétrage manuel des services invité sélectionnés

Les additions invité de VirtualBox contiennent plusieurs pilotes différents. Si pour une raison quelconque vous ne souhaitez pas tous les initialiser, vous pouvez installer les additions invité en utilisant la commande suivante:

```
sh ./VBoxLinuxAdditions-x86.run no_setup
```

(en remplaçant par `VBoxLinuxAdditions-amd64` sur un invité 64 bits).
Après cela, vous devrez au moins compiler les modules noyau en lançant la commande

```
/usr/lib/VBoxGuestAdditions/vboxadd setup
```

en tant qu'administrateur (vous devrez remplacer *lib* par *lib64* sur certains invités 64 bits), et sur les invités anciens sans le service udev vous devrez ajouter le service *vboxadd* au niveau d'exécution par défaut pour garantir que les modules seront chargés.
Pour initialiser le service de synchronisation de temps, lancez la commande

```
/usr/lib/VBoxGuestAdditions/vboxadd-service setup
```

et ajoutez le service vboxadd-service au niveau d'exécution par défaut. Pour initialiser la partie OpenGL de X11 des additions invité, lancez la commande

```
/usr/lib/VBoxGuestAdditions/vboxadd-x11 setup
```

(vous n'avez besoin d'activer aucun service pour cela).
Pour recompiler les modules du noyau de l'invité, utilisez cette commande:

```
/usr/lib/VBoxGuestAdditions/vboxadd setup
```

Après la compilation vous devriez redémarrer votre invité pour garantir que les nouveaux modules seront bien utilisés.

4.2.2.3 Accélération vidéo et modes graphiques haute résolution

Sur les invités Linux, l'accélération graphique de VirtualBox est disponible à travers le système X Window. En général, sur les distributions Linux actuelles, il s'agira du serveur X.Org. Pendant le processus d'installation, X sera initialisé pour utiliser le pilote graphique fourni avec les additions invité.

Pour les invités Linux et Solaris, la version 1.3 du serveur X.org est nécessaire pour le redimensionnement (la fonctionnalité a été désactivée sur les invités Fedora 9 du fait d'un bogue dans leur serveur X). La version du serveur peut être vérifiée avec `Xorg -version`.

Vous pouvez aussi envoyer des astuces du mode graphique en utilisant l'outil `VBoxManage`.

Si vous n'utilisez que des systèmes invités Linux récents, vous pouvez sauter le reste de cette section. Sur les systèmes invités anciens, tous les modes graphiques initialisés avant l'installation seront utilisés. Si ces modes ne satisfont pas vos exigences, vous pouvez modifier vos paramètres en éditant le fichier de configuration du serveur X, situé en général dans `/etc/X11/xorg.conf`.

VirtualBox peut utiliser n'importe quel mode graphique X par défaut adapté à la mémoire vidéo virtuelle allouée à la machine virtuelle comme décrit au chapitre chapitre 3.3, *Paramètres généraux*, page 46. Vous pouvez aussi ajouter vos propres modes au fichier de configuration du serveur X. Vous n'avez besoin que de les ajouter à la liste des modes de la sous-section Display de la section Screen. Par exemple, la section montrée ici a une résolution personnalisée en mode ajouté 2048x800:

```
Section "Screen"
        Identifier    "Default Screen"
        Device        "VirtualBox graphics card"
        Monitor       "Generic Monitor"
        DefaultDepth  24
        SubSection "Display"
                Depth         24
                Modes         "2048x800" "800x600" "640x480"
        EndSubSection
EndSection
```

4.2.2.4 Mettre à jour les additions Invité Linux

Vous pouvez simplement mettre à jour les additions invité en refaisant la procédure d'installation avec une image de CD-ROM mise à jour. Ceci remplacera les pilotes par les versions mises à jour. Vous devriez redémarrer après la mise à jour des additions invité.

4.2.2.5 Désinstaller les additions invité Linux

Si vous avez installé une version des additions invité sur votre machine virtuelle et si vous souhaitez la supprimer sans en installer de nouvelles, vous pouvez le faire en insérant l'image du CD des additions invité dans le lecteur de CD-ROM virtuel comme décrit ci-dessus et en lançant l'installeur des additions invité actuelles avec le paramètre uninstall depuis le chemin de l'invité où est monté l'image de CD:

```
sh ./VBoxLinuxAdditions-x86.run uninstall
```

(en remplaçant `VBoxLinuxAdditions-amd64` sur un invité 64 bits). Si cela fonctionnera normalement sans problèmes, il se peut que vous deviez faire du nettoyage manuel de l'invité (en particulier du fichier XFree86Config ou xorg.conf) dans certains cas, surtout si la version des additions installée ou si le système d'exploitation invité était très vieux, ou si vous avez fait vos propres modifications sur le réglage des additions invité après avoir installé.

À partir de la version 3.1.0, vous pouvez désinstaller les additions en appelant

```
/opt/VBoxGuestAdditions-4.2.12_RPMFusion/uninstall.sh
```

en remplaçant `/opt/VBoxGuestAdditions-4.2.12_RPMFusion` par le répertoire d'installation des additions invité.

4.2.3 Additions Invité pour Solaris

Comme les additions invité Windows, les additions invité de Solaris ont la forme de pilotes de périphériques et d'applications système qui peuvent être installés sur le système d'exploitation invité.

Les distributions Solaris suivantes sont officiellement supportées:

- OpenSolaris Nevada (construction 82 et supérieur; ceci inclut OpenSolaris 2008.05, 2008.11 et 2009.06);
- OpenSolaris Indiana (Developer Preview 2 et supérieur);
- Solaris 10 (u5 et supérieur).

Il se peut que d'autres distributions fonctionnent si elles se basent sur des versions de logiciels comparables.

4.2.3.1 Installer les additions Solaris

Les additions invité VirtualBox pour Solaris sont fournis sur la même ISO de CD-ROM que les additions pour Windows et Linux décrits ci-dessus. Ils sont aussi fournis avec un programme d'installation vous guidant à travers le processus d'initialisation.

L'installation implique les étapes suivantes:

1. Monter le fichier `VBoxGuestAdditions.iso` en tant que votre lecteur de CD-ROM virtuel de votre invité Solaris, exactement de la même façon que celle décrite pour un invité Windows au chapitre 4.2.1.1, *Installation*, page 60.

   Si le lecteur CD-ROM de l'invité n'est pas monté, (on le voit sur certaines versions de Solaris 10), exécutez en tant qu'administrateur:

   ```
   svcadm restart volfs
   ```

2. Aller dans le répertoire où est monté votre lecteur CD-ROM et exécutez en tant qu'administrateur:

   ```
   pkgadd -G -d ./VBoxSolarisAdditions.pkg
   ```

3. Choisissez 1 et confirmer l'installation du paquet des additions invité. Après que l'installation est terminée, reconnectez le serveur X de votre invité pour activer les additions invité X11.

4.2.3.2 Désinstaller les additions Solaris

Vous pouvez supprimer en toute sécurité les additions invité Solaris en supprimant le paquet de l'invité. Ouvrez une session de terminal administrateur et exécutez:

```
pkgrm SUNWvboxguest
```

4.2.3.3 Mettre à jour les additions Solaris

Les additions invité devraient être mis à jour en désinstallant d'abord les additions invité existants puis en installant les nouveaux. Tenter d'installer les nouvelles additions invité sans supprimer celles existantes n'est pas possible.

4.2.4 Additions Invité pour OS/2

VirtualBox offre aussi un ensemble de pilotes qui améliorent le fonctionnement d'OS/2 dans une machine virtuelle. Du fait de restrictions d'OS/2 lui-même, cette variante des additions invité a un ensemble de fonctionnalités limité; voir le chapitre 14, *Known limitations*, page 201 pour les détails.

Les additions invité OS/2 sont fournies sur la même ISO de CD-ROM comme ceux pour les autres plateformes. Par conséquent, montez l'ISO sous OS/2 comme décrit précédemment. Les additions invité OS/2 se situent dans le répertoire `\32bit\OS2`.

Comme nous ne fournissons pas d'installateur automatique pour le moment, merci de vous référer au fichier `readme.txt` de ce répertoire qui décrit comment installer les additions invité OS/2 à la main.

4.3 Dossiers partagés

Avec la fonctionnalité dossiers partagés de VirtualBox, vous pouvez accéder à des fichiers de votre système hôte à partir de l'intérieur du système invité, presque comme le feraient les partages ordinaires sur les réseaux Windows – sauf que les dossiers partagés, tant que les additions invité

sont installés, n'exigent pas de réseau –. Les dossiers partagés sont supportés avec des invités Windows (2000 ou supérieur), Linux et Solaris.

Les dossiers partagés doivent résider physiquement sur *l'hôte* et sont alors partagés avec l'invité; le partage se fait en utilisant un service spécial de l'hôte et un pilote de système de fichiers pour l'invité, les deux étant fournis par VirtualBox. Pour les invités Windows, les dossiers partagés sont implémentés comme un redirecteur pseudo-réseau; pour les invités Linux et Solaris, les additions invité fournissent un pilote de système de fichiers virtuel qui gère la communication avec l'hôte.

Pour partager un dossier hôte avec une machine virtuelle dans VirtualBox, vous devez spécifier le chemin vers ce dossiers et lui choisir un nom de partage que l'invité peut utiliser pour y accéder. Aussi, créez d'abord le dossier partagé sur l'hôte, puis à l'intérieur de l'invité, connectez-vous-y.

Il y a plusieurs façons de paramétrer les dossiers partagés pour une machine virtuelle particulière:

- Dans l'interface graphique de la machine virtuelle en fonction, vous pouvez sélectionner Dossiers partagés depuis le menu Périphériques, ou cliquer sur l'icône de dossier de la barre de statut dans le coin en bas à droite de la fenêtre de la machine virtuelle.
- Si une machine virtuelle n'est pas actuellement en fonction, vous pouvez configurer les dossiers partagés dans la boîte de dialogue Paramètres de chaque machine virtuelle.
- Depuis la ligne de commande, vous pouvez créer des dossiers partagés en utilisant l'interface en ligne de commande VBoxManage; voir le chapitre 8, *VBoxManage*, page 110. La commande est comme suit:

  ```
  VBoxManage sharedfolder add "nom VM" --name "sharename" --hostpath "C:\test"
  ```

Il y a deux types de partages:

1. Les partages de VM qui ne sont disponibles que pour la VM pour la quelle ils ont été définis;
2. Les partages avec VM transitoires, qui peuvent être ajoutés et supprimés au moment de l'exécution et qui ne demeurent pas après qu'une VM a été arrêtée; pour ces derniers, ajoutez l'option `--transient` à la ligne de commande ci-dessus.

Les dossiers partagés ont un accès en lecture/écriture aux fichiers du chemin de l'hôte par défaut. Pour obliger l'invité à n'avoir un accès qu'en lecture seule, créez un répertoire partagé en lecture seule. Vous pouvez faire cela, soit en utilisant la GUI, soit en indiquant le paramètre `--readonly` lorsque vous créez le dossier partagé avec VBoxManage.

4.3.1 Montage manuel

Vous pouvez monter le dossier partagé depuis l'intérieur d'une VM de la même façon que vous monteriez un partage réseau ordinaire:

- Sur un invité Windows, à partir de VirtualBox 1.5.0, on peut naviguer dans les dossiers partagés et ils sont donc visibles dans l'explorateur Windows. Donc, pour attacher le dossier partagé de l'hôte à votre invité Windows, ouvrez l'explorateur Windows et cherchez-le sous Mes emplacements réseau -> Tout le réseau -> Dossiers partagés de VirtualBox. En faisant un clic droit sur un dossier partagé et en cliquant sur Connecter un lecteur réseau dans le menu qui s'affiche, vous pouvez affecter une lettre de lecteur à ce dossier partagé.

  En alternative, sur la ligne de commande Windows, utilisez ce qui suit:

  ```
  net use x: \\vboxsvr\nom_partage
  ```

  Alors que `vboxsvr` est un nom corrigé (notez que vboxsrv devrait aussi fonctionner), remplacez x: par la lettre de lecteur que vous voulez utiliser pour le partage, et `nom_partage` par le nom partagé spécifié avec `VBoxManage`.

- Sur un invité Linux, utilisez la commande suivante:

  ```
  mount -t vboxsf [-o OPTIONS] nom_partage point_montage
  ```

  Pour monter un dossier partagé au démarrage, ajoutez l'entrée suivante à /etc/fstab:

  ```
  sharename   mountpoint   vboxsf   defaults  0   0
  ```

- Sur un invité Solaris, utilisez la commande suivante:

  ```
  mount -F vboxfs [-o OPTIONS] nom_partage point_montage
  ```

  Remplacez `nom_partage` (utilisez des minuscules) par le nom réseau spécifié avec `VBoxManage` ou la GUI, et point_montage par le chemin où vous voulez que le partage soit monté sur l'invité (comme `/mnt/share`). Les règles de montage habituelles s'appliquent, à savoir créez ce répertoire au préalable s'il n'existe pas encore.

  Voici un exemple de montage de dossier partagé pour l'utilisateur jack sur OpenSolaris:

  ```
  $ id
  uid=5000(jack) gid=1(other)
  $ mkdir /export/home/jack/mount
  $ pfexec mount -F vboxfs -o uid=5000,gid=1 jackshare /export/home/jack/mount
  $ cd ~/mount
  $ ls
  sharedfile1.mp3 sharedfile2.txt
  $
  ```

  Au-delà des options de montage fournies par la commande `mount`, celles suivantes sont disponibles:

  ```
  iocharset CHARSET
  ```

  pour régler l'encodage utilisé pour les opérations E/S (utf8 par défaut) et

  ```
  convertcp CHARSET
  ```

  pour spécifier l'encodage utilisé pour le nom du dossier partagé (utf8 par défaut).

  Les options génériques de mount (documentées dans la page de manuel de mount) s'appliquent également. Celles particulièrement utiles sont les options `gid` et `mode`, car elles autorisent l'accès par des utilisateurs normaux (en mode lecture/écriture selon les réglages) même si l'administrateur a monté le système de fichiers.

4.3.2 Montage automatique

À partir de la version 3.3.0, VirtualBox supporte le montage automatique des dossiers partagés. Les additions invité installées prendront en compte tous les dossiers partagés qui sont marqués comme devant être montés automatiquement dès que l'utilisateur est connecté sur le système d'exploitation invité. Cela le rend plus pratique, plutôt que de monter les dossiers partagés à la main comme décrit au chapitre 4.3.1, *Montage manuel*, page 69.

> **Note:** Le montage automatique n'est actuellement supporté que sur les invités Windows, Linux et Solaris.

Sur les invités Windows, un dossier monté automatiquement sera représenté par sa propre lettre de lecteur (comme `E:`), selon les lettres de lecteur qui restent disponibles sur le système.

Sur les invités Linux et Solaris, les dossiers partagés automatiquement montés sont montés dans le répertoire `/media`, avec le préfixe `sf_`, donc le dossier partagé `myfiles` serait monté dans `/media/sf_myfiles` sur Linux et dans `/mnt/sf_myfiles` sur Solaris.

Pour modifier le préfixe `sf_` d'une machine virtuelle donnée, réglez la valeur de sa propriété invité `/VirtualBox/GuestAdd/SharedFolders/MountPrefix` sur une autre valeur; voir le chapitre 4.6, *Propriétés invité*, page 73 pour des détails.

Pour qu'un utilisateur ait accès complet aux dossiers partagés automatiquement montés sur l'invité, cet utilisateur a besoin de faire partie du groupe nouvellement créé vboxsf, qui est créé par l'installeur des additions invité de VirtualBox. S'il n'est pas dans ce groupe, il aura un accès en lecture seule.

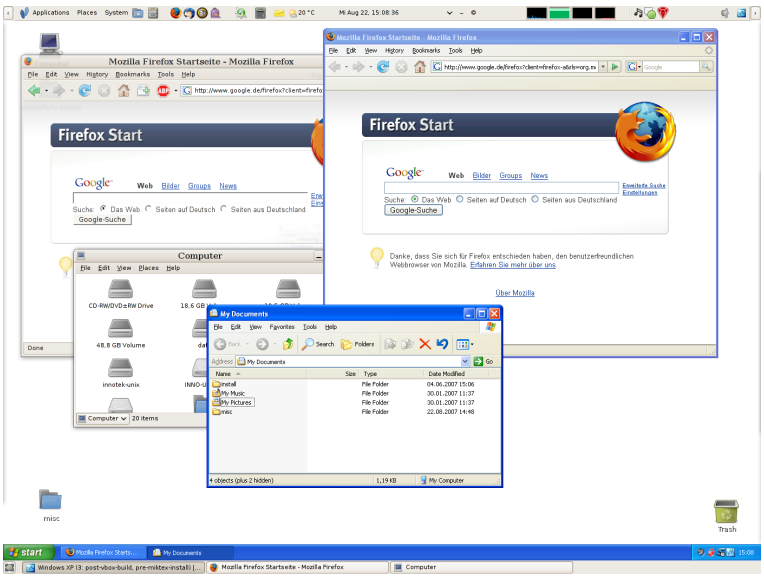
Pour appliquer les modifications, par exemple l'ajout ou la suppression d'un nouveau dossier monté automatiquement, alors qu'une VM est en fonction, il faut redémarrer l'OS invité. Cependant, cela ne touche pas chapitre 4.3.1, *Montage manuel*, page 69.

4.4 Fenêtres intégrées

Avec la fonctionnalité fenêtres intégrées de VirtualBox, vous pouvez faire apparaître les fenêtres affichées dans une machine virtuelle côte à côte près des fenêtres de votre hôte. Cette fonctionnalité est supportée pour les systèmes d'exploitation suivants (fournis lorsque les additions invité sont installées):

- Invités Windows (support ajoutés avec VirtualBox 1.5);
- Invités Linux ou Solaris/OpenSolaris avec une version 1.3 ou supérieure du serveur X.org [1] (support ajouté avec VirtualBox 1.6). L'exception est Fedora 9, du fait d'un bogue dans leur serveur X.

Après que l'intégration des fenêtres a été activée (voir ci-dessous), VirtualBox supprime l'affichage du fond d'écran du bureau de votre invité, vous permettant de lancer les fenêtres de votre système d'exploitation invité de façon intégrée à côté des fenêtres de votre hôte:



Pour activer ce mode, après avoir démarré la machine virtuelle, appuyez sur la touche Hôte (en principe la touche Contrôle droite) simultanément avec L. Cela agrandira la taille de l'affichage de la VM jusqu'à la taille de l'écran de votre hôte et masquera le fond d'écran du système d'exploitation invité. Pour revenir à l'affichage normal de la VM (désactivant ainsi l'intégration des fenêtres), ré-appuyez sur les touches Hôte et L.

[1] La version du serveur X n'est pas la même que la version de toute la suite X.org. Vous pouvez taper `X -version` dans un terminal pour connaître le niveau de version du serveur X.org actuellement installée.

4.5 Le graphisme avec l'accélération matérielle

4.5.1 Accélération 3D matérielle (OpenGL et Direct3D 8/9)

Les additions invité de VirtualBox contiennent un support matériel expérimental du 3D pour les invités Windows, Linux et Solaris.[2]

Avec cette fonctionnalité, si une application à l'intérieur de votre machine virtuelle utilise des fonctionnalités 3D passant par les interfa ces de programmation d'OpenGL ou de Direct3D 8/9, au lieu de les émuler de manière logicielle (ce qui serait lent), VirtualBox essaiera d'utiliser le matériel 3D de votre hôte. Ceci fonctionne pour toutes les plateformes supportées (Windows, Mac, Linux, Solaris), à condition que votre système d'exploitation hôte puisse d'abord lui-même utiliser votre matériel 3D accéléré.

L'accélération 3D nécessite actuellement les conditions suivantes:

1. Elle n'est disponible que pour certains invités Windows, Linux et Solaris. En particulier:
   - Pour les invités Windows, le support se réduit aux versions 32 bits d'XP et de Vista. OpenGL et Direct3D 8/9 sont supportés (expérimental).
   - OpenGL sur Linux exige un noyau 2.6.27 et supérieur ainsi qu'une version du serveur X.org 1.5 et supérieure. Ubuntu 8.10 et Fedora 10 ont été testées et confirmées comme fonctionnant.
   - OpenGL sur les invités Solaris exige une version de X.org 1.5 et supérieur.
2. Les additions invité doivent être installés.

   Note: Pour que l'accélération Direct 3D fonctionne dans un invité Windows, VirtualBox doit remplacer les fichiers du système Windows de la machine virtuelle. Il en résulte que l'installation du programme des additions invité offre l'accélération Direct 3D en tant qu'option qui doit être explicitement activée.Vous devez aussi installer les additions invité en mode sécurisé; voir le chapitre 14, *Known limitations*, page 201 pour les détails.

3. Vu que le support 3D est encore pour l'instant expérimental, il est désactivé par défaut et vous devez **l'activer à la main** dans les paramètres de la VM (voir le chapitre chapitre 3.3, *Paramètres généraux*, page 46).

   Note: Il se peut que l'activation de l'accélération 3D expose des trous de sécurité à des logiciels malveillants en fonction sur l'invité. Le code tiers utilisé par VirtualBox à cette fin (Chromium) n'est pas assez endurci pour empêcher n'importe quelle opération 3D risquée sur l'hôte.

Techniquement, VirtualBox implémente cela en installant un pilote 3D matériel supplémentaire dans votre invité lorsque vous installez les additions invité. Ce pilote agit comme un pilote de matériel 3D et signale au système d'exploitation invité que le matériel (virtuel) est capable d'une accélération 3D matérielle. Lorsqu'une application sur l'invité demande l'accélération matérielle par les interfaces de programmation d'OpenGL ou de Direct3D, les requêtes sont envoyées à l'hôte par un tunnel de communication spécial implémenté par VirtualBox, puis l'hôte effectue l'opération 3D demandée par les interfaces de programmation de l'hôte.

[2] Le support d'OpenGL pour les invités Windows a été ajouté avec VirtualBox 2.1; le support pour Linux et Solaris a suivi avec VirtualBox 2.2. Avec VirtualBox 3.0, le support Direct3D 8/9 a été ajouté pour les invités Windows. OpenGL 2.0 est maintenant également supporté.

4.5.2 Accélération vidéo 2D pour Windows

À partir de la version 3.1, les additions invité de VirtualBox contiennent le support expérimental de l'accélération 2D matérielle pour les invités Windows.

Avec cette fonctionnalité, si une application (telle qu'un lecteur vidéo) dans votre VM utilise les superpositions vidéos 2D pour jouer un clip animé, VirtualBox essaiera alors d'utiliser le matériel d'accélération graphique de votre hôte au lieu de l'étirement des superpositions ou la conversion de couleurs logiciels (ce qui serait lent). Cela fonctionne actuellement pour les plateformes d'hôtes Windows, Linux et Mac, à condition que votre système d'exploitation puisse lui-même utiliser l'accélération graphique 2D.

L'accélération graphique 2D fonctionne actuellement sous les conditions suivantes:

1. Elle n'est disponible que pour les invités Windows (XP ou supérieur).
2. Les additions invité doivent être installés.
3. Comme le support 2D est encore pour l'instant expérimental, il est par défaut désactivé et vous devez **l'activer à la main** dans les paramètres de la VM (voir le chapitre 3.3, *Paramètres généraux*, page 46).

Techniquement, VirtualBox implémente cela en offrant dans le pilote graphique invité les capacités de superposition graphique DirectDraw. Le pilote envoie toutes les commandes de superposition à l'hôte à travers un tunnel de communication spécial implémenté par VirtualBox. De l'autre côté, OpenGL est alors utilisé pour implémenter échelonnement de transformation d'espace de couleurs

4.6 Propriétés invité

À partir de la version 2.1, VirtualBox vous permet d'exiger certaines propriétés de la part d'un invité en fonction, à condition que les additions invité de VirtualBox soient installées et que la VM soit en fonction. Ceci est intéressant pour deux choses:

1. Un certain nombre de caractéristiques prédéfinies d'une VM peuvent être automatiquement maintenues par l'hôte et récupérées sur l'hôte, comme la surveillance de performances et des statiques de la VM.
2. Des chaînes de données de votre choix peuvent être échangées entre l'invité et l'hôte. Cela fonctionne dans les deux sens.

Pour faire cela, VirtualBox établit un canal de communication privé entre les additions invité de VirtualBox et l'hôte, et le logiciel des deux côtés peut utiliser ce canal pour échanger des chaînes de données à des fins de votre choix. Les propriétés de l'invité sont simplement des chaînes de touches auxquelles est attachée une valeur. Elles peuvent être paramétrées (on peut écrire dessus) soit depuis l'hôte, soit depuis l'invité, et on peut les lire des deux côtés.

En plus d'établir un mécanisme général de valeurs de lecture et d'écriture, le paramétrage de propriétés d'invité prédéfinies est automatiquement maintenu par les additions invité de VirtualBox pour permettre de récupérer des données de l'invité intéressantes telles que le système d'exploitation exact de l'invité et le niveau du pack service, la version des additions invité installée, les utilisateurs actuellement connectés à l'OS invité, les statistiques du réseau et davantage. Ces propriétés prédéfinies sont toutes précédées de `/VirtualBox/` et organisées dans une arborescence de clés hiérarchiques.

Certaines de ces informations apparaissent en fonctionnement quand vous sélectionnez Boîte de dialogue d'informations de session depuis le menu Machine d'une machine virtuelle.

Une façon plus flexible d'utiliser ce canal est de passer par la commande de paramétrage `VBoxManage guestproperty`; voir le chapitre 8.29, *VBoxManage guestproperty*, page 139

pour les détails. Par exemple, pour que toutes les propriétés invité soient disponibles pour une VM en cours de fonctionnement donnée, listées avec leurs valeurs respectives, utilisez ceci:

```
$ VBoxManage guestproperty enumerate "Windows Vista III"
VirtualBox Command Line Management Interface Version 4.2.12
(C) 2005-2013 Oracle Corporation
All rights reserved.

Name: /VirtualBox/GuestInfo/OS/Product, value: Windows Vista Business Edition,
    timestamp: 1229098278843087000, flags:
Name: /VirtualBox/GuestInfo/OS/Release, value: 6.0.6001,
    timestamp: 1229098278950553000, flags:
Name: /VirtualBox/GuestInfo/OS/ServicePack, value: 1,
    timestamp: 1229098279122627000, flags:
Name: /VirtualBox/GuestAdd/InstallDir,
    value: C:/Program Files/Oracle/VirtualBox
    Guest Additions, timestamp: 1229098279269739000, flags:
Name: /VirtualBox/GuestAdd/Revision, value: 40720,
    timestamp: 1229098279345664000, flags:
Name: /VirtualBox/GuestAdd/Version, value: 4.2.12,
    timestamp: 1229098279479515000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxControl.exe, value: 4.2.12r40720,
    timestamp: 1229098279651731000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxHook.dll, value: 4.2.12r40720,
    timestamp: 1229098279804835000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxDisp.dll, value: 4.2.12r40720,
    timestamp: 1229098279880611000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxMRXNP.dll, value: 4.2.12r40720,
    timestamp: 1229098279882618000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxService.exe, value: 4.2.12r40720,
    timestamp: 1229098279883195000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxTray.exe, value: 4.2.12r40720,
    timestamp: 1229098279885027000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxGuest.sys, value: 4.2.12r40720,
    timestamp: 1229098279886838000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxMouse.sys, value: 4.2.12r40720,
    timestamp: 1229098279890600000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxSF.sys, value: 4.2.12r40720,
    timestamp: 1229098279893056000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxVideo.sys, value: 4.2.12r40720,
    timestamp: 1229098279895767000, flags:
Name: /VirtualBox/GuestInfo/OS/LoggedInUsers, value: 1,
    timestamp: 1229099826317660000, flags:
Name: /VirtualBox/GuestInfo/OS/NoLoggedInUsers, value: false,
    timestamp: 1229098455580553000, flags:
Name: /VirtualBox/GuestInfo/Net/Count, value: 1,
    timestamp: 1229099826299785000, flags:
Name: /VirtualBox/HostInfo/GUI/LanguageID, value: C,
    timestamp: 1229098151272771000, flags:
Name: /VirtualBox/GuestInfo/Net/0/V4/IP, value: 192.168.2.102,
    timestamp: 1229099826300088000, flags:
Name: /VirtualBox/GuestInfo/Net/0/V4/Broadcast, value: 255.255.255.255,
    timestamp: 1229099826300220000, flags:
Name: /VirtualBox/GuestInfo/Net/0/V4/Netmask, value: 255.255.255.0,
    timestamp: 1229099826300350000, flags:
Name: /VirtualBox/GuestInfo/Net/0/Status, value: Up,
    timestamp: 1229099826300524000, flags:
Name: /VirtualBox/GuestInfo/OS/LoggedInUsersList, value: username,
    timestamp: 1229099826317386000, flags:
```

Pour chercher la valeur d'une seule propriété, utilisez la sous-commande get comme ceci:

```
$ VBoxManage guestproperty get "Windows Vista III"
         "/VirtualBox/GuestInfo/OS/Product"
VirtualBox Command Line Management Interface Version 4.2.12
(C) 2005-2013 Oracle Corporation
All rights reserved.
```

```
Value: Windows Vista Business Edition
```

Pour ajouter ou modifier des propriétés depuis l'invité, utilisez l'outil `VBoxControl`. Cet outil est inclu avec les additions invité de VirtualBox 2.2 ou supérieur. Lorsqu'il est lancé depuis un invité Linux, cet outil exige les privilèges administrateur pour des raisons de sécurité:

```
$ sudo VBoxControl guestproperty enumerate
VirtualBox Guest Additions Command Line Management Interface Version 4.2.12
(C) 2009-2013 Oracle Corporation
All rights reserved.

Name: /VirtualBox/GuestInfo/OS/Release, value: 2.6.28-18-generic,
    timestamp: 1265813265835667000, flags: <NULL>
Name: /VirtualBox/GuestInfo/OS/Version, value: #59-Ubuntu SMP Thu Jan 28 01:23:03 UTC 2010,
    timestamp: 1265813265836305000, flags: <NULL>
      ...
```

Pour des besoins plus complexes, vous pouvez utiliser les interfaces de programmation de VirtualBox; voir le chapitre 11, *VirtualBox programming interfaces*, page 181.

4.7 Contrôle invité

À partir de la version 3.2, les additions invité de VirtualBox permettent le démarrage d'applications à l'intérieur d'une VM depuis le système hôte.

Pour que cela fonctionne, l'application doit être installée dans l'invité; aucun logiciel supplémentaire ne doit être installé sur l'hôte. En outre, la sortie en mode texte (sur stdout et stderr) peut être affiché sur l'hôte pour un traitement ultérieur avec des options pour spécifier les droits de l'utilisateur et une valeur de timeout (en millisecondes) pour limiter le temps pendant lequel l'application peut s'exécuter.

Cette fonctionnalité peut être utilisée pour automatiser le déploiement de logiciels dans l'invité.

Pour utiliser cette fonctionnalité, utilisez la ligne de commande de VirtualBox. Voir le chapitre 8.30, *VBoxManage guestcontrol*, page 140 pour des détails.

4.8 Faire de la montgolfière avec la mémoire

À partir de la version 3.2, les additions invité de VirtualBox peuvent modifier la quantité de mémoire d'une machine virtuelle alors que la machine est en fonction. Vu la façon dont ceci est implémenté, cette fonctionnalité est appelée faire de la montgolfière.

Normalement, pour modifier la quantité de mémoire allouée à une machine virtuelle, il faut éteindre complètement la machine virtuelle et modifier les paramètres de la machine virtuelle. Avec la pratique de la montgolfière, on peut donner de la mémoire qui a été allouée à une machine virtuelle à une autre machine virtuelle sans devoir éteindre la machine. Cela peut être utile pour démarrer temporairement une autre machine virtuelle, ou dans des environnements plus compliqués pour la gestion sophistiquée de la mémoire de beaucoup de machines virtuelles qui peuvent être en fonction en même temps, en fonction de la façon dont la mémoire est utilisée par les invités.

Quand on demande la mémoire en montgolfière, les additions invité de VirtualBox (qui se lancent dans l'invité) allouent de la mémoire physique à partir du système d'exploitation invité au niveau du noyau et verrouillent vers le bas cette mémoire dans l'invité. Cela garantit que l'invité n'utilisera plus cette mémoire: aucune application invitée ne peut l'allouer, et le système d'exploitation invité ne l'utilisera pas non plus. VirtualBox peut alors réutiliser cette mémoire et la donner à une seconde machine.

La mémoire rendue disponible par le mécanisme de la montgolfière n'est disponible que pour une réutilisation par VirtualBox. Elle n'est pas libérée pour l'hôte. La demande de la montgolfière depuis un invité en fonction n'augmentera donc pas la quantité de mémoire libre et non allouée de l'hôte.

En fait, la pratique de la montgolfière est donc un mécanisme de réaffectation de mémoire pour plusieurs machines virtuelles alors qu'elles sont en fonction.

Pour l'instant, la pratique de la montgolfière avec de la mémoire n'est supportée que dans VBoxManage, l'outil de VirtualBox en ligne de commande. Utilisez la commande suivante pour augmenter ou diminuer la taille du ballon de la mémoire dans une machine virtuelle en fonction où les additions invité sont installés:

```
VBoxManage controlvm "nom VM" guestmemoryballoon <n>
```

où `"nom VM"` est le nom ou l'UUID de la machine virtuelle en question et `<n>` est la quantité de mémoire à allouer depuis l'invité, en mégaoctets; voir le chapitre 8.12, *VBoxManage controlvm*, page 129 pour plus d'informations..

Vous pouvez aussi régler un ballon par défaut qui sera automatiquement demandée depuis la VM à chaque fois qu'elle a été démarrée avec la commande suivante:

```
VBoxManage modifyvm "nom VM" --guestmemoryballoon <n>
```

Par défaut, aucun ballon de mémoire n'est alloué. C'est un paramètre de la VM, comme d'autres paramètres de modifyvm, et il peut donc être réglé alors que machine éteinte; voir le chapitre 8.7, *VBoxManage modifyvm*, page 120.

Note: VirtualBox ne supporte la pratique de la montgolfière que sur les hôtes 64 bits, la pratique de la montgolfière n'est pas supportée sur les hôtes Mac OS X.

4.9 Fusion de page

La fusion de page est une technologie nouvelle pour améliorer encore plus la densité de la VM sur l'hôte, et donc une manière de partager les ressources. Elle a été introduite tout d'abord avec VirtualBox 3.2 et est actuellement limitée aux VMs qui utilisent Windows 2000 et supérieur. Dans un scénario classique, des douzaines, voire des centaines de VMs très similaires sont consolidées sur un ordinateur hôte puissant et le niveau de consolidation est limité le plus souvent par la quantité de RAM qui peut être installée sur un système pour un coût raisonnable. Souvent, du fait d'un épuisement de RAM, on ne peut démarrer de VMs supplémentaires, bien que les processeurs de l'hôte offrent encore de la capacité. Pour dépasser cette limite, les hyperviseurs peuvent bénéficier du fait que souvent, les VMs se ressemblent (comme plusieurs VMs exécutant Windows XP Pack Service 2) et donc, elles contiennent une quantité de cellules de RAM identiques. L'hyperviseur peut chercher de telles données en double en mémoire, éliminer les redondances et libérer ainsi de la mémoire supplémentaire.

Les hyperviseurs traditionnels utilisent une technique souvent appelée le partage de mémoire ou la synchronisation de la même page quand ils parcourent toute la mémoire et calculent des sommes de contrôle (hashes) pour chaque page de mémoire. Puis, ils cherchent des pages avec des hashes identiques et ils comparent le contenu des pages (si deux pages donnent le mêmes hash, c'est très vraisemblablement que les pages ont un contenu identique). Les pages identiques sont éliminées afin que toutes les VMs pointent vers la même page tant qu'aucune VM n'essaie de modifier la page. Si une telle page est modifiée, le doublon précédemment éliminé est à nouveau alloué. Tout cela est complètement transparent pour la machine virtuelle. Cependant, l'algorithme classique comporte plusieurs inconvénients. Avant tout, il est plutôt long pour scanner la mémoire complète (surtout lorsque le système n'est pas en veille) donc la mémoire supplémentaire ne devient disponibles qu'après du temps (cela peut prendre plusieurs

jours!). En outre, tout l'algorithme de partage de page consomme en général des ressources processeur significatives et augmente la surcharge de virtualisation de 10 à 20%.

La fusion de page dans VirtualBox utilise les additions invité de VirtualBox pour identifier les cellules de mémoire qui sont le plus vraisemblablement identiques entre les VMs et donc effectue la plupart des sauvegardes possibles de partage de mémoire presque immédiatement et presque sans surcharge. La fusion de page a également beaucoup moins de chances d'être faussée par de la mémoire identique en l'éliminant pour apprendre quelques secondes plus tard que la mémoire a maintenant changé, devant alors effectuer une ré-allocation très coûteuse et souvent perturbatrice.

Vous pouvez activer la fusion de page pour une VM en utilisant:

```
VBoxManage modifyvm "nom VM" --pagefusion on
```

Vous pouvez observer l'opération de fusion de page en utilisant certaines unités. `RAM/VMM/Shared` affiche la quantité totale de pages fusionnées alors que l'unité `Guest/RAM/Usage/Shared` par VM retournera la quantité de mémoire fusionnée pour une machine donnée. Merci de vous reporter au chapitre 8.32, *VBoxManage metrics*, page 144 pour des informations sur la façon de demander des unités.

Note: VirtualBox ne supporte la fusion de page que sur des systèmes d'exploitation hôtes 64 bits. Les hôtes Mac OS X ne sont pas actuellement supportés. La fusion de page n'est disponible que pour les invités Windows 2000 et supérieur avec les additions invité actuelles.

5 Stockage virtuel

Vu que la machine virtuelle s'attendra très probablement à voir un disque dur virtuel construit dans son ordinateur virtuel, VirtualBox doit pouvoir présenter un support de stockage « réel » à l'invité comme un disque dur virtuel. Il y a actuellement trois méthodes pour effectuer cela :

1. Le plus souvent, VirtualBox utilisera les grands fichiers images sur un vrai disque dur et les présenter à un invité comme un disque dur virtuel. Ceci est décrit au chapitre 5.2, *Fichiers images de disque (VDI, VMDK, VHD, HDD)*, page 81.

2. Alternativement, si vous avez des serveurs de stockage iSCSI, vous pouvez aussi attacher de tels serveurs à VirtualBox; ceci est décrit au chapitre 5.11, *Serveurs iSCSI*, page 90.

3. Enfin, en fonctionnalité expérimentale, vous pouvez autoriser une machine virtuelle à accéder directement à un de vos disques hôtes; cette fonctionnalité avancée est décrite au chapitre 9.8.1, *Using a raw host hard disk from a guest*, page 157.

Chacun de ces périphériques de stockage virtuels (fichier image, cible iSCSI ou disque dur physique) devra être connecté au contrôleur de disque dur virtuel que présente VirtualBox à une machine virtuelle. Ceci est expliqué dans la prochaine section.

5.1 Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS

Dans un vrai PC, les disques durs et les lecteurs de CD/DVD sont connectés à un périphérique appelé contrôleur de disque dur, qui dirige les opérations du disque dur et les transferts de données. VirtualBox peut émuler les trois types les plus courants de contrôleurs de disque dur qu'on trouve généralement sur les PCs d'aujourd'hui : IDE, SATA (AHCI) et SCSI.[1]

- Les contrôleurs **IDE (ATA)** sont utilisés depuis les années 80. Au départ, ce type d'interface ne fonctionnait qu'avec les disques durs, mais il a été étendu ensuite aussi pour supporter les lecteurs de CD-ROM et d'autres types de médias amovibles. Dans un PC physique, ce standard utilise des nappes de 40 ou 80 broches. Chacune de ces nappes connecte deux périphériques à un contrôleur, ce que l'on a appelé traditionnellement « master » (maître) et « slave » (esclave). Les contrôleurs de disque dur ont en général deux connecteurs pour de telles nappes ; il s'en suit que la plupart des PCs supportent jusqu'à quatre périphériques.

  Dans VirtualBox, chaque machine virtuelle a un contrôleur IDE activé par défaut, ce qui vous permet d'attacher jusqu'à quatre périphériques de stockage virtuels à la machine. (Par défaut, un des quatre - le second maître - est pré-configuré pour être le lecteur virtuel de CD/DVD de la machine, mais vous pouvez modifier cela.) [2])

  Donc même si votre système d'exploitation invité ne supporte pas les périphériques SCSI ou SATA, il devrait toujours pouvoir voir le contrôleur IDE par défaut activé.

[1] Le support SATA a été ajouté avec VirtualBox 1.6 ; le support expérimental de SCSI a été ajouté avec la version 2.1 puis complètement ajouté avec la 2.2. De façon générale, l'attachements de supports de stockage a été rendu beaucoup plus flexible avec VirtualBox 3.1 ; voir ci-dessous.

[2] L'affectation du lecteur CD/DVD de la machine au maître secondaire a été corrigée avant VirtualBox 3.1 ; on peut maintenant la modifier ; et le lecteur peut être branché sur d'autres slots du contrôleur IDE, et il peut y avoir plus d'un lecteurs comme ça.

Vous pouvez aussi sélectionner précisément le type de matériel de contrôleur IDE que VirtualBox devrait présenter à la machine virtu elle (PIIX3, PIIX4 ou ICH6). Cela ne constitue aucune différence en termes de performances mais si vous importez une machine virtuelle à partir d'un autre produit de virtualisation, il se peut que le système d'exploitation de cette machine s'attende à un contrôleur particulier et plante s'il ne le trouve pas.

Après que vous ayez créé une nouvelle machine virtuelle avec l'assistant « Nouvelle machine » de l'interface graphique, vous verrez en général un contrôleur IDE dans les paramètres de « Stockage » de la machine, où le lecteur de CD/DVD virtuel sera attaché à l'un des quatre ports de ce contrôleur.

- **Serial ATA (SATA)** est un standard plus récent apparu en 2003. Par rapport à l'IDE, il supporte à la fois des vitesses beaucoup plus élevées et davantage de périphériques par contrôleur de disque dur. En outre, avec du matériel physique, vous pouvez ajouter des périphériques et les supprimer alors que le système est en fonction. L'interface standard pour les contrôleurs SATA s'appelle Advanced Host Controller Interface (**AHCI**).

  Pour des questions de compatibilité, les contrôleurs AHCI par défaut voient les disques qui y sont attachés dans un « mode IDE de compatibilité », sauf si le support SATA est explicitement demandé. Le « mode IDE de compatibilité » signifie seulement que les lecteurs peuvent être vus et utilisés par le BIOS de l'ordinateur. Mais, les disques attachés à ces emplacement travailleront en mode AHCI pleine vitesse une fois que le système invité aura chargé son pilote de périphérique AHCI.

  Comme un vrai contrôleur SATA, le contrôleur virtuel SATA de VirtualBox agit plus vite et consomme aussi moins de ressources processeur que le contrôleur IDE virtuel. En outre, ceci vous permet de connecter jusqu'à 30 disques durs virtuels à une machine contre à peine trois pour l'IDE (avec le lecteur DVD déjà attaché). Parmi eux, les quatre premiers (numérotés de 0 à 3 dans l'interface graphique sont gérés par défaut en mode IDE de compatibilité.

  C'est pour cette raison qu'à partir de la version 3.2 et en fonction du système d'exploitation invité sélectionné, VirtualBox utilise SATA par défaut pour les machines virtuelles nouvellement créées. Un contrôleur SATA est créé par défaut et le disque par défaut qui a été créé avec une nouvelle machine virtuelle est attaché à ce contrôleur.

  Avertissement: Le contrôleur SATA et les disques virtuels qui y sont attachés (y compris ceux en mode compatibilité IDE) ne seront pas vus par un système d'exploitation qui n'a pas de support périphérique pour AHCI. En particulier, **il n'y a pas de support pour AHCI sur les Windows antérieurs à Windows Vista**, donc Windows XP (même SP2) ne verra pas de tels disques sauf si vous installez les pilotes supplémentaires. Il est possible de basculer entre IDE et SATA après l'installation en installant les pilotes SATA et en modifiant le type de contrôleur dans la boîte de dialogue des paramètres de la VM. [a]

  [a] VirtualBox recommande les pilotes Intel Matrix qu'on peut télécharger sur http://downloadcenter.intel.com/Product_Filter.aspx?ProductID=2101

Pour ajouter un contrôleur SATA à une machine pour laquelle il n'a pas été activé par défaut (soit parce qu'elle a été créée par une version antérieure de VirtualBox soit parce que le SATA n'est pas supporté par le système d'exploitation par défaut sélectionné), allez sur l'onglet «stockage» de la boîte de dialogue des paramètres de la machine, cliquez sur le bouton «Ajouter un contrôleur» sous la case «Arborescence de stockage» puis sélectionnez «Ajouter un contrôleur SATA». Après quoi, le contrôleur supplémentaire apparaîtra comme périphérique PCI séparé dans la machine virtuelle et vous pouvez y ajouter des disques virtuels.

Pour modifier les paramètres du mode de compatibilité IDE du contrôleur SATA, merci de voir le chapitre 8.18, *VBoxManage storagectl*, page 134.

- **SCSI** est un autre standard

  industriel, signifiant « Small Computer System Interface ». Il a été établi dès 1986 comme une interface générique pour le transfert de données entre tous types de périphérique, y compris les périphériques de stockage. Aujourd'hui, le SCSI est toujours utilisé pour connecter des disques durs et des périphériques de bande magnétique, mais la plupart du temps, il a été relégué en matériel de secours. Il est encore couramment utilisé sur des stations de travail et des serveurs haute performance.

  Pour des raisons principalement liées à la compatibilité avec d'autres logiciels de virtualisation, VirtualBox supporte éventuellement les contrôleurs LsiLogic et BusLogic, sur lesquels vous pouvez attacher jusqu'à 16 disques durs virtuels.

  Pour activer un contrôleur SCSI, sur l'onglet « Stockage » d'une boîte de dialogue de paramètres d'une machine virtuelle, cliquez sur le bouton « Ajouter un contrôleur » sous la case à cocher « arborescence de stockage », puis sélectionnez « Ajouter un contrôleur SCSI ». Après quoi le contrôleur supplémentaire apparaîtra comme un périphérique PCI distinct dans la machine virtuelle.

  Avertissement: Comme avec les autres types de contrôleur, un contrôleur SCSI ne sera vu par les systèmes d'exploitation qu'avec le support pour un tel périphérique. Windows 2003 et supérieur incluent les pilotes pour le contrôleur LSI Logic, tandis que Windows NT 4.0 et Windows 2000 incluent les pilotes pour le contrôleur BusLogic. Windows XP n'inclut les pilotes pour aucun contrôleur.

- **SCSI attaché en série (Serial Attached SCSI, SAS)** est un autre standard de bus qui utilise le jeux de commandes de SCSI. Mais contrairement à SCSI, avec des périphériques physiques, ce sont des câbles série qui sont utilisés au lieu de câbles parallèle, ce qui simplifie les connexions de périphériques physiques. En quelque sorte, le SAS est au SCSI ce que le SCSI est à l'IDE: il permet davantage de connexions, plus fiables et plus rapides.

  Pour supporter les invités de haut niveau qui exigent des contrôleurs SAS, VirtualBox émule un contrôleur SAS LsiLogic que vous pouvez activer de la même façon qu'un contrôleur SCSI. Pour l'instant, vous pouvez connecter jusqu'à huit périphériques au contrôleur SAS.[3]

  Avertissement: Comme avec SATA, le contrôleur SAS ne sera vu que par les systèmes d'exploitation qui le supportent. En particulier, **il n'y a pas de support pour SAS dans Windows avant Windows Vista**, donc Windows XP (même SP2) ne verra pas de tels disques sauf si vous installez des pilotes supplémentaires.

En résumé, VirtualBox vous donne les catégories suivantes d'emplacement de stockage virtuels :

1. Quatre emplacements attachés au contrôleur IDE traditionnel, qui sont toujours toujours présents (en général l'un d'eux est en général un lecteur de CD/DVD virtuel);
2. 30 emplacements attachés au contrôleur SATA, s'il est activé et si votre système d'exploitation peut le voir ; ces emplacements peuvent être soit
   a) En mode compatibilité IDE (par défaut les emplacements 0 à 3) ou

[3] Le support du contrôleur LSI Logic SAS a été ajouté avec VirtualBox 3.2.

b) En mode SATA ;

3. 15 emplacements attachés au contrôleur SCSI, s'il est activé et supporté par le système d'exploitation invité;

4. Huit emplacements attachés au contrôleur SAS, s'il est activé et supporté par le système d'exploitation invité.

Étant donné le vaste choix de contrôleurs de stockage, il se peut que vous vous demandiez lequel choisir. En général, vous devriez éviter l'IDE, sauf si c'est le seul contrôleur supporté par votre invité. Que vous utilisez SATA, SCSI, ou SAS, il n'y a pas de différence réelle.

5.2 Fichiers images de disque (VDI, VMDK, VHD, HDD)

Les fichiers images de disque résident sur le système hôte et sont vus par les systèmes invités comme des disques durs d'une certaine géométrie. Lorsqu'un système d'exploitation lit depuis ou écrit sur un disque dur, VirtualBox redirige la demande sur le fichier image.

Remarquez que quand vous créez un fichier image, vous devez spécifier sa taille, qui représente une géométrie fixe du disque virtuelle. Il n'est donc pas possible de modifier la taille du disque dur virtuel ultérieurement.

VirtualBox supporte quatre variantes de fichiers images de disque :

- Normalement, VirtualBox utilise son propre format de contenu pour les disques durs invités - fichiers Virtual Disk Image (VDI) -. En particulier, ce format sera utilisée quand vous créerez une nouvelle machine virtuelle avec un nouveau disque.

- VirtualBox supporte aussi complètement le format de contenu populaire et libre VMDK utilisé par beaucoup d'autres produits de virtualisation, en particulier par Vmware.[4]

- VirtualBox supporte aussi pleinement le format VHD utilisé par Microsoft.

- Les fichiers images de Parallels version 2 (format HDD) sont aussi supportés.[5] Faute de documentation sur le format, les formats récents (3 et 4) ne sont pas supportés. Vous pouvez cependant convertir de tels fichiers images vers le format de la version 2 en utilisant les outils fournis par Parallels.

Indépendamment du format de disque, comme il a été brièvement mentionné au chapitre 1.6, *Créer votre première machine virtuelle*, page 17, il y a deux options pour créer une image de disque : taille statique ou extension dynamique.

- Si vous créez une **image à la taille fixe** de, par exemple, 10 Gio, un fichier image d'à peu près la même taille sera créé sur votre système hôte. Remarquez que la création d'une image à taille statique peut prendre beaucoup de temps selon la taille de l'image et les performances d'écriture de votre disque dur.

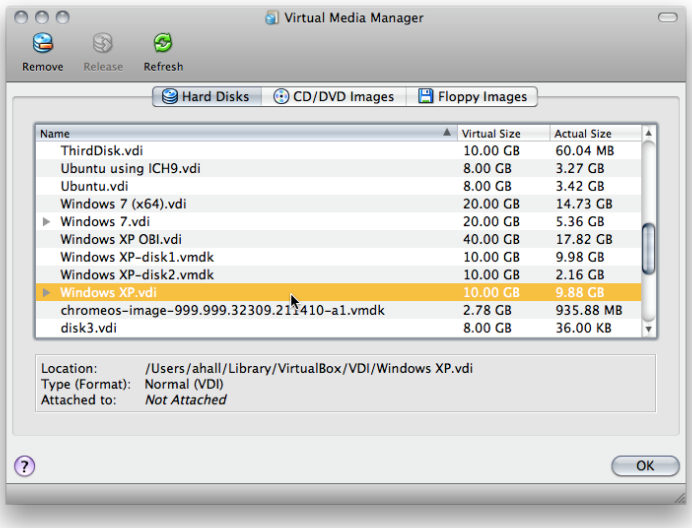
- Pour une gestion du stockage plus flexible, utilisez une **image à extension dynamique**. Celle ci sera au départ très petite et n'occupera pas d'espace pour des secteurs du disque virtuel non utilisés, mais le fichier image grossira chaque fois qu'on écrira sur un secteur du disque pour la première fois. Si ce format prend moins d'espace disque au début, le fait que VirtualBox doive étendre en permanence le fichier image consomme des ressources machines supplémentaires, donc jusqu'à ce que le disque ne soit entièrement rempli, les opérations d'écriture sont plus lentes qu'avec des disques à taille statique. Cependant, après qu'un disque dynamique a été étendu entièrement, la différence de performances pour les opérations de lecture et d'écriture est négligeable.

[4] Le support initial pour VMDK a été ajouté avec VirtualBox 1.4 ; depuis la version 2.1, VirtualBox supporte complètement VMDK, ce qui veut dire que vous pouvez créer des instantanés et utiliser toutes les autres fonctionnalités avancées décrites ci-dessus pour les images VDI également avec VMDK.

[5] Le support a été ajouté à VirtualBox 3.1.

5.3 Le gestionnaire de médias virtuels

VirtualBox conserve un registre interne de toutes les images disponibles de disque, CD/DVD-ROM et de disquette. Vous pouvez voir et modifier ce registre dans le gestionnaire de médias virtuels, auquel vous pouvez accéder depuis le menu « Fichier » dans la fenêtre principale de VirtualBox :



La fenêtre vous montre toutes les images actuellement enregistrées avec VirtualBox, regroupées de façon pratique en trois onglets pour les trois formats possibles. Ces formats sont :

- Les images de disque dur, soit au format Virtual Disk Image (VDI) de VirtualBox lui-même, soit dans les formats tiers listés ci-dessus ;
- Les images de CD/DVD au format standard ISO ;
- Des images de disquette au format standard RAW.

Comme vous pouvez le voir dans la capture d'écran ci-dessus, pour chaque image, le gestionnaire de médias virtuels vous montre le chemin complet vers le fichier image et d'autres informations telles que la machine virtuelle à laquelle l'image est actuellement attachée, s'il y en a une.

Le gestionnaire de médias virtuels vous permet de

- créer de nouvelles images de disque dur en utilisant le bouton **"Nouveau"** ; cela ouvrira l'assistant « Créer une image de disque » déjà décrit au chapitre 1.6, *Créer votre première machine virtuelle*, page 17;
- importer des fichiers images existantes depuis votre disque dur sur VirtualBox en utilisant le bouton **"Ajouter"** ;
- **supprimer** une image du registre(et éventuellement effacer le fichier image en même temps) ;
- **"libérer"** une image, c'est-à-dire, la détacher d'une machine virtuelle si elle y est actuellement attachée comme un disque dur.

Nous vous recommandons de maintenir deux dossiers spéciaux sur votre système pour conserver les images : un pour les fichiers images de disques durs (qui peut, en cas d'images à extension dynamique, atteindre des tailles considérables), et un pour les fichiers ISO (qui ont probablement été téléchargées sur Internet).

Vous pouvez copier des fichiers images de disque dur sur d'autres systèmes hôtes et les importer depuis des machines virtuelles, bien que certains systèmes invités (surtout Windows 2000 et XP) exigeront que la nouvelle machine virtuelle soit paramétrée de la même manière que l'ancienne. .

Note: Ne faites pas simplement des copies d'images de disques virtuels. Si vous importer ainsi une seconde copie dans une machine virtuelle, VirtualBox se plaindra avec une erreur, puisque VirtualBox attribue un identifiant unique (UUID) à chaque image de disque pour être sûr qu'il n'est utilisé qu'une seule fois. Voir le chapitre 5.6, *Cloner des images de disque*, page 87 pour des instructions à ce sujet. De même, si vous voulez copier une machine virtuelle sur un autre système, VirtualBox a une fonction d'importation/exportation qui pourrait mieux convenir à vos besoins ; voir le chapitre 1.12, *Importer et exporter des machines virtuelles*, page 29.

5.4 Modes spéciaux d'écriture d'images

Pour chaque image de disque virtuel supportée par VirtualBox, vous pouvez utiliser des commandes spéciales pour définir comment des opérations d'écriture depuis la machine virtuelle devraient modifier l'image et comment les instantanés devraient la modifier. Cela vaut pour tous les formats d'images précités (VDI, VMDK, VHD ou HDD) et indépendamment du fait que l'image soit de taille statique ou étendue de façon dynamique.

1. Avec des **images normales** (le réglage par défaut), il n'y a pas de restrictions sur la façon dont les invités peuvent lire et écrire sur le disque.

   Quand vous faites un instantané de votre machine virtuelle comme décrit au chapitre 1.8, *Instantanés*, page 24, l'état d'une telle image de « disque dur normal » sera enregistrée avec le instantané, et quand vous restaurerez le instantané, son état sera entièrement réinitialisé.

   (pour être précis sur un plan technique, le fichier image n'est pas « réinitialisé » en tant que tel. Par contre, quand on fait un instantané, VirtualBox « gèle » le fichier image et n'écrit plus dessus. Pour les opérations d'écriture depuis la VM, un second fichier image « de différenciation » est créé qui ne reçoit que les modifications de l'image d'origine ; voir la prochaine section pour des détails).

   Si vous pouvez attacher la même image « normale » à plusieurs machines virtuelles, une seule de ces machine virtuelle attachée au même fichier image peut être lancée simultanément, sans quoi il y aurait un conflit si plusieurs machines écrivaient sur le même fichier image.[6]

2. À l'inverse, les disques durs « write-through » ne sont pas du tout concernés par les instantanés : leur état *n'est pas* sauvegardé quand on crée un instantané, et il n'est pas restauré quand on le restaure.

   Pour *créer* une image de disque VDI en « write-through », utilisez la commande `VBoxManage createhd` ; voir le chapitre 8.21, *VBoxManage createhd*, page 135. Pour marquer une image *existante* comme dynamique, utilisez la commande `VBoxManage modifyhd` ; voir chapitre 8.22, *VBoxManage modifyhd*, page 136.

[6] Cette restriction est plus allégée maintenant qu'avec VirtualBox 2.2. Auparavant, toute image de disque « normal » ne pouvait être *attachée* qu'à une seule machine. Maintenant, on peut à plus d'une machine, tant que seule une d'entre elles est en fonction.

3. **Les disques durs partageables** sont une variante des disques durs « write-through ». En principe, ils se comportent exactement de la même façon, c'est-à-dire que leur état *n'est pas* sauvegardé quand on prend un instantané, et il n'est pas restauré lors de la restauration d'un instantané. La différence n'apparaît que si on attache de tels disques à plusieurs VMs. Les VMs partageables peuvent être attachés à plusieurs VMs, qui peuvent se lancer concommitamment. Elles sont ainsi convenables pour être utilisées par des systèmes de fichiers cluster entre des VMs et des applications identiques qui sont préparées explicitement pour accéder concommitamment à un disque. Seules les images à taille fixe peuvent être utilisées de cette manière, les images à taille dynamique sont rejetées.

   C'est une fonctionnalité avancée, une mauvaise utilisation peut conduire à une perte de données – les systèmes de fichiers réguliers ne sont pas préparés pour gérer des modifications simultanes par plusieurs éléments.

   Pour *créer* une image de disque au format VDI en tant que « partageable », utilisez la commande `VBoxManage createhd` ; voir le chapitre 8.21, *VBoxManage createhd*, page 135. Pour marquer une image *existante* comme partageable, utilisez `VBoxManage modifyhd` ; voir chapitre 8.22, *VBoxManage modifyhd*, page 136.

4. Enfin, les images immuables ne se souviennent des accès en écriture que de façon temporaire pendant que la machine virtuelle est en fonction ; toutes les modifications sont perdues quand la machine virtuelle est rallumée la fois suivante. Il en résulte que, contrairement aux images « normal », la même image immuable peut être utilisée avec plusieurs machines virtuelles sans restrictions.

   Créer une image immuable n'a pas beaucoup de sens puisqu'elle serait vide au départ et perdrait son contenu à chaque redémarrage de la machine (sauf si vous voulez vraiment avoir un disque non formaté quand la machine démarre). Il en résulte que normalement, vous créeriez d'abord une image « normal », puis lorsque vous considérez que son contenu est utile, vous la marqueriez plus tard comme immuable en utilisant `VBoxManage modifyhd` ; merci de voir de nouveau le chapitre 8.22, *VBoxManage modifyhd*, page 136. Sinon, ouvrez une image existante en mode « immuable » en utilisant `VBoxManage openmedium`.

   Si vous faites un instantané d'une machine avec des images immuables, à chaque arrêt de la machine, ces images seront réinitialisées à l'état du dernier (l'actuel) instantané (et pas à l'état de l'image immuable d'origine).

   Note: En guise d'exception spéciale, les images immuables *ne sont pas* réinitialisées si elles sont attachées à une machine dont on a fait le dernier instantané alors que la machine était en fonction (ce que l'on appelle un instantané « en ligne »). Il en résulte que si le instantané actuel de la machine est comme un instantané « en ligne », ses images immuables se comportent exactement comme les images normales décrites précédemment. Pour réactiver la réinitialisation automatique de telles images, effacez le instantané actuel de la machine.

   À nouveau, techniquement, VirtualBox n'écrit jamais directement sur une image immuable. Toutes les opérations d'écriture depuis la machine seront redirigées vers une image de différenciation ; la prochaine fois que la VM sera allumée, l'image de différenciation est réinitialisée de sorte que chaque fois que la VM démarre, ses images immuables ont exactement le même contenu.[7] L'image de différenciation n'est réinitialisée que lorsque la machine est allumée depuis l'intérieur de VirtualBox, pas quand vous redémarrez en demandant un

[7] Ce comportement a aussi changé avec VirtualBox 2.2. Auparavant, les images de différenciation étaient désactivées quand la session de la machine *se terminait* ; maintenant, elles sont désactivées à chaque fois que la machine est allumée.

redémarrage depuis la machine. C'est aussi pourquoi les images immuables se comportent comme décrit ci-dessus quand les instantanés sont également présents, ce qui utilise des images de différenciation.

Si la non prise en compte automatique des images de différenciation au démarrage de la VM ne correspond pas à vos besoins, vous pouvez la désactiver en utilisant le paramètre `autoreset` de `VBoxManage modifyhd` ; voir le chapitre 8.22, *VBoxManage modifyhd*, page 136 pour des détails.

Pour illustrer les variations entre les divers types du point de vue des instantanés : Supposons que vous ayez installé votre système d'exploitation invité dans votre VM et que vous ayez fait un instantané. Imaginons que vous ayez infecté par accident votre VM avec un virus et que vous aimeriez revenir au instantané. Avec une image de disque dur normale, vous restaurez simplement le instantané et l'état précédent de l'image de votre disque dur sera restaurée (et votre infection par un virus sera annulée). Avec un disque dur immuable, tout ce qu'il y a à faire est d'éteindre et d'allumer votre VM, et l'infection par le virus sera désactivée. Néanmoins, avec une image de disque write-through, vous ne pouvez pas facilement annuler l'infection par le virus via la virtualisation, mais vous devrez désinfecter votre machine virtuelle comme un vrai ordinateur.

Vous pourriez encore toujours trouver les images write-though utiles si vous voulez préserver des données critiques indépendamment des instantanés, et comme vous pouvez attacher plus d'une image à une VM, il se peut que vous vouliez avoir une immuable pour le système d'exploitation et une write-through pour vos fichiers de données.

5.5 Images de différenciation

La section précédente traitait des images de différenciation et de la façon dont elles sont utilisées avec les instantanés, les images immuables et les attachements de plusieurs disques. Pour l'utilisateur de VirtualBox curieux, cette section décrit avec davantage de détails comment elles fonctionnent.

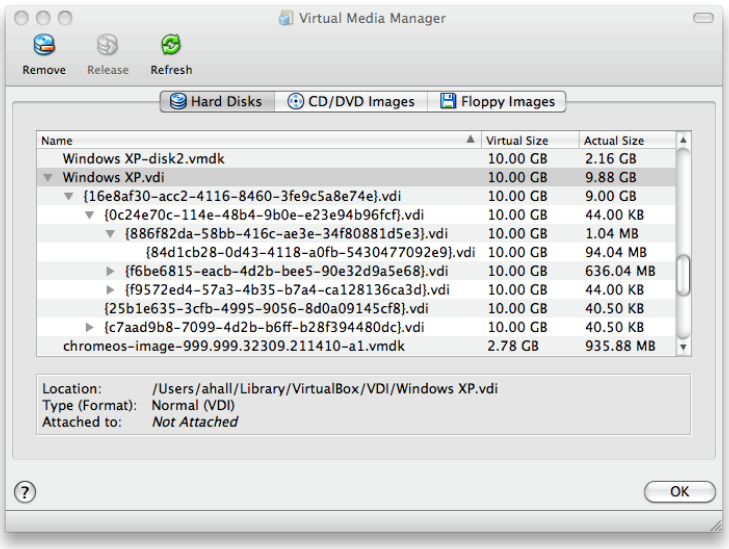
Une image de différenciation est une image de disque spéciale qui ne conserve que les différences avec une autre image. Elle est en soi inutile, elle doit toujours se référer à une autre image. L'image de différenciation est alors vue en général comme une « fille » qui conserve les différences par rapport à son « parent ».

Quand une image de différenciation est active, elle reçoit toutes les opérations d'écriture depuis la machine virtuelle à la place de son parent. L'image de différenciation ne contient que les secteurs du disque virtuel qui a changé depuis que l'image de différenciation a été créée. Quand la machine lit un secteur depuis un tel disque dur virtuel, il regarde d'abord dans l'image de différenciation. Si le secteur est présent, il est retourné depuis celui-ci ; sinon VirtualBox regarde dans le parent. En d'autres termes, le parent devient en « lecture seule » ; on n'y écrit plus, mais on lit à partir de celui ci si un secteur n'a pas changé.

Les images de différenciation peuvent être chaînées. Si vous créez une autre image de différenciation pour un disque virtuel qui a déjà une image de différenciation, elle devient un « petit-fils » du parent d'origine. La première image de différenciation devient alors en lecture seule, et les opérations d'écriture ne vont que sur l'image de différenciation de second niveau. Lors de la lecture à partir du disque virtuel, VirtualBox doit d'abord regarder d'abord dans la deuxième image de différenciation, puis dans la première si le secteur n'a pas été trouvé puis dans l'image d'origine.

Il peut y avoir un nombre illimité d'images de différenciation et chaque image peut avoir plus d'un enfant. Il en résulte que les images de différenciation peuvent constituer une arborescence complexe avec des parents, des « frères » et des enfants selon la complexité de la configuration de votre machine. Les opérations d'écriture vont toujours sur l'image de différenciation « active » attachée à la machine, et pour des opérations de lecture, il se peut que VirtualBox ait besoin

de de regarder dans presque tous les parents de la chaîne jusqu'à ce qu'il trouve le secteur en question. Vous pouvez regarder une telle arborescence dans le gestionnaire de médias virtuels :



Dans toutes ces situations, du point de vue de la machine virtuelle, le disque dur virtuel se comporte comme n'importe quel autre disque. Pendant que la machine virtuelle est en fonction, il y a une légere surcharge d'E/S en cours d'exécution car il se peut que VirtualBox doive regarder des secteurs plusieurs fois. Cela n'est cependant pas observable puisque les tables avec des informations de secteurs sont toujours conservées en mémoire et peuvent être inspectées rapidement.

Les images de différenciation sont utilisées dans les situations suivantes :

1. **Les instantanés.** Quand vous créez un instantané, comme expliqué dans la section précédente, VirtualBox « gèle » les images attachées à la machine virtuelle et crée des images de différenciation pour chacune d'elles (pour être précis, une pour chaque image non en mode « write-through »). Du point de vue de la machine virtuelle, les disques virtuels continuent d'agir comme avant mais toutes les opérations d'écriture vont sur les images de différenciation. Chaque fois que vous créez un autre instantané, pour chaque attachement de disque dur, une autre image de différenciation est créée et attachée, constituant une chaîne ou une arborescence.

   Dans la capture d'écran ci-dessus, vous voyez que l'image de disque d'origine est maintenant attachée à un instantané, représentant l'état du disque quand le instantané a été fait.

   Si vous **restaurez** maintenant un instantané - c'est-à-dire si vous voulez revenir à l'état exact de la machine qui a été stocké dans le instantané -, ce qui suit se produit :

   a) VirtualBox copie les paramètres de la machine virtuelle qui ont été copiés dans le instantané vers la machine virtuelle. Il en résulte que si vous avez fait des changements sur la configuration de la machine depuis que vous avez fait le instantané, elles sont annulées.

   b) Si l'instantané a été pris alors que la machine était en fonction, son contenu a un état de machine sauvegardé et cet état est restauré ; après la restauration du instantané, la machine sera alors en état « sauvegardée » et reprendra l'exécution là où se trouve le démarrage suivant. Sinon la machine sera dans l'état « Coupée » et fera un démarrage complet.

c) Pour chaque image de disque attachée à la machine, l'image de différenciation qui conserve toutes les opérations d'écriture depuis que le instantané actuel a été pris est projetée et l'image parente d'origine est à nouveau activée. (Si vous avez restauré le instantané « racine », elle sera l'image de disque racine de chaque élément attaché ; sinon une autre image de différenciation proviennent d'elle.) Ceci restaure en fait l'ancien état de la machine.

Si vous **effacez** ultérieurement un instantané pour libérer de l'espace disque, pour chaque attachement de disque, une des images de différenciation devient obsolète. Dans ce cas, l'image de différenciation de l'attachement du disque ne peut pas être simplement effacée. VirtualBox doit au contraire regarder chaque secteur de l'image de différenciation et doit le copier vers son parent ; cela s'appelle du « merging » d'image et peut être un processus potentiellement long selon la taille de l'image de différenciation. Il peut aussi nécessiter temporairement une quantité de d'espace disque supplémentaire substantielle, avant que l'image de différenciation devenue obsolète avec l'opération de merging ne seit effacée.

2. **Images immuables.** Quand on bascule une image en mode « immuable », une image de différenciation est créée. Comme avec les instantanés, l'image parent devient alors en lecture seule et l'image de différenciation reçoit toutes les opérations d'écriture. Chaque fois qu'on démarre la machine virtuelle, toutes les images immuables qui y sont attachées ont leur propre image de différenciation qui apparaît, réinitialisant effectivement le disque virtuel de la machine virtuelle à chaque redémarrage.

5.6 Cloner des images de disque

Vous pouvez dupliquer des fichiers images de disque dur sur le même hôte pour rapidement créer une seconde machine virtuelle avec le même paramétrage de système d'exploitation. Cependant, *vous ne devriez faire de copies d'images de disques durs virtuels qu'en utilisant* l'outil fourni avec VirtualBox ; voir le chapitre 8.23, *VBoxManage clonehd*, page 137. Car VirtualBox donne un un numéro d'identité unique (UUID) à chaque image de disque, qui est également stocké dans l'image, et VirtualBox refusera de fonctionner avec deux images qui utilisent le même numéro Si vous essayez accidentellement de réimporter une image de disque que vous avez copiée normalement, vous pouvez faire une deuxième copie en utilisant l'outil de VirtualBox et l'importer à la place.

Remarquez que les distributions Linux récentes identifient le disque d'amorçage à partir de l'ID du disque. Les signalements de l'ID VirtualBox sont déterminés à partir de l'UUID de l'image du disque virtuel. Donc si vous clonez une image de disque et si vous essayez de démarrer sur l'image copiée, il se pourrait que l'invité ne puisse pas déterminer son propre disque d'amorçage vu que l'UUID a changé. Dans ce cas, vous devez adapter l'ID du disque dans votre script de chargeur de démarrage (par exemple `/boot/grub/menu.lst`). L'ID du disque ressemble à ceci :

```
scsi-SATA_VBOX_HARDDISK_VB5cfdb1e2-c251e503
```

Vous pouvez déterminer l'ID de l'image copiée avec

```
hdparm -i /dev/sda
```

5.7 Images de disque et mise en cache E/S

VirtualBox peut éventuellement désactiver la mise en cache E/S qu'effectuerait sinon le système d'exploitation hôte sur les fichiers images de disque.

Traditionnellement, VirtualBox ouvre les fichiers images de disque comme des normaux, ce qui fait qu'ils sont mis en cache par le système d'exploitation hôte, comme n'importe quel autre

fichier. Le principal avantage de ceci est la vitesse : quand l'OS invité écrit sur le disque et lorsque le cache de l'OS hôte utilise l'écriture différée, l'opération d'écriture peut être reportée tout en étant effectuée rapidement sur l'OS invité, tandis que l'hôte peut effectuer l'opération de façon non synchronisée. En outre, quand vous démarrez une VM une deuxième fois et si vous avez assez de mémoire disponible pour que l'OS l'utilise pour la mise en cache, il se peut que de grandes parties du disque virtuel restent dans la mémoire du système, la VM peut accéder aux données beaucoup plus rapidement.

Remarquez que cela ne s'applique que sur des fichiers images ; la mise en tampon n'affecte jamais les disques virtuels résidant sur des supports de stockage iSCSI distants, ce qui est le scénario le plus courant sur des configurations d'entreprise. (voir chapitre 5.11, *Serveurs iSCSI*, page 90).

Si la mise en tampon est un paramètre par défaut utile pour virtualiser quelques machines sur un ordinateur de bureau, cette approche comporte certains inconvénients :

1. L'écriture différée à travers le cache de l'OS hôte est moins sécurisée. Quand l'OS invité écrit des données, il considère qu'elles sont écrites même si elles ne sont pas encore arrivé sur un disque physique. Si pour une raison quelconque l'écriture ne se produit pas (panne de courant, plantage de l'hôte), les chances de perdre des données sont accrues.

2. Les fichiers images de disque ont tendance à être très gros. Les mettre en cache peut donc rapidement utiliser jusqu'à l'ensemble du cache de l'OS hôte. Selon l'efficacité de la mise en cache de l'OS hôte, cela peut ralentir énormément l'hôte, surtout si plusieurs VMs fonctionnent en même temps. Par exemple, sur des hôtes Linux, il peut résulter de la mise en cache de l'hôte que Linux diffère toutes les écritures jusqu'à ce que le cache de l'hôte soit presque plein, alors l'écriture de tous ces changements en une seule fois peut éventuellement figer l'exécution de la VM pendant quelques minutes. Il peut s'en suivre des erreurs E/S dans l'invité du fait du timeout de requête des E/S sur ceux-ci.

3. La mémoire physique est souvent gaspillée vu que les systèmes d'exploitation invités ont leurs propres caches E/S, ce qui peut aboutir à ce que les données soient mises en cache deux fois (à la fois dans les caches de l'invité et de l'hôte) pour un résultat limité.

Il s'en suit que, à partir de la version 3.2, VirtualBox vous permet éventuellement de désactiver la mise en cache E/S de l'hôte des fichiers images de disque. Dans ce cas, VirtualBox utilise son propre petit cache pour mettre les écritures en tampon, mais il n'y a pas de mise en cache de lecture puisque ceci est déjà fait par l'OS invité. En outre, VirtualBox supporte complètement l'E/S asynchrone pour ses contrôleurs virtuels SATA, SCSI et SAS à travers divers fils d'E/S.

Comme l'E/S asynchrone n'est pas supportée par les contrôleurs IDE, pour des raisons de performance, il se peut que vous souhaitiez laisser la mise en cache activée pour les contrôleurs IDE virtuels de votre VM.

C'est pourquoi VirtualBox vous permet de configurer si le cache E/S de l'hôte est utilisé pour chaque contrôleur E/S séparément. Soit décochez la case « Utiliser le cache E/S de l'hôte » des paramètres « Stockage » pour un contrôleur de stockage virtuel donné, soit utilisez la commande VBoxManage suivante pour désactiver le cache E/S de l'hôte pour un contrôleur de stockage virtuel :

```
VBoxManage storagectl <vm> --name <nomcontrôleur> --hostiocache off
```

Voir chapitre 8.18, *VBoxManage storagectl*, page 134 pour des détails.

De même, pour les raisons évoquées ci-dessus, VirtualBox utilise maintenant par défaut des contrôleurs SATA pour les nouvelles machines virtuelles.

Note: La désactivation des caches E/S de l'hôte donnera de faibles performances actuellement avec les fichiers VHD et sparse VMDK. Voir le chapitre 14, *Known limitations*, page 201 pour les détails.

5.8 Limiting bandwidth for disk images

Starting with version 4.0, VirtualBox allows for limiting the maximum bandwidth used for asynchronous I/O. Additionally it supports sharing limits through bandwidth groups for several images. It is possible to have more than one such limit.

Limits are configured through `VBoxManage`. The example below creates a bandwidth group named “Limit”, sets the limit to 20 MB/s and assigns the group to the attached disks of the VM:

```
VBoxManage bandwidthctl "VM name" --name Limit --add disk --limit 20
VBoxManage storageattach "VM name" --controller "SATA" --port 0 --device 0 --type hdd
                                   --medium disk1.vdi --bandwidthgroup Limit
VBoxManage storageattach "VM name" --controller "SATA" --port 1 --device 0 --type hdd
                                   --medium disk2.vdi --bandwidthgroup Limit
```

All disks in a group share the bandwidth limit, meaning that in the example above the bandwidth of both images combined can never exceed 20 MB/s. However if one disk doesn't require bandwidth the other can use the remaining bandwidth of its group.

The limits for each group can be changed while the VM is running, with changes being picked up immediately. The example below changes the limit for the group created in the example above to 10 MB/s:

```
VBoxManage bandwidthctl "VM name" --name Limit --limit 10
```

5.9 Opération sur le lecteur de CD/DVD

Le(s) lecteur(s) de CD/DVD par défaut ne supporte(nt) que la lecture. La configuration du média peut être modifiée au moment de l'exécution. Vous pouvez choisir entre trois options pour fournir les données au média :

- **Lecteur hôte** définit que l'invité peut lire depuis le média dans le lecteur hôte. Les changements de média des lecteurs hôtes sont signalés à l'invité.
- Le **fichier image** donne à l'invité l'accès en lecture seule aux données de l'image (souvent mentionnée comme image ISO). Un changement de média est signalé lors du basculement vers une image différente ou de la sélection d'une autre option.
- **EVide** est pour un lecteur sans média inséré. Le lecteur répond comme d'habitude à la situation, mais aucune donnée ne peut être lue.

Comme déjà mentionné, les signalements de changements de média dépendent des options sélectionnées pour le média. Les changements de média peuvent être empêchés par l'invité et VirtualBox le répercute en verrouillant le lecteur hôte si nécessaire. Vous pouvez forcer la suppression d'un média dans une telle situation par l'interface graphique de VirtualBox ou l'outil en ligne de commande VBoxManage. En fait, cela revient à une éjection d'urgence fournie par beaucoup de lecteurs CD/DVD avec tous les effets colatéraux associés. L'OS invité peut renvoyer des messages d'erreur dans ce cas, comme sur du vrai matériel. Utilisez ceci avec prudence.

Dans tous les cas, seuls des médias de données sont supportées pour les lecteurs de CD/DVD. Cela signifie que tous les formats de CD de données et tous les formats DVD peuvent en principe être utilisés. Comme les lecteurs DVD hôte refusent de lire des médias vidéos DVD chiffrés, vous ne pouvez pas jouer de telles vidéos avec l'émulation de CD/DVD régulière. Il se peut que vous réussissiez à le faire fonctionner avec le support expérimental passthrough décrit au chapitre 5.10, *Écrire des CDs et des DVDs en utilisant le lecteur hôte*, page 90.

Les formats de CD audio et de CD vidéo ne sont pas supportés, ce qui signifie que vous ne pouvez pas jouer de tels médias depuis une machine virtuelle.

5.10 Écrire des CDs et des DVDs en utilisant le lecteur hôte

Quand vous attachez le lecteur CD/DVD de votre hôte à une machine virtuelle (voir le chapitre 3.6, *Paramètres de stockage*, page 50), cela donne en principe à la machine un accès en lecture seule au lecteur hôte. Cela empêche l'invité d'écrire sur le lecteur hôte. En particulier, vous ne pouvez pas graver de CDs et de DVDs depuis l'invité de cette façon.

En fonctionnalité expérimentale (qui ne fonctionne actuellement que pour les médias de données, les formats de CD audio, vidéo ne sont pas supportés), il est possible de donner à l'invité l'accès aux fonctionnalités d'écriture de CD/DVD du lecteur hôte (si disponibles). Il y a une case à cocher « Passthrough » dans la boîte de dialogue de l'interface graphique pour configurer le média attaché à un contrôleur de stockage, ou vous pouvez utiliser la ligne de commande :

```
VBoxManage storageattach <uuid|nomvm>
                         --storagectl <nom>
                         --port <numéro>
                         --device <numéro>
                         [--type <dvddrive|hdd|fdd>
                          --medium <none|emptydrive|uuid|filename|host:<drive>>]
                         [--passthrough <on|off>]
                         [--forceunmount]
```

Voir aussi chapitre 8.17, *VBoxManage storageattach*, page 132.

Même si pass-through est activé, des commandes risquées telles que la mise à jour du firmware du lecteur seront bloquées. Sur certains lecteurs hôtes, la fonctionnalité pass-through permet de jouer des médias vidéos DVD chiffrés.

Sur les hôtes Solaris, le path-through exige de lancer VirtualBox avec les droits du vrai administrateur du fait de mesures de sécurité renforcées par l'hôte.

5.11 Serveurs iSCSI

iSCSI signifie « Internet SCSI » et c'est un standard qui permet d'utiliser le protocole SCSI à travers des connexions Internet ((TCP/IP). Surtout avec l'arrivée de l'Internet haut débit (Gigabit Ethernet), il devient envisageable d'attacher des serveurs de stockage iSCSI simplement en tant que disques durs distants à un réseau d'ordinateurs. Dans la terminologie iSCSI, le serveur qui fournit les ressources de stockage est appelé une « cible iSCSI », tandis que le client qui se connecte au serveur et accède à ses ressources est appelé « l'initiateur iSCSI ».

VirtualBox peut présenter de façon intégréte à une machine virtuelle du stockage distant iSCSI comme un disque dur virtuel. Le système d'exploitation invité ne verra aucune différence entre une image de disque virtuelle (fichier VDI) et une cible iSCSI. Pour faire cela, VirtualBox a un initiateur iSCSI intégré.

Le support iSCSI de VirtualBox a été développé en fonction du standard iSCSI et il devrait fonctionner avec toutes les cibles iSCSI standards conformes au modèle. Pour utiliser une cible iSCSI avec VirtualBox, vous devez d'abord l'enregistrer en tant que disque dur virtuel avec `VBoxManage` ; voir chapitre 8.17, *VBoxManage storageattach*, page 132. La cible apparaîtra dans la liste d'images de disques, comme décrit au chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82, et peut ainsi être attachée à l'un des trois emplacements de disque dur de la VM de façon ordinaire.

6 Réseau virtuel

Comme il a été brièvement indiqué au chapitre 3.8, *Paramètres réseau*, page 53, VirtualBox fournit huit cartes Ethernet PCI virtuelles pour chaque machine virtuelle. Pour chacune de ces cartes, vous pouvez sélectionner individuellement

1. 1. le matériel qui sera virtualisé ou
2. 2. le mode de virtualisation avec lequel la carte virtuelle agira par rapport à votre matériel réseau physique sur l'hôte.

Quatre des cartes réseau peuvent être configurées dans la section « Réseau » des paramètres de la boîte de dialogue dans l'interface graphique de VirtualBox. Vous pouvez configurer les huit cartes réseau en ligne de commande avec VBoxManage modifyvm ; voir chapitre 8.7, *VBoxManage modifyvm*, page 120.

Ce chapitre explique les divers paramètres réseau avec plus détails.

6.1 Matériel de réseau virtuel

Pour chaque carte, vous pouvez sélectionner individuellement le type de *matériel* qui sera présenté à la machine virtuelle. VirtualBox peut virtualiser les six types de matériels de carte suivants :

- AMD PCNet PCI II (Am79C970A);
- AMD PCNet FAST III (Am79C973, celle par défaut) ;
- Intel PRO/1000 MT Desktop (82540EM);
- Intel PRO/1000 T Server (82543GC);
- Intel PRO/1000 MT Server (82545EM);
- Paravirtualized network adapter (virtio-net).

La PCNet FAST III est celle par défaut parce qu'elle est supportée par presque tous les systèmes d'exploitation d'ordinateur et le gestionnaire de démarrage GNU GRUB. Sauf que les adaptateurs de la famille Intel PRO/1000 sont choisis pour certains types de système d'exploitation invités qui n'incluent plus les pilotes de la carte PCNet, tels que Windows Vista.

Celle de type Intel PRO/1000 MT Desktop fonctionne avec Windows Vista et les versions supérieures. La variante Server T de la carte Intel PRO/1000 est reconnue par les invités Windows XP sans l'installation de pilotes supplémentaires. La variante Server T facilite les importations OVF à partir d'autres plateformes.

L'« **adaptateur réseau para-virtualisé (virtio-net** » est spécial. Si vous le sélectionnez, VirtualBox *ne virtualise pas* du matériel réseau classique (qui est supporté par les systèmes d'exploitation habituels hors du PC). VirtualBox s'attend par contre à ce qu'une interface logicielle spéciale pour les environnements virtualisés soient fournies par l'invité, évitant ainsi la complexité d'émuler du matériel réseau et améliorant la performance réseau. À partir de la version 3.1, VirtualBox fournit le support des pilotes de réseau « virtio » industriels, qui font partie du projet libre KVM.

Les pilotes de réseau « virtio » sont disponibles pour les systèmes d'exploitation invités suivants :

- Les noyaux Linux version 2.6.25 ou supérieur peuvent être configurés pour fournir le support de virtio ; certaines distributions ont aussi porté virtio dans des noyaux plus anciens.
- Pour Windows 2000, XP et Vista, les pilotes virtio peuvent être téléchargés et installés à partir de la page Internet du projet KVM.[1]

VirtualBox a aussi un support pour les **cadres appelés jumbo**, par exemple les paquets réseaux avec plus de 1500 bytes de données fournies pour que vous utilisiez la virtualisation de la carte réseau Intel et le réseau bridgé. En d'autres termes, les cadres jumbo ne sont pas supportés avec les périphériques réseau AMD ; dans ces cas les paquets jumbo seront refusés sans message à la fois dans la direction de la transmission et de la réception. Les systèmes d'exploitation invités qui essaient d'utiliser cette fonctionnalité verront cela comme une perte de paquets, ce qui peut provoquer un comportement inattendu de certaines applications dans l'invité. Cela n'entraîne pas de problèmes avec des systèmes d'exploitation invités dans leur configuration par défaut, vu que les cadres jumbo doivent être explicitement activés.

6.2 Introduction aux modes réseaux

Chacun des huit adaptateurs réseau peut être configuré séparément pour agir dans l'un des cinq modes suivants :

Non attaché Dans ce mode, VirtualBox indique à l'invité qu'une carte réseau est présente mais qu'il n'y a pas de connexion – comme si aucun câble Ethernet n'était branché sur la carte. - De cette façon, il est possible de « débrancher » le câble Ethernet virtuel et de couper la connexion, ce qui peut être utile pour informer un système d'exploitation invité qu'aucune connexion réseau n'est disponible et l'obliger à une reconfiguration.

Network Address Translation (NAT) (traduction d'adresse réseau)

Si tout ce que vous voulez faire est de naviguer sur Internet, télécharger des fichiers et lire les messages à l'intérieur de l'invité, ce mode par défaut devrait vous suffire et vous pouvez sauter en toute sécurité le reste de cette section. Merci de remarquer qu'il y a certaines limitations quand on utilise le partage de fichiers Windows (voir le chapitre 6.3.3, *Limites du NAT*, page 94 pour des détails).

Réseau bridgé C'est pour des besoins réseau plus avancés tels que des simulations de réseau et le fonctionnement de serveurs dans un invité. Lorsqu'il est activé, VirtualBox se connecte à une de vos cartes réseaux installées et échange directement des paquets réseaux, contournant la pile réseau de votre système d'exploitation hôte.

Réseau interne Ceci peut être utilisé pour créer un genre différent de réseau de type logiciel, visible par les machines virtuelles sélectionnées mais pas pour les applications en fonction sur l'hôte ou dans le monde extérieur.

Réseau privé-hôte Ceci peut être utilisé pour créer un réseau contenant l'hôte et un ensemble de machines virtuelles, sans qu'il y ait besoin de l'interface physique de l'hôte. À la place, une interface réseau virtuelle (comme une interface loopback) est créée sur l'hôte, fournissant la connectivité parmi les machines virtuelles et l'hôte.

Réseau générique Les modes rarement utilisés partagent la même interface réseau générique mais ils permettent à l'utilisateur de sélectionner un pilote qui peut être inclu dans VirtualBox ou distribué dans un pack d'extension.

Pour l'heure, il y a deux sous-modes disponibles :

[1] http://www.linux-kvm.org/page/WindowsGuestDrivers.

Tunnel UDP On peut l'utiliser pour connecter directement des machines virtuelles en fonction sur des hôtes différents, facilement et de manière transparente, via une infrastructure réseau existante.

Réseau VDE (Virtual Distributed Ethernet) Cette option peut être utilisée pour se connecter à un switch Virtual Distributed Ethernet (Ethernet distribué virtuel) d'un hôte Linux ou FreeBSD. Pour l'instant, il implique de compiler VirtualBox à partir des sources, car les paquets d'Oracle ne l'incluent pas.

Les sections suivantes décrivent les modes de réseau disponibles avec plus de détails.

6.3 Network Address Translation (NAT)

La traduction d'adresses réseaux (NAT) est la façon la plus simple d'accéder à un réseau externe depuis une machine virtuelle. Habituellement, cela ne demande aucune configuration sur le réseau hôte et le système invité. Pour cette raison, c'est le mode réseau par défaut de VirtualBox.

Une machine virtuelle où le NAT est activé agit comme un vrai ordinateur qui se connecte à Internet à travers un routeur. Le « routeur », dans ce cas, est le moteur réseau de VirtualBox qui dresse le plan du trafic venant et à destination de la machine virtuelle de façon intégréte. L'inconvénient du mode NAT est que, tout comme se comporte un réseau privé derrière un routeur, la machine virtuelle est invisible et injoignable à partir et à l'extérieur sur Internet ; vous ne pouvez pas lancer de serveur de cette façon sauf si vous réglez le forwarding (routage) de ports (décrit ci-dessous).

Les trames réseau envoyés par le système d'exploitation invité sont reçus par le moteur NAT de VirtualBox, qui extrait les données TCP/IP et les réexpédie en utilisant le système d'exploitation de l'hôte. Sur une application de l'hôte ou sur un autre ordinateur du même réseau que l'hôte, on a l'impression que les données ont été envoyées par l'application VirtualBox sur l'hôte, en utilisant une adresse IP appartenant à l'hôte. VirtualBox écoute les réponses aux paquets envoyés, puis les réemballe et les renvoie sur la machine invitée sur son réseau privé.

La machine virtuelle reçoit son adresse réseau et sa configuration sur le réseau privé à partir d'un serveur DHCP intégré dans VirtualBox. L'adresse IP ainsi assignée à la machine virtuelle est en général sur un réseau complètement différent de l'hôte. Comme plus d'une carte d'une machine virtuelle peut être paramétrée pour utiliser NAT, la première carte est connectée au réseau privé 10.0.2.0, la deuxième au réseau 10.0.3.0 et ainsi de suite. Si vous devez modifier la plage d'IP assignées à l'hôte pour une raison quelconque, merci de vous reporter au chapitre 9.11, *Fine-tuning the VirtualBox NAT engine*, page 162.

6.3.1 Configurer la redirection de ports avec NAT

Comme la machine virtuelle est connectée à un réseau privé interne à VirtualBox et qu'il est invisible pour l'hôte, les services réseau de l'invité ne sont pas accessibles pour la machine hôte ou pour d'autres ordinateurs du même réseau. Néanmoins, comme un routeur physique, VirtualBox peut effectuer un **transfert de ports** sélectionné. Cela signifie que VirtualBox écoute certains ports sur l'hôte et renvoie tous les paquets qui y arrivent vers l'invité, sur un port identique ou différent.

Pour une application de l'hôte ou d'autres machines physiques (ou virtuelles) du réseau, c'est comme si le service derrière le proxy fonctionnait finalement sur l'hôte. Cela signifie aussi que vous ne pouvez pas exécuter le même service sur les mêmes ports que sur l'hôte. Néanmoins, vous en tirez les avantages de l'exécution d'un service dans une machine virtuelle : par exemple les services de la machine hôte ou d'autres machines virtuelles ne peuvent pas être compromis ou plantés par une vulnérabilité ou un bogue du service, et le service peut s'exécuter sur un système d'exploitation différent du système hôte.

Vous pouvez paramétrer un service invité que vous souhaitez mettre derrière un proxy en utilisant l'outil en ligne de commande `VBoxManage` ; pour des détails, merci de vous référer au chapitre 8.7, *VBoxManage modifyvm*, page 120.

Vous devrez savoir quels ports sur l'invité le service utilise et décider quels ports utiliser sur l'hôte (vous voudrez souvent, mais pas toujours, utiliser les mêmes ports sur l'invité et sur l'hôte). Vous pouvez utiliser n'importe quel port de l'hôte qui n'est pas déjà utilisé par un service. Par exemple, pour paramétrer des connexions entrantes NAT vers un serveur `ssh` de l'invité, utilisez la commande suivante :

```
VBoxManage modifyvm "nom VM" --natpf1 "guestssh,tcp,,2222,,22"
```

Avec l'exemple ci-dessus, tout le trafic `tcp` arrivant sur le port 2222 sur n'importe quelle interface de l'hôte sera transféré sur le port 22 dans l'invité. Le nom du protocole `tcp` est un attribut obligatoire définissant quel protocole devrait être utilisé pour la redirection (on pourrait aussi utiliser (`udp`). Le nom `guestssh` est purement descriptif et sera généré automatiquement si vous n'en spécifiez pas. Le numéro après `--natpf` nomme la carte réseau, comme dans d'autres paramétrages de VboxManage.

Pour supprimer à nouveau cette règle de redirection, utilisez la commande suivante :

```
VBoxManage modifyvm "nom VM" --natpf1 delete "guestssh"
```

Si pour une raison quelconque l'invité utilise une adresse IP statique non attribuée par le serveur DHCP intégré, il faut spécifier l'IP de l'invité lors de l'enregistrement de la règle de redirection :

```
VBoxManage modifyvm "nom VM" --natpf1
      "guestssh,tcp,,2222,10.0.2.19,22"
```

Cet exemple est identique au précédent, sauf qu'on dira au moteur NAT qu'on peut trouver l'invité sur l'adresse 10.0.2.19.

Pour rediriger *tout* le trafic entrant d'une interface spécifique de l'hôte vers l'invité, spécifiez l'IP de cette interface hôte comme ceci :

```
VBoxManage modifyvm "nom VM" --natpf1 "guestssh,tcp,127.0.0.1,2222,,22"
```

Ceci redirige tout le trafic TCP arrivant sur l'interface localhost (127.0.0.1) via le port 2222 vers le port 22 de l'invité.

Il n'est pas possible de configurer des connexions NAT entrantes pendant que la VM est en fonction. Néanmoins, vous pouvez modifier les paramètres d'une VM actuellement sauvegardée (ou éteinte) dans un instantané.

6.3.2 Amorçage PXE avec NAT

L'amorçage PXE est à présent supporté par le mode NAT. Le serveur DHCP NAT fournit un fichier d'amorçage nommé sous la forme `nomvm.pxe` si le répertoire `TFTP` existe dans le répertoire où est conservé le fichier `VirtualBox.xml` de l'utilisateur. C'est à l'utilisateur de fournir `nomvm.pxe`.

6.3.3 Limites du NAT

Il existe quatre **limites** du mode NAT dont l'utilisateur devrait être conscient :

Limite du protocole ICMP : Certains outils de débogage réseau fréquemment utilisés (tels que `ping` ou tracerouting) utilisent le protocole ICMP pour envoyer/recevoir des messages. Si le support d'ICMP a été amélioré avec VirtualBox 2.1 (`ping` devrait maintenant fonctionner), il se peut que d'autres outils ne fonctionnent pas de façon fiable.

La réception de broadcasts (diffusion de messages) UDP n'est pas fiable : L'invité ne reçoit pas de façon fiable les broadcasts puisque, pour économiser des ressources, il n'écoute qu'un certain temps après que l'invité ait envoyé des données UDP sur un port particulier. Par conséquent, la résolution de nom NetBios basée sur les broadcasts ne fonctionne pas toujours (mais WINS fonctionne toujours). Vous pouvez utiliser, en contournement, l'IP numérique du serveur désiré noté `\\server\share`.

Les protocoles tels que GRE ne sont pas supportés : Les protocoles autres que TCP et UDP ne sont pas supportés. Cela signifie que certains produits VPN (par exemple PPTP de Microsoft) ne peuvent pas être utilisés. Il y a d'autres produits VPN qui utilisent simplement TCP ou UDP.

Redirection des ports de l'hôte < 1024 impossible : Sur les hôtes basés sur Unix (comme Linux, Solaris, Mac OS X) il n'est pas possible de solliciter des ports inférieurs à 1024 depuis des applications non lancées par `root`. Il en résulte que si vous essayez de configurer une telle redirection de ports, la VM refusera de démarrer.

Ces limitations ne devraient normalement pas perturber l'utilisation standard du réseau. Mais la présence du NAT a aussi des effets subtils qui peuvent interagir avec des protocoles qui, normalement, fonctionnent. NFS est un exemple où le serveur est souvent configuré pour refuser les connexions provenant de ports non privilégiés (c'est-à-dire des ports non inférieurs à 1024).

6.4 Réseau bridgé

Avec le réseau bridgé, VirtualBox utilise un pilote de périphérique de votre système *hôte* qui filtre les données depuis l'adaptateur de votre réseau physique. Ce pilote est donc appelé un pilote « net filter » (filtre de réseau). Ceci permet à VirtualBox d'intercepter des données du réseau physique et d'injecter des données dedans, créant de fait une nouvelle interface réseau logicielle. Quand un invité utilise une telle nouvelle interface logicielle, le système hôte voit l'invité comme si il était physiquement connectés à l'interface par un câble réseau : l'hôte peut envoyer des données à l'invité par cette interface et reçoit des données à partir d'elle. Cela signifie que vous pouvez régler un routage ou un pont entre l'invité et le reste de votre réseau.

Pour que cela fonctionne, VirtualBox a besoin d'un pilote de périphérique sur votre système hôte. La façon dont fonctionne le réseau bridgé a été entièrement réécrite avec VirtualBox 2.0 et 2.1, selon le système d'exploitation hôte.[2]

> **Note:** Même si TAP n'est plus nécessaire sur Linux avec le réseau bridgé, vous *pouvez* toujours utiliser des interfaces TAP pour certains réglages avancés, puisque vous pouvez connecter une VM à n'importe quelle interface hôte – ce qui pourrait aussi être une interface TAP.

Pour activer le réseau bridgé, tout ce que vous avez besoin de faire est d'ouvrir la boîte de dialogue Paramètres de la fenêtre d'une machine virtuelle, aller sur la page « Réseau » et sélectionner « Réseau bridgé » dans la liste déroulante du champ « Attaché à ». Enfin, sélectionnez l'interface hôte désirée depuis la liste au bas de la page, qui contient les interfaces réseau physiques de vos systèmes. Sur un MacBook typique, par exemple, cela vous permettra de choisir entre « en1:

[2] Pour les hôtes Mac OS X et Solaris, les pilotes net filter étaient déjà ajoutés à VirtualBox 2.0 (en tant que support dès le départ pour Host Interface Networking (réseau par interface hôte) sur ces plateformes). Avec VirtualBox 2.1, les pilotes net filter ont également été ajoutés pour les hôtes Windows et Linux, en remplaçant les mécanismes présents auparavant dans VirtualBox pour ces plateformes ; surtout sur Linux, la méthode plus récente sous Linux exigeait de créer des interfaces et des ponts (bridge) TAP, ce qui était complexe et ce qui changeait d'une distribution à l'autre. Rien de tout cela n'est nécessaire aujourd'hui. Le réseau bridgé s'appelait avant « Host Interface Networking » (réseau avec interface de l'hôte), et il a été renommé dans la version 2.2 sans modification de son fonctionnement.

AirPort » (qui est l'interface tans fil) et « en0: Ethernet », qui représente l'interface avec un câble réseau.

En fonction de votre système d'exploitation hôte, vous devriez garder à l'esprit les limites suivantes :

- Sur des hôtes **Macintosh**, la fonctionnalité est limitée quand on utilise AirPort (le réseau sans fil de Mac) pour le réseau bridgé. Actuellement, VirtualBox ne supporte que l'IPv4 via AirPort. Pour d'autres protocoles tels qu'IPv6 et IPX, vous devez choisir une interface filaire.

- Sur les hôtes **Linux**, la fonctionnalité est limitée lors de l'utilisation d'interfaces sans fil pour le réseau bridgé. VirtualBox ne supporte actuellement que l'IPv4 via le sans fil. Pour les autres protocoles tels que IPv6 et IPX, vous devez choisir une interface filaire.

  En outre, régler le MTU à moins de 1500 bytes sur les interfaces filaires fournies par le pilote sky2 sur la Marvell Yukon II EC Ultra Ethernet NIC est connu pour entraîner des pertes de paquets dans certaines conditions.

- Sur les hôtes **Solaris**, il n'y a pas de support pour utiliser les interfaces sans fil. Le filtrage du trafic invité en utilisant IPFilter n'est pas non plus totalement supporté du fait de restrictions techniques du sous-système de réseau de Solaris 11.

  Avec VirtualBox 2.0.4 et supérieur, il est possible d'utiliser des interfaces réseaux virtuelles Crossbow (Crossbow Virtual Network Interfaces, VNICs) avec le réseau bridgé, mais avec les précautions suivantes :

  – Vous ne pouvez pas partager un VNIC entre plusieurs interfaces réseau virtuelles, ainsi chaque interface réseau invité doit avoir son propre VNIC.

  – La VNIC et l'interface réseau de l'invité qui utilise le VNIC doivent se voir assigner des adresses MAC.

  Lors de l'utilisation des interfaces VLAN avec VirtualBox, elles doivent être nommées selon le schéma de nommage du hack PPA (comme “e1000g513001”), car sinon, il se peut que l'invité reçoive des paquets sous un format inattendu.

6.5 Réseau interne

Le réseau interne est similaire au réseau bridgé dans le sens où la VM peut communiquer directement avec le monde extérieur. Cependant le « monde extérieur » est limité aux autres VMs qui se connectent au même réseau interne.

Même si techniquement, tout ce que vous pouvez faire en utilisant le réseau interne est aussi faisable en utilisant le réseau bridgé, il y a des avantages en matière de sécurité à utiliser le réseau interne. En mode réseau bridgé, tout le trafic passe par une interface physique du système hôte. Il est donc possible d'attacher un détecteur de paquets (tel que Wireshark) à l'interface hôte et d'enregistrer tout le trafic qui passe par lui. Si, pour une raison quelconque, vous préférez que deux ou davantage de VMs de la même machine communiquent de façon privée, tout en cachant leurs données à la fois au système hôte et à et à l'utilisateur, le réseau bridgé n'est donc pas une option.

Les réseaux internes sont créés automatiquement selon les besoins, ainsi il n'y a pas de configuration centrale. Chaque réseau interne est simplement identifié par son nom. Une fois qu'il y a plus d'une carte réseau virtuelle active avec le même ID de réseau interne, le pilote de support de VirtualBox va automatiquement « brancher » les cartes et agir comme un switch réseau. Le pilote du support de VirtualBox implémente une switch Ethernet complet et supporte à la fois les trames broadcast/multicast et le mode promiscuité.

Afin d'attacher la carte réseau d'une machine virtuelle à un réseau interne, réglez son mode réseau sur « réseau interne ». Il y a deux façons d'accomplir cela :

- Vous pouvez utiliser la boîte de dialogue « Paramètres » de la VM dans l'interface graphique. Dans la catégorie « Réseau » de la boîte de dialogue des paramètres, sélectionnez « Réseau interne » dans la liste déroulante des modes réseau. Maintenant, sélectionnez le nom d'un réseau interne existant depuis le menu déroulant en-dessous ou entrez un nouveau nom dans le champ d'édition.

- Vous pouvez utiliser

  ```
  VBoxManage modifyvm "nom VM" --nic<x> intnet
  ```

  Vous pouvez éventuellement spécifier un nom de réseau avec la commande

  ```
  VBoxManage modifyvm "nom VM" --intnet<x> "network name"
  ```

  Si vous ne spécifiez pas de nom de réseau, la carte réseau sera attachée au réseau `intnet` par défaut.

Sauf si vous configurez les cartes réseau (virtuelles) des systèmes d'exploitation invités qui participent au réseau interne pour utiliser des adresses IP statiques, il se peut que vous vouliez utiliser le serveur DHCP construit dans VirtualBox pour gérer les adresses IP pour le réseau interne. Merci de voir le chapitre 8.34, *VBoxManage dhcpserver*, page 146 pour des détails.

Par mesure de sécurité, l'implémentation Linux du réseau interne n'autorise que les VMs en fonction sous le même ID utilisateur à établir un réseau interne.

6.6 Réseau privé avec l'hôte (Host-only)

Le réseau privé avec l'hôte est un autre mode réseau qui a été ajouté avec la version 2.2 de VirtualBox. On peut le concevoir comme un hybride entre les modes réseau bridgé et interne : comme avec le réseau bridgé, les machines virtuelles peuvent se parler entre elles et avec l'hôte comme si elles étaient connectées sur un switch ethernet physique. De la même façon, comme avec le réseau interne cependant, il n'est pas besoin qu'une interface réseau physique soit présente et les machines virtuelles ne peuvent pas parler au monde extérieur à l'hôte puisqu'elles ne sont pas connectées à une interface réseau physique.

Quand on utilise le réseau privé avec l'hôte, VirtualBox crée à la place une nouvelle interface logicielle sur l'hôte qui apparaît ensuite à côté de vos interfaces réseau existantes. En d'autres termes, alors qu'avec un réseau bridgé, on utilise une interface réseau existante pour y attacher les machines virtuelles, avec le mode hôte privé on crée une nouvelle interface « loopback » sur l'hôte. Et alors qu'avec le réseau interne, on ne peut pas voir le trafic entre les machines virtuelles, on peut intercepter le trafic sur l'interface « loopback » de l'hôte.

Le réseau avec hôte privé est particulièrement utile pour les applications virtuelles pré-configurées, où plusieurs machines virtuelles sont emballées ensembles et destinées à coopérer. Par exemple, il se peut qu'une machine virtuelle contienne un serveur Web et la seconde une base de données, et puisqu'elles sont prévues pour se parler, l'application peut demander à VirtualBox de régler un réseau avec hôte privé pour les deux. Un second réseau (bridgé) connecterait alors le serveur Web au monde extérieur pour donner accès aux données, mais le monde extérieur ne peut pas se connecter à la base de données.

Pour passer l'interface réseau de la machine virtuelle en mode « avec hôte privé » :

- soit allez sur la page « réseau » dans les paramètres de la machine virtuelle dans l'interface graphique et sélectionnez « Hôte privé » ou

- en ligne de commande, tapez `VBoxManage modifyvm "nom VM" --nic<x> hostonly` ; voir le chapitre 8.7, *VBoxManage modifyvm*, page 120 pour les détails.

Pour le réseau privé avec hôte, comme avec le réseau interne, il se peut que vous trouviez le serveur DHCP construit dans VirtualBox utile. On peut l'activer puis gérer les adresses IP dynamiques dans le réseau avec hôte privé puisque sinon vous devriez configurer toutes les adresses IP de manière statique.

- Dans l'interface graphique de VirtualBox, vous pouvez configurer toutes ces options dans les paramètres globaux avec « Fichier » -> « Paramètres » -> « Réseau », qui liste tous les réseaux avec hôte privé présentement utilisés. Cliquez sur un nom de réseau puis sur le bouton « Éditer » à droite et vous pouvez modifier les paramètres de l'adaptateur et du DHCP.
- Alternativement, vous pouvez utiliser `VBoxManage dhcpserver` en ligne de commande ; voir le chapitre 8.34, *VBoxManage dhcpserver*, page 146 pour des détails.

6.7 Réseau par tunnel UDP

Ce mode réseau permet de connecter entre elles des machines virtuelles en fonction sur plusieurs hôtes.

Techniquement, cela se fait en encapsulant des cadres (frames) Ethernet envoyés ou reçus par la carte réseau invitée dans un chiffreur de données (data datagrams) UDP/IP, et en les envoyant à l'hôte à travers un réseau disponible.

Le mode Tunnel UDP prend trois paramètres :

port UDP source Le port sur lequel écoute l'hôte. Les déchiffreurs de données arrivant sur ce port issus de n'importe quelle adresse source seront redirigés vers la partie réceptrice d'une carte réseau invitée.

adresse de destination L'adresse IP de l'hôte cible des données transmises.

Port UDP de destination Numéro de port vers lequel sont envoyées les données transmises.

Lorsqu'on connecte deux machines virtuelles sur deux hôtes différents, leurs adresses IP doivent être échangées. Sur un hôte unique, les ports sources et de destination doivent être échangés.

Dans l'exemple suivant, l'hôte 1 utilise l'adresse IP 10.0.0.1 et l'hôte 2 utilise l'adresse IP 10.0.0.2. Configuration en ligne de commande :

```
VBoxManage modifyvm "VM 01 on host 1" --nic<x> generic
VBoxManage modifyvm "VM 01 on host 1" --nicgenericdrv<x> UDPTunnel
VBoxManage modifyvm "VM 01 on host 1" --nicproperty<x> dest=10.0.0.2
VBoxManage modifyvm "VM 01 on host 1" --nicproperty<x> sport=10001
VBoxManage modifyvm "VM 01 on host 1" --nicproperty<x> dport=10002
```

et

```
VBoxManage modifyvm "VM 02 on host 2" --nic<y> generic
VBoxManage modifyvm "VM 02 on host 2" --nicgenericdrv<y> UDPTunnel
VBoxManage modifyvm "VM 02 on host 2" --nicproperty<y> dest=10.0.0.1
VBoxManage modifyvm "VM 02 on host 2" --nicproperty<y> sport=10002
VBoxManage modifyvm "VM 02 on host 2" --nicproperty<y> dport=10001
```

Bien entendu, vous pouvez connecter deux machines virtuelles sur le même hôte en réglant le paramètre d'adresse de destination sur 127.0.0.1 sur les deux. Dans ce cas, il agira comme le « Réseau interne », mais l'hôte peut voir le trafic réseau, ce qu'il ne pourrait pas faire en réseau interne normal.

Note: Sur les hôtes basés sur Unix (comme Linux, Solaris, Mac OS X), il n'est pas possible de diriger (bind) sur des ports inférieurs à 1024, à partir d'applications qui ne sont pas exécutées par `root`. Il s'en suit que si vous essayez de configurer un tel port source UDP, la VM refusera de démarrer.

6.8 Réseau VDE

Virtual Distributed Ethernet (Ethernet virtuel distribué, VDE[3]) est une infrastructure réseau virtuelle flexible, reliant plusieurs hôtes de manière sécurisée. Elle permet de basculer entre L2/L3, y compris avec un protocole d'arborescence de passerelles, entre VLANs et l'émulation WAN. C'est une partie facultative de VirtualBox qui n'est incluse que dans code source.

Les blocs de construction de base de l'infrastructure sont des switches VDE, des plugs (prises) VDE et des fils VDE, qui connectent les switches entre eux.

Le pilote VDE de VirtualBox prend un paramètre :

Réseau VDE Le nom du socket du switch du réseau VDE vers lequel se connectera la VM.

L'exemple de base suivant montre comment connecter une machine virtuelle à un switch VDE :

1. Créez un switch VDE :

   ```
   vde_switch -s /tmp/switch1
   ```

2. Configuration en ligne de commande :

   ```
   VBoxManage modifyvm "nom VM" --nic<x> generic
   ```

   ```
   VBoxManage modifyvm "nom VM" --nicgenericdrv<x> VDE
   ```

   Pour se connecter au port du switch automatiquement affecté, utilisez :

   ```
   VBoxManage modifyvm "nom VM" --nicproperty<x> network=/tmp/switch1
   ```

   Pour se connecter à un port spécifique du switch <n>, utilisez :

   ```
   VBoxManage modifyvm "nom VM" --nicproperty<x> network=/tmp/switch1[<n>]
   ```

   La dernière option peut être utile pour des VLANs.

3. Plan facultatif entre un port de switch VDE et un VLAN : (du switch en console)

   ```
   vde$ vlan/create <VLAN>
   ```

   ```
   vde$ port/setvlan <port> <VLAN>
   ```

VDE n'est disponible que sur des hôtes Linux et FreeBSD. Il n'est disponible que si le logiciel VDE et la bibliothèque du plugin VDE du projet VirtualSquare sont installés sur le système hôte. Pour plus d'informations sur le paramétrage des réseaux VDE, merci de voir la documentation qui accompagne le logiciel.[4]

6.9 Limiting bandwidth for network I/O

Starting with version 4.2, VirtualBox allows for limiting the maximum bandwidth used for network transmission. Several network adapters of one VM may share limits through bandwidth groups. It is possible to have more than one such limit.

Limits are configured through `VBoxManage`. The example below creates a bandwidth group named "Limit", sets the limit to 20 Mbit/s and assigns the group to the first and second adapters of the VM:

```
VBoxManage bandwidthctl "VM name" add Limit --type network --limit 20m
VBoxManage modifyvm "VM name" --nicbandwidthgroup1 Limit
VBoxManage modifyvm "VM name" --nicbandwidthgroup2 Limit
```

[3] VDE est un projet développé par Renzo Davoli, professeur associé à l'Université de Bologna en Italie.
[4] http://wiki.virtualsquare.org/wiki/index.php/VDE_Basic_Networking.

All adapters in a group share the bandwidth limit, meaning that in the example above the bandwidth of both adapters combined can never exceed 20 Mbit/s. However, if one adapter doesn't require bandwidth the other can use the remaining bandwidth of its group.

The limits for each group can be changed while the VM is running, with changes being picked up immediately. The example below changes the limit for the group created in the example above to 100 Kbit/s:

```
VBoxManage bandwidthctl "VM name" set Limit --limit 100k
```

7 Remote virtual machines

7.1 Remote display (VRDP support)

VirtualBox can display virtual machines remotely, meaning that a virtual machine can execute on one machine even though the machine will be displayed on a second computer, and the machine will be controlled from there as well, as if the virtual machine was running on that second computer.

For maximum flexibility, starting with VirtualBox 4.0, VirtualBox implements remote machine display through a generic extension interface, the VirtualBox Remote Desktop Extension (VRDE). The base open-source VirtualBox package only provides this interface, while implementations can be supplied by third parties with VirtualBox extension packages, which must be installed separately from the base package. See chapitre 1.5, *Installer et lancer VirtualBox*, page 15 for more information.

Oracle provides support for the **VirtualBox Remote Display Protocol (VRDP)** in such a VirtualBox extension package. When this package is installed, VirtualBox versions 4.0 and later support VRDP the same way as binary (non-open-source) versions of VirtualBox before 4.0 did.

VRDP is a backwards-compatible extension to Microsoft's Remote Desktop Protocol (RDP). Typically graphics updates and audio are sent from the remote machine to the client, while keyboard and mouse events are sent back. As a result, you can use any standard RDP client to control the remote VM.

Even when the extension is installed, the VRDP server is disabled by default. It can easily be enabled on a per-VM basis either in the VirtualBox Manager in the "Display" settings (see chapitre 3.5, *Paramètres d'affichage*, page 50) or with `VBoxManage`:

```
VBoxManage modifyvm "VM name" --vrde on
```

If you use `VBoxHeadless` (described further below), VRDP support will be automatically enabled since VBoxHeadless has no other means of output.

7.1.1 Common third-party RDP viewers

Since VRDP is backwards-compatible to RDP, you can use any standard RDP viewer to connect to such a remote virtual machine (examples follow below). For this to work, you must specify the **IP address** of your *host* system (not of the virtual machine!) as the server address to connect to, as well as the **port number** that the RDP server is using.

By default, VRDP uses TCP port `3389`. You will need to change the default port if you run more than one VRDP server, since the port can only be used by one server at a time; you might also need to change it on Windows hosts since the default port might already be used by the RDP server that is built into Windows itself. Ports 5000 through 5050 are typically not used and might be a good choice.

The port can be changed either in the "Display" settings of the graphical user interface or with `--vrdeport` option of the `VBoxManage modifyvm` command. You can specify a comma-separated list of ports or ranges of ports. Use a dash between two port numbers to specify a range. The VRDP server will bind to **one** of available ports from the specified list. For example, `VBoxManage modifyvm "VM name" --vrdeport 5000,5010-5012` will configure the server to bind to one of the ports 5000, 5010, 5011 or 5012. See chapitre 8.7, *VBoxManage modifyvm*, page 120 for details.

The actual port used by a running VM can be either queried with `VBoxManage showvminfo` command or seen in the GUI on the "Runtime" tab of the "Session Information Dialog", which is accessible via the "Machine" menu of the VM window.

Here follow examples for the most common RDP viewers:

- On Windows, you can use the Microsoft Terminal Services Connector (`mstsc.exe`) that ships with Windows. You can start it by bringing up the "Run" dialog (press the Windows key and "R") and typing "mstsc". You can also find it under "Start" -> "All Programs" -> "Accessories" -> "Remote Desktop Connection". If you use the "Run" dialog, you can type in options directly:

  ```
  mstsc 1.2.3.4[:3389]
  ```

  Replace "1.2.3.4" with the host IP address, and 3389 with a different port if necessary.

  Note: When connecting to localhost in order to test the connection, the addresses `localhost` and `127.0.0.1` might not work using `mstsc.exe`. Instead, the address `127.0.0.2[:3389]` has to be used.

- On other systems, you can use the standard open-source `rdesktop` program. This ships with most Linux distributions, but VirtualBox also comes with a modified variant of rdesktop for remote USB support (see chapitre 7.1.4, *Remote USB*, page 104 below).

  With rdesktop, use a command line such as the following:

  ```
  rdesktop -a 16 -N 1.2.3.4:3389
  ```

  As said for the Microsoft viewer above, replace "1.2.3.4" with the host IP address, and 3389 with a different port if necessary. The `-a 16` option requests a color depth of 16 bits per pixel, which we recommend. (For best performance, after installation of the guest operating system, you should set its display color depth to the same value). The `-N` option enables use of the NumPad keys.

- If you run the KDE desktop, you might prefer `krdc`, the KDE RDP viewer. The command line would look like this:

  ```
  krdc --window --high-quality rdp:/1.2.3.4[:3389]
  ```

  Again, replace "1.2.3.4" with the host IP address, and 3389 with a different port if necessary. The "rdp:/" bit is required with krdc to switch it into RDP mode.

- With Sun Ray thin clients you can use `uttsc`, which is part of the Sun Ray Windows Connector package. See the corresponding documentation for details.

7.1.2 VBoxHeadless, the remote desktop server

While any VM started from the VirtualBox Manager is capable of running virtual machines remotely, it is not convenient to have to run the full-fledged GUI if you never want to have VMs displayed locally in the first place. In particular, if you are running server hardware whose only purpose is to host VMs, and all your VMs are supposed to run remotely over VRDP, then it is pointless to have a graphical user interface on the server at all – especially since, on a Linux or Solaris host, the VirtualBox manager comes with dependencies on the Qt and SDL libraries. This is inconvenient if you would rather not have the X Window system on your server at all.

VirtualBox therefore comes with yet another front-end called `VBoxHeadless`, which produces no visible output on the host at all, but instead only delivers VRDP data. This front-end has no dependencies on the X Window system on Linux and Solaris hosts.[1]

To start a virtual machine with `VBoxHeadless`, you have two options:

[1] Before VirtualBox 1.6, the headless server was called `VBoxVRDP`. For the sake of backwards compatibility, the VirtualBox installation still installs an executable with that name as well.

- You can use

  ```
  VBoxManage startvm "VM name" --type headless
  ```

  The extra `--type` option causes VirtualBox to use `VBoxHeadless` as the front-end to the internal virtualization engine instead of the Qt front-end.

- The alternative is to use `VBoxHeadless` directly, as follows:

  ```
  VBoxHeadless --startvm <uuid|name>
  ```

  This way of starting the VM is preferred because you can see more detailed error messages, especially for early failures before the VM execution is started. If you have trouble with `VBoxManage startvm`, it can help greatly to start `VBoxHeadless` directly to diagnose the problem cause.

Note that when you use `VBoxHeadless` to start a VM, since the headless server has no other means of output, the VRDP server will *always* be enabled, regardless of whether you had enabled the VRDP server in the VM's settings. If this is undesirable (for example because you want to access the VM via `ssh` only), start the VM like this:

```
VBoxHeadless --startvm <uuid|name> --vrde=off
```

To have the VRDP server enabled depending on the VM configuration, as the other front-ends would, use this:

```
VBoxHeadless --startvm <uuid|name> --vrde=config
```

7.1.3 Step by step: creating a virtual machine on a headless server

The following instructions may give you an idea how to create a virtual machine on a headless server over a network connection. We will create a virtual machine, establish an RDP connection and install a guest operating system – all without having to touch the headless server. All you need is the following:

1. VirtualBox on a server machine with a supported host operating system. The VirtualBox extension pack for the VRDP server must be installed (see the previous section). For the following example, we will assume a Linux server.

2. An ISO file accessible from the server, containing the installation data for the guest operating system to install (we will assume Windows XP in the following example).

3. A terminal connection to that host through which you can access a command line (e.g. via `ssh`).

4. An RDP viewer on the remote client; see chapitre 7.1.1, *Common third-party RDP viewers*, page 101 above for examples.

Note again that on the server machine, since we will only use the headless server, neither Qt nor SDL nor the X Window system will be needed.

1. On the headless server, create a new virtual machine:

   ```
   VBoxManage createvm --name "Windows XP" --ostype WindowsXP --register
   ```

   Note that if you do not specify `--register`, you will have to manually use the `registervm` command later.

   Note further that you do not need to specify `--ostype`, but doing so selects some sane default values for certain VM parameters, for example the RAM size and the type of the virtual network device. To get a complete list of supported operating systems you can use

```
VBoxManage list ostypes
```

2. Make sure the settings for this VM are appropriate for the guest operating system that we will install. For example:

```
VBoxManage modifyvm "Windows XP" --memory 256 --acpi on --boot1 dvd --nic1 nat
```

3. Create a virtual hard disk for the VM (in this case, 10GB in size):

```
VBoxManage createhd --filename "WinXP.vdi" --size 10000
```

4. Add an IDE Controller to the new VM:

```
VBoxManage storagectl "Windows XP" --name "IDE Controller"
      --add ide --controller PIIX4
```

5. Set the VDI file created above as the first virtual hard disk of the new VM:

```
VBoxManage storageattach "Windows XP" --storagectl "IDE Controller"
      --port 0 --device 0 --type hdd --medium "WinXP.vdi"
```

6. Attach the ISO file that contains the operating system installation that you want to install later to the virtual machine, so the machine can boot from it:

```
VBoxManage storageattach "Windows XP" --storagectl "IDE Controller"
      --port 0 --device 1 --type dvddrive --medium /full/path/to/iso.iso
```

7. Start the virtual machine using VBoxHeadless:

```
VBoxHeadless --startvm "Windows XP"
```

   If everything worked, you should see a copyright notice. If, instead, you are returned to the command line, then something went wrong.

8. On the client machine, fire up the RDP viewer and try to connect to the server (see chapitre 7.1.1, *Common third-party RDP viewers*, page 101 above for how to use various common RDP viewers).

   You should now be seeing the installation routine of your guest operating system remotely in the RDP viewer.

7.1.4 Remote USB

As a special feature on top of the VRDP support, VirtualBox supports remote USB devices over the wire as well. That is, the VirtualBox guest that runs on one computer can access the USB devices of the remote computer on which the VRDP data is being displayed the same way as USB devices that are connected to the actual host. This allows for running virtual machines on a VirtualBox host that acts as a server, where a client can connect from elsewhere that needs only a network adapter and a display capable of running an RDP viewer. When USB devices are plugged into the client, the remote VirtualBox server can access them.

For these remote USB devices, the same filter rules apply as for other USB devices, as described with chapitre 3.10.1, *Paramètres USB*, page 54. All you have to do is specify “Remote” (or “Any”) when setting up these rules.

Accessing remote USB devices is only possible if the RDP client supports this extension. On Linux and Solaris hosts, the VirtualBox installation provides a suitable VRDP client called `rdesktop-vrdp`. Recent versions of `uttsc`, a client tailored for the use with Sun Ray thin clients, also support accessing remote USB devices. RDP clients for other platforms will be provided in future VirtualBox versions.

To make a remote USB device available to a VM, `rdesktop-vrdp` should be started as follows:

```
rdesktop-vrdp -r usb -a 16 -N my.host.address
```

Note that `rdesktop-vrdp` can access USB devices only through `/proc/bus/usb`. Please refer to chapitre 12.6.7, *USB not working*, page 194 for further details on how to properly set up the permissions. Furthermore it is advisable to disable automatic loading of any host driver on the remote host which might work on USB devices to ensure that the devices are accessible by the RDP client. If the setup was properly done on the remote host, plug/unplug events are visible on the VBox.log file of the VM.

7.1.5 RDP authentication

For each virtual machine that is remotely accessible via RDP, you can individually determine if and how client connections are authenticated. For this, use `VBoxManage modifyvm` command with the `--vrdeauthtype` option; see chapitre 8.7, *VBoxManage modifyvm*, page 120 for a general introduction. Three methods of authentication are available:

- The "null" method means that there is no authentication at all; any client can connect to the VRDP server and thus the virtual machine. This is, of course, very insecure and only to be recommended for private networks.
- The "external" method provides external authentication through a special authentication library. VirtualBox ships with two such authentication libraries:
  1. The default authentication library, `VBoxAuth`, authenticates against user credentials of the hosts. Depending on the host platform, this means:
     - On Linux hosts, `VBoxAuth.so` authenticates users against the host's PAM system.
     - On Windows hosts, `VBoxAuth.dll` authenticates users against the host's WinLogon system.
     - On Mac OS X hosts, `VBoxAuth.dylib` authenticates users against the host's directory service.[2]

     In other words, the "external" method per default performs authentication with the user accounts that exist on the host system. Any user with valid authentication credentials is accepted, i.e. the username does not have to correspond to the user running the VM.
  2. An additional library called `VBoxAuthSimple` performs authentication against credentials configured in the "extradata" section of a virtual machine's XML settings file. This is probably the simplest way to get authentication that does not depend on a running and supported guest (see below). The following steps are required:

     a) Enable `VBoxAuthSimple` with the following command:

     ```
     VBoxManage setproperty vrdeauthlibrary "VBoxAuthSimple"
     ```

     b) To enable the library for a particular VM, you must then switch authentication to external:

     ```
     VBoxManage modifyvm <vm> --vrdeauthtype external
     ```

     Replace `<vm>` with the VM name or UUID.

     c) You will then need to configure users and passwords by writing items into the machine's extradata. Since the XML machine settings file, into whose "extradata" section the password needs to be written, is a plain text file, VirtualBox uses hashes to encrypt passwords. The following command must be used:

     ```
     VBoxManage setextradata <vm> "VBoxAuthSimple/users/<user>" <hash>
     ```

[2] Support for Mac OS X was added in version 3.2.

Replace <vm> with the VM name or UUID, <user> with the user name who should be allowed to log in and <hash> with the encrypted password. As an example, to obtain the hash value for the password "secret", you can use the following command:

```
VBoxManage internalcommands passwordhash "secret"
```

This will print

```
2bb80d537b1da3e38bd30361aa855686bde0eacd7162fef6a25fe97bf527a25b
```

You can then use VBoxManage setextradata to store this value in the machine's "extradata" section.

As example, combined together, to set the password for the user "john" and the machine "My VM" to "secret", use this command:

```
VBoxManage setextradata "My VM" "VBoxAuthSimple/users/john"
    2bb80d537b1da3e38bd30361aa855686bde0eacd7162fef6a25fe97bf527a25b
```

- Finally, the "guest" authentication method performs authentication with a special component that comes with the Guest Additions; as a result, authentication is not performed on the host, but with the *guest* user accounts.

  This method is currently still in testing and not yet supported.

In addition to the methods described above, you can replace the default "external" authentication module with any other module. For this, VirtualBox provides a well-defined interface that allows you to write your own authentication module. This is described in detail in the VirtualBox Software Development Kit (SDK) reference; please see chapitre 11, *VirtualBox programming interfaces*, page 181 for details.

7.1.6 RDP encryption

RDP features data stream encryption, which is based on the RC4 symmetric cipher (with keys up to 128bit). The RC4 keys are being replaced in regular intervals (every 4096 packets).

RDP provides different authentication methods:

1. Historically, RDP4 authentication was used, with which the RDP client does not perform any checks in order to verify the identity of the server it connects to. Since user credentials can be obtained using a "man in the middle" (MITM) attack, RDP4 authentication is insecure and should generally not be used.

2. RDP5.1 authentication employs a server certificate for which the client possesses the public key. This way it is guaranteed that the server possess the corresponding private key. However, as this hard-coded private key became public some years ago, RDP5.1 authentication is also insecure.

3. RDP5.2 authentication uses the Enhanced RDP Security, which means that an external security protocol is used to secure the connection. RDP4 and RDP5.1 use Standard RDP Security. VRDP server supports Enhanced RDP Security with TLS protocol and, as a part of TLS handshake, sends the server certificate to the client.

   The `Security/Method` VRDE property sets the desired security method, which is used for a connection. Valid values are:

   - `Negotiate` - both Enhanced (TLS) and Standard RDP Security connections are allowed. The security method is negotiated with the client. This is the default setting.
   - `RDP` - only Standard RDP Security is accepted.
   - `TLS` - only Enhanced RDP Security is accepted. The client must support TLS.

For example the following command allows a client to use either Standard or Enhanced RDP Security connection:

```
vboxmanage modifyvm NAME --vrdeproperty "Security/Method=negotiate"
```

If the `Security/Method` property is set to either `Negotiate` or `TLS`, the TLS protocol will be automatically used by the server, if the client supports TLS. However in order to use TLS the server must possess the Server Certificate, the Server Private Key and the Certificate Authority (CA) Certificate. The following example shows how to generate a server certificate.

a) Create a CA self signed certificate:

```
openssl req -new -x509 -days 365 -extensions v3_ca -keyout ca_key_private.pem -out ca_cert.pem
```

b) Generate a server private key and a request for signing:

```
openssl genrsa -out server_key_private.pem
openssl req -new -key server_key_private.pem -out server_req.pem
```

c) Generate the server certificate:

```
openssl x509 -req -days 365 -in server_req.pem -CA ca_cert.pem -CAkey ca_key_private.pem -set_se
```

The server must be configured to access the required files:

```
vboxmanage modifyvm NAME --vrdeproperty "Security/CACertificate=path/ca_cert.pem"

vboxmanage modifyvm NAME --vrdeproperty "Security/ServerCertificate=path/server_cert.pem"

vboxmanage modifyvm NAME --vrdeproperty "Security/ServerPrivateKey=path/server_key_private.pem"
```

As the client that connects to the server determines what type of encryption will be used, with rdesktop, the Linux RDP viewer, use the `-4` or `-5` options.

7.1.7 Multiple connections to the VRDP server

The VRDP server of VirtualBox supports multiple simultaneous connections to the same running VM from different clients. All connected clients see the same screen output and share a mouse pointer and keyboard focus. This is similar to several people using the same computer at the same time, taking turns at the keyboard.

The following command enables multiple connection mode:

```
VBoxManage modifyvm "VM name" --vrdemulticon on
```

7.1.8 Multiple remote monitors

To access two or more remote VM displays you have to enable the VRDP multiconnection mode (see chapitre 7.1.7, *Multiple connections to the VRDP server*, page 107).

The RDP client can select the virtual monitor number to connect to using the `domain` logon parameter (`-d`). If the parameter ends with `@` followed by a number, VirtualBox interprets this number as the screen index. The primary guest screen is selected with `@1`, the first secondary screen is `@2`, etc.

The Microsoft RDP6 client does not let you specify a separate domain name. Instead, use `domain\username` in the `Username:` field – for example, `@2\name`. `name` must be supplied, and must be the name used to log in if the VRDP server is set up to require credentials. If it is not, you may use any text as the username.

7.1.9 VRDP video redirection

Starting with VirtualBox 3.2, the VRDP server can redirect video streams from the guest to the RDP client. Video frames are compressed using the JPEG algorithm allowing a higher compression ratio than standard RDP bitmap compression methods. It is possible to increase the compression ratio by lowering the video quality.

The VRDP server automatically detects video streams in a guest as frequently updated rectangular areas. As a result, this method works with any guest operating system without having to install additional software in the guest; in particular, the Guest Additions are not required.

On the client side, however, currently only the Windows 7 Remote Desktop Connection client supports this feature. If a client does not support video redirection, the VRDP server falls back to regular bitmap updates.

The following command enables video redirection:

```
VBoxManage modifyvm "VM name" --vrdevideochannel on
```

The quality of the video is defined as a value from 10 to 100 percent, representing a JPEG compression level (where lower numbers mean lower quality but higher compression). The quality can be changed using the following command:

```
VBoxManage modifyvm "VM name" --vrdevideochannelquality 75
```

7.1.10 VRDP customization

With VirtualBox 4.0 it is possible to disable display output, mouse and keyboard input, audio, remote USB or clipboard individually in the VRDP server.

The following commands change corresponding server settings:

```
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableDisplay=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableInput=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableUSB=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableAudio=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableClipboard=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableUpstreamAudio=1
```

To reenable a feature use a similar command without the trailing 1. For example:

```
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableDisplay=
```

These properties were introduced with VirtualBox 3.2.10. However, in the 3.2.x series, it was necessary to use the following commands to alter these settings instead:

```
VBoxManage setextradata "VM name" "VRDP/Feature/Client/DisableDisplay" 1
VBoxManage setextradata "VM name" "VRDP/Feature/Client/DisableInput" 1
VBoxManage setextradata "VM name" "VRDP/Feature/Client/DisableUSB" 1
VBoxManage setextradata "VM name" "VRDP/Feature/Client/DisableAudio" 1
VBoxManage setextradata "VM name" "VRDP/Feature/Client/DisableClipboard" 1
```

To reenable a feature use a similar command without the trailing 1. For example:

```
VBoxManage setextradata "VM name" "VRDP/Feature/Client/DisableDisplay"
```

7.2 Teleporting

Starting with version 3.1, VirtualBox supports “teleporting” – that is, moving a virtual machine over a network from one VirtualBox host to another, while the virtual machine is running. This works regardless of the host operating system that is running on the hosts: you can teleport virtual machines between Solaris and Mac hosts, for example.

Teleporting requires that a machine be currently running on one host, which is then called the **"source"**. The host to which the virtual machine will be teleported will then be called the **"target"**; the machine on the target is then configured to wait for the source to contact the target. The machine's running state will then be transferred from the source to the target with minimal downtime.

Teleporting happens over any TCP/IP network; the source and the target only need to agree on a TCP/IP port which is specified in the teleporting settings.

At this time, there are a few prerequisites for this to work, however:

1. On the target host, you must configure a virtual machine in VirtualBox with exactly the same hardware settings as the machine on the source that you want to teleport. This does not apply to settings which are merely descriptive, such as the VM name, but obviously for teleporting to work, the target machine must have the same amount of memory and other hardware settings. Otherwise teleporting will fail with an error message.

2. The two virtual machines on the source and the target must share the same storage (hard disks as well as floppy and CD/DVD images). This means that they either use the same iSCSI targets or that the storage resides somewhere on the network and both hosts have access to it via NFS or SMB/CIFS.

   This also means that neither the source nor the target machine can have any snapshots.

Then perform the following steps:

1. On the *target* host, configure the virtual machine to wait for a teleport request to arrive when it is started, instead of actually attempting to start the machine. This is done with the following VBoxManage command:

   ```
   VBoxManage modifyvm <targetvmname> --teleporter on --teleporterport <port>
   ```

   where `<targetvmname>` is the name of the virtual machine on the target host and `<port>` is a TCP/IP port number to be used on both the source and the target hosts. For example, use 6000. For details, see chapitre 8.7.5, *Teleporting settings*, page 126.

2. Start the VM on the target host. You will see that instead of actually running, it will show a progress dialog. indicating that it is waiting for a teleport request to arrive.

3. Start the machine on the *source* host as usual. When it is running and you want it to be teleported, issue the following command on the source host:

   ```
   VBoxManage controlvm <sourcevmname> teleport --host <targethost> --port <port>
   ```

   where `<sourcevmname>` is the name of the virtual machine on the source host (the machine that is currently running), `<targethost>` is the host or IP name of the target host on which the machine is waiting for the teleport request, and `<port>` must be the same number as specified in the command on the target host. For details, see chapitre 8.12, *VBoxManage controlvm*, page 129.

For testing, you can also teleport machines on the same host; in that case, use "localhost" as the hostname on both the source and the target host.

Note: In rare cases, if the CPUs of the source and the target are very different, teleporting can fail with an error message, or the target may hang. This may happen especially if the VM is running application software that is highly optimized to run on a particular CPU without correctly checking that certain CPU features are actually present. VirtualBox filters what CPU capabilities are presented to the guest operating system. Advanced users can attempt to restrict these virtual CPU capabilities with the `VBoxManage --modifyvm --cpuid` command; see chapitre 8.7.5, *Teleporting settings*, page 126.

8 VBoxManage

8.1 Introduction

As briefly mentioned in chapitre 1.13, *Interfaces alternatives*, page 30, VBoxManage is the command-line interface to VirtualBox. With it, you can completely control VirtualBox from the command line of your host operating system. VBoxManage supports all the features that the graphical user interface gives you access to, but it supports a lot more than that. It exposes really all the features of the virtualization engine, even those that cannot (yet) be accessed from the GUI.

You will need to use the command line if you want to

- use a different user interface than the main GUI (for example, VBoxSDL or the VBoxHeadless server);
- control some of the more advanced and experimental configuration settings for a VM.

There are two main things to keep in mind when using `VBoxManage`: First, `VBoxManage` must always be used with a specific "subcommand", such as "list" or "createvm" or "startvm". All the subcommands that `VBoxManage` supports are described in detail in chapitre 8, *VBoxManage*, page 110.

Second, most of these subcommands require that you specify a particular virtual machine after the subcommand. There are two ways you can do this:

- You can specify the VM name, as it is shown in the VirtualBox GUI. Note that if that name contains spaces, then you must enclose the entire name in double quotes (as it is always required with command line arguments that contain spaces).

  For example:

  ```
  VBoxManage startvm "Windows XP"
  ```

- You can specify the UUID, which is the internal unique identifier that VirtualBox uses to refer to the virtual machine. Assuming that the aforementioned VM called "Windows XP" has the UUID shown below, the following command has the same effect as the previous:

  ```
  VBoxManage startvm 670e746d-abea-4ba6-ad02-2a3b043810a5
  ```

You can type `VBoxManage list vms` to have all currently registered VMs listed with all their settings, including their respective names and UUIDs.

Some typical examples of how to control VirtualBox from the command line are listed below:

- To create a new virtual machine from the command line and immediately register it with VirtualBox, use `VBoxManage createvm` with the `--register` option,[1] like this:

  ```
  $ VBoxManage createvm --name "SUSE 10.2" --register
  VirtualBox Command Line Management Interface Version 4.2.12
  (C) 2005-2013 Oracle Corporation
  All rights reserved.

  Virtual machine 'SUSE 10.2' is created.
  UUID: c89fc351-8ec6-4f02-a048-57f4d25288e5
  Settings file: '/home/username/.VirtualBox/Machines/SUSE 10.2/SUSE 10.2.xml'
  ```

[1] For details, see chapitre 8.6, *VBoxManage createvm*, page 120.

As can be seen from the above output, a new virtual machine has been created with a new UUID and a new XML settings file.

- To show the configuration of a particular VM, use `VBoxManage showvminfo`; see chapitre 8.4, *VBoxManage showvminfo*, page 119 for details and an example.
- To change settings while a VM is powered off, use `VBoxManage modifyvm`, e.g. as follows:

  ```
  VBoxManage modifyvm "Windows XP" --memory "512MB"
  ```

  For details, see chapitre 8.7, *VBoxManage modifyvm*, page 120.
- To change the storage configuration (e.g. to add a storage controller and then a virtual disk), use `VBoxManage storagectl` and `VBoxManage storageattach`; see chapitre 8.18, *VBoxManage storagectl*, page 134 and chapitre 8.17, *VBoxManage storageattach*, page 132 for details.
- To control VM operation, use one of the following:
  - To start a VM that is currently powered off, use `VBoxManage startvm`; see chapitre 8.11, *VBoxManage startvm*, page 129 for details.
  - To pause or save a VM that is currently running or change some of its settings, use `VBoxManage controlvm`; see chapitre 8.12, *VBoxManage controlvm*, page 129 for details.

8.2 Commands overview

When running VBoxManage without parameters or when supplying an invalid command line, the below syntax diagram will be shown. Note that the output will be slightly different depending on the host platform; when in doubt, check the output of `VBoxManage` for the commands available on your particular host.

```
Usage:

  VBoxManage [<general option>] <command>


General Options:

  [-v|--version]            print version number and exit
  [-q|--nologo]             suppress the logo
  [--settingspw <pw>]       provide the settings password
  [--settingspwfile <file>] provide a file containing the settings password


Commands:

  list [--long|-l]          vms|runningvms|ostypes|hostdvds|hostfloppies|
                            bridgedifs|dhcpservers|hostinfo|
                            hostcpuids|hddbackends|hdds|dvds|floppies|
                            usbhost|usbfilters|systemproperties|extpacks|
                            groups

  showvminfo                <uuid>|<name> [--details]
                            [--machinereadable]
  showvminfo                <uuid>|<name> --log <idx>

  registervm                <filename>

  unregistervm              <uuid>|<name> [--delete]
```

```
createvm                 --name <name>
                         [--groups <group>, ...]
                         [--ostype <ostype>]
                         [--register]
                         [--basefolder <path>]
                         [--uuid <uuid>]

modifyvm                 <uuid|name>
                         [--name <name>]
                         [--groups <group>, ...]
                         [--ostype <ostype>]
                         [--memory <memorysize in MB>]
                         [--pagefusion on|off]
                         [--vram <vramsize in MB>]
                         [--acpi on|off]
                         [--ioapic on|off]
                         [--pae on|off]
                         [--hpet on|off]
                         [--hwvirtex on|off]
                         [--hwvirtexexcl on|off]
                         [--nestedpaging on|off]
                         [--largepages on|off]
                         [--vtxvpid on|off]
                         [--synthcpu on|off]
                         [--cpuidset <leaf> <eax> <ebx> <ecx> <edx>]
                         [--cpuidremove <leaf>]
                         [--cpuidremoveall]
                         [--hardwareuuid <uuid>]
                         [--cpus <number>]
                         [--cpuhotplug on|off]
                         [--plugcpu <id>]
                         [--unplugcpu <id>]
                         [--cpuexecutioncap <1-100>]
                         [--rtcuseutc on|off]
                         [--monitorcount <number>]
                         [--accelerate3d on|off]
                         [--firmware bios|efi|efi32|efi64]
                         [--chipset ich9|piix3]
                         [--bioslogofadein on|off]
                         [--bioslogofadeout on|off]
                         [--bioslogodisplaytime <msec>]
                         [--bioslogoimagepath <imagepath>]
                         [--biosbootmenu disabled|menuonly|messageandmenu]
                         [--biossystemtimeoffset <msec>]
                         [--biospxedebug on|off]
                         [--boot<1-4> none|floppy|dvd|disk|net>]
                         [--nic<1-N> none|null|nat|bridged|intnet|
                                     generic]
                         [--nictype<1-N> Am79C970A|Am79C973]
                         [--cableconnected<1-N> on|off]
                         [--nictrace<1-N> on|off]
                         [--nictracefile<1-N> <filename>]
                         [--nicproperty<1-N> name=[value]]
                         [--nicspeed<1-N> <kbps>]
                         [--nicbootprio<1-N> <priority>]
                         [--nicpromisc<1-N> deny|allow-vms|allow-all]
                         [--nicbandwidthgroup<1-N> none|<name>]
                         [--bridgeadapter<1-N> none|<devicename>]
                         [--intnet<1-N> <network name>]
                         [--natnet<1-N> <network>|default]
                         [--nicgenericdrv<1-N> <driver>
                         [--natsettings<1-N> [<mtu>],[<socksnd>],
                                             [<sockrcv>],[<tcpsnd>],
                                             [<tcprcv>]]
                         [--natpf<1-N> [<rulename>],tcp|udp,[<hostip>],
                                       <hostport>,[<guestip>],<guestport>]
```

```
                        [--natpf<1-N> delete <rulename>]
                        [--nattftpprefix<1-N> <prefix>]
                        [--nattftpfile<1-N> <file>]
                        [--nattftpserver<1-N> <ip>]
                        [--natbindip<1-N> <ip>
                        [--natdnspassdomain<1-N> on|off]
                        [--natdnsproxy<1-N> on|off]
                        [--natdnshostresolver<1-N> on|off]
                        [--nataliasmode<1-N> default|[log],[proxyonly],
                                             [sameports]]
                        [--macaddress<1-N> auto|<mac>]
                        [--mouse ps2|usb|usbtablet
                        [--keyboard ps2|usb
                        [--uart<1-N> off|<I/O base> <IRQ>]
                        [--uartmode<1-N> disconnected|
                                         server <pipe>|
                                         client <pipe>|
                                         file <file>|
                                         <devicename>]
                        [--lpt<1-N> off|<I/O base> <IRQ>]
                        [--lptmode<1-N> <devicename>]
                        [--guestmemoryballoon <balloonsize in MB>]
                        [--audio none|null|dsound|solaudio|oss|
                                 oss|coreaudio]
                        [--audiocontroller ac97|hda|sb16]
                        [--clipboard disabled|hosttoguest|guesttohost|
                                     bidirectional]
                        [--draganddrop disabled|hosttoguest
                        [--vrde on|off]
                        [--vrdeextpack default|<name>
                        [--vrdeproperty <name=[value]>]
                        [--vrdeport <hostport>]
                        [--vrdeaddress <hostip>]
                        [--vrdeauthtype null|external|guest]
                        [--vrdeauthlibrary default|<name>
                        [--vrdemulticon on|off]
                        [--vrdereusecon on|off]
                        [--vrdevideochannel on|off]
                        [--vrdevideochannelquality <percent>]
                        [--usb on|off]
                        [--usbehci on|off]
                        [--snapshotfolder default|<path>]
                        [--teleporter on|off]
                        [--teleporterport <port>]
                        [--teleporteraddress <address|empty>
                        [--teleporterpassword <password>]
                        [--teleporterpasswordfile  <file>|stdin]
                        [--tracing-enabled on|off]
                        [--tracing-config <config-string>]
                        [--tracing-allow-vm-access on|off]
                        [--autostart-enabled on|off]
                        [--autostart-delay <seconds>]

clonevm                 <uuid>|<name>
                        [--snapshot <uuid>|<name>]
                        [--mode machine|machineandchildren|all]
                        [--options link|keepallmacs|keepnatmacs|
                                   keepdisknames]
                        [--name <name>]
                        [--groups <group>, ...]
                        [--basefolder <basefolder>]
                        [--uuid <uuid>]
                        [--register]

import                  <ovf/ova>
                        [--dry-run|-n]
                        [--options keepallmacs|keepnatmacs]
```

```
                        [more options]
                        (run with -n to have options displayed
                         for a particular OVF)

export                  <machines> --output|-o <name>.<ovf/ova>
                        [--legacy09|--ovf09|--ovf10|--ovf20]
                        [--manifest]
                        [--vsys <number of virtual system>]
                                [--product <product name>]
                                [--producturl <product url>]
                                [--vendor <vendor name>]
                                [--vendorurl <vendor url>]
                                [--version <version info>]
                                [--eula <license text>]
                                [--eulafile <filename>]

startvm                 <uuid>|<name>...
                        [--type gui|sdl|headless]

controlvm               <uuid>|<name>
                        pause|resume|reset|poweroff|savestate|
                        acpipowerbutton|acpisleepbutton|
                        keyboardputscancode <hex> [<hex> ...]|
                        setlinkstate<1-N> on|off |
                        nic<1-N> null|nat|bridged|intnet|generic
                                 [<devicename>] |
                        nictrace<1-N> on|off |
                        nictracefile<1-N> <filename> |
                        nicproperty<1-N> name=[value] |
                        nicpromisc<1-N> deny|allow-vms|allow-all |
                        natpf<1-N> [<rulename>],tcp|udp,[<hostip>],
                                    <hostport>,[<guestip>],<guestport> |
                        natpf<1-N> delete <rulename> |
                        guestmemoryballoon <balloonsize in MB> |
                        usbattach <uuid>|<address> |
                        usbdetach <uuid>|<address> |
                        clipboard disabled|hosttoguest|guesttohost|
                                  bidirectional |
                        draganddrop disabled|hosttoguest |
                        vrde on|off |
                        vrdeport <port> |
                        vrdeproperty <name=[value]> |
                        vrdevideochannelquality <percent> |
                        setvideomodehint <xres> <yres> <bpp>
                                         [[<display>] [<enabled:yes|no> |
                                           [<xorigin> <yorigin>]]] |
                        screenshotpng <file> [display] |
                        setcredentials <username>
                                       --passwordfile <file> | <password>
                                       <domain>
                                       [--allowlocallogon <yes|no>] |
                        teleport --host <name> --port <port>
                                 [--maxdowntime <msec>]
                                 [--passwordfile <file> |
                                  --password <password>] |
                        plugcpu <id> |
                        unplugcpu <id> |
                        cpuexecutioncap <1-100>

discardstate            <uuid>|<name>

adoptstate              <uuid>|<name> <state_file>

snapshot                <uuid>|<name>
                        take <name> [--description <desc>] [--pause] |
                        delete <uuid>|<name> |
                        restore <uuid>|<name> |
```

```
                        restorecurrent |
                        edit <uuid>|<name>|--current
                             [--name <name>]
                             [--description <desc>] |
                        list [--details|--machinereadable]
                        showvminfo <uuid>|<name>

closemedium             disk|dvd|floppy <uuid>|<filename>
                        [--delete]

storageattach           <uuid|vmname>
                        --storagectl <name>
                        [--port <number>]
                        [--device <number>]
                        [--type dvddrive|hdd|fdd]
                        [--medium none|emptydrive|additions|
                                  <uuid>|<filename>|host:<drive>|iscsi]
                        [--mtype normal|writethrough|immutable|shareable|
                                 readonly|multiattach]
                        [--comment <text>]
                        [--setuuid <uuid>]
                        [--setparentuuid <uuid>]
                        [--passthrough on|off]
                        [--tempeject on|off]
                        [--nonrotational on|off]
                        [--discard on|off]
                        [--bandwidthgroup <name>]
                        [--forceunmount]
                        [--server <name>|<ip>]
                        [--target <target>]
                        [--tport <port>]
                        [--lun <lun>]
                        [--encodedlun <lun>]
                        [--username <username>]
                        [--password <password>]
                        [--initiator <initiator>]
                        [--intnet]

storagectl              <uuid|vmname>
                        --name <name>
                        [--add ide|sata|scsi|floppy|sas]
                        [--controller LSILogic|LSILogicSAS|BusLogic|
                                      IntelAHCI|PIIX3|PIIX4|ICH6|I82078]
                        [--sataportcount <1-30>]
                        [--hostiocache on|off]
                        [--bootable on|off]
                        [--remove]

bandwidthctl            <uuid|vmname>
                        add <name> --type disk|network
                            --limit <megabytes per second>[k|m|g|K|M|G] |
                        set <name>
                            --limit <megabytes per second>[k|m|g|K|M|G] |
                        remove <name> |
                        list [--machinereadable]
                        (limit units: k=kilobit, m=megabit, g=gigabit,
                                      K=kilobyte, M=megabyte, G=gigabyte)

showhdinfo              <uuid>|<filename>

createhd                --filename <filename>
                        [--size <megabytes>|--sizebyte <bytes>]
                        [--diffparent <uuid>|<filename>
                        [--format VDI|VMDK|VHD] (default: VDI)
                        [--variant Standard,Fixed,Split2G,Stream,ESX]

modifyhd                <uuid>|<filename>
```

```
                        [--type normal|writethrough|immutable|shareable|
                               readonly|multiattach]
                        [--autoreset on|off]
                        [--compact]
                        [--resize <megabytes>|--resizebyte <bytes>]

clonehd                 <uuid>|<filename> <uuid>|<outputfile>
                        [--format VDI|VMDK|VHD|RAW|<other>]
                        [--variant Standard,Fixed,Split2G,Stream,ESX]
                        [--existing]

convertfromraw          <filename> <outputfile>
                        [--format VDI|VMDK|VHD]
                        [--variant Standard,Fixed,Split2G,Stream,ESX]
                        [--uuid <uuid>]
convertfromraw          stdin <outputfile> <bytes>
                        [--format VDI|VMDK|VHD]
                        [--variant Standard,Fixed,Split2G,Stream,ESX]
                        [--uuid <uuid>]

getextradata            global|<uuid>|<name>
                        <key>|enumerate

setextradata            global|<uuid>|<name>
                        <key>
                        [<value>] (no value deletes key)

setproperty             machinefolder default|<folder> |
                        vrdeauthlibrary default|<library> |
                        websrvauthlibrary default|null|<library> |
                        vrdeextpack null|<library> |
                        autostartdbpath null|<folder> |
                        loghistorycount <value>

usbfilter               add <index,0-N>
                        --target <uuid>|<name>|global
                        --name <string>
                        --action ignore|hold (global filters only)
                        [--active yes|no] (yes)
                        [--vendorid <XXXX>] (null)
                        [--productid <XXXX>] (null)
                        [--revision <IIFF>] (null)
                        [--manufacturer <string>] (null)
                        [--product <string>] (null)
                        [--remote yes|no] (null, VM filters only)
                        [--serialnumber <string>] (null)
                        [--maskedinterfaces <XXXXXXXX>]

usbfilter               modify <index,0-N>
                        --target <uuid>|<name>|global
                        [--name <string>]
                        [--action ignore|hold] (global filters only)
                        [--active yes|no]
                        [--vendorid <XXXX>|""]
                        [--productid <XXXX>|""]
                        [--revision <IIFF>|""]
                        [--manufacturer <string>|""]
                        [--product <string>|""]
                        [--remote yes|no] (null, VM filters only)
                        [--serialnumber <string>|""]
                        [--maskedinterfaces <XXXXXXXX>]

usbfilter               remove <index,0-N>
                        --target <uuid>|<name>|global

sharedfolder            add <vmname>|<uuid>
                        --name <name> --hostpath <hostpath>
```

```
                        [--transient] [--readonly] [--automount]

sharedfolder            remove <vmname>|<uuid>
                        --name <name> [--transient]

debugvm                 <uuid>|<name>
                        dumpguestcore --filename <name> |
                        info <item> [args] |
                        injectnmi |
                        log [--release|--debug] <settings> ...|
                        logdest [--release|--debug] <settings> ...|
                        logflags [--release|--debug] <settings> ...|
                        osdetect |
                        osinfo |
                        getregisters [--cpu <id>] <reg>|all ... |
                        setregisters [--cpu <id>] <reg>=<value> ... |
                        show [--human-readable|--sh-export|--sh-eval|
                              --cmd-set]
                            <logdbg-settings|logrel-settings>
                            [[opt] what ...] |
                        statistics [--reset] [--pattern <pattern>]
                        [--descriptions]

metrics                 list [*|host|<vmname> [<metric_list>]]
                                              (comma-separated)

metrics                 setup
                        [--period <seconds>] (default: 1)
                        [--samples <count>] (default: 1)
                        [--list]
                        [*|host|<vmname> [<metric_list>]]

metrics                 query [*|host|<vmname> [<metric_list>]]

metrics                 enable
                        [--list]
                        [*|host|<vmname> [<metric_list>]]

metrics                 disable
                        [--list]
                        [*|host|<vmname> [<metric_list>]]

metrics                 collect
                        [--period <seconds>] (default: 1)
                        [--samples <count>] (default: 1)
                        [--list]
                        [--detach]
                        [*|host|<vmname> [<metric_list>]]

dhcpserver              add|modify --netname <network_name> |
                        [--ip <ip_address>
                        --netmask <network_mask>
                        --lowerip <lower_ip>
                        --upperip <upper_ip>]
                        [--enable | --disable]

dhcpserver              remove --netname <network_name> |

extpack                 install [--replace] <tarball> |
                        uninstall [--force] <name> |
                        cleanup
```

Each time VBoxManage is invoked, only one command can be executed. However, a command might support several subcommands which then can be invoked in one single call. The following sections provide detailed reference information on the different commands.

8.3 VBoxManage list

The `list` command gives relevant information about your system and information about VirtualBox's current settings.

The following subcommands are available with `VBoxManage list`:

- `vms` lists all virtual machines currently registered with VirtualBox. By default this displays a compact list with each VM's name and UUID; if you also specify `--long` or `-l`, this will be a detailed list as with the `showvminfo` command (see below).
- `runningvms` lists all currently running virtual machines by their unique identifiers (UUIDs) in the same format as with `vms`.
- `ostypes` lists all guest operating systems presently known to VirtualBox, along with the identifiers used to refer to them with the `modifyvm` command.
- `hostdvds`, `hostfloppies`, respectively, list DVD, floppy, bridged networking and host-only networking interfaces on the host, along with the name used to access them from within VirtualBox.
- `bridgedifs`, `hostonlyifs` and `dhcpservers`, respectively, list bridged network interfaces, host-only network interfaces and DHCP servers currently available on the host. Please see chapitre 6, *Réseau virtuel*, page 91 for details on these.
- `hostinfo` displays information about the host system, such as CPUs, memory size and operating system version.
- `hostcpuids` dumps the CPUID parameters for the host CPUs. This can be used for a more fine grained analyis of the host's virtualization capabilities.
- `hddbackends` lists all known virtual disk back-ends of VirtualBox. For each such format (such as VDI, VMDK or RAW), this lists the back-end's capabilities and configuration.
- `hdds`, `dvds` and `floppies` all give you information about virtual disk images currently in use by VirtualBox, including all their settings, the unique identifiers (UUIDs) associated with them by VirtualBox and all files associated with them. This is the command-line equivalent of the Virtual Media Manager; see chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82.
- `usbhost` supplies information about USB devices attached to the host, notably information useful for constructing USB filters and whether they are currently in use by the host.
- `usbfilters` lists all global USB filters registered with VirtualBox – that is, filters for devices which are accessible to all virtual machines – and displays the filter parameters.
- `systemproperties` displays some global VirtualBox settings, such as minimum and maximum guest RAM and virtual hard disk size, folder settings and the current authentication library in use.
- `extpacks` displays all VirtualBox extension packs currently installed; see chapitre 1.5, *Installer et lancer VirtualBox*, page 15 and chapitre 8.35, *VBoxManage extpack*, page 147 for more information.

8.4 VBoxManage showvminfo

The `showvminfo` command shows information about a particular virtual machine. This is the same information as `VBoxManage list vms --long` would show for all virtual machines.

You will get information similar to the following:

```
$ VBoxManage showvminfo "Windows XP"
VirtualBox Command Line Management Interface Version 4.2.12
(C) 2005-2013 Oracle Corporation
All rights reserved.

Name:            Windows XP
Guest OS:        Other/Unknown
UUID:            1bf3464d-57c6-4d49-92a9-a5cc3816b7e7
Config file:     /home/username/.VirtualBox/Machines/Windows XP/Windows XP.xml
Memory size:     512MB
VRAM size:       12MB
Number of CPUs:  2
Synthetic Cpu:   off
Boot menu mode:  message and menu
Boot Device (1): DVD
Boot Device (2): HardDisk
Boot Device (3): Not Assigned
Boot Device (4): Not Assigned
ACPI:            on
IOAPIC:          on
PAE:             on
Time offset:     0 ms
Hardw. virt.ext: on
Hardw. virt.ext exclusive: on
Nested Paging:   on
VT-x VPID:       off
State:           powered off (since 2009-10-20T14:52:19.000000000)
Monitor count:   1
3D Acceleration: off
2D Video Acceleration: off
Teleporter Enabled: off
Teleporter Port: 0
Teleporter Address:
Teleporter Password:
Storage Controller      (0): IDE Controller
Storage Controller Type (0): PIIX4
Storage Controller      (1): Floppy Controller 1
Storage Controller Type (1): I82078
IDE Controller (0, 0): /home/user/windows.vdi (UUID: 46f6e53a-4557-460a-9b95-68b0f17d744b)
IDE Controller (0, 1): /home/user/openbsd-cd46.iso (UUID: 4335e162-59d3-4512-91d5-b63e94eebe0b)
Floppy Controller 1 (0, 0): /home/user/floppy.img (UUID: 62ac6ccb-df36-42f2-972e-22f836368137)
NIC 1:           disabled
NIC 2:           disabled
NIC 3:           disabled
NIC 4:           disabled
NIC 5:           disabled
NIC 6:           disabled
NIC 7:           disabled
NIC 8:           disabled
UART 1:          disabled
UART 2:          disabled
Audio:           disabled (Driver: Unknown)
Clipboard Mode:  Bidirectional
VRDE:            disabled
USB:             disabled

USB Device Filters:
<none>

Shared folders:
<none>
```

```
Statistics update:  disabled
```

8.5 VBoxManage registervm / unregistervm

The `registervm` command allows you to import a virtual machine definition in an XML file into VirtualBox. The machine must not conflict with one already registered in VirtualBox and it may not have any hard or removable disks attached. It is advisable to place the definition file in the machines folder before registering it.

> **Note:** When creating a new virtual machine with `VBoxManage createvm` (see below), you can directly specify the `--register` option to avoid having to register it separately.

The `unregistervm` command unregisters a virtual machine. If `--delete` is also specified, the following files will automatically be deleted as well:

1. all hard disk image files, including differencing files, which are used by the machine and not shared with other machines;
2. saved state files that the machine created, if any (one if the machine was in "saved" state and one for each online snapshot);
3. the machine XML file and its backups;
4. the machine log files, if any;
5. the machine directory, if it is empty after having deleted all the above.

8.6 VBoxManage createvm

This command creates a new XML virtual machine definition file.

The `--name <name>` parameter is required and must specify the name of the machine. Since this name is used by default as the file name of the settings file (with the extension `.xml`) and the machine folder (a subfolder of the `.VirtualBox/Machines` folder), it must conform to your host operating system's requirements for file name specifications. If the VM is later renamed, the file and folder names will change automatically.

However, if the `--basefolder <path>` option is used, the machine folder will be named `<path>`. In this case, the names of the file and the folder will not change if the virtual machine is renamed.

By default, this command only creates the XML file without automatically registering the VM with your VirtualBox installation. To register the VM instantly, use the optional `--register` option, or run `VBoxManage registervm` separately afterwards.

8.7 VBoxManage modifyvm

This command changes the properties of a registered virtual machine which is not running. Most of the properties that this command makes available correspond to the VM settings that VirtualBox graphical user interface displays in each VM's "Settings" dialog; these were described in chapitre 3, *Configurer des machines virtuelles*, page 43. Some of the more advanced settings, however, are only available through the `VBoxManage` interface.

These commands require that the machine is powered off (neither running nor in “saved” state). Some machine settings can also be changed while a machine is running; those settings will then have a corresponding subcommand with the `VBoxManage controlvm` subcommand (see chapitre 8.12, *VBoxManage controlvm*, page 129).

8.7.1 General settings

The following general settings are available through `VBoxManage modifyvm`:

- `--name <name>`: This changes the VM's name and possibly renames the internal virtual machine files, as described with `VBoxManage createvm` above.
- `--ostype <ostype>`: This specifies what guest operating system is supposed to run in the VM. To learn about the various identifiers that can be used here, use `VBoxManage list ostypes`.
- `--memory <memorysize>`: This sets the amount of RAM, in MB, that the virtual machine should allocate for itself from the host. See the remarks in chapitre 1.6, *Créer votre première machine virtuelle*, page 17 for more information.
- `--vram <vramsize>`: This sets the amount of RAM that the virtual graphics card should have. See chapitre 3.5, *Paramètres d'affichage*, page 50 for details.
- `--acpi on|off`; `--ioapic on|off`: These two determine whether the VM should have ACPI and I/O APIC support, respectively; see chapitre 3.4.1, *Onglet Carte mère*, page 47 for details.
- `--hardwareuuid <uuid>`: The UUID presented to the guest via memory tables (DMI/SMBIOS), hardware and guest properties. By default this is the same as the VM uuid. Useful when cloning a VM. Teleporting takes care of this automatically.
- `--cpus <cpucount>`: This sets the number of virtual CPUs for the virtual machine (see chapitre 3.4.2, *Onglet processeur*, page 49). If CPU hot-plugging is enabled (see below), this then sets the *maximum* number of virtual CPUs that can be plugged into the virtual machines.
- `--rtcuseutc on|off`: This option lets the real-time clock (RTC) operate in UTC time (see chapitre 3.4.1, *Onglet Carte mère*, page 47).
- `--cpuhotplug on|off`: This enables CPU hot-plugging. When enabled, virtual CPUs can be added to and removed from a virtual machine while it is running. See chapitre 9.5, *CPU hot-plugging*, page 154 for more information.
- `--plugcpu|unplugcpu <id>`: If CPU hot-plugging is enabled (see above), this adds a virtual CPU to the virtual machines (or removes one). `<id>` specifies the index of the virtual CPU to be added or removed and must be a number from 0 to the maximum no. of CPUs configured with the `--cpus` option. CPU 0 can never be removed.
- `--cpuexecutioncap <1-100>`: This setting controls how much cpu time a virtual CPU can use. A value of 50 implies a single virtual CPU can use up to 50% of a single host CPU.
- `--synthcpu on|off`: This setting determines whether VirtualBox will expose a synthetic CPU to the guest to allow live migration between host systems that differ significantly.
- `--pae on|off`: This enables/disables PAE (see chapitre 3.4.2, *Onglet processeur*, page 49).

- `--hpet on|off`: This enables/disables a High Precision Event Timer (HPET) which can replace the legacy system timers. This is turned off by default. Note that Windows supports a HPET only from Vista onwards.

- `--hwvirtex on|off`: This enables or disables the use of hardware virtualization extensions (Intel VT-x or AMD-V) in the processor of your host system; see chapitre 10.3, *Hardware vs. software virtualization*, page 175.

- `--hwvirtexexcl on|off`: This specifies whether VirtualBox will make exclusive use of the hardware virtualization extensions (Intel VT-x or AMD-V) in the processor of your host system; see chapitre 10.3, *Hardware vs. software virtualization*, page 175. If you wish to simultaneously share these extensions with other hypervisors, then you must disable this setting. Doing so has negative performance implications.

- `--nestedpaging on|off`: If hardware virtualization is enabled, this additional setting enables or disables the use of the nested paging feature in the processor of your host system; see chapitre 10.3, *Hardware vs. software virtualization*, page 175.

- `--largepages on|off`: If hardware virtualization *and* nested paging are enabled, for Intel VT-x only, an additional performance improvement of up to 5% can be obtained by enabling this setting. This causes the hypervisor to use large pages to reduce TLB use and overhead.

- `--vtxvpid on|off`: If hardware virtualization is enabled, for Intel VT-x only, this additional setting enables or disables the use of the tagged TLB (VPID) feature in the processor of your host system; see chapitre 10.3, *Hardware vs. software virtualization*, page 175.

- `--accelerate3d on|off`: This enables, if the Guest Additions are installed, whether hardware 3D acceleration should be available; see chapitre 4.5.1, *Accélération 3D matérielle (OpenGL et Direct3D 8/9)*, page 72.

- You can influence the BIOS logo that is displayed when a virtual machine starts up with a number of settings. Per default, a VirtualBox logo is displayed.

  With `--bioslogofadein on|off` and `--bioslogofadeout on|off`, you can determine whether the logo should fade in and out, respectively.

  With `--bioslogodisplaytime <msec>` you can set how long the logo should be visible, in milliseconds.

  With `--bioslogoimagepath <imagepath>` you can, if you are so inclined, replace the image that is shown, with your own logo. The image must be an uncompressed 256 color BMP file.

- `--biosbootmenu disabled|menuonly|messageandmenu`: This specifies whether the BIOS allows the user to select a temporary boot device. `menuonly` suppresses the message, but the user can still press F12 to select a temporary boot device.

- `--boot<1-4> none|floppy|dvd|disk|net`: This specifies the boot order for the virtual machine. There are four "slots", which the VM will try to access from 1 to 4, and for each of which you can set a device that the VM should attempt to boot from.

- `--snapshotfolder default|<path>`: This allows you to specify the folder in which snapshots will be kept for a virtual machine.

- `--firmware efi|bios`: Specifies which firmware is used to boot particular virtual machine: EFI or BIOS. Use EFI only if your fully understand what you're doing.

- `--guestmemoryballoon <size>` sets the default size of the guest memory balloon, that is, memory allocated by the VirtualBox Guest Additions from the guest operating system and returned to the hypervisor for re-use by other virtual machines. <size> must be specified in megabytes. The default size is 0 megabytes. For details, see chapitre 4.8, *Faire de la montgolfière avec la mémoire*, page 75.

8.7.2 Networking settings

The following networking settings are available through `VBoxManage modifyvm`. With all these settings, the decimal number directly following the option name ("1-N" in the list below) specifies the virtual network adapter whose settings should be changed.

- `--nic<1-N> none|null|nat|bridged|intnet|hostonly|generic` : With this, you can set, for each of the VM's virtual network cards, what type of networking should be available. They can be not present (`none`), not connected to the host (`null`), use network address translation (`nat`), bridged networking (`bridged`) or communicate with other virtual machines using internal networking (`intnet`), host-only networking (`hostonly`), or access rarely used sub-modes (`generic`). These options correspond to the modes which are described in detail in chapitre 6.2, *Introduction aux modes réseaux*, page 92.

- `--nictype<1-N> Am79C970A|Am79C973|82540EM|82543GC|82545EM|virtio`: This allows you, for each of the VM's virtual network cards, to specify which networking hardware VirtualBox presents to the guest; see chapitre 6.1, *Matériel de réseau virtuel*, page 91.

- `--cableconnected<1-N> on|off`: This allows you to temporarily disconnect a virtual network interface, as if a network cable had been pulled from a real network card. This might be useful for resetting certain software components in the VM.

- With the "nictrace" options, you can optionally trace network traffic by dumping it to a file, for debugging purposes.

  With `--nictrace<1-N> on|off`, you can enable network tracing for a particular virtual network card.

  If enabled, you must specify with `--nictracefile<1-N> <filename>` what file the trace should be logged to.

- `--bridgeadapter<1-N> none|<devicename>`: If bridged networking has been enabled for a virtual network card (see the `--nic` option above; otherwise this setting has no effect), use this option to specify which host interface the given virtual network interface will use. For details, please see chapitre 6.4, *Réseau bridgé*, page 95.

- `--hostonlyadapter<1-N> none|<devicename>`: If host-only networking has been enabled for a virtual network card (see the –nic option above; otherwise this setting has no effect), use this option to specify which host-only networking interface the given virtual network interface will use. For details, please see chapitre 6.6, *Réseau privé avec l'hôte (Host-only)*, page 97.

- `--intnet<1-N> network`: If internal networking has been enabled for a virtual network card (see the `--nic` option above; otherwise this setting has no effect), use this option to specify the name of the internal network (see chapitre 6.5, *Réseau interne*, page 96).

- `--macaddress<1-N> auto|<mac>`: With this option you can set the MAC address of the virtual network card. Normally, each virtual network card is assigned a random address by VirtualBox at VM creation.

- `--nicgenericdrv<1-N> <backend driver>`: If generic networking has been enabled for a virtual network card (see the `--nic` option above; otherwise this setting has no effect), this mode allows you to access rarely used networking sub-modes, such as VDE network or UDP Tunnel.

- `--nicproperty<1-N> <paramname>="paramvalue"`: This option, in combination with "nicgenericdrv" allows you to pass parameters to rarely-used network backends.

  Those parameters are backend engine-specific, and are different between UDP Tunnel and the VDE backend drivers. For example, please see chapitre 6.7, *Réseau par tunnel UDP*, page 98.

8.7.2.1 NAT Networking settings.

The following NAT networking settings are available through `VBoxManage modifyvm`. With all these settings, the decimal number directly following the option name ("1-N" in the list below) specifies the virtual network adapter whose settings should be changed.

- `--natpf<1-N> [<name>],tcp|udp,[<hostip>],<hostport>,[<guestip>],<guestport>`: This option defines a NAT port-forwarding rule (please see chapitre 6.3.1, *Configurer la redirection de ports avec NAT*, page 93 for details).

- `--natpf<1-N> delete <name>`: This option deletes a NAT port-forwarding rule (please see chapitre 6.3.1, *Configurer la redirection de ports avec NAT*, page 93 for details).

- `--nattftpprefix<1-N> <prefix>`: This option defines a prefix for the built-in TFTP server, i.e. where the boot file is located (please see chapitre 6.3.2, *Amorçage PXE avec NAT*, page 94 and chapitre 9.11.2, *Configuring the boot server (next server) of a NAT network interface*, page 162 for details).

- `--nattftpfile<1-N> <bootfile>`: This option defines the TFT boot file (please see chapitre 9.11.2, *Configuring the boot server (next server) of a NAT network interface*, page 162 for details).

- `--nattftpserver<1-N> <tftpserver>`: This option defines the TFTP server address to boot from (please see chapitre 9.11.2, *Configuring the boot server (next server) of a NAT network interface*, page 162 for details).

- `--natdnspassdomain<1-N> on|off`: This option specifies whether the built-in DHCP server passes the domain name for network name resolution.

- `--natdnsproxy<1-N> on|off`: This option makes the NAT engine proxy all guest DNS requests to the host's DNS servers (please see chapitre 9.11.5, *Enabling DNS proxy in NAT mode*, page 163 for details).

- `--natdnshostresolver<1-N> on|off`: This option makes the NAT engine use the host's resolver mechanisms to handle DNS requests (please see chapitre 9.11.5, *Enabling DNS proxy in NAT mode*, page 163 for details).

- `--natnatsettings<1-N> [<mtu>],[<socksnd>],[<sockrcv>],[<tcpsnd>],[<tcprcv>]`: This option controls several NAT settings (please see chapitre 9.11.3, *Tuning TCP/IP buffers for NAT*, page 162 for details).

- `--nataliasmode<1-N> default|[log],[proxyonly],[sameports]`: This option defines behaviour of NAT engine core: log - enables logging, proxyonly - switches of aliasing mode makes NAT transparent, sameports enforces NAT engine to send packets via the same port as they originated on, default - disable all mentioned modes above . (please see chapitre 9.11.7, *Configuring aliasing of the NAT engine*, page 163 for details).

8.7.3 Serial port, audio, clipboard, remote desktop and USB settings

The following other hardware settings are available through `VBoxManage modifyvm`:

- `--uart<1-N> off|<I/O base> <IRQ>`: With this option you can configure virtual serial ports for the VM; see chapitre 3.9, *Ports série*, page 53 for an introduction.
- `--uartmode<1-N> <arg>`: This setting controls how VirtualBox connects a given virtual serial port (previously configured with the `--uartX` setting, see above) to the host on which the virtual machine is running. As described in detail in chapitre 3.9, *Ports série*, page 53, for each such port, you can specify `<arg>` as one of the following options:
  - `disconnected`: Even though the serial port is shown to the guest, it has no "other end" – like a real COM port without a cable.
  - `server <pipename>`: On a Windows host, this tells VirtualBox to create a named pipe on the host named `<pipename>` and connect the virtual serial device to it. Note that Windows requires that the name of a named pipe begin with `\\.\pipe\`.

    On a Linux host, instead of a named pipe, a local domain socket is used.
  - `client <pipename>`: This operates just like `server ...`, except that the pipe (or local domain socket) is not created by VirtualBox, but assumed to exist already.
  - `<devicename>`: If, instead of the above, the device name of a physical hardware serial port of the host is specified, the virtual serial port is connected to that hardware port. On a Windows host, the device name will be a COM port such as `COM1`; on a Linux host, the device name will look like `/dev/ttyS0`. This allows you to "wire" a real serial port to a virtual machine.
- `--audio none|null|oss`: With this option, you can set whether the VM should have audio support.
- `--clipboard disabled|hosttoguest|guesttohost|bidirectional`: With this setting, you can select whether the guest operating system's clipboard should be shared with the host; see chapitre 3.3, *Paramètres généraux*, page 46. This requires that the Guest Additions be installed in the virtual machine.
- `--monitorcount <count>`: This enables multi-monitor support; see chapitre 3.5, *Paramètres d'affichage*, page 50.
- `--usb on|off`: This option enables or disables the VM's virtual USB controller; see chapitre 3.10.1, *Paramètres USB*, page 54 for details.
- `--usbehci on|off`: This option enables or disables the VM's virtual USB 2.0 controller; see chapitre 3.10.1, *Paramètres USB*, page 54 for details.

8.7.4 Remote machine settings

The following settings that affect remote machine behavior are available through `VBoxManage modifyvm`:

- `--vrde on|off`: With the VirtualBox graphical user interface, this enables or disables the VirtualBox remote desktop extension (VRDE) server. Note that if you are using `VBoxHeadless` (see chapitre 7.1.2, *VBoxHeadless, the remote desktop server*, page 102), VRDE is enabled by default.
- `--vrdeport default|<ports>`: A port or a range of ports the VRDE server can bind to; "default" or "0" means port 3389, the standard port for RDP. You can specify a comma-separated list of ports or ranges of ports. Use a dash between two port numbers to specify

a range. The VRDE server will bind to **one** of available ports from the specified list. Only one machine can use a given port at a time. For example, the option `--vrdeport 5000,5010-5012` will tell the server to bind to one of following ports: 5000, 5010, 5011 or 5012.

- `--vrdeaddress <IP address>`: The IP address of the host network interface the VRDE server will bind to. If specified, the server will accept connections only on the specified host network interface.
- `--vrdeauthtype null|external|guest`: This allows you to choose whether and how authorization will be performed; see chapitre 7.1.5, *RDP authentication*, page 105 for details.
- `--vrdemulticon on|off`: This enables multiple connections to the same VRDE server, if the server supports this feature; see chapitre 7.1.7, *Multiple connections to the VRDP server*, page 107.
- `--vrdereusecon on|off`: This specifies the VRDE server behavior when multiple connections are disabled. When this option is enabled, the server will allow a new client to connect and will drop the existing connection. When this option is disabled (this is the default setting), a new connection will not be accepted if there is already a client connected to the server.
- `--vrdevideochannel on|off`: This enables video redirection, if it is supported by the VRDE server; see chapitre 7.1.9, *VRDP video redirection*, page 108.
- `--vrdevideochannelquality <percent>`: Sets the image quality for video redirection; see chapitre 7.1.9, *VRDP video redirection*, page 108.

8.7.5 Teleporting settings

With the following commands for `VBoxManage modifyvm` you can configure a machine to be a target for teleporting. See chapitre 7.2, *Teleporting*, page 108 for an introduction.

- `--teleporter on|off`: With this setting you turn on or off whether a machine waits for a teleporting request to come in on the network when it is started. If “on”, when the machine is started, it does not boot the virtual machine as it would normally; instead, it then waits for a teleporting request to come in on the port and address listed with the next two parameters.
- `--teleporterport <port>`, `--teleporteraddress <address>`: these must be used with –teleporter and tell the virtual machine on which port and address it should listen for a teleporting request from another virtual machine. `<port>` can be any free TCP/IP port number (e.g. 6000); `<address>` can be any IP address or hostname and specifies the TCP/IP socket to bind to. The default is “0.0.0.0”, which means any address.
- `--teleporterpassword <password>`: if this optional argument is given, then the teleporting request will only succeed if the source machine specifies the same password as the one given with this command.

> **Note:** Currently, the password is stored without encryption (i.e. in clear text) in the XML machine configuration file.

- `--cpuid <leaf> <eax> <ebx> <ecx> <edx>`: Advanced users can use this command before a teleporting operation to restrict the virtual CPU capabilities that VirtualBox presents to the guest operating system. This must be run on both the source and the target machines involved in the teleporting and will then modify what the guest sees when it executes the `CPUID` machine instruction. This might help with misbehaving applications that wrongly assume that certain CPU capabilities are present. The meaning of the parameters is hardware dependent; please refer to the AMD or Intel processor manuals.

8.8 VBoxManage clonevm

This command creates a full or linked copy of an existing virtual machine.

The `clonevm` subcommand takes at least the name of the virtual machine which should be cloned. The following additional settings can be used to further configure the clone VM operation:

- `--snapshot <uuid>|<name>`: Select a specific snapshot where the clone operation should refer to. Default is referring to the current state.
- `--mode machine|machineandchildren|all`: Selects the cloning mode of the operation. If `machine` is selected (the default), the current state of the VM without any snapshots is cloned. In the `machineandchildren` mode the snapshot provided by `--snapshot` and all child snapshots are cloned. If `all` is the selected mode all snapshots and the current state are cloned.
- `--options link|keepallmacs|keepnatmacs|keepdisknames`: Allows additional fine tuning of the clone operation. The first option defines that a linked clone should be created, which is only possible for a machine clone from a snapshot. The next two options allow to define how the MAC addresses of every virtual network card should be handled. They can either be reinitialized (the default), left unchanged (`keepallmacs`) or left unchanged when the network type is NAT (`keepnatmacs`). If you add `keepdisknames` all new disk images are called like the original once, otherwise they are renamed.
- `--name <name>`: Select a new name for the new virtual machine. Default is “Original Name Clone”.
- `--basefolder <basefolder>`: Select the folder where the new virtual machine configuration should be saved in.
- `--uuid <uuid>`: Select the UUID the new VM should have. This id has to be unique in the VirtualBox instance this clone should be registered. Default is creating a new UUID.
- `--register`: Automatically register the new clone in this VirtualBox installation. If you manually want register the new VM later, see chapitre 8.5, *VBoxManage registervm / unregistervm*, page 120 for instructions how to do so.

8.9 VBoxManage import

This command imports a virtual appliance in OVF format by copying the virtual disk images and creating virtual machines in VirtualBox. See chapitre 1.12, *Importer et exporter des machines virtuelles*, page 29 for an introduction to appliances.

The `import` subcommand takes at least the path name of an OVF file as input and expects the disk images, if needed, in the same directory as the OVF file. A lot of additional command-line options are supported to control in detail what is being imported and modify the import parameters, but the details depend on the content of the OVF file.

It is therefore recommended to first run the import subcommand with the `--dry-run` or `-n` option. This will then print a description of the appliance's contents to the screen how it would be imported into VirtualBox, together with the optional command-line options to influence the import behavior.

As an example, here is the screen output with a sample appliance containing a Windows XP guest:

```
VBoxManage import WindowsXp.ovf --dry-run
Interpreting WindowsXp.ovf...
OK.
Virtual system 0:
 0: Suggested OS type: "WindowsXP"
    (change with "--vsys 0 --ostype <type>"; use "list ostypes" to list all)
 1: Suggested VM name "Windows XP Professional_1"
    (change with "--vsys 0 --vmname <name>")
 3: Number of CPUs: 1
    (change with "--vsys 0 --cpus <n>")
 4: Guest memory: 956 MB (change with "--vsys 0 --memory <MB>")
 5: Sound card (appliance expects "ensoniq1371", can change on import)
    (disable with "--vsys 0 --unit 5 --ignore")
 6: USB controller
    (disable with "--vsys 0 --unit 6 --ignore")
 7: Network adapter: orig bridged, config 2, extra type=bridged
 8: Floppy
    (disable with "--vsys 0 --unit 8 --ignore")
 9: SCSI controller, type BusLogic
    (change with "--vsys 0 --unit 9 --scsitype {BusLogic|LsiLogic}";
    disable with "--vsys 0 --unit 9 --ignore")
10: IDE controller, type PIIX4
    (disable with "--vsys 0 --unit 10 --ignore")
11: Hard disk image: source image=WindowsXp.vmdk,
      target path=/home/user/disks/WindowsXp.vmdk, controller=9;channel=0
    (change controller with "--vsys 0 --unit 11 --controller <id>";
    disable with "--vsys 0 --unit 11 --ignore")
```

As you can see, the individual configuration items are numbered, and depending on their type support different command-line options. The import subcommand can be directed to ignore many such items with a `--vsys X --unit Y --ignore` option, where X is the number of the virtual system (zero unless there are several virtual system descriptions in the appliance) and Y the item number, as printed on the screen.

In the above example, Item #1 specifies the name of the target machine in VirtualBox. Items #9 and #10 specify hard disk controllers, respectively. Item #11 describes a hard disk image; in this case, the additional `--controller` option indicates which item the disk image should be connected to, with the default coming from the OVF file.

You can combine several items for the same virtual system behind the same `--vsys` option. For example, to import a machine as described in the OVF, but without the sound card and without the USB controller, and with the disk image connected to the IDE controller instead of the SCSI controller, use this:

```
VBoxManage import WindowsXp.ovf
      --vsys 0 --unit 5 --ignore --unit 6 --ignore --unit 11 --controller 10
```

8.10 VBoxManage export

This command exports one or more virtual machines from VirtualBox into a virtual appliance in OVF format, including copying their virtual disk images to compressed VMDK. See chapitre 1.12, *Importer et exporter des machines virtuelles*, page 29 for an introduction to appliances.

The `export` command is simple to use: list the machine (or the machines) that you would like to export to the same OVF file and specify the target OVF file after an additional `--output` or `-o` option. Note that the directory of the target OVF file will also receive the exported disk images in

the compressed VMDK format (regardless of the original format) and should have enough disk space left for them.

Beside a simple export of a given virtual machine, you can append several product information to the appliance file. Use `--product`, `--producturl`, `--vendor`, `--vendorurl` and `--version` to specify this additional information. For legal reasons you may add a license text or the content of a license file by using the `--eula` and `--eulafile` option respectively. As with OVF import, you must use the `--vsys X` option to direct the previously mentioned options to the correct virtual machine.

For virtualization products which aren't fully compatible with the OVF standard 1.0 you can enable a OVF 0.9 legacy mode with the `--legacy09` option.

8.11 VBoxManage startvm

This command starts a virtual machine that is currently in the "Powered off" or "Saved" states.

> **Note:** This is provided for backwards compatibility only. We recommend to start virtual machines directly by running the respective front-end, as you might otherwise miss important error and state information that VirtualBox may display on the console. This is especially important for front-ends other than `VirtualBox`, our graphical user interface, because those cannot display error messages in a popup window. See chapitre 7.1.2, *VBoxHeadless, the remote desktop server*, page 102 for more information.

The optional `--type` specifier determines whether the machine will be started in a window (GUI mode, which is the default) or whether the output should go through `VBoxHeadless`, with VRDE enabled or not; see chapitre 7.1.2, *VBoxHeadless, the remote desktop server*, page 102 for more information. The list of types is subject to change, and it's not guaranteed that all types are accepted by any product variant.

The following values are allowed:

gui Starts a VM showing a GUI window. This is the default.

headless Starts a VM without a window for remote display only.

8.12 VBoxManage controlvm

The `controlvm` subcommand allows you to change the state of a virtual machine that is currently running. The following can be specified:

- `VBoxManage controlvm <vm> pause` temporarily puts a virtual machine on hold, without changing its state for good. The VM window will be painted in gray to indicate that the VM is currently paused. (This is equivalent to selecting the "Pause" item in the "Machine" menu of the GUI.)
- Use `VBoxManage controlvm <vm> resume` to undo a previous `pause` command. (This is equivalent to selecting the "Resume" item in the "Machine" menu of the GUI.)
- `VBoxManage controlvm <vm> reset` has the same effect on a virtual machine as pressing the "Reset" button on a real computer: a cold reboot of the virtual machine, which will restart and boot the guest operating system again immediately. The state of the VM is not saved beforehand, and data may be lost. (This is equivalent to selecting the "Reset" item in the "Machine" menu of the GUI.)

- `VBoxManage controlvm <vm> poweroff` has the same effect on a virtual machine as pulling the power cable on a real computer. Again, the state of the VM is not saved beforehand, and data may be lost. (This is equivalent to selecting the "Close" item in the "Machine" menu of the GUI or pressing the window's close button, and then selecting "Power off the machine" in the dialog.)

  After this, the VM's state will be "Powered off". From there, it can be started again; see chapitre 8.11, *VBoxManage startvm*, page 129.

- `VBoxManage controlvm <vm> savestate` will save the current state of the VM to disk and then stop the VM. (This is equivalent to selecting the "Close" item in the "Machine" menu of the GUI or pressing the window's close button, and then selecting "Save the machine state" in the dialog.)

  After this, the VM's state will be "Saved". From there, it can be started again; see chapitre 8.11, *VBoxManage startvm*, page 129.

- `VBoxManage controlvm <vm> teleport --hostname <name> --port <port> [--password <password>]` makes the machine the source of a teleporting operation and initiates a teleport to the given target. See chapitre 7.2, *Teleporting*, page 108 for an introduction. If the optional password is specified, it must match the password that was given to the `modifyvm` command for the target machine; see chapitre 8.7.5, *Teleporting settings*, page 126 for details.

A few extra options are available with `controlvm` that do not directly affect the VM's running state:

- The `setlinkstate<1-N>` operation connects or disconnects virtual network cables from their network interfaces.

- `nic<1-N> null|nat|bridged|intnet|hostonly|generic`: With this, you can set, for each of the VM's virtual network cards, what type of networking should be available. They can be not connected to the host (`null`), use network address translation (`nat`), bridged networking (`bridged`) or communicate with other virtual machines using internal networking (`intnet`) or host-only networking (`hostonly`) or access to rarely used sub-modes (`generic`). These options correspond to the modes which are described in detail in chapitre 6.2, *Introduction aux modes réseaux*, page 92.

- `usbattach` and `usbdettach` make host USB devices visible to the virtual machine on the fly, without the need for creating filters first. The USB devices can be specified by UUID (unique identifier) or by address on the host system.

  You can use `VBoxManage list usbhost` to locate this information.

- `vrde on|off` lets you enable or disable the VRDE server, if it is installed.

- `vrdeport default|<ports>` changes the port or a range of ports that the VRDE server can bind to; "default" or "0" means port 3389, the standard port for RDP. For details, see the description for the `--vrdeport` option in chapitre 8.7.3, *Serial port, audio, clipboard, remote desktop and USB settings*, page 125.

- `setvideomodehint` requests that the guest system change to a particular video mode. This requires that the Guest Additions be installed, and will not work for all guest systems.

- `screenshotpng` takes a screenshot of the guest display and saves it in PNG format.

- The `setcredentials` operation is used for remote logons in Windows guests. For details, please refer to chapitre 9.2, *Automated guest logons*, page 150.

- The `guestmemoryballoon` operation changes the size of the guest memory balloon, that is, memory allocated by the VirtualBox Guest Additions from the guest operating system and returned to the hypervisor for re-use by other virtual machines. This must be specified in megabytes. For details, see chapitre 4.8, *Faire de la montgolfière avec la mémoire*, page 75.
- The `cpuexecutioncap <1-100>`: This operation controls how much cpu time a virtual CPU can use. A value of 50 implies a single virtual CPU can use up to 50% of a single host CPU.

8.13 VBoxManage discardstate

This command discards the saved state of a virtual machine which is not currently running, which will cause its operating system to restart next time you start it. This is the equivalent of pulling out the power cable on a physical machine, and should be avoided if possible.

8.14 VBoxManage adoptstate

If you have a saved state file (`.sav`) that is seperate from the VM configuration, you can use this command to "adopt" the file. This will change the VM to saved state and when you start it, VirtualBox will attempt to restore it from the saved state file you indicated. This command should only be used in special setups.

8.15 VBoxManage snapshot

This command is used to control snapshots from the command line. A snapshot consists of a complete copy of the virtual machine settings, copied at the time when the snapshot was taken, and optionally a virtual machine saved state file if the snapshot was taken while the machine was running. After a snapshot has been taken, VirtualBox creates differencing hard disk for each normal hard disk associated with the machine so that when a snapshot is restored, the contents of the virtual machine's virtual hard disks can be quickly reset by simply dropping the pre-existing differencing files.

The `take` operation takes a snapshot of the current state of the virtual machine. You must supply a name for the snapshot and can optionally supply a description. The new snapshot is inserted into the snapshots tree as a child of the current snapshot and then becomes the new current snapshot.

The `delete` operation deletes a snapshot (specified by name or by UUID). This can take a while to finish since the differencing images associated with the snapshot might need to be merged with their child differencing images.

The `restore` operation will restore the given snapshot (specified by name or by UUID) by resetting the virtual machine's settings and current state to that of the snapshot. The previous current state of the machine will be lost. After this, the given snapshot becomes the new "current" snapshot so that subsequent snapshots are inserted under the snapshot from which was restored.

The `restorecurrent` operation is a shortcut to restore the current snapshot (i.e. the snapshot from which the current state is derived). This subcommand is equivalent to using the "restore" subcommand with the name or UUID of the current snapshot, except that it avoids the extra step of determining that name or UUID.

With the `edit` operation, you can change the name or description of an existing snapshot.

With the `showvminfo` operation, you can view the virtual machine settings that were stored with an existing snapshot.

8.16 VBoxManage closemedium

This commands removes a hard disk, DVD or floppy image from a VirtualBox media registry.[2]

Optionally, you can request that the image be deleted. You will get appropriate diagnostics that the deletion failed, however the image will become unregistered in any case.

8.17 VBoxManage storageattach

This command attaches/modifies/removes a storage medium connected to a storage controller that was previously added with the `storagectl` command (see the previous section). The syntax is as follows:

```
VBoxManage storageattach    <uuid|vmname>
                            --storagectl <name>
                            [--port <number>]
                            [--device <number>]
                            [--type dvddrive|hdd|fdd]
                            [--medium none|emptydrive|
                                      <uuid>|<filename>|host:<drive>|iscsi]
                            [--mtype normal|writethrough|immutable|shareable]
                            [--comment <text>]
                            [--setuuid <uuid>]
                            [--setparentuuid <uuid>]
                            [--passthrough on|off]
                            [--tempeject on|off]
                            [--bandwidthgroup name|none]
                            [--forceunmount]
                            [--server <name>|<ip>]
                            [--target <target>]
                            [--tport <port>]
                            [--lun <lun>]
                            [--encodedlun <lun>]
                            [--username <username>]
                            [--password <password>]
                            [--intnet]
```

A number of parameters are commonly required; the ones at the end of the list are required only for iSCSI targets (see below).

The common parameters are:

uuid|vmname The VM UUID or VM Name. Mandatory.

storagectl Name of the storage controller. Mandatory. The list of the storage controllers currently attached to a VM can be obtained with `VBoxManage showvminfo`; see chapitre 8.4, *VBoxManage showvminfo*, page 119.

port The number of the storage controller's port which is to be modified. Mandatory, unless the storage controller has only a single port.

device The number of the port's device which is to be modified. Mandatory, unless the storage controller has only a single device per port.

type Define the type of the drive to which the medium is being attached/detached/modified. This argument can only be omitted if the type of medium can be determined from either the medium given with the `--medium` argument or from a previous medium attachment.

[2] Before VirtualBox 4.0, it was necessary to call VBoxManage openmedium before a medium could be attached to a virtual machine; that call "registered" the medium with the global VirtualBox media registry. With VirtualBox 4.0 this is no longer necessary; media are added to media registries automatically. The "closemedium" call has been retained, however, to allow for explicitly removing a medium from a registry.

medium Specifies what is to be attached. The following values are supported:

- "none": Any existing device should be removed from the given slot.
- "emptydrive": For a virtual DVD or floppy drive only, this makes the device slot behaves like a removeable drive into which no media has been inserted.
- If a UUID is specified, it must be the UUID of a storage medium that is already known to VirtualBox (e.g. because it has been attached to another virtual machine). See chapitre 8.3, *VBoxManage list*, page 118 for how to list known media. This medium is then attached to the given device slot.
- If a filename is specified, it must be the full path of an existing disk image (ISO, RAW, VDI, VMDK or other), which is then attached to the given device slot.
- "host:<drive>": For a virtual DVD or floppy drive only, this connects the given device slot to the specified DVD or floppy drive on the host computer.
- "iscsi": For virtual hard disks only, this allows for specifying an iSCSI target. In this case, more parameters must be given; see below.

Some of the above changes, in particular for removeable media (floppies and CDs/DVDs), can be effected while a VM is running. Others (device changes or changes in hard disk device slots) require the VM to be powered off.

mtype Defines how this medium behaves with respect to snapshots and write operations. See chapitre 5.4, *Modes spéciaux d'écriture d'images*, page 83 for details.

comment Any description that you want to have stored with this medium (optional; for example, for an iSCSI target, "Big storage server downstairs"). This is purely descriptive and not needed for the medium to function correctly.

setuuid, setparentuuid Modifies the UUID or parent UUID of a medium before attaching it to a VM. This is an expert option. Inappropriate use can make the medium unusable or lead to broken VM configurations if any other VM is referring to the same media already. The most frequently used variant is `--setuuid ""`, which assigns a new (random) UUID to an image. This is useful to resolve the duplicate UUID errors if one duplicated an image using file copy utilities.

passthrough For a virtual DVD drive only, you can enable DVD writing support (currently experimental; see chapitre 5.9, *Opération sur le lecteur de CD/DVD*, page 89).

tempeject For a virtual DVD drive only, you can configure the behavior for guest-triggered medium eject. If this is set to "on", the eject has only temporary effects. If the VM is powered off and restarted the originally configured medium will be still in the drive.

bandwidthgroup Sets the bandwidth group to use for the given device; see chapitre 5.8, *Limiting bandwidth for disk images*, page 89.

forceunmount For a virtual DVD or floppy drive only, this forcibly unmounts the DVD/CD/Floppy or mounts a new DVD/CD/Floppy even if the previous one is locked down by the guest for reading. Again, see chapitre 5.9, *Opération sur le lecteur de CD/DVD*, page 89 for details.

When "iscsi" is used with the `--medium` parameter for iSCSI support – see chapitre 5.11, *Serveurs iSCSI*, page 90 –, additional parameters must or can be used:

server The host name or IP address of the iSCSI target; required.

target Target name string. This is determined by the iSCSI target and used to identify the storage resource; required.

port TCP/IP port number of the iSCSI service on the target (optional).

lun Logical Unit Number of the target resource (optional). Often, this value is zero.

username, password Username and password for target authentication, if required (optional).

> **Note:** Currently, username and password are stored without encryption (i.e. in clear text) in the XML machine configuration file.

intnet If specified, connect to the iSCSI target via Internal Networking. This needs further configuration which is described in chapitre 9.8.3, *Accès à des cibles iSCSI à travers le réseau interne*, page 160.

8.18 VBoxManage storagectl

This command attaches/modifies/removes a storage controller. After this, virtual media can be attached to the controller with the `storageattach` command (see the next section).

The syntax is as follows:

```
VBoxManage storagectl       <uuid|vmname>
                            --name <name>
                            [--add <ide/sata/scsi/floppy>]
                            [--controller <LsiLogic|LSILogicSAS|BusLogic|
                                           IntelAhci|PIIX3|PIIX4|ICH6|I82078>]
                            [--sataideemulation<1-4> <1-30>]
                            [--sataportcount <1-30>]
                            [--hostiocache on|off]
                            [--bootable on|off]
                            [--remove]
```

where the parameters mean:

uuid|vmname The VM UUID or VM Name. Mandatory.

name Name of the storage controller. Mandatory.

add Define the type of the system bus to which the storage controller must be connected.

controller Allows to choose the type of chipset being emulated for the given storage controller.

sataideemulation This specifies which SATA ports should operate in IDE emulation mode. As explained in chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78, by default, this is the case for SATA ports 1-4; with this command, you can map four IDE channels to any of the 30 supported SATA ports.

sataportcount This determines how many ports the SATA controller should support.

hostiocache Configures the use of the host I/O cache for all disk images attached to this storage controller. For details, please see chapitre 5.7, *Images de disque et mise en cache E/S*, page 87.

bootable Selects whether this controller is bootable.

remove Removes the storage controller from the VM config.

8.19 VBoxManage bandwidthctl

This command creates/deletes/modifies bandwidth groups of the given virtual machine:

```
VBoxManage bandwidthctl    <uuid|vmname>
                          --name <name>
                          [--add disk
                          [--delete]
                          [--limit MB/s]
```

See chapitre 5.8, *Limiting bandwidth for disk images*, page 89 for an introduction to bandwidth limits. The parameters mean:

uuid|vmname The VM UUID or VM Name. Mandatory.

name Name of the bandwidth group. Mandatory.

add Creates a new bandwdith group with the given type.

delete Deletes a bandwdith group if it isn't used anymore.

limit Sets the limit for the given group to the specified amount. Can be changed while the VM is running.

8.20 VBoxManage showhdinfo

This command shows information about a virtual hard disk image, notably its size, its size on disk, its type and the virtual machines which use it.

> **Note:** For compatibility with earlier versions of VirtualBox, the "showvdiinfo" command is also supported and mapped internally to the "showhdinfo" command.

The disk image must be specified either by its UUID (if the medium is registered) or by its filename. Registered images can be listed by `VBoxManage list hdds` (see chapitre 8.3, *VBoxManage list*, page 118 for more information). A filename must be specified as valid path, either as an absolute path or as a relative path starting from the current directory.

8.21 VBoxManage createhd

This command creates a new virtual hard disk image. The syntax is as follows:

```
VBoxManage createhd         --filename <filename>
                            --size <megabytes>
                            [--format VDI|VMDK|VHD] (default: VDI)
                            [--variant Standard,Fixed,Split2G,Stream,ESX]
```

where the parameters mean:

filename Allows to choose a file name. Mandatory.

size Allows to define the image capacity, in 1 MiB units. Mandatory.

format Allows to choose a file format for the output file different from the file format of the input file.

variant Allows to choose a file format variant for the output file. It is a comma-separated list of variant flags. Not all combinations are supported, and specifying inconsistent flags will result in an error message.

Note: For compatibility with earlier versions of VirtualBox, the "createvdi" command is also supported and mapped internally to the "createhd" command.

8.22 VBoxManage modifyhd

With the `modifyhd` command, you can change the characteristics of a disk image after it has been created:

```
VBoxManage modifyhd          <uuid>|<filename>
                             [--type normal|writethrough|immutable|shareable|
                                     readonly|multiattach]
                             [--autoreset on|off]
                             [--compact]
                             [--resize <megabytes>|--resizebyte <bytes>]
```

Note: Despite the "hd" in the subcommand name, the command works with all disk images, not only hard disks. For compatibility with earlier versions of VirtualBox, the "modifyvdi" command is also supported and mapped internally to the "modifyhd" command.

The disk image to modify must be specified either by its UUID (if the medium is registered) or by its filename. Registered images can be listed by `VBoxManage list hdds` (see chapitre 8.3, *VBoxManage list*, page 118 for more information). A filename must be specified as valid path, either as an absolute path or as a relative path starting from the current directory.

The following options are available:

- With the `--type` argument, you can change the type of an existing image between the normal, immutable, write-through and other modes; see chapitre 5.4, *Modes spéciaux d'écriture d'images*, page 83 for details.
- For immutable (differencing) hard disks only, the `--autoreset on|off` option determines whether the disk is automatically reset on every VM startup (again, see chapitre 5.4, *Modes spéciaux d'écriture d'images*, page 83). The default is "on".
- With the `--compact` option, can be used to compact disk images, i.e. remove blocks that only contains zeroes. This will shrink a dynamically allocated image again; it will reduce the *physical* size of the image without affecting the logical size of the virtual disk. Compaction works both for base images and for diff images created as part of a snapshot.

  For this operation to be effective, it is required that free space in the guest system first be zeroed out using a suitable software tool. For Windows guests, you can use the `sdelete` tool provided by Microsoft. Execute `sdelete -c` in the guest to zero the free disk space before compressing the virtual disk image. For Linux, use the `zerofree` utility which supports ext2/ext3 filesystems.

  Please note that compacting is currently only available for VDI images. A similar effect can be achieved by zeroing out free blocks and then cloning the disk to any other dynamically allocated format. You can use this workaround until compacting is also supported for disk formats other than VDI.
- The `--resize` option allows you to change the capacity of an existing image; this adjusts the *logical* size of a virtual disk without affecting the physical size much.[3] This currently

[3] Image resizing was added with VirtualBox 4.0.

works only for expanding the capacity of VDI and VHD formats, and only for the dynamically allocated variants. For example, if you originally created a 10G disk which is now full, you can use the `--resize 15360` command to add 5 GByte more space to the virtual disk without having to create a new image and copy all data from within a virtual machine.

8.23 VBoxManage clonehd

This command duplicates a registered virtual hard disk image to a new image file with a new unique identifier (UUID). The new image can be transferred to another host system or imported into VirtualBox again using the Virtual Media Manager; see chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82 and chapitre 5.6, *Cloner des images de disque*, page 87. The syntax is as follows:

```
VBoxManage clonehd          <uuid>|<filename> <outputfile>
                            [--format VDI|VMDK|VHD|RAW|<other>]
                            [--variant Standard,Fixed,Split2G,Stream,ESX]
                            [--existing]
```

The disk image to clone as well as the target image must be described either by its UUIDs (if the mediums are registered) or by its filename. Registered images can be listed by `VBoxManage list hdds` (see chapitre 8.3, *VBoxManage list*, page 118 for more information). A filename must be specified as valid path, either as an absolute path or as a relative path starting from the current directory.

The following options are available:

format Allow to choose a file format for the output file different from the file format of the input file.

variant Allow to choose a file format variant for the output file. It is a comma-separated list of variant flags. Not all combinations are supported, and specifying inconsistent flags will result in an error message.

existing Perform the clone operation to an already existing destination medium. Only the portion of the source medium which fits into the destination medium is copied. This means if the destination medium is smaller than the source only a part of it is copied, and if the destination medium is larger than the source the remaining part of the destination medium is unchanged.

Note: For compatibility with earlier versions of VirtualBox, the “clonevdi” command is also supported and mapped internally to the “clonehd” command.

8.24 VBoxManage convertfromraw

This command converts a raw disk image to a VirtualBox Disk Image (VDI) file. The syntax is as follows:

```
VBoxManage convertfromraw   <filename> <outputfile>
                            [--format VDI|VMDK|VHD]
                            [--variant Standard,Fixed,Split2G,Stream,ESX]
VBoxManage convertfromraw   stdin <outputfile> <bytes>
                            [--format VDI|VMDK|VHD]
                            [--variant Standard,Fixed,Split2G,Stream,ESX]
```

where the parameters mean:

format Select the disk image format to create. Default is VDI.

variant Allow to choose a file format variant for the output file. It is a comma-separated list of variant flags. Not all combinations are supported, and specifying inconsistent flags will result in an error message.

The second form forces VBoxManage to read the content for the disk image from standard input (useful for using that command in a pipe).

> **Note:** For compatibility with earlier versions of VirtualBox, the "convertdd" command is also supported and mapped internally to the "convertfromraw" command.

8.25 VBoxManage getextradata/setextradata

These commands let you attach and retrieve string data to a virtual machine or to a VirtualBox configuration (by specifying `global` instead of a virtual machine name). You must specify a key (as a text string) to associate the data with, which you can later use to retrieve it. For example:

```
VBoxManage setextradata Fedora5 installdate 2006.01.01
VBoxManage setextradata SUSE10 installdate 2006.02.02
```

would associate the string "2006.01.01" with the key installdate for the virtual machine Fedora5, and "2006.02.02" on the machine SUSE10. You could retrieve the information as follows:

```
VBoxManage getextradata Fedora5 installdate
```

which would return

```
VirtualBox Command Line Management Interface Version 4.2.12
(C) 2005-2013 Oracle Corporation
All rights reserved.

Value: 2006.01.01
```

8.26 VBoxManage setproperty

This command is used to change global settings which affect the entire VirtualBox installation. Some of these correspond to the settings in the "Global settings" dialog in the graphical user interface. The following properties are available:

machinefolder This specifies the default folder in which virtual machine definitions are kept; see chapitre 10.1, *Where VirtualBox stores its files*, page 171 for details.

vrdeauthlibrary This specifies which library to use when "external" authentication has been selected for a particular virtual machine; see chapitre 7.1.5, *RDP authentication*, page 105 for details.

websrvauthlibrary This specifies which library the web service uses to authenticate users. For details about the VirtualBox web service, please refer to the separate VirtualBox SDK reference (see chapitre 11, *VirtualBox programming interfaces*, page 181).

vrdelibrary This specifies which library implements the VirtualBox Remote Desktop Extension.

hwvirtexenabled This selects whether or not hardware virtualization support is enabled by default.

8.27 VBoxManage usbfilter add/modify/remove

The `usbfilter` commands are used for working with USB filters in virtual machines, or global filters which affect the whole VirtualBox setup. Global filters are applied before machine-specific filters, and may be used to prevent devices from being captured by any virtual machine. Global filters are always applied in a particular order, and only the first filter which fits a device is applied. So for example, if the first global filter says to hold (make available) a particular Kingston memory stick device and the second to ignore all Kingston devices, that memory stick will be available to any machine with an appropriate filter, but no other Kingston device will.

When creating a USB filter using `usbfilter add`, you must supply three or four mandatory parameters. The index specifies the position in the list at which the filter should be placed. If there is already a filter at that position, then it and the following ones will be shifted back one place. Otherwise the new filter will be added onto the end of the list. The `target` parameter selects the virtual machine that the filter should be attached to or use "global" to apply it to all virtual machines. `name` is a name for the new filter and for global filters, `action` says whether to allow machines access to devices that fit the filter description ("hold") or not to give them access ("ignore"). In addition, you should specify parameters to filter by. You can find the parameters for devices attached to your system using `VBoxManage list usbhost`. Finally, you can specify whether the filter should be active, and for local filters, whether they are for local devices, remote (over an RDP connection) or either.

When you modify a USB filter using `usbfilter modify`, you must specify the filter by index (see the output of `VBoxManage list usbfilters` to find global filter indexes and that of `VBoxManage showvminfo` to find indexes for individual machines) and by target, which is either a virtual machine or "global". The properties which can be changed are the same as for `usbfilter add`. To remove a filter, use `usbfilter remove` and specify the index and the target.

8.28 VBoxManage sharedfolder add/remove

This command allows you to share folders on the host computer with guest operating systems. For this, the guest systems must have a version of the VirtualBox Guest Additions installed which supports this functionality.

Shared folders are described in detail in chapitre 4.3, *Dossiers partagés*, page 68.

8.29 VBoxManage guestproperty

The "guestproperty" commands allow you to get or set properties of a running virtual machine. Please see chapitre 4.6, *Propriétés invité*, page 73 for an introduction. As explained there, guest properties are arbitrary key/value string pairs which can be written to and read from by either the guest or the host, so they can be used as a low-volume communication channel for strings, provided that a guest is running and has the Guest Additions installed. In addition, a number of values whose keys begin with "/VirtualBox/" are automatically set and maintained by the Guest Additions.

The following subcommands are available (where `<vm>`, in each case, can either be a VM name or a VM UUID, as with the other VBoxManage commands):

- `enumerate <vm> [--patterns <pattern>]`: This lists all the guest properties that are available for the given VM, including the value. This list will be very limited if the guest's service process cannot be contacted, e.g. because the VM is not running or the Guest Additions are not installed.

  If `--patterns <pattern>` is specified, it acts as a filter to only list properties that match the given pattern. The pattern can contain the following wildcard characters:

  - `*` (asterisk): represents any number of characters; for example, "`/VirtualBox*`" would match all properties beginning with "/VirtualBox".
  - `?` (question mark): represents a single arbitrary character; for example, "`fo?`" would match both "foo" and "for".
  - `|` (pipe symbol): can be used to specify multiple alternative patterns; for example, "`s*|t*`" would match anything starting with either "s" or "t".

- `get <vm>`: This retrieves the value of a single property only. If the property cannot be found (e.g. because the guest is not running), this will print

  ```
  No value set!
  ```

- `set <vm> <property> [<value> [--flags <flags>]]`: This allows you to set a guest property by specifying the key and value. If `<value>` is omitted, the property is deleted. With `--flags` you can optionally specify additional behavior (you can combine several by separating them with commas):
  - `TRANSIENT`: the value will not be stored with the VM data when the VM exits;
  - `TRANSRESET`: the value will be deleted as soon as the VM restarts and/or exits;
  - `RDONLYGUEST`: the value can only be changed by the host, but the guest can only read it;
  - `RDONLYHOST`: reversely, the value can only be changed by the guest, but the host can only read it;
  - `READONLY`: a combination of the two, the value cannot be changed at all.

- `wait <vm> <pattern> --timeout <timeout>`: This waits for a particular value described by "pattern" to change or to be deleted or created. The pattern rules are the same as for the "enumerate" subcommand above.

8.30 VBoxManage guestcontrol

The "guestcontrol" commands allow you to control certain things inside a guest from the host. Please see chapitre 4.7, *Contrôle invité*, page 75 for an introduction.

Generally, the syntax is as follows:

```
VBoxManage guestcontrol <command>
```

The following subcommands are available (where `<vm>`, in each case, can either be a VM name or a VM UUID, as with the other VBoxManage commands):

- `execute`, which allows for executing a program/script (process) which is already installed and runnable on the guest. This command only works while a VM is up and running and has the following syntax:

  ```
  VBoxManage guestcontrol <vmname>|<uuid> exec[ute]
              --image <path to program>
              --username <name> --password <password>
              [--dos2unix]
              [--environment "<NAME>=<VALUE> [<NAME>=<VALUE>]"]
              [--timeout <msec>] [--unix2dos] [--verbose]
              [--wait-exit] [--wait-stdout] [--wait-stderr]
              -- [[<argument1>] ... [<argumentN>]]
  ```

  where the parameters mean:

  uuid|vmname The VM UUID or VM name. Mandatory.

–image "<path to program>" Absolute path and process name of process to execute in the guest, e.g. `C:\Windows\System32\calc.exe`

–username <name> Name of the user the process should run under. This user must exist on the guest OS.

–password <password> Password of the user account specified with `--username`. If not given, an empty password is assumed.

–dos2unix Converts output from DOS/Windows guests to UNIX-compatible line endings (CR + LF -> LF). Not implemented yet.

–environment "<NAME>=<VALUE>" One or more environment variables to be set or unset.

By default, the new process in the guest will be created with the standard environment of the guest OS. This option allows for modifying that environment. To set/modify a variable, a pair of `NAME=VALUE` must be specified; to unset a certain variable, the name with no value must set, e.g. `NAME=`.

Arguments containing spaces must be enclosed in quotation marks. More than one `--environment` at a time can be specified to keep the command line tidy.

–timeout <msec> Value (in milliseconds) that specifies the time how long the started process is allowed to run and how long VBoxManage waits for getting output from that process. If no timeout is specified, VBoxManage will wait forever until the started process ends or an error occured.

–unix2dos Converts output from a UNIX/Linux guests to DOS-/Windows-compatible line endings (LF -> CR + LF). Not implemented yet.

–verbose Tells VBoxManage to be more verbose.

–wait-exit Waits until the process ends and outputs its exit code along with the exit reason/flags.

–wait-stdout Waits until the process ends and outputs its exit code along with the exit reason/flags. While waiting VBoxManage retrieves the process output collected from stdout.

–wait-stderr Waits until the process ends and outputs its exit code along with the exit reason/flags. While waiting VBoxManage retrieves the process output collected from stderr.

[– [<argument1s> ... [<argumentNs>]]]

One or more arguments to pass to the process being executed.

Arguments containing spaces must be enclosed in quotation marks.

Note: On Windows there are certain limitations for graphical applications; please see chapitre 14, *Known limitations*, page 201 for more information.

Examples:

```
VBoxManage --nologo guestcontrol "My VM" execute --image "/bin/ls"
          --username foo --password bar --wait-exit --wait-stdout -- -l /usr

VBoxManage --nologo guestcontrol "My VM" execute --image "c:\\windows\\system32\\ipconfig.exe"
          --username foo --password bar --wait-exit --wait-stdout
```

Note that the double backslashes in the second example are only required on Unix hosts.

Starting at VirtualBox 4.1.2 guest process execution by default is limited to serve up to 5 guest processes at a time. If a new guest process gets started which would exceed this limit,

the oldest not running guest process will be discarded in order to be able to run that new process. Also, retrieving output from this old guest process will not be possible anymore then. If all 5 guest processes are still active and running, starting a new guest process will result in an appropriate error message.

To raise or lower the guest process execution limit, either the guest property /VirtualBox/GuestAdd/VBoxService/–control-procs-max-kept or VBoxService' command line by specifying `--control-procs-max-kept` needs to be modified. A restart of the guest OS is required afterwards. To serve unlimited guest processes, a value of `0` needs to be set (not recommended).

- `copyto`, which allows copying files from the host to the guest (only with installed Guest Additions 4.0 and later).

```
VBoxManage guestcontrol <vmname>|<uuid> copyto|cp
           <source on host> <destination on guest>
           --username <name> --password <password>
           [--dryrun] [--follow] [--recursive] [--verbose]
```

where the parameters mean:

uuid|vmname The VM UUID or VM name. Mandatory.

source on host Absolute path of source file(s) on host to copy over to the guest, e.g. `C:\Windows\System32\calc.exe`. This also can be a wildcard expression, e.g. `C:\Windows\System32\*.dll`

destination on guest Absolute destination path on the guest, e.g. `C:\Temp`

–username <name> Name of the user the copy process should run under. This user must exist on the guest OS.

–password <password> Password of the user account specified with `--username`. If not given, an empty password is assumed.

–dryrun Tells VBoxManage to only perform a dry run instead of really copying files to the guest.

–follow Enables following symlinks on the host's source.

–recursive Recursively copies files/directories of the specified source.

–verbose Tells VBoxManage to be more verbose.

–flags <flags> Additional flags to set. This is not used at the moment.

- `createdirectory`, which allows copying files from the host to the guest (only with installed Guest Additions 4.0 and later).

```
VBoxManage guestcontrol <vmname>|<uuid> createdir[ectory]|mkdir|md
           <directory to create on guest>
           [--username "<name>"] [--password "<password>"]
           [--parents] [--mode <mode>] [--verbose]
```

where the parameters mean:

uuid|vmname The VM UUID or VM name. Mandatory.

directory to create on guest Absolute path of directory/directories to create on guest, e.g. `D:\Foo\Bar`. Parent directories need to exist (e.g. in this example `D:\Foo`) when switch `--parents` is omitted. The specified user must have appropriate rights to create the specified directory.

–username <name> Name of the user the copy process should run under. This user must exist on the guest OS.

–password <password> Password of the user account specified with `--username`. If not given, an empty password is assumed.

–parents Also creates not yet existing parent directories of the specified directory, e.g. if the directory `D:\Foo` of `D:\Foo\Bar` does not exist yet it will be created. Without specifying `--parent` the action would have failed.

–mode <mode> Sets the permission mode of the specified directory. Only octal modes (e.g. `0755`) are supported right now.

–verbose Tells VBoxManage to be more verbose.

- `stat`, which displays file or file system status on the guest.

```
VBoxManage guestcontrol <vmname>|<uuid> stat
            <file element(s) to check on guest>
            [--username "<name>"] [--password "<password>"]
            [--verbose]
```

where the parameters mean:

uuid|vmname The VM UUID or VM name. Mandatory.

file element(s) to check on guest Absolute path of directory/directories to check on guest, e.g. `/home/foo/a.out`. The specified user must have appropriate rights to access the given file element(s).

–username <name> Name of the user the copy process should run under. This user must exist on the guest OS.

–password <password> Password of the user account specified with `--username`. If not given, an empty password is assumed.

–verbose Tells VBoxManage to be more verbose.

- `updateadditions`, which allows for updating an already installed Guest Additions version on the guest (only already installed Guest Additions 4.0 and later).

```
VBoxManage guestcontrol updateadditions <vmname>|<uuid>
            [--source "<guest additions .ISO file to use>"] [--verbose]
```

where the parameters mean:

uuid|vmname The VM UUID or VM name. Mandatory.

–source "<guest additions .ISO file to use>" Full path to an alternative VirtualBox Guest Additions .ISO file to use for the Guest Additions update.

–verbose Tells VBoxManage to be more verbose.

8.31 VBoxManage debugvm

The "debugvm" commands are for experts who want to tinker with the exact details of virtual machine execution. Like the VM debugger described in chapitre 12.1.3, *The built-in VM debugger*, page 183, these commands are only useful if you are very familiar with the details of the PC architecture and how to debug software.

The subcommands of "debugvm" all operate on a running virtual machine. The following are available:

- With `dumpguestcore --filename <name>`, you can create a system dump of the running VM, which will be written into the given file. This file will have the standard ELF core format (with custom sections); see chapitre 12.1.4, *VM core format*, page 185.

  This corresponds to the `writecore` command in the debugger.

- The `info` command is used to display info items relating to the VMM, device emulations and associated drivers. This command takes one or two arguments: the name of the info item, optionally followed by a string containing arguments specific to the info item. The `help` info item provides a listning of the available items and hints about any optional arguments.

  This corresponds to the `info` command in the debugger.

- The `injectnmi` command causes a non-maskable interrupt (NMI) in the guest, which might be useful for certain debugging scenarios. What happens exactly is dependent on the guest operating system, but an NMI can crash the whole guest operating system. Do not use unless you know what you're doing.

- The `osdetect` command makes the VMM's debugger facility (re-)detection the guest operation system.

  This corresponds to the `detect` command in the debugger.

- The `osinfo` command is used to display info about the operating system (OS) detected by the VMM's debugger facility.

- The `getregisters` command is used to display CPU and device registers. The command takes a list of registers, each having one of the following forms:

  - `register-set.register-name.sub-field`
  - `register-set.register-name`
  - `cpu-register-name.sub-field`
  - `cpu-register-name`
  - `all`

  The `all` form will cause all registers to be shown (no sub-fields). The registers names are case-insensitive. When requesting a CPU register the register set can be omitted, it will be selected using the value of the `--cpu` option (defaulting to 0).

- The `setregisters` command is used to change CPU and device registers. The command takes a list of register assignments, each having one of the following forms:

  - `register-set.register-name.sub-field=value`
  - `register-set.register-name=value`
  - `cpu-register-name.sub-field=value`
  - `cpu-register-name=value`

  The value format should be in the same style as what `getregisters` displays, with the exception that both octal and decimal can be used instead of hexadecimal. The register naming and the default CPU register set are handled the same way as with the `getregisters` command.

- The `statistics` command can be used to display VMM statistics on the command line. The `--reset` option will reset statistics. The affected statistics can be filtered with the `--pattern` option, which accepts DOS/NT-style wildcards (`?` and `*`).

8.32 VBoxManage metrics

This command supports monitoring the usage of system resources. Resources are represented by various metrics associated with the host system or a particular VM. For example, the host system

has a CPU/Load/User metric that shows the percentage of time CPUs spend executing in user mode over a specific sampling period.

Metric data is collected and retained internally; it may be retrieved at any time with the VBoxManage metrics query subcommand. The data is available as long as the background VBoxSVC process is alive. That process terminates shortly after all VMs and frontends have been closed.

By default no metrics are collected at all. Metrics collection does not start until VBoxManage metrics setup is invoked with a proper sampling interval and the number of metrics to be retained. The interval is measured in seconds. For example, to enable collecting the host processor and memory usage metrics every second and keeping the 5 most current samples, the following command can be used:

```
VBoxManage metrics setup --period 1 --samples 5 host CPU/Load,RAM/Usage
```

Metric collection can only be enabled for started VMs. Collected data and collection settings for a particular VM will disappear as soon as it shuts down. Use VBoxManage metrics list subcommand to see which metrics are currently available. You can also use --list option with any subcommand that modifies metric settings to find out which metrics were affected.

Note that the VBoxManage metrics setup subcommand discards all samples that may have been previously collected for the specified set of objects and metrics.

To enable or disable metrics collection without discarding the data VBoxManage metrics enable and VBoxManage metrics disable subcommands can be used. Note that these subcommands expect metrics, not submetrics, like CPU/Load or RAM/Usage as parameters. In other words enabling CPU/Load/User while disabling CPU/Load/Kernel is not supported.

The host and VMs have different sets of associated metrics. Available metrics can be listed with VBoxManage metrics list subcommand.

A complete metric name may include an aggregate function. The name has the following form: Category/Metric[/SubMetric][:aggregate]. For example, RAM/Usage/Free:min stands for the minimum amount of available memory over all retained data if applied to the host object.

Subcommands may apply to all objects and metrics or can be limited to one object or/and a list of metrics. If no objects or metrics are given in the parameters, the subcommands will apply to all available metrics of all objects. You may use an asterisk ("*") to explicitly specify that the command should be applied to all objects or metrics. Use "host" as the object name to limit the scope of the command to host-related metrics. To limit the scope to a subset of metrics, use a metric list with names separated by commas.

For example, to query metric data on the CPU time spent in user and kernel modes by the virtual machine named "test", you can use the following command:

```
VBoxManage metrics query test CPU/Load/User,CPU/Load/Kernel
```

The following list summarizes the available subcommands:

list This subcommand shows the parameters of the currently existing metrics. Note that VM-specific metrics are only available when a particular VM is running.

setup This subcommand sets the interval between taking two samples of metric data and the number of samples retained internally. The retained data is available for displaying with the query subcommand. The --list option shows which metrics have been modified as the result of the command execution.

enable This subcommand "resumes" data collection after it has been stopped with disable subcommand. Note that specifying submetrics as parameters will not enable underlying metrics. Use --list to find out if the command did what was expected.

disable This subcommand "suspends" data collection without affecting collection parameters or collected data. Note that specifying submetrics as parameters will not disable underlying metrics. Use `--list` to find out if the command did what was expected.

query This subcommand retrieves and displays the currently retained metric data.

> **Note:** The `query` subcommand does not remove or "flush" retained data. If you query often enough you will see how old samples are gradually being "phased out" by new samples.

collect This subcommand sets the interval between taking two samples of metric data and the number of samples retained internally. The collected data is displayed periodically until Ctrl-C is pressed unless the `--detach` option is specified. With the `--detach` option, this subcommand operates the same way as `setup` does. The `--list` option shows which metrics match the specified filter.

8.33 VBoxManage hostonlyif

With "hostonlyif" you can change the IP configuration of a host-only network interface. For a description of host-only networking, please refer to chapitre 6.6, *Réseau privé avec l'hôte (Host-only)*, page 97. Each host-only interface is identified by a name and can either use the internal DHCP server or a manual IP configuration (both IP4 and IP6).

8.34 VBoxManage dhcpserver

The "dhcpserver" commands allow you to control the DHCP server that is built into VirtualBox. You may find this useful when using internal or host-only networking. (Theoretically, you can enable it for a bridged network as well, but that will likely cause conflicts with other DHCP servers in your physical network.)

Use the following command line options:

- If you use internal networking for a virtual network adapter of a virtual machine, use `VBoxManage dhcpserver add --netname <network_name>`, where `<network_name>` is the same network name you used with `VBoxManage modifyvm <vmname> --intnet<X> <network_name>`.
- If you use host-only networking for a virtual network adapter of a virtual machine, use `VBoxManage dhcpserver add --ifname <hostonly_if_name>` instead, where `<hostonly_if_name>` is the same host-only interface name you used with `VBoxManage modifyvm <vmname> --hostonlyadapter<X> <hostonly_if_name>`.

  Alternatively, you can also use the –netname option as with internal networks if you know the host-only network's name; you can see the names with `VBoxManage list hostonlyifs` (see chapitre 8.3, *VBoxManage list*, page 118 above).

The following additional parameters are required when first adding a DHCP server:

- With `--ip`, specify the IP address of the DHCP server itself.
- With `--netmask`, specify the netmask of the network.
- With `--lowerip` and `--upperip`, you can specify the lowest and highest IP address, respectively, that the DHCP server will hand out to clients.

Finally, you must specify `--enable` or the DHCP server will be created in the disabled state, doing nothing.

After this, VirtualBox will automatically start the DHCP server for given internal or host-only network as soon as the first virtual machine which uses that network is started.

Reversely, use `VBoxManage dhcpserver remove` with the given `--netname <network_name>` or `--ifname <hostonly_if_name>` to remove the DHCP server again for the given internal or host-only network.

To modify the settings of a DHCP server created earlier with `VBoxManage dhcpserver add`, you can use `VBoxManage dhcpserver modify` for a given network or host-only interface name.

8.35 VBoxManage extpack

The "extpack" command allows you to add or remove VirtualBox extension packs, as described in chapitre 1.5, *Installer et lancer VirtualBox*, page 15.

- To add a new extension pack, use `VBoxManage extpack install <tarball>`. This command will fail if an older version of the same extension pack is already installed. The optional `--replace` parameter can be used to uninstall the old package before the new package is installed.
- To remove a previously installed extension pack, use `VBoxManage extpack uninstall <name>`. You can use `VBoxManage list extpacks` to show the names of the extension packs which are currently installed; please see chapitre 8.3, *VBoxManage list*, page 118 also. The optional `--force` parameter can be used to override the refusal of an extension pack to be uninstalled.
- The `VBoxManage extpack cleanup` command can be used to remove temporary files and directories that may have been left behind if a previous install or uninstall command failed.

9 Advanced topics

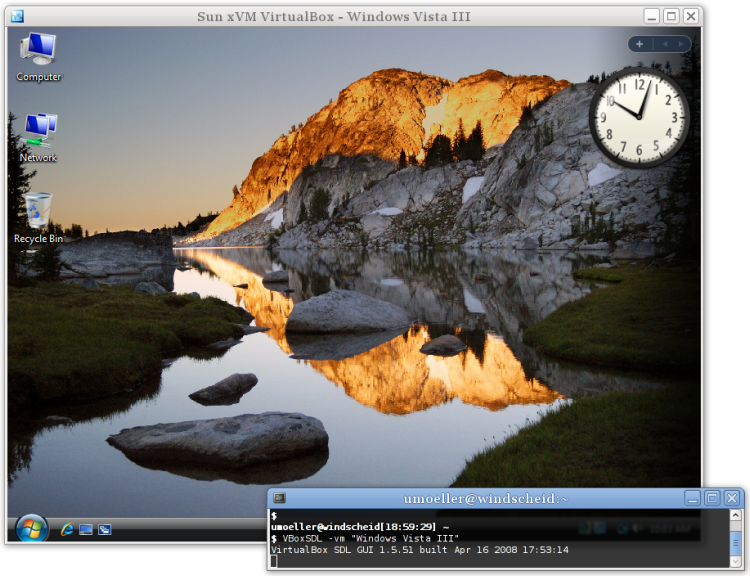
9.1 VBoxSDL, the simplified VM displayer

9.1.1 Introduction

VBoxSDL is a simple graphical user interface (GUI) that lacks the nice point-and-click support which VirtualBox, our main GUI, provides. VBoxSDL is currently primarily used internally for debugging VirtualBox and therefore not officially supported. Still, you may find it useful for environments where the virtual machines are not necessarily controlled by the same person that uses the virtual machine.

Note: VBoxSDL is not available on the Mac OS X host platform.

As you can see in the following screenshot, VBoxSDL does indeed only provide a simple window that contains only the "pure" virtual machine, without menus or other controls to click upon and no additional indicators of virtual machine activity:



To start a virtual machine with VBoxSDL instead of the VirtualBox GUI, enter the following on a command line:

```
VBoxSDL --startvm <vm>
```

where `<vm>` is, as usual with VirtualBox command line parameters, the name or UUID of an existing virtual machine.

9.1.2 Secure labeling with VBoxSDL

When running guest operating systems in fullscreen mode, the guest operating system usually has control over the whole screen. This could present a security risk as the guest operating

system might fool the user into thinking that it is either a different system (which might have a higher security level) or it might present messages on the screen that appear to stem from the host operating system.

In order to protect the user against the above mentioned security risks, the secure labeling feature has been developed. Secure labeling is currently available only for VBoxSDL. When enabled, a portion of the display area is reserved for a label in which a user defined message is displayed. The label height in set to 20 pixels in VBoxSDL. The label font color and background color can be optionally set as hexadecimal RGB color values. The following syntax is used to enable secure labeling:

```
VBoxSDL --startvm "VM name"
      --securelabel --seclabelfnt ~/fonts/arial.ttf
      --seclabelsiz 14 --seclabelfgcol 00FF00 --seclabelbgcol 00FFFF
```

In addition to enabling secure labeling, a TrueType font has to be supplied. To use another font size than 12 point use the parameter `--seclabelsiz`.

The label text can be set with

```
VBoxManage setextradata "VM name" "VBoxSDL/SecureLabel" "The Label"
```

Changing this label will take effect immediately.

Typically, full screen resolutions are limited to certain “standard” geometries such as 1024 x 768. Increasing this by twenty lines is not usually feasible, so in most cases, VBoxSDL will chose the next higher resolution, e.g. 1280 x 1024 and the guest’s screen will not cover the whole display surface. If VBoxSDL is unable to choose a higher resolution, the secure label will be painted on top of the guest’s screen surface. In order to address the problem of the bottom part of the guest screen being hidden, VBoxSDL can provide custom video modes to the guest that are reduced by the height of the label. For Windows guests and recent Solaris and Linux guests, the VirtualBox Guest Additions automatically provide the reduced video modes. Additionally, the VESA BIOS has been adjusted to duplicate its standard mode table with adjusted resolutions. The adjusted mode IDs can be calculated using the following formula:

```
reduced_modeid = modeid + 0x30
```

For example, in order to start Linux with 1024 x 748 x 16, the standard mode 0x117 (1024 x 768 x 16) is used as a base. The Linux video mode kernel parameter can then be calculated using:

```
vga = 0x200 | 0x117 + 0x30
vga = 839
```

The reason for duplicating the standard modes instead of only supplying the adjusted modes is that most guest operating systems require the standard VESA modes to be fixed and refuse to start with different modes.

When using the X.org VESA driver, custom modelines have to be calculated and added to the configuration (usually in /etc/X11/xorg.conf. A handy tool to determine modeline entries can be found at http://www.tkk.fi/Misc/Electronics/faq/vga2rgb/calc.html.)

9.1.3 Releasing modifiers with VBoxSDL on Linux

When switching from a X virtual terminal (VT) to another VT using Ctrl-Alt-Fx while the VBoxSDL window has the input focus, the guest will receive Ctrl and Alt keypress events without receiving the corresponding key release events. This is an architectural limitation of Linux. In order to reset the modifier keys, it is possible to send `SIGUSR1` to the VBoxSDL main thread (first entry in the `ps` list). For example, when switching away to another VT and saving the virtual machine from this terminal, the following sequence can be used to make sure the VM is not saved with stuck modifiers:

```
kill -usr1 <pid>
VBoxManage controlvm "Windows 2000" savestate
```

9.2 Automated guest logons

VirtualBox provides Guest Addition modules for Windows, Linux and Solaris to enable automated logons on the guest.

When a guest operating system is running in a virtual machine, it might be desirable to perform coordinated and automated logons using credentials from a master logon system. (With "credentials", we are referring to logon information consisting of user name, password and domain name, where each value might be empty.)

9.2.1 Automated Windows guest logons

Since Windows NT, Windows has provided a modular system logon subsystem ("Winlogon") which can be customized and extended by means of so-called GINA modules (Graphical Identification and Authentication). With Windows Vista and Windows 7, the GINA modules were replaced with a new mechanism called "credential providers". The VirtualBox Guest Additions for Windows come with both, a GINA and a credential provider module, and therefore enable any Windows guest to perform automated logons.

To activate the VirtualBox GINA or credential provider module, install the Guest Additions with using the command line switch `/with_autologon`. All the following manual steps required for installing these modules will be then done by the installer.

To manually install the VirtualBox GINA module, extract the Guest Additions (see chapitre 4.2.1.4, *Extraction manuelle de fichiers*, page 61) and copy the file `VBoxGINA.dll` to the Windows `SYSTEM32` directory. Then, in the registry, create the following key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL
```

with a value of `VBoxGINA.dll`.

> **Note:** The VirtualBox GINA module is implemented as a wrapper around the standard Windows GINA module (`MSGINA.DLL`). As a result, it will most likely not work correctly with 3rd party GINA modules.

To manually install the VirtualBox credential provider module, extract the Guest Additions (see chapitre 4.2.1.4, *Extraction manuelle de fichiers*, page 61) and copy the file `VBoxCredProv.dll` to the Windows `SYSTEM32` directory. Then, in the registry, create the following keys:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\
           Authentication\Credential Providers\{275D3BCC-22BB-4948-A7F6-3A3054EBA92B}

HKEY_CLASSES_ROOT\CLSID\{275D3BCC-22BB-4948-A7F6-3A3054EBA92B}

HKEY_CLASSES_ROOT\CLSID\{275D3BCC-22BB-4948-A7F6-3A3054EBA92B}\InprocServer32
```

with all default values (the key named `(Default)` in each key) set to `VBoxCredProv`. After that a new string named

```
HKEY_CLASSES_ROOT\CLSID\{275D3BCC-22BB-4948-A7F6-3A3054EBA92B}\InprocServer32\ThreadingModel
```

with a value of `Apartment` has to be created.

To set credentials, use the following command on a *running* VM:

```
VBoxManage controlvm "Windows XP" setcredentials "John Doe" "secretpassword" "DOMTEST"
```

While the VM is running, the credentials can be queried by the VirtualBox logon modules (GINA or credential provider) using the VirtualBox Guest Additions device driver. When Windows is in "logged out" mode, the logon modules will constantly poll for credentials and if they

are present, a logon will be attempted. After retrieving the credentials, the logon modules will erase them so that the above command will have to be repeated for subsequent logons.

For security reasons, credentials are not stored in any persistent manner and will be lost when the VM is reset. Also, the credentials are "write-only", i.e. there is no way to retrieve the credentials from the host side. Credentials can be reset from the host side by setting empty values.

Depending on the particular variant of the Windows guest, the following restrictions apply:

1. For **Windows XP guests,** the logon subsystem needs to be configured to use the classic logon dialog as the VirtualBox GINA module does not support the XP-style welcome dialog.

2. For **Windows Vista and Windows 7 guests,** the logon subsystem does not support the so-called Secure Attention Sequence (`CTRL+ALT+DEL`). As a result, the guest's group policy settings need to be changed to not use the Secure Attention Sequence. Also, the user name given is only compared to the true user name, not the user friendly name. This means that when you rename a user, you still have to supply the original user name (internally, Windows never renames user accounts).

3. Auto-logon handling of the built-in Windows Remote Desktop Service (formerly known as Terminal Services) is disabled by default. To enable it, create the registry key

   `HKEY_LOCAL_MACHINE\SOFTWARE\Oracle\VirtualBox Guest Additions\AutoLogon`

   with a `DWORD` value of `1`.

The following command forces VirtualBox to keep the credentials after they were read by the guest and on VM reset:

```
VBoxManage setextradata "Windows XP" VBoxInternal/Devices/VMMDev/0/Config/KeepCredentials 1
```

Note that this is a potential security risk as a malicious application running on the guest could request this information using the proper interface.

9.2.2 Automated Linux/Unix guest logons

Starting with version 3.2, VirtualBox provides a custom PAM module (Pluggable Authentication Module) which can be used to perform automated guest logons on platforms which support this framework. Virtually all modern Linux/Unix distributions rely on PAM.

The `pam_vbox.so` module itself **does not** do an actual verification of the credentials passed to the guest OS; instead it relies on other modules such as `pam_unix.so` or `pam_unix2.so` down in the PAM stack to do the actual validation using the credentials retrieved by `pam_vbox.so`. Therefore `pam_vbox.so` has to be on top of the authentication PAM service list.

> **Note:** The `pam_vbox.so` only supports the `auth` primitive. Other primitives such as `account`, `session` or `password` are not supported.

The `pam_vbox.so` module is shipped as part of the Guest Additions but it is not installed and/or activated on the guest OS by default. In order to install it, it has to be copied from `/opt/VBoxGuestAdditions-<version>/lib/VBoxGuestAdditions/` to the security modules directory, usually `/lib/security/` on 32-bit guest Linuxes or `/lib64/security/` on 64-bit ones. Please refer to your guest OS documentation for the correct PAM module directory.

For example, to use `pam_vbox.so` with a Ubuntu Linux guest OS and GDM (the GNOME Desktop Manager) to logon users automatically with the credentials passed by the host, the guest OS has to be configured like the following:

1. The `pam_vbox.so` module has to be copied to the security modules directory, in this case it is `/lib/security`.

2. Edit the PAM configuration file for GDM found at `/etc/pam.d/gdm`, adding the line `auth requisite pam_vbox.so` at the top. Additionaly, in most Linux distributions there is a file called `/etc/pam.d/common-auth`. This file is included in many other services (like the GDM file mentioned above). There you also have to add the line `auth requisite pam_vbox.so`.

3. If authentication against the shadow database using `pam_unix.so` or `pam_unix2.so` is desired, the argument `try_first_pass` for `pam_unix.so` or `use_first_pass` for `pam_unix2.so` is needed in order to pass the credentials from the VirtualBox module to the shadow database authentication module. For Ubuntu, this needs to be added to `/etc/pam.d/common-auth`, to the end of the line referencing `pam_unix.so`. This argument tells the PAM module to use credentials already present in the stack, i.e. the ones provided by the VirtualBox PAM module.

Avertissement: An incorrectly configured PAM stack can effectively prevent you from logging into your guest system!

To make deployment easier, you can pass the argument `debug` right after the `pam_vbox.so` statement. Debug log output will then be recorded using syslog.

Avertissement: At present, the GDM display manager only retrieves credentials at startup so unless the credentials have been supplied to the guest before GDM starts, automatic logon will not work. This limitation needs to be addressed by the GDM developers or another display manager must be used.

9.3 Advanced configuration for Windows guests

9.3.1 Automated Windows system preparation

Beginning with Windows NT 4.0, Microsoft offers a “system preparation” tool (in short: Sysprep) to prepare a Windows system for deployment or redistribution. Whereas Windows 2000 and XP ship with Sysprep on the installation medium, the tool also is available for download on the Microsoft web site. In a standard installation of Windows Vista and 7, Sysprep is already included. Sysprep mainly consists of an executable called `sysprep.exe` which is invoked by the user to put the Windows installation into preparation mode.

Starting with VirtualBox 3.2.2, the Guest Additions offer a way to launch a system preparation on the guest operating system in an automated way, controlled from the host system. To achieve that, see chapitre 4.7, *Contrôle invité*, page 75 for using the feature with the special identifier `sysprep` as the program to execute, along with the user name `sysprep` and password `sysprep` for the credentials. Sysprep then gets launched with the required system rights.

Note: Specifying the location of "sysprep.exe" is **not possible** – instead the following paths are used (based on the operating system):

- `C:\sysprep\sysprep.exe` for Windows NT 4.0, 2000 and XP
- `%WINDIR%\System32\Sysprep\sysprep.exe` for Windows Vista, 2008 Server and 7

The Guest Additions will automatically use the appropriate path to execute the system preparation tool.

9.4 Advanced configuration for Linux and Solaris guests

9.4.1 Manual setup of selected guest services on Linux

The VirtualBox Guest Additions contain several different drivers. If for any reason you do not wish to set them all up, you can install the Guest Additions using the following command:

```
sh ./VBoxLinuxAdditions.run no_setup
```

After this, you will need to at least compile the kernel modules by running the command

```
/usr/lib/VBoxGuestAdditions/vboxadd setup
```

as root (you will need to replace *lib* by *lib64* on some 64bit guests), and on older guests without the udev service you will need to add the *vboxadd* service to the default runlevel to ensure that the modules get loaded.

To setup the time synchronization service, run the command

```
/usr/lib/VBoxGuestAdditions/vboxadd-service setup
```

and add the service vboxadd-service to the default runlevel. To set up the X11 and OpenGL part of the Guest Additions, run the command

```
/usr/lib/VBoxGuestAdditions/vboxadd-x11 setup
```

(you do not need to enable any services for this).

To recompile the guest kernel modules, use this command:

```
/usr/lib/VBoxGuestAdditions/vboxadd setup
```

After compilation you should reboot your guest to ensure that the new modules are actually used.

9.4.2 Guest graphics and mouse driver setup in depth

This section assumes that you are familiar with configuring the X.Org server using xorg.conf and optionally the newer mechanisms using hal or udev and xorg.conf.d. If not you can learn about them by studying the documentation which comes with X.Org.

The VirtualBox Guest Additions come with drivers for X.Org versions

- X11R6.8/X11R6.9 and XFree86 version 4.3 (vboxvideo_drv_68.o and vboxmouse_drv_68.o)
- X11R7.0 (vboxvideo_drv_70.so and vboxmouse_drv_70.so)
- X11R7.1 (vboxvideo_drv_71.so and vboxmouse_drv_71.so)

- X.Org Server versions 1.3 and later (vboxvideo_drv_13.so and vboxmouse_drv_13.so and so on).

By default these drivers can be found in the directory
`/opt/VBoxGuestAdditions-<version>/lib/VBoxGuestAdditions`
and the correct versions for the X server are symbolically linked into the X.Org driver directories.

For graphics integration to work correctly, the X server must load the vboxvideo driver (many recent X server versions look for it automatically if they see that they are running in VirtualBox) and for an optimal user experience the guest kernel drivers must be loaded and the Guest Additions tool VBoxClient must be running as a client in the X session. For mouse integration to work correctly, the guest kernel drivers must be loaded and in addition, in X servers from X.Org X11R6.8 to X11R7.1 and in XFree86 version 4.3 the right vboxmouse driver must be loaded and associated with /dev/mouse or /dev/psaux; in X.Org server 1.3 or later a driver for a PS/2 mouse must be loaded and the right vboxmouse driver must be associated with /dev/vboxguest.

The VirtualBox guest graphics driver can use any graphics configuration for which the virtual resolution fits into the virtual video memory allocated to the virtual machine (minus a small amount used by the guest driver) as described in chapitre 3.5, *Paramètres d'affichage*, page 50. The driver will offer a range of standard modes at least up to the default guest resolution for all active guest monitors. In X.Org Server 1.3 and later the default mode can be changed by setting the output property VBOX_MODE to "<width>x<height>" for any guest monitor. When VBoxClient and the kernel drivers are active this is done automatically when the host requests a mode change. The driver for older versions can only receive new modes by querying the host for requests at regular intervals.

With pre-1.3 X Servers you can also add your own modes to the X server configuration file. You simply need to add them to the "Modes" list in the "Display" subsection of the "Screen" section. For example, the section shown here has a custom 2048x800 resolution mode added:

```
Section "Screen"
        Identifier    "Default Screen"
        Device        "VirtualBox graphics card"
        Monitor       "Generic Monitor"
        DefaultDepth  24
        SubSection "Display"
                Depth         24
                Modes         "2048x800" "800x600" "640x480"
        EndSubSection
EndSection
```

9.5 CPU hot-plugging

With virtual machines running modern server operating systems, VirtualBox supports CPU hot-plugging.[1] Whereas on a physical computer this would mean that a CPU can be added or removed while the machine is running, VirtualBox supports adding and removing virtual CPUs while a virtual machine is running.

CPU hot-plugging works only with guest operating systems that support it. So far this applies only to Linux and Windows Server 2008 x64 Data Center Edition. Windows supports only hot-add while Linux supports hot-add and hot-remove but to use this feature with more than 8 CPUs a 64bit Linux guest is required.

At this time, CPU hot-plugging requires using the VBoxManage command-line interface. First, hot-plugging needs to be enabled for a virtual machine:

```
VBoxManage modifyvm "VM name" --cpuhotplug on
```

[1] Support for CPU hot-plugging was introduced with VirtualBox 3.2.

After that, the –cpus option specifies the maximum number of CPUs that the virtual machine can have:

```
VBoxManage modifyvm "VM name" --cpus 8
```

When the VM is off, you can then add and remove virtual CPUs with the modifyvm –plugcpu and –unplugcpu subcommands, which take the number of the virtual CPU as a parameter, like this:

```
VBoxManage modifyvm "VM name" --plugcpu 3
VBoxManage modifyvm "VM name" --unplugcpu 3
```

Note that CPU 0 can never be removed.

While the VM is running, CPUs can be added with the `controlvm plugcpu/unplugcpu` commands instead:

```
VBoxManage controlvm "VM name" plugcpu 3
VBoxManage controlvm "VM name" unplugcpu 3
```

See chapitre 8.7, *VBoxManage modifyvm*, page 120 and chapitre 8.12, *VBoxManage controlvm*, page 129 for details.

With Linux guests, the following applies: To prevent ejection while the CPU is still used it has to be ejected from within the guest before. The Linux Guest Additions contain a service which receives hot-remove events and ejects the CPU. Also, after a CPU is added to the VM it is not automatically used by Linux. The Linux Guest Additions service will take care of that if installed. If not a CPU can be started with the following command:

```
echo 1 > /sys/devices/system/cpu/cpu<id>/online
```

9.6 PCI passthrough

When running on Linux hosts, with a recent enough kernel (at least version `2.6.31`) experimental host PCI devices passthrough is available.[2]

> **Note:** The PCI passthrough module is shipped as a VirtualBox extension package, which must be installed separately. See chapitre 1.5, *Installer et lancer VirtualBox*, page 15 for more information.

Essentially this feature allows to directly use physical PCI devices on the host by the guest even if host doesn't have drivers for this particular device. Both, regular PCI and some PCI Express cards, are supported. AGP and certain PCI Express cards are not supported at the moment if they rely on GART (Graphics Address Remapping Table) unit programming for texture management as it does rather nontrivial operations with pages remapping interfering with IOMMU. This limitation may be lifted in future releases.

To be fully functional, PCI passthrough support in VirtualBox depends upon an IOMMU hardware unit which is not yet too widely available. If the device uses bus mastering (i.e. it performs DMA to the OS memory on its own), then an IOMMU is required, otherwise such DMA transactions may write to the wrong physical memory address as the device DMA engine is programmed using a device-specific protocol to perform memory transactions. The IOMMU functions as translation unit mapping physical memory access requests from the device using knowledge of the guest physical address to host physical addresses translation rules.

Intel's solution for IOMMU is marketed as "Intel Virtualization Technology for Directed I/O" (VT-d), and AMD's one is called AMD-Vi. So please check if your motherboard datasheet has appropriate technology. Even if your hardware doesn't have a IOMMU, certain PCI cards may

[2] Experimental support for PCI passthrough was introduced with VirtualBox 4.1.

work (such as serial PCI adapters), but the guest will show a warning on boot and the VM execution will terminate if the guest driver will attempt to enable card bus mastering.

It is very common that the BIOS or the host OS disables the IOMMU by default. So before any attempt to use it please make sure that

1. Your motherboard has an IOMMU unit.
2. Your CPU supports the IOMMU.
3. The IOMMU is enabled in the BIOS.
4. The VM must run with VT-x/AMD-V and nested paging enabled.
5. Your Linux kernel was compiled with IOMMU support (including DMA remapping, see `CONFIG_DMAR` kernel compilation option). The PCI stub driver (`CONFIG_PCI_STUB`) is required as well.
6. Your Linux kernel recognizes and uses the IOMMU unit (`intel_iommu=on` boot option could be needed). Search for DMAR and PCI-DMA in kernel boot log.

Once you made sure that the host kernel supports the IOMMU, the next step is to select the PCI card and attach it to the guest. To figure out the list of available PCI devices, use the `lspci` command. The output will look like this

```
01:00.0 VGA compatible controller: ATI Technologies Inc Cedar PRO [Radeon HD 5450]
01:00.1 Audio device: ATI Technologies Inc Manhattan HDMI Audio [Mobility Radeon HD 5000 Series]
02:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL8111/8168B PCI Express Gigabit Eth
03:00.0 SATA controller: JMicron Technology Corp. JMB362/JMB363 Serial ATA Controller (rev 03)
03:00.1 IDE interface: JMicron Technology Corp. JMB362/JMB363 Serial ATA Controller (rev 03)
06:00.0 VGA compatible controller: nVidia Corporation G86 [GeForce 8500 GT] (rev a1)
```

The first column is a PCI address (in format `bus:device.function`). This address could be used to identify the device for further operations. For example, to attach a PCI network controller on the system listed above to the second PCI bus in the guest, as device 5, function 0, use the following command:

```
VBoxManage modifyvm "VM name" --pciattach 02:00.0@01:05.0
```

To detach same device, use

```
VBoxManage modifyvm "VM name" --pcidetach 02:00.0
```

Please note that both host and guest could freely assign a different PCI address to the card attached during runtime, so those addresses only apply to the address of the card at the moment of attachment (host), and during BIOS PCI init (guest).

If the virtual machine has a PCI device attached, certain limitations apply:

1. Only PCI cards with non-shared interrupts (such as using MSI on host) are supported at the moment.
2. No guest state can be reliably saved/restored (as the internal state of the PCI card could not be retrieved).
3. Teleportation (live migration) doesn't work (for the same reason).
4. No lazy physical memory allocation. The host will preallocate the whole RAM required for the VM on startup (as we cannot catch physical hardware accesses to the physical memory).

9.7 Advanced display configuration

9.7.1 Custom VESA resolutions

Apart from the standard VESA resolutions, the VirtualBox VESA BIOS allows you to add up to 16 custom video modes which will be reported to the guest operating system. When using Windows guests with the VirtualBox Guest Additions, a custom graphics driver will be used instead of the fallback VESA solution so this information does not apply.

Additional video modes can be configured for each VM using the extra data facility. The extra data key is called CustomVideoMode<x> with x being a number from 1 to 16. Please note that modes will be read from 1 until either the following number is not defined or 16 is reached. The following example adds a video mode that corresponds to the native display resolution of many notebook computers:

```
VBoxManage setextradata "VM name" "CustomVideoMode1" "1400x1050x16"
```

The VESA mode IDs for custom video modes start at 0x160. In order to use the above defined custom video mode, the following command line has be supplied to Linux:

```
vga = 0x200 | 0x160
vga = 864
```

For guest operating systems with VirtualBox Guest Additions, a custom video mode can be set using the video mode hint feature.

9.7.2 Configuring the maximum resolution of guests when using the graphical frontend

When guest systems with the Guest Additions installed are started using the graphical frontend (the normal VirtualBox application), they will not be allowed to use screen resolutions greater than the host's screen size unless the user manually resizes them by dragging the window, switching to fullscreen or seamless mode or sending a video mode hint using VBoxManage. This behavior is what most users will want, but if you have different needs, it is possible to change it by issuing one of the following commands from the command line:

```
VBoxManage setextradata global GUI/MaxGuestResolution any
```

will remove all limits on guest resolutions.

```
VBoxManage setextradata global GUI/MaxGuestResolution >width,height<
```

manually specifies a maximum resolution.

```
VBoxManage setextradata global GUI/MaxGuestResolution auto
```

restores the default settings. Note that these settings apply globally to all guest systems, not just to a single machine.

9.8 Advanced storage configuration

9.8.1 Using a raw host hard disk from a guest

Starting with version 1.4, as an alternative to using virtual disk images (as described in detail in chapitre 5, *Stockage virtuel*, page 78), VirtualBox can also present either entire physical hard disks or selected partitions thereof as virtual disks to virtual machines.

With VirtualBox, this type of access is called "raw hard disk access"; it allows a guest operating system to access its virtual hard disk without going through the host OS file system. The actual

performance difference for image files vs. raw disk varies greatly depending on the overhead of the host file system, whether dynamically growing images are used and on host OS caching strategies. The caching indirectly also affects other aspects such as failure behavior, i.e. whether the virtual disk contains all data written before a host OS crash. Consult your host OS documentation for details on this.

> **Avertissement:** Raw hard disk access is for expert users only. Incorrect use or use of an outdated configuration can lead to **total loss of data** on the physical disk. Most importantly, *do not* attempt to boot the partition with the currently running host operating system in a guest. This will lead to severe data corruption.

Raw hard disk access – both for entire disks and individual partitions – is implemented as part of the VMDK image format support. As a result, you will need to create a special VMDK image file which defines where the data will be stored. After creating such a special VMDK image, you can use it like a regular virtual disk image. For example, you can use the Virtual Media Manager (chapitre 5.3, *Le gestionnaire de médias virtuels*, page 82) or `VBoxManage` to assign the image to a virtual machine.

9.8.1.1 Access to entire physical hard disk

While this variant is the simplest to set up, you must be aware that this will give a guest operating system direct and full access to an *entire physical disk*. If your *host* operating system is also booted from this disk, please take special care to not access the partition from the guest at all. On the positive side, the physical disk can be repartitioned in arbitrary ways without having to recreate the image file that gives access to the raw disk.

To create an image that represents an entire physical hard disk (which will not contain any actual data, as this will all be stored on the physical disk), on a Linux host, use the command

```
VBoxManage internalcommands createrawvmdk -filename /path/to/file.vmdk
      -rawdisk /dev/sda
```

This creates the image `/path/to/file.vmdk` (must be absolute), and all data will be read and written from `/dev/sda`.

On a Windows host, instead of the above device specification, use e.g. `\\.\PhysicalDrive0`. On a Mac OS X host, instead of the above device specification use e.g. `/dev/disk1`. Note that on OS X you can only get access to an entire disk if no volume is mounted from it.

Creating the image requires read/write access for the given device. Read/write access is also later needed when using the image from a virtual machine.

Just like with regular disk images, this does not automatically attach the newly created image to a virtual machine. This can be done with e.g.

```
VBoxManage storageattach WindowsXP --storagectl "IDE Controller"
      --port 0 --device 0 --type hdd --medium /path/to/file.vmdk
```

When this is done the selected virtual machine will boot from the specified physical disk.

9.8.1.2 Access to individual physical hard disk partitions

This “raw partition support” is quite similar to the “full hard disk” access described above. However, in this case, any partitioning information will be stored inside the VMDK image, so you can e.g. install a different boot loader in the virtual hard disk without affecting the host’s partitioning information. While the guest will be able to *see* all partitions that exist on the physical disk, access will be filtered in that reading from partitions for which no access is allowed the partitions will only yield zeroes, and all writes to them are ignored.

To create a special image for raw partition support (which will contain a small amount of data, as already mentioned), on a Linux host, use the command

```
VBoxManage internalcommands createrawvmdk -filename /path/to/file.vmdk
      -rawdisk /dev/sda -partitions 1,5
```

As you can see, the command is identical to the one for "full hard disk" access, except for the additional `-partitions` parameter. This example would create the image `/path/to/file.vmdk` (which, again, must be absolute), and partitions 1 and 5 of `/dev/sda` would be made accessible to the guest.

VirtualBox uses the same partition numbering as your Linux host. As a result, the numbers given in the above example would refer to the first primary partition and the first logical drive in the extended partition, respectively.

On a Windows host, instead of the above device specification, use e.g. `\\.\PhysicalDrive0`. On a Mac OS X host, instead of the above device specification use e.g. `/dev/disk1`. Note that on OS X you can only use partitions which are not mounted (eject the respective volume first). Partition numbers are the same on Linux, Windows and Mac OS X hosts.

The numbers for the list of partitions can be taken from the output of

```
VBoxManage internalcommands listpartitions -rawdisk /dev/sda
```

The output lists the partition types and sizes to give the user enough information to identify the partitions necessary for the guest.

Images which give access to individual partitions are specific to a particular host disk setup. You cannot transfer these images to another host; also, whenever the host partitioning changes, the image *must be recreated*.

Creating the image requires read/write access for the given device. Read/write access is also later needed when using the image from a virtual machine. If this is not feasible, there is a special variant for raw partition access (currently only available on Linux hosts) that avoids having to give the current user access to the entire disk. To set up such an image, use

```
VBoxManage internalcommands createrawvmdk -filename /path/to/file.vmdk
      -rawdisk /dev/sda -partitions 1,5 -relative
```

When used from a virtual machine, the image will then refer not to the entire disk, but only to the individual partitions (in the example `/dev/sda1` and `/dev/sda5`). As a consequence, read/write access is only required for the affected partitions, not for the entire disk. During creation however, read-only access to the entire disk is required to obtain the partitioning information.

In some configurations it may be necessary to change the MBR code of the created image, e.g. to replace the Linux boot loader that is used on the host by another boot loader. This allows e.g. the guest to boot directly to Windows, while the host boots Linux from the "same" disk. For this purpose the `-mbr` parameter is provided. It specifies a file name from which to take the MBR code. The partition table is not modified at all, so a MBR file from a system with totally different partitioning can be used. An example of this is

```
VBoxManage internalcommands createrawvmdk -filename /path/to/file.vmdk
      -rawdisk /dev/sda -partitions 1,5 -mbr winxp.mbr
```

The modified MBR will be stored inside the image, not on the host disk.

The created image can be attached to a storage controller in a VM configuration as usual.

9.8.2 Configuring the hard disk vendor product data (VPD)

VirtualBox reports vendor product data for its virtual hard disks which consist of hard disk serial number, firmware revision and model number. These can be changed using the following commands:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/SerialNumber" "serial"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/FirmwareRevision" "firmware"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/ModelNumber" "model"
```

The serial number is a 20 byte alphanumeric string, the firmware revision an 8 byte alphanumeric string and the model number a 40 byte alphanumeric string. Instead of "Port0" (referring to the first port), specify the desired SATA hard disk port.

The above commands apply to virtual machines with an AHCI (SATA) controller. The commands for virtual machines with an IDE controller are:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/piix3ide/0/Config/PrimaryMaster/SerialNumber" "serial"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/piix3ide/0/Config/PrimaryMaster/FirmwareRevision" "firmware"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/piix3ide/0/Config/PrimaryMaster/ModelNumber" "model"
```

For hard disks it's also possible (experimental!) to mark the drive as having a non-rotational medium with:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/NonRotational" "1"
```

Additional three parameters are needed for CD/DVD drives to report the vendor product data:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/ATAPIVendorId" "vendor"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/ATAPIProductId" "product"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/ATAPIRevision" "revision"
```

The vendor id is an 8 byte alphanumeric string, the product id an 16 byte alphanumeric string and the revision a 4 byte alphanumeric string. Instead of "Port0" (referring to the first port), specify the desired SATA hard disk port.

9.8.3 Accès à des cibles iSCSI à travers le réseau interne

En tant que fonctionnalité expérimentale, VirtualBox permet l'accès à une cible iSCSI en exécutant dans une machine virtuelle ce qui est configuré pour utiliser le mode Réseau interne (comme décrit au chapitre 6.5, *Réseau interne*, page 96). Le paramétrage de la machine virtuelle qui utilise une telle cible iSCSI se fait comme décrit ci-dessous. La seule différence est que l'adresse IP de la cible doit être spécifiée comme adresse IP numérique.

La pile IP qui accède au réseau interne doit être configurée dans la machine virtuelle qui accède à la cible iSCSI. Vous devez choisir une adresse IP statique libre et une adresse MAC non utilisée par d'autres machines virtuelles. Dans l'exemple ci-dessous, adaptez le nom de la machine virtuelle, l'adresse MAC et la configuration IP et le nom du réseau interne (« MyIntNet ») selon vos besoins. Vous devez exécuter les sept commandes suivantes :

```
VBoxManage setextradata "nom VM" VBoxInternal/Devices/IntNetIP/0/Trusted 1
VBoxManage setextradata "nom VM" VBoxInternal/Devices/IntNetIP/0/Config/MAC 08:00:27:01:02:0f
VBoxManage setextradata "nom VM" VBoxInternal/Devices/IntNetIP/0/Config/IP 10.0.9.1
VBoxManage setextradata "nom VM" VBoxInternal/Devices/IntNetIP/0/Config/Netmask 255.255.255.0
VBoxManage setextradata "nom VM" VBoxInternal/Devices/IntNetIP/0/LUN#0/Driver IntNet
VBoxManage setextradata "nom VM" VBoxInternal/Devices/IntNetIP/0/LUN#0/Config/Network MyIntNet
VBoxManage setextradata "nom VM" VBoxInternal/Devices/IntNetIP/0/LUN#0/Config/IsService 1
```

Enfin, le disque iSCSI doit être enregistré avec l'option `--intnet` pour dire à l'initiateur iSCSI d'utiliser le réseau interne :

```
VBoxManage addiscsidisk --server 10.0.9.30 --target iqn.2008-12.com.sun:sampletarget --intnet
```

L'adresse cible doit être spécifiée en tant qu'adresse IP numérique vu qu'il n'y a pas de résolution DNS pour le réseau interne.

La machine virtuelle avec la cible iSCSI devrait être démarrée avant que la machine qui l'utilise ne soit allumée. Si une machine virtuelle utilisant un disque iSCSI est démarrée sans que la cible iSCSI ne soit allumée, cela peut prendre jusqu'à 200 secondes pour détecter cette situation. La VM échouera pour s'allumer.

9.9 Launching more than 120 VMs on Solaris hosts

Solaris hosts have a fixed number of IPC semaphores IDs per process preventing users from starting more than 120 VMs. While trying to launch more VMs you would be shown a "Cannot create IPC semaphore" error.

In order to run more VMs, you will need to bump the semaphore ID limit of the VBoxSVC process. Execute as root the `prctl` command as shown below. The process ID of VBoxSVC can be obtained using the `ps` list command.

```
prctl -r -n project.max-sem-ids -v 2048 <pid-of-VBoxSVC>
```

9.10 Legacy commands for using serial ports

Starting with version 1.4, VirtualBox provided support for virtual serial ports, which, at the time, was rather complicated to set up with a sequence of `VBoxManage setextradata` statements. Since version 1.5, that way of setting up serial ports is no longer necessary and *deprecated*. To set up virtual serial ports, use the methods now described in chapitre 3.9, *Ports série*, page 53.

> **Note:** For backwards compatibility, the old `setextradata` statements, whose description is retained below from the old version of the manual, take *precedence* over the new way of configuring serial ports. As a result, if configuring serial ports the new way doesn't work, make sure the VM in question does not have old configuration data such as below still active.

The old sequence of configuring a serial port used the following 6 commands:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/Config/IRQ" 4
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/Config/IOBase" 0x3f8
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/LUN#0/Driver" Char
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/LUN#0/AttachedDriver/Driver" NamedPipe
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/LUN#0/AttachedDriver/Config/Location" "\\.\pipe\vboxCOM1"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/LUN#0/AttachedDriver/Config/IsServer" 1
```

This sets up a serial port in the guest with the default settings for COM1 (IRQ 4, I/O address 0x3f8) and the `Location` setting assumes that this configuration is used on a Windows host, because the Windows named pipe syntax is used. Keep in mind that on Windows hosts a named pipe must always start with `\\.\pipe\`. On Linux the same config settings apply, except that the path name for the `Location` can be chosen more freely. Local domain sockets can be placed anywhere, provided the user running VirtualBox has the permission to create a new file in the directory. The final command above defines that VirtualBox acts as a server, i.e. it creates the named pipe itself instead of connecting to an already existing one.

9.11 Fine-tuning the VirtualBox NAT engine

9.11.1 Configuring the address of a NAT network interface

In NAT mode, the guest network interface is assigned to the IPv4 range `10.0.x.0/24` by default where `x` corresponds to the instance of the NAT interface +2. So `x` is 2 when there is only one NAT instance active. In that case the guest is assigned to the address `10.0.2.15`, the gateway is set to `10.0.2.2` and the name server can be found at `10.0.2.3`.

If, for any reason, the NAT network needs to be changed, this can be achieved with the following command:

```
VBoxManage modifyvm "VM name" --natnet1 "192.168/16"
```

This command would reserve the network addresses from `192.168.0.0` to `192.168.254.254` for the first NAT network instance of "VM name". The guest IP would be assigned to `192.168.0.15` and the default gateway could be found at `192.168.0.2`.

9.11.2 Configuring the boot server (next server) of a NAT network interface

For network booting in NAT mode, by default VirtualBox uses a built-in TFTP server at the IP address 10.0.2.3. This default behavior should work fine for typical remote-booting scenarios. However, it is possible to change the boot server IP and the location of the boot image with the following commands:

```
VBoxManage modifyvm "VM name" --nattftpserver1 10.0.2.2
VBoxManage modifyvm "VM name" --nattftpfile1 /srv/tftp/boot/MyPXEBoot.pxe
```

9.11.3 Tuning TCP/IP buffers for NAT

The VirtualBox NAT stack performance is often determined by its interaction with the host's TCP/IP stack and the size of several buffers (`SO_RCVBUF` and `SO_SNDBUF`). For certain setups users might want to adjust the buffer size for a better performance. This can by achieved using the following commands (values are in kilobytes and can range from 8 to 1024):

```
VBoxManage modifyvm "VM name" --natsettings1 16000,128,128,0,0
```

This example illustrates tuning the NAT settings. The first parameter is the MTU, then the size of the socket's send buffer and the size of the socket's receive buffer, the initial size of the TCP send window, and lastly the initial size of the TCP receive window. Note that specifying zero means fallback to the default value.

Each of these buffers has a default size of 64KB and default MTU is 1500.

9.11.4 Binding NAT sockets to a specific interface

By default, VirtualBox's NAT engine will route TCP/IP packets through the default interface assigned by the host's TCP/IP stack. (The technical reason for this is that the NAT engine uses sockets for communication.) If, for some reason, you want to change this behavior, you can tell the NAT engine to bind to a particular IP address instead. Use the following command:

```
VBoxManage modifyvm "VM name" --natbindip1 "10.45.0.2"
```

After this, all outgoing traffic will be sent through the interface with the IP address 10.45.0.2. Please make sure that this interface is up and running prior to this assignment.

9.11.5 Enabling DNS proxy in NAT mode

The NAT engine by default offers the same DNS servers to the guest that are configured on the host. In some scenarios, it can be desirable to hide the DNS server IPs from the guest, for example when this information can change on the host due to expiring DHCP leases. In this case, you can tell the NAT engine to act as DNS proxy using the following command:

```
VBoxManage modifyvm "VM name" --natdnsproxy1 on
```

9.11.6 Using the host's resolver as a DNS proxy in NAT mode

For resolving network names, the DHCP server of the NAT engine offers a list of registered DNS servers of the host. If for some reason you need to hide this DNS server list and use the host's resolver settings, thereby forcing the VirtualBox NAT engine to intercept DNS requests and forward them to host's resolver, use the following command:

```
VBoxManage modifyvm "VM name" --natdnshostresolver1 on
```

Note that this setting is similar to the DNS proxy mode, however whereas the proxy mode just forwards DNS requests to the appropriate servers, the resolver mode will interpret the DNS requests and use the host's DNS API to query the information and return it to the guest.

9.11.7 Configuring aliasing of the NAT engine

By default, the NAT core uses aliasing and uses random ports when generating an alias for a connection. This works well for the most protocols like SSH, FTP and so on. Though some protocols might need a more transparent behavior or may depend on the real port number the packet was sent from. It is possible to change the NAT mode via the VBoxManage frontend with the following commands:

```
VBoxManage modifyvm "VM name" --nataliasmode1 proxyonly
```

and

```
VBoxManage modifyvm "Linux Guest" --nataliasmode1 sameports
```

The first example disables aliasing and switches NAT into transparent mode, the second example enforces preserving of port values. These modes can be combined if necessary.

9.12 Configuring the BIOS DMI information

The DMI data VirtualBox provides to guests can be changed for a specific VM. Use the following commands to configure the DMI BIOS information:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSVendor"        "BIOS Vendor"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSVersion"       "BIOS Version"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSReleaseDate"   "BIOS Release Date"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSReleaseMajor"  1
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSReleaseMinor"  2
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSFirmwareMajor" 3
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSFirmwareMinor" 4
VBoxManage setextradata "VM name"
```

```
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemVendor"      "System Vendor"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemProduct"     "System Product"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemVersion"     "System Version"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemSerial"      "System Serial"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemSKU"         "System SKU"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemFamily"      "System Family"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemUuid"
                                          "9852bf98-b83c-49db-a8de-182c42c7226b"
```

If a DMI string is not set, the default value of VirtualBox is used. To set an empty string use "<EMPTY>".

Note that in the above list, all quoted parameters (DmiBIOSVendor, DmiBIOSVersion but not DmiBIOSReleaseMajor) are expected to be strings. If such a string is a valid number, the parameter is treated as number and the VM will most probably refuse to start with an VERR_CFGM_NOT_STRING error. In that case, use "string:<value>", for instance

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemSerial"      "string:1234"
```

Changing this information can be necessary to provide the DMI information of the host to the guest to prevent Windows from asking for a new product key. On Linux hosts the DMI BIOS information can be obtained with

```
dmidecode -t0
```

and the DMI system information can be obtained with

```
dmidecode -t1
```

9.13 Fine-tuning timers and time synchronization

9.13.1 Configuring the guest time stamp counter (TSC) to reflect guest execution

By default, VirtualBox keeps all sources of time visible to the guest synchronized to a single time source, the monotonic host time. This reflects the assumptions of many guest operating systems, which expect all time sources to reflect “wall clock” time. In special circumstances it may be useful however to make the TSC (time stamp counter) in the guest reflect the time actually spent executing the guest.

This special TSC handling mode can be enabled on a per-VM basis, and for best results must be used only in combination with hardware virtualization. To enable this mode use the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/TM/TSCTiedToExecution" 1
```

To revert to the default TSC handling mode use:

```
VBoxManage setextradata "VM name" "VBoxInternal/TM/TSCTiedToExecution"
```

Note that if you use the special TSC handling mode with a guest operating system which is very strict about the consistency of time sources you may get a warning or error message about the timing inconsistency. It may also cause clocks to become unreliable with some guest operating systems depending on they use the TSC.

9.13.2 Accelerate or slow down the guest clock

For certain purposes it can be useful to accelerate or to slow down the (virtual) guest clock. This can be achieved as follows:

```
VBoxManage setextradata "VM name" "VBoxInternal/TM/WarpDrivePercentage" 200
```

The above example will double the speed of the guest clock while

```
VBoxManage setextradata "VM name" "VBoxInternal/TM/WarpDrivePercentage" 50
```

will halve the speed of the guest clock. Note that changing the rate of the virtual clock can confuse the guest and can even lead to abnormal guest behavior. For instance, a higher clock rate means shorter timeouts for virtual devices with the result that a slightly increased response time of a virtual device due to an increased host load can cause guest failures. Note further that any time synchronization mechanism will frequently try to resynchronize the guest clock with the reference clock (which is the host clock if the VirtualBox Guest Additions are active). Therefore any time synchronization should be disabled if the rate of the guest clock is changed as described above (see chapitre 9.13.3, *Tuning the Guest Additions time synchronization parameters*, page 165).

9.13.3 Tuning the Guest Additions time synchronization parameters

The VirtualBox Guest Additions ensure that the guest's system time is synchronized with the host time. There are several parameters which can be tuned. The parameters can be set for a specific VM using the following command:

```
VBoxManage guestproperty set VM_NAME "/VirtualBox/GuestAdd/VBoxService/PARAMETER" VALUE
```

where `PARAMETER` is one of the following:

`--timesync-interval` Specifies the interval at which to synchronize the time with the host. The default is 10000 ms (10 seconds).

`--timesync-min-adjust` The minimum absolute drift value measured in milliseconds to make adjustments for. The default is 1000 ms on OS/2 and 100 ms elsewhere.

`--timesync-latency-factor` The factor to multiply the time query latency with to calculate the dynamic minimum adjust time. The default is 8 times, that means in detail: Measure the time it takes to determine the host time (the guest has to contact the VM host service which may take some time), multiply this value by 8 and do an adjustment only if the time difference between host and guest is bigger than this value. Don't do any time adjustment otherwise.

`--timesync-max-latency` The max host timer query latency to accept. The default is 250 ms.

`--timesync-set-threshold` The absolute drift threshold, given as milliseconds where to start setting the time instead of trying to smoothly adjust it. The default is 20 minutes.

`--timesync-set-start` Set the time when starting the time sync service.

`--timesync-set-on-restore 0|1` Set the time after the VM was restored from a saved state when passing 1 as parameter (default). Disable by passing 0. In the latter case, the time will be adjusted smoothly which can take a long time.

All these parameters can be specified as command line parameters to VBoxService as well.

9.14 Installing the alternate bridged networking driver on Solaris 11 hosts

Starting with VirtualBox 4.1, VirtualBox ships a new network filter driver that utilizes Solaris 11's Crossbow functionality. By default, this new driver is installed for Solaris 11 hosts (builds 159 and above) that has support for it.

To force installation of the older STREAMS based network filter driver, execute as root execute the below command before installing the VirtualBox package:

```
touch /etc/vboxinst_vboxflt
```

To force installation of the Crossbow based network filter driver, execute as root the below command before installing the VirtualBox package:

```
touch /etc/vboxinst_vboxbow
```

To check which driver is currently being used by VirtualBox, execute:

```
modinfo | grep vbox
```

If the output contains "vboxbow", it indicates VirtualBox is using the Crossbow network filter driver, while the name "vboxflt" indicates usage of the older STREAMS network filter.

9.15 VirtualBox VNIC templates for VLANs on Solaris 11 hosts

VirtualBox supports VNIC (Virtual Network Interface) templates for configuring VMs over VLANs.[3] A VirtualBox VNIC template is a VNIC whose name starts with "vboxvnic_template".

Here is an example of how to use a VNIC template to configure a VLAN for VMs. Create a VirtualBox VNIC template, by executing as root:

```
dladm create-vnic -t -l nge0 -v 23 vboxvnic_template0
```

This will create a temporary VNIC over interface "nge0" with the VLAN ID 23. To create VNIC templates that are persistent across host reboots, skip the `-t` parameter in the above command. You may check the current state of links using:

```
$ dladm show-link
LINK        CLASS     MTU    STATE    BRIDGE     OVER
nge0        phys      1500   up       --         --
nge1        phys      1500   down     --         --
vboxvnic_template0 vnic 1500 up       --         nge0

$ dladm show-vnic
LINK         OVER         SPEED  MACADDRESS        MACADDRTYPE         VID
vboxvnic_template0 nge0   1000   2:8:20:25:12:75   random              23
```

Once the VNIC template is created, all VMs that need to be part of VLAN 23 over the physical interface "nge0" can use the same VNIC template. This makes managing VMs on VLANs simpler and efficient, as the VLAN details are not stored as part of every VM's configuration but rather picked up via the VNIC template which can be modified anytime using `dladm`. Apart from the VLAN ID, VNIC templates can be created with additional properties such as bandwidth limits, CPU fanout etc. Refer to your Solaris network documentation on how to accomplish this. These additional properties, if any, are also applied to VMs which use the VNIC template.

[3] Support for Crossbow based bridged networking was introduced with VirtualBox 4.1 and requires Solaris 11 build 159 or above.

9.16 Configuring multiple host-only network interfaces on Solaris hosts

By default VirtualBox provides you with one host-only network interface. Adding more host-only network interfaces on Solaris hosts requires manual configuration. Here's how to add two more host-only network interfaces.

You first need to stop all running VMs and unplumb all existing "vboxnet" interfaces. Execute the following commands as root:

```
ifconfig vboxnet0 unplumb
```

Once you make sure all vboxnet interfaces are unplumbed, remove the driver using:

```
rem_drv vboxnet
```

then edit the file `/platform/i86pc/kernel/drv/vboxnet.conf` and add a line for the new interfaces:

```
name="vboxnet" parent="pseudo" instance=1;
name="vboxnet" parent="pseudo" instance=2;
```

Add as many of these lines as required and make sure "instance" number is uniquely incremented. Next reload the vboxnet driver using:

```
add_drv vboxnet
```

Now plumb all the interfaces using `ifconfig vboxnetX plumb` (where X can be 0, 1 or 2 in this case) and once plumbed you can then configure the interface like any other network interface.

To make your newly added interfaces' settings persistent across reboots you will need to edit the files `/etc/netmasks`, and if you are using NWAM `/etc/nwam/llp` and add the appropriate entries to set the netmask and static IP for each of those interfaces. The VirtualBox installer only updates these configuration files for the one "vboxnet0" interface it creates by default.

9.17 Configuring the VirtualBox CoreDumper on Solaris hosts

VirtualBox is capable of producing its own core files when things go wrong and for more extensive debugging. Currently this is only available on Solaris hosts.

The VirtualBox CoreDumper can be enabled using the following command:

```
VBoxManage setextradata "VM name" VBoxInternal2/CoreDumpEnabled 1
```

You can specify which directory to use for core dumps with this command:

```
VBoxManage setextradata "VM name" VBoxInternal2/CoreDumpDir <path-to-directory>
```

Make sure the directory you specify is on a volume with sufficient free space and that the VirtualBox process has sufficient permissions to write files to this directory. If you skip this command and don't specify any core dump directory, the current directory of the VirtualBox executable will be used (which would most likely fail when writing cores as they are protected with root permissions). It is recommended you explicity set a core dump directory.

You must specify when the VirtualBox CoreDumper should be triggered. This is done using the following commands:

```
VBoxManage setextradata "VM name" VBoxInternal2/CoreDumpReplaceSystemDump 1
VBoxManage setextradata "VM name" VBoxInternal2/CoreDumpLive 1
```

At least one of the above two commands will have to be provided if you have enabled the VirtualBox CoreDumper.

Setting `CoreDumpReplaceSystemDump` sets up the VM to override the host's core dumping mechanism and in the event of any crash only the VirtualBox CoreDumper would produce the core file.

Setting `CoreDumpLive` sets up the VM to produce cores whenever the VM receives a `SIGUSR2` signal. After producing the core file, the VM will not be terminated and will continue to run. You can then take cores of the VM process using:

```
kill -s SIGUSR2 <VM-process-id>
```

Core files produced by the VirtualBox CoreDumper are of the form `core.vb.<ProcessName>.<ProcessID>`, e.g.`core.vb.VBoxHeadless.11321`.

9.18 Locking down the VirtualBox manager GUI

There are several advanced customization settings for locking down the VirtualBox manager, that is, removing some features that the user should not see.

```
VBoxManage setextradata global GUI/Customizations OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

`noSelector` Don't allow to start the VirtualBox manager. Trying to do so will show a window containing a proper error message.

`noMenuBar` VM windows will not contain a menu bar.

`noStatusBar` VM windows will not contain a status bar.

To disable any GUI customization do

```
VBoxManage setextradata global GUI/Customizations
```

To disable all host key combinations, open the preferences and change the host key to *None*. This might be useful when using VirtualBox in a kiosk mode.

Furthermore, you can disallow certain actions when terminating a VM. To disallow specific actions, type:

```
VBoxManage setextradata "VM name" GUI/RestrictedCloseActions OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

`SaveState` Don't allow the user to save the VM state when terminating the VM.

`Shutdown` Don't allow the user to shutdown the VM by sending the ACPI power-off event to the guest.

`PowerOff` Don't allow the user to power off the VM.

`Restore` Don't allow the user to return to the last snapshot when powering off the VM.

Any combination of the above is allowed. If all options are specified, the VM cannot be shut down at all.

9.19 Starting the VirtualBox web service automatically

The VirtualBox web service (`vboxwebsrv`) is used for controlling VirtualBox remotely. It is documented in detail in the VirtualBox Software Development Kit (SDK); please see chapitre 11, *VirtualBox programming interfaces*, page 181. As the client base using this interface is growing, we added start scripts for the various operation systems we support. The following describes how to use them.

- On Mac OS X, launchd is used. An example configuration file can be found in `$HOME/Library/LaunchAgents/org.virtualbox.vboxwebsrv.plist`. It can be enabled by changing the `Disabled` key from `true` to `false`. To manually start the service use the following command:

  ```
  launchctl load ~/Library/LaunchAgents/org.virtualbox.vboxwebsrv.plist
  ```

  For additional information on how launchd services could be configured see `http://developer.apple.com/mac/library/documentation/MacOSX/Conceptual/BPSystemStartup/BPSystemStartup.html`.

9.20 Memory Ballooning Service

Starting with VirtualBox 4.0.8 a new host executable called `VBoxBalloonCtrl` is available to automatically take care of a VM's configured memory balloon (see chapitre 4.8, *Faire de la montgolfière avec la mémoire*, page 75 for an introduction to memory ballooning). This is especially useful for server environments where VMs may dynamically require more or less memory during runtime.

VBoxBalloonCtrl periodically checks a VM's current memory balloon and its free guest RAM and automatically adjusts the current memory balloon by inflating or deflating it accordingly. This handling only applies to running VMs having recent Guest Additions installed.

To set up VBoxBalloonCtrl and adjust the maximum ballooning size a VM can reach the following parameters will be checked in the following order:

- specified via VBoxBalloonCtrl command line parameter `--balloon-max`
- per-VM parameter using

  ```
  VBoxManage setextradata "VM-Name" VBoxInternal/Guest/BalloonSizeMax <Size in MB>
  ```

- global parameter for all VMs using

  ```
  VBoxManage setextradata global VBoxInternal/Guest/BalloonSizeMax <Size in MB>
  ```

> **Note:** If no maximum ballooning size is specified by at least one of the parameters above, no ballooning will be performed at all.

For more options and parameters check the built-in command line help accessible with `--help`.

9.21 Starting virtual machines during system boot

Starting with VirtualBox 4.2.0 it is possible to start VMs automatically during system boot on Linux and Mac OS X for all users.

9.21.1 Linux: starting the autostart service via `init`

On Linux, the autostart service is activated by setting two variables in `/etc/default/virtualbox`. The first one is `VBOXAUTOSTART_DB` which contains an absolute path to the autostart database directory. The directory should have write access for every user who should be able to start virtual machines automatically. Furthermore the directory should have the sticky bit set. The second variable is `VBOXAUTOSTART_CONFIG` which points the service to the autostart configuration file which is used during boot to determine whether to allow individual users to start a VM automatically and configure startup delays. The config file can be placed in `/etc/vbox` and contains several options. One is `default_policy` which controls whether the autostart service allows or denies to start a VM for users which are not in the exception list. The exception list starts with `exception_list` and contains a comma seperated list with usernames. Furthermore a separate startup delay can be configured for every user to avoid overloading the host. A sample configuration is given below:

```
# Default policy is to deny starting a VM, the other option is "allow".
default_policy = deny
# Users which are allowed are given below.
# If the default policy is to allow starting a VM is not allowed for the users below.
exception_list = bob, alice, joe

bob   = 30  # Bobs machines will be started 30 seconds after the autostart service started
alice = 180 # Alice machines will be started 180 seconds after the autostart service started
joe   = 240 # Joes machines will be started 240 seconds after the autostart service started
```

Every user who wants to enable autostart for individual machines has to set the path to the autostart database directory with

```
VBoxManage setproperty autostartdbpath <Autostart directory>
```

9.21.2 Solaris: starting the autostart service via SMF

On Solaris hosts, the VirtualBox autostart daemon is integrated into the SMF framework. To enable it you have to point the service to an existing configuration file which has the same format as on Linux (see chapitre 9.21.1, *Linux: starting the autostart service via `init`*, page 170):

```
svccfg -s svc:/application/virtualbox/autostart:default setprop config/config=/etc/vbox/autostart.cfg
```

When everything is configured correctly you can start the VirtualBox autostart service with the following command:

```
svcadm enable svc:/application/virtualbox/autostart:default
```

For more information about SMF, please refer to the Solaris documentation.

9.21.3 Mac OS X: starting the autostart service via launchd

On Mac OS X, launchd is used to start the VirtualBox autostart service. An example configuration file can be found in `/Applications/VirtualBox.app/Contents/MacOS/org.virtualbox.vboxautostar`. To enable the service copy the file to `/Library/LaunchDaemons` and change the `Disabled` key from `true` to `false`. Furthermore replace the second parameter to an existing configuration file which has the same format as on Linux (see chapitre 9.21.1, *Linux: starting the autostart service via `init`*, page 170). To manually start the service use the following command:

```
launchctl load /Library/LaunchDaemons/org.virtualbox.vboxautostart.plist
```

For additional information on how launchd services could be configured see http://developer.apple.com/mac/library/documentation/MacOSX/Conceptual/BPSystemStartup/BPSystemStartup.html.

10 Technical background

The contents of this chapter are not required to use VirtualBox successfully. The following is provided as additional information for readers who are more familiar with computer architecture and technology and wish to find out more about how VirtualBox works "under the hood".

10.1 Where VirtualBox stores its files

In VirtualBox, a virtual machine and its settings are described in a virtual machine settings file in XML format. In addition, most virtual machine have one or more virtual hard disks, which are typically represented by disk images (e.g. in VDI format). Where all these files are stored depends on which version of VirtualBox created the machine.

10.1.1 Machines created by VirtualBox version 4.0 or later

Starting with version 4.0, by default, each virtual machine has one directory on your host computer where all the files of that machine are stored – the XML settings file (with a `.vbox` file extension) and its disk images.

By default, this "machine folder" is placed in a common folder called "VirtualBox VMs", which VirtualBox creates in the current system user's home directory. The location of this home directory depends on the conventions of the host operating system:

- On Windows, this is `%HOMEDRIVE%%HOMEPATH%`; typically something like `C:\Documents and Settings\Username\`.
- On Mac OS X, this is `/Users/username`.
- On Linux and Solaris, this is `/home/username`.

For simplicity, we will abbreviate this as `$HOME` below. Using that convention, the common folder for all virtual machines is `$HOME/VirtualBox VMs`.

As an example, when you create a virtual machine called "Example VM", you will find that VirtualBox creates

1. the folder `$HOME/VirtualBox VMs/Example VM/` and, in that folder,
2. the settings file `Example VM.vbox` and
3. the virtual disk image `Example VM.vdi`.

This is the default layout if you use the "Create new virtual machine" wizard as described in chapitre 1.6, *Créer votre première machine virtuelle*, page 17. Once you start working with the VM, additional files will show up: you will find log files in a subfolder called `Logs`, and once you have taken snapshots, they will appear in a `Snapshots` subfolder. For each VM, you can change the location of its snapsnots folder in the VM settings.

You can change the default machine folder by selecting "Preferences" from the "File" menu in the VirtualBox main window. Then, in the window that pops up, click on the "General" tab. Alternatively, use `VBoxManage setproperty machinefolder`; see chapitre 8.26, *VBoxManage setproperty*, page 138.

10.1.2 Machines created by VirtualBox versions before 4.0

If you have upgraded to VirtualBox 4.0 from an earlier version of VirtualBox, you probably have settings files and disks in the earlier file system layout.

Before version 4.0, VirtualBox separated the machine settings files from virtual disk images. The machine settings files had an `.xml` file extension and resided in a folder called "Machines" under the global VirtualBox configuration directory (see the next section). So, for example, on Linux, this was the hidden `$HOME/.VirtualBox/Machines` directory. The default hard disks folder was called "HardDisks" and resided in the `.VirtualBox` folder as well. Both locations could be changed by the user in the global preferences. (The concept of a "default hard disk folder" has been abandoned with VirtualBox 4.0, since disk images now reside in each machine's folder by default.)

The old layout had several severe disadvantages.

1. It was very difficult to move a virtual machine from one host to another because the files involved did not reside in the same folder. In addition, the virtual media of all machines were registered with a global registry in the central VirtualBox settings file (`$HOME/.VirtualBox/VirtualBox.xml`).

   To move a machine to another host, it was therefore not enough to move the XML settings file and the disk images (which were in different locations), but the hard disk entries from the global media registry XML had to be meticulously copied as well, which was close to impossible if the machine had snapshots and therefore differencing images.

2. Storing virtual disk images, which can grow very large, under the hidden `.VirtualBox` directory (at least on Linux and Solaris hosts) made many users wonder where their disk space had gone.

Whereas new VMs created with VirtualBox 4.0 or later will conform to the new layout, for maximum compatibility, old VMs are *not* converted to the new layout. Otherwise machine settings would be irrevocably broken if a user downgraded from 4.0 back to an older version of VirtualBox.

10.1.3 Global configuration data

In addition to the files of the virtual machines, VirtualBox maintains global configuration data. On Windows, Linux and Solaris, this is in `$HOME/.VirtualBox` (which makes it hidden on Linux and Solaris), whereas on a Mac this resides in `$HOME/Library/VirtualBox`.

VirtualBox creates this configuration directory automatically if necessary. Optionally, you can supply an alternate configuration directory by setting the `VBOX_USER_HOME` environment variable. (Since the global `VirtualBox.xml` settings file points to all other configuration files, this allows for switching between several VirtualBox configurations entirely.)

Most importantly, in this directory, VirtualBox stores its global settings file, another XML file called `VirtualBox.xml`. This includes global configuration options and the list of registered virtual machines with pointers to their XML settings files. (Neither the location of this file nor its directory has changed with VirtualBox 4.0.)

Before VirtualBox 4.0, all virtual media (disk image files) were also contained in a global registry in this settings file. For compatibility, this media registry still exists if you upgrade VirtualBox and there are media from machines which were created with a version before 4.0. If you have no such machines, then there will be no global media registry; with VirtualBox 4.0, each machine XML file has its own media registry.

Also before VirtualBox 4.0, the default "Machines" folder and the default "HardDisks" folder resided under the VirtualBox configuration directory (e.g. `$HOME/.VirtualBox/Machines` on Linux). If you are upgrading from a VirtualBox version before 4.0, files in these directories are not automatically moved in order not to break backwards compatibility.

10.1.4 Summary of 4.0 configuration changes

	Before 4.0	4.0 or above
Default machines folder	`$HOME/.VirtualBox/Machines`	`$HOME/VirtualBox VMs`
Default disk image location	`$HOME/.VirtualBox/HardDisks`	In each machine's folder
Machine settings file extension	`.xml`	`.vbox`
Media registry	Global `VirtualBox.xml` file	Each machine settings file
Media registration	Explicit open/close required	Automatic on attach

10.1.5 VirtualBox XML files

VirtualBox uses XML for both the machine settings files and the global configuration file, `VirtualBox.xml`.

All VirtualBox XML files are versioned. When a new settings file is created (e.g. because a new virtual machine is created), VirtualBox automatically uses the settings format of the current VirtualBox version. These files may not be readable if you downgrade to an earlier version of VirtualBox. However, when VirtualBox encounters a settings file from an earlier version (e.g. after upgrading VirtualBox), it attempts to preserve the settings format as much as possible. It will only silently upgrade the settings format if the current settings cannot be expressed in the old format, for example because you enabled a feature that was not present in an earlier version of VirtualBox.[1] In such cases, VirtualBox backs up the old settings file in the virtual machine's configuration directory. If you need to go back to the earlier version of VirtualBox, then you will need to manually copy these backup files back.

We intentionally do not document the specifications of the VirtualBox XML files, as we must reserve the right to modify them in the future. We therefore strongly suggest that you do not edit these files manually. VirtualBox provides complete access to its configuration data through its the `VBoxManage` command line tool (see chapitre 8, *VBoxManage*, page 110) and its API (see chapitre 11, *VirtualBox programming interfaces*, page 181).

10.2 VirtualBox executables and components

VirtualBox was designed to be modular and flexible. When the VirtualBox graphical user interface (GUI) is opened and a VM is started, at least three processes are running:

1. `VBoxSVC`, the VirtualBox service process which always runs in the background. This process is started automatically by the first VirtualBox client process (the GUI, `VBoxManage`, `VBoxHeadless`, the web service or others) and exits a short time after the last client exits. The service is responsible for bookkeeping, maintaining the state of all VMs, and for providing communication between VirtualBox components. This communication is implemented via COM/XPCOM.

> **Note:** When we refer to "clients" here, we mean the local clients of a particular `VBoxSVC` server process, not clients in a network. VirtualBox employs its own client/server design to allow its processes to cooperate, but all these processes run under the same user account on the host operating system, and this is totally transparent to the user.

[1] As an example, before VirtualBox 3.1, it was only possible to enable or disable a single DVD drive in a virtual machine. If it was enabled, then it would always be visible as the secondary master of the IDE controller. With VirtualBox 3.1, DVD drives can be attached to arbitrary slots of arbitrary controllers, so they could be the secondary slave of an IDE controller or in a SATA slot. If you have a machine settings file from an earlier version and upgrade VirtualBox to 3.1 and then move the DVD drive from its default position, this cannot be expressed in the old settings format; the XML machine file would get written in the new format, and a backup file of the old format would be kept.

2. The GUI process, `VirtualBox`, a client application based on the cross-platform Qt library. When started without the `--startvm` option, this application acts as the VirtualBox manager, displaying the VMs and their settings. It then communicates settings and state changes to `VBoxSVC` and also reflects changes effected through other means, e.g., `VBoxManage`.

3. If the `VirtualBox` client application is started with the `--startvm` argument, it loads the VMM library which includes the actual hypervisor and then runs a virtual machine and provides the input and output for the guest.

Any VirtualBox front-end (client) will communicate with the service process and can both control and reflect the current state. For example, either the VM selector or the VM window or VBoxManage can be used to pause the running VM, and other components will always reflect the changed state.

The VirtualBox GUI application is only one of several available front ends (clients). The complete list shipped with VirtualBox is:

1. `VirtualBox`, the Qt front end implementing the manager and running VMs;

2. `VBoxManage`, a less user-friendly but more powerful alternative, described in chapitre 8, *VBoxManage*, page 110.

3. `VBoxSDL`, a simple graphical front end based on the SDL library; see chapitre 9.1, *VBoxSDL, the simplified VM displayer*, page 148.

4. `VBoxHeadless`, a VM front end which does not directly provide any video output and keyboard/mouse input, but allows redirection via VirtualBox Remote Desktop Extension; see chapitre 7.1.2, *VBoxHeadless, the remote desktop server*, page 102.

5. `vboxwebsrv`, the VirtualBox web service process which allows for controlling a VirtualBox host remotely. This is described in detail in the VirtualBox Software Development Kit (SDK) reference; please see chapitre 11, *VirtualBox programming interfaces*, page 181 for details.

6. The VirtualBox Python shell, a Python alternative to VBoxManage. This is also described in the SDK reference.

Internally, VirtualBox consists of many more or less separate components. You may encounter these when analyzing VirtualBox internal error messages or log files. These include:

- IPRT, a portable runtime library which abstracts file access, threading, string manipulation, etc. Whenever VirtualBox accesses host operating features, it does so through this library for cross-platform portability.

- VMM (Virtual Machine Monitor), the heart of the hypervisor.

- EM (Execution Manager), controls execution of guest code.

- REM (Recompiled Execution Monitor), provides software emulation of CPU instructions.

- TRPM (Trap Manager), intercepts and processes guest traps and exceptions.

- HWACCM (Hardware Acceleration Manager), provides support for VT-x and AMD-V.

- PDM (Pluggable Device Manager), an abstract interface between the VMM and emulated devices which separates device implementations from VMM internals and makes it easy to add new emulated devices. Through PDM, third-party developers can add new virtual devices to VirtualBox without having to change VirtualBox itself.

- PGM (Page Manager), a component controlling guest paging.

- PATM (Patch Manager), patches guest code to improve and speed up software virtualization.
- TM (Time Manager), handles timers and all aspects of time inside guests.
- CFGM (Configuration Manager), provides a tree structure which holds configuration settings for the VM and all emulated devices.
- SSM (Saved State Manager), saves and loads VM state.
- VUSB (Virtual USB), a USB layer which separates emulated USB controllers from the controllers on the host and from USB devices; this also enables remote USB.
- DBGF (Debug Facility), a built-in VM debuger.
- VirtualBox emulates a number of devices to provide the hardware environment that various guests need. Most of these are standard devices found in many PC compatible machines and widely supported by guest operating systems. For network and storage devices in particular, there are several options for the emulated devices to access the underlying hardware. These devices are managed by PDM.
- Guest Additions for various guest operating systems. This is code that is installed from within a virtual machine; see chapitre 4, *Additions invité*, page 58.
- The "Main" component is special: it ties all the above bits together and is the only public API that VirtualBox provides. All the client processes listed above use only this API and never access the hypervisor components directly. As a result, third-party applications that use the VirtualBox Main API can rely on the fact that it is always well-tested and that all capabilities of VirtualBox are fully exposed. It is this API that is described in the VirtualBox SDK mentioned above (again, see chapitre 11, *VirtualBox programming interfaces*, page 181).

10.3 Hardware vs. software virtualization

VirtualBox allows software in the virtual machine to run directly on the processor of the host, but an array of complex techniques is employed to intercept operations that would interfere with your host. Whenever the guest attempts to do something that could be harmful to your computer and its data, VirtualBox steps in and takes action. In particular, for lots of hardware that the guest believes to be accessing, VirtualBox simulates a certain "virtual" environment according to how you have configured a virtual machine. For example, when the guest attempts to access a hard disk, VirtualBox redirects these requests to whatever you have configured to be the virtual machine's virtual hard disk – normally, an image file on your host.

Unfortunately, the x86 platform was never designed to be virtualized. Detecting situations in which VirtualBox needs to take control over the guest code that is executing, as described above, is difficult. There are two ways in which to achive this:

- Since 2006, Intel and AMD processors have had support for so-called **"hardware virtualization"**. This means that these processors can help VirtualBox to intercept potentially dangerous operations that a guest operating system may be attempting and also makes it easier to present virtual hardware to a virtual machine.

  These hardware features differ between Intel and AMD processors. Intel named its technology **VT-x**; AMD calls theirs **AMD-V**. The Intel and AMD support for virtualization is very different in detail, but not very different in principle.

> **Note:** On many systems, the hardware virtualization features first need to be enabled in the BIOS before VirtualBox can use them.

- As opposed to other virtualization software, for many usage scenarios, VirtualBox does not *require* hardware virtualization features to be present. Through sophisticated techniques, VirtualBox virtualizes many guest operating systems entirely in **software**. This means that you can run virtual machines even on older processors which do not support hardware virtualization.

Even though VirtualBox does not always require hardware virtualization, enabling it is *required* in the following scenarios:

- Certain rare guest operating systems like OS/2 make use of very esoteric processor instructions that are not supported with our software virtualization. For virtual machines that are configured to contain such an operating system, hardware virtualization is enabled automatically.
- VirtualBox's 64-bit guest support (added with version 2.0) and multiprocessing (SMP, added with version 3.0) both require hardware virtualization to be enabled. (This is not much of a limitation since the vast majority of today's 64-bit and multicore CPUs ship with hardware virtualization anyway; the exceptions to this rule are e.g. older Intel Celeron and AMD Opteron CPUs.)

Avertissement: Do not run other hypervisors (open-source or commercial virtualization products) together with VirtualBox! While several hypervisors can normally be *installed* in parallel, do not attempt to *run* several virtual machines from competing hypervisors at the same time. VirtualBox cannot track what another hypervisor is currently attempting to do on the same host, and especially if several products attempt to use hardware virtualization features such as VT-x, this can crash the entire host. Also, within VirtualBox, you can mix software and hardware virtualization when running multiple VMs. In certain cases a small performance penalty will be unavoidable when mixing VT-x and software virtualization VMs. We recommend not mixing virtualization modes if maximum performance and low overhead are essential. This does *not* apply to AMD-V.

10.4 Details about software virtualization

Implementing virtualization on x86 CPUs with no hardware virtualization support is an extraordinarily complex task because the CPU architecture was not designed to be virtualized. The problems can usually be solved, but at the cost of reduced performance. Thus, there is a constant clash between virtualization performance and accuracy.

The x86 instruction set was originally designed in the 1970s and underwent significant changes with the addition of protected mode in the 1980s with the 286 CPU architecture and then again with the Intel 386 and its 32-bit architecture. Whereas the 386 did have limited virtualization support for real mode operation (V86 mode, as used by the "DOS Box" of Windows 3.x and OS/2 2.x), no support was provided for virtualizing the entire architecture.

In theory, software virtualization is not overly complex. In addition to the four privilege levels ("rings") provided by the hardware (of which typically only two are used: ring 0 for kernel mode and ring 3 for user mode), one needs to differentiate between "host context" and "guest context".

In "host context", everything is as if no hypervisor was active. This might be the active mode if another application on your host has been scheduled CPU time; in that case, there is a host ring 3 mode and a host ring 0 mode. The hypervisor is not involved.

In "guest context", however, a virtual machine is active. So long as the guest code is running in ring 3, this is not much of a problem since a hypervisor can set up the page tables properly

and run that code natively on the processor. The problems mostly lie in how to intercept what the guest's kernel does.

There are several possible solutions to these problems. One approach is full software emulation, usually involving recompilation. That is, all code to be run by the guest is analyzed, transformed into a form which will not allow the guest to either modify or see the true state of the CPU, and only then executed. This process is obviously highly complex and costly in terms of performance. (VirtualBox contains a recompiler based on QEMU which can be used for pure software emulation, but the recompiler is only activated in special situations, described below.)

Another possible solution is paravirtualization, in which only specially modified guest OSes are allowed to run. This way, most of the hardware access is abstracted and any functions which would normally access the hardware or privileged CPU state are passed on to the hypervisor instead. Paravirtualization can achieve good functionality and performance on standard x86 CPUs, but it can only work if the guest OS can actually be modified, which is obviously not always the case.

VirtualBox chooses a different approach. When starting a virtual machine, through its ring-0 support kernel driver, VirtualBox has set up the host system so that it can run most of the guest code natively, but it has inserted itself at the "bottom" of the picture. It can then assume control when needed – if a privileged instruction is executed, the guest traps (in particular because an I/O register was accessed and a device needs to be virtualized) or external interrupts occur. VirtualBox may then handle this and either route a request to a virtual device or possibly delegate handling such things to the guest or host OS. In guest context, VirtualBox can therefore be in one of three states:

- Guest ring 3 code is run unmodified, at full speed, as much as possible. The number of faults will generally be low (unless the guest allows port I/O from ring 3, something we cannot do as we don't want the guest to be able to access real ports). This is also referred to as "raw mode", as the guest ring-3 code runs unmodified.

- For guest code in ring 0, VirtualBox employs a nasty trick: it actually reconfigures the guest so that its ring-0 code is run in ring 1 instead (which is normally not used in x86 operating systems). As a result, when guest ring-0 code (actually running in ring 1) such as a guest device driver attempts to write to an I/O register or execute a privileged instruction, the VirtualBox hypervisor in "real" ring 0 can take over.

- The hypervisor (VMM) can be active. Every time a fault occurs, VirtualBox looks at the offending instruction and can relegate it to a virtual device or the host OS or the guest OS or run it in the recompiler.

  In particular, the recompiler is used when guest code disables interrupts and VirtualBox cannot figure out when they will be switched back on (in these situations, VirtualBox actually analyzes the guest code using its own disassembler). Also, certain privileged instructions such as LIDT need to be handled specially. Finally, any real-mode or protected-mode code (e.g. BIOS code, a DOS guest, or any operating system startup) is run in the recompiler entirely.

Unfortunately this only works to a degree. Among others, the following situations require special handling:

1. Running ring 0 code in ring 1 causes a lot of additional instruction faults, as ring 1 is not allowed to execute any privileged instructions (of which guest's ring-0 contains plenty). With each of these faults, the VMM must step in and emulate the code to achieve the desired behavior. While this works, emulating thousands of these faults is very expensive and severely hurts the performance of the virtualized guest.

2. There are certain flaws in the implementation of ring 1 in the x86 architecture that were never fixed. Certain instructions that *should* trap in ring 1 don't. This affect for example the

LGDT/SGDT, LIDT/SIDT, or POPF/PUSHF instruction pairs. Whereas the "load" operation is privileged and can therefore be trapped, the "store" instruction always succeed. If the guest is allowed to execute these, it will see the true state of the CPU, not the virtualized state. The CPUID instruction also has the same problem.

3. A hypervisor typically needs to reserve some portion of the guest's address space (both linear address space and selectors) for its own use. This is not entirely transparent to the guest OS and may cause clashes.

4. The SYSENTER instruction (used for system calls) executed by an application running in a guest OS always transitions to ring 0. But that is where the hypervisor runs, not the guest OS. In this case, the hypervisor must trap and emulate the instruction even when it is not desirable.

5. The CPU segment registers contain a "hidden" descriptor cache which is not software-accessible. The hypervisor cannot read, save, or restore this state, but the guest OS may use it.

6. Some resources must (and can) be trapped by the hypervisor, but the access is so frequent that this creates a significant performance overhead. An example is the TPR (Task Priority) register in 32-bit mode. Accesses to this register must be trapped by the hypervisor, but certain guest operating systems (notably Windows and Solaris) write this register very often, which adversely affects virtualization performance.

To fix these performance and security issues, VirtualBox contains a Code Scanning and Analysis Manager (CSAM), which disassembles guest code, and the Patch Manager (PATM), which can replace it at runtime.

Before executing ring 0 code, CSAM scans it recursively to discover problematic instructions. PATM then performs *in-situ* patching, i.e. it replaces the instruction with a jump to hypervisor memory where an integrated code generator has placed a more suitable implementation. In reality, this is a very complex task as there are lots of odd situations to be discovered and handled correctly. So, with its current complexity, one could argue that PATM is an advanced *in-situ* recompiler.

In addition, every time a fault occurs, VirtualBox analyzes the offending code to determine if it is possible to patch it in order to prevent it from causing more faults in the future. This approach works well in practice and dramatically improves software virtualization performance.

10.5 Details about hardware virtualization

With Intel VT-x, there are two distinct modes of CPU operation: VMX root mode and non-root mode.

- In root mode, the CPU operates much like older generations of processors without VT-x support. There are four privilege levels ("rings"), and the same instruction set is supported, with the addition of several virtualization specific instruction. Root mode is what a host operating system without virtualization uses, and it is also used by a hypervisor when virtualization is active.

- In non-root mode, CPU operation is significantly different. There are still four privilege rings and the same instruction set, but a new structure called VMCS (Virtual Machine Control Structure) now controls the CPU operation and determines how certain instructions behave. Non-root mode is where guest systems run.

Switching from root mode to non-root mode is called "VM entry", the switch back is "VM exit". The VMCS includes a guest and host state area which is saved/restored at VM entry and exit. Most importantly, the VMCS controls which guest operations will cause VM exits.

The VMCS provides fairly fine-grained control over what the guests can and can't do. For example, a hypervisor can allow a guest to write certain bits in shadowed control registers, but not others. This enables efficient virtualization in cases where guests can be allowed to write control bits without disrupting the hypervisor, while preventing them from altering control bits over which the hypervisor needs to retain full control. The VMCS also provides control over interrupt delivery and exceptions.

Whenever an instruction or event causes a VM exit, the VMCS contains information about the exit reason, often with accompanying detail. For example, if a write to the CR0 register causes an exit, the offending instruction is recorded, along with the fact that a write access to a control register caused the exit, and information about source and destination register. Thus the hypervisor can efficiently handle the condition without needing advanced techniques such as CSAM and PATM described above.

VT-x inherently avoids several of the problems which software virtualization faces. The guest has its own completely separate address space not shared with the hypervisor, which eliminates potential clashes. Additionally, guest OS kernel code runs at privilege ring 0 in VMX non-root mode, obviating the problems by running ring 0 code at less privileged levels. For example the SYSENTER instruction can transition to ring 0 without causing problems. Naturally, even at ring 0 in VMX non-root mode, any I/O access by guest code still causes a VM exit, allowing for device emulation.

The biggest difference between VT-x and AMD-V is that AMD-V provides a more complete virtualization environment. VT-x requires the VMX non-root code to run with paging enabled, which precludes hardware virtualization of real-mode code and non-paged protected-mode software. This typically only includes firmware and OS loaders, but nevertheless complicates VT-x hypervisor implementation. AMD-V does not have this restriction.

Of course hardware virtualization is not perfect. Compared to software virtualization, the overhead of VM exits is relatively high. This causes problems for devices whose emulation requires high number of traps. One example is the VGA device in 16-color modes, where not only every I/O port access but also every access to the framebuffer memory must be trapped.

10.6 Nested paging and VPIDs

In addition to "plain" hardware virtualization, your processor may also support additional sophisticated techniques:[2]

- A newer feature called **"nested paging"** implements some memory management in hardware, which can greatly accelerate hardware virtualization since these tasks no longer need to be performed by the virtualization software.

  With nested paging, the hardware provides another level of indirection when translating linear to physical addresses. Page tables function as before, but linear addresses are now translated to "guest physical" addresses first and not physical addresses directly. A new set of paging registers now exists under the traditional paging mechanism and translates from guest physical addresses to host physical addresses, which are used to access memory.

  Nested paging eliminates the overhead caused by VM exits and page table accesses. In essence, with nested page tables the guest can handle paging without intervention from the hypervisor. Nested paging thus significantly improves virtualization performance.

  On AMD processors, nested paging has been available starting with the Barcelona (K10) architecture – they call it now "rapid virtualization indexing" (RVI). Intel added support for nested paging, which they call "extended page tables" (EPT), with their Core i7 (Nehalem) processors.

[2] VirtualBox 2.0 added support for AMD's nested paging; support for Intel's EPT and VPIDs was added with version 2.1.

If nested paging is enabled, the VirtualBox hypervisor can also use **large pages** to reduce TLB usage and overhead. This can yield a performance improvement of up to 5%. To enable this feature for a VM, you need to use the `VBoxManage modifyvm --largepages` command; see chapitre 8.7, *VBoxManage modifyvm*, page 120.

- On Intel CPUs, another hardware feature called **"Virtual Processor Identifiers" (VPIDs)** can greatly accelerate context switching by reducing the need for expensive flushing of the processor's Translation Lookaside Buffers (TLBs).

  To enable these features for a VM, you need to use the `VBoxManage modifyvm --vtxvpid` and `--largepages` commands; see chapitre 8.7, *VBoxManage modifyvm*, page 120.

11 VirtualBox programming interfaces

VirtualBox comes with comprehensive support for third-party developers. The so-called "Main API" of VirtualBox exposes the entire feature set of the virtualization engine. It is completely documented and available to anyone who wishes to control VirtualBox programmatically.

The Main API is made available to C++ clients through COM (on Windows hosts) or XPCOM (on other hosts). Bridges also exist for SOAP, Java and Python.

All programming information (documentation, reference information, header and other interface files as well as samples) have been split out to a separate **Software Development Kit (SDK),** which is available for download from http://www.virtualbox.org. In particular, the SDK comes with a "Programming Guide and Reference" in PDF format, which contains, among other things, the information that was previously in this chapter of the User Manual.

12 Troubleshooting

This chapter provides answers to commonly asked questions. In order to improve your user experience with VirtualBox, it is recommended to read this section to learn more about common pitfalls and get recommendations on how to use the product.

12.1 Procedures and tools

12.1.1 Categorizing and isolating problems

More often than not, a virtualized guest behaves like a physical system. Any problems that a physical machine would encounter, a virtual machine will encounter as well. If, for example, Internet connectivity is lost due to external issues, virtual machines will be affected just as much as physical ones.

If a true VirtualBox problem is encountered, it helps to categorize and isolate the problem first. Here are some of the questions that should be answered before reporting a problem:

1. Is the problem specific to a certain guest OS? Specific release of a guest OS? Especially with Linux guest related problems, the issue may be specific to a certain distribution and version of Linux.

2. Is the problem specific to a certain host OS? Problems are usually not host OS specific (because most of the VirtualBox code base is shared across all supported platforms), but especially in the areas of networking and USB support, there are significant differences between host platforms. Some GUI related issues are also host specific.

3. Is the problem specific to certain host hardware? This category of issues is typically related to the host CPU. Because of significant differences between VT-x and AMD-V, problems may be specific to one or the other technology. The exact CPU model may also make a difference (even for software virtualization) because different CPUs support different features, which may affect certain aspects of guest CPU operation.

4. Is the problem specific to a certain virtualization mode? Some problems may only occur in software virtualization mode, others may be specific to hardware virtualization.

5. Is the problem specific to guest SMP? That is, is it related to the number of virtual CPUs (VCPUs) in the guest? Using more than one CPU usually significantly affects the internal operation of a guest OS.

6. Is the problem specific to the Guest Additions? In some cases, this is a given (e.g., a shared folders problem), in other cases it may be less obvious (for example, display problems). And if the problem is Guest Additions specific, is it also specific to a certain version of the Additions?

7. Is the problem specific to a certain environment? Some problems are related to a particular environment external to the VM; this usually involves network setup. Certain configurations of external servers such as DHCP or PXE may expose problems which do not occur with other, similar servers.

8. Is the problem a regression? Knowing that an issue is a regression usually makes it significantly easier to find the solution. In this case, it is crucial to know which version is affected and which is not.

12.1.2 Collecting debugging information

For problem determination, it is often important to collect debugging information which can be analyzed by VirtualBox support. This section contains information about what kind of information can be obtained.

Every time VirtualBox starts up a VM, a so-called **"release log file"** is created containing lots of information about the VM configuration and runtime events. The log file is called `VBox.log` and resides in the VM log file folder. Typically this will be a directory like this:

```
$HOME/VirtualBox VMs/{machinename}/Logs
```

When starting a VM, the configuration file of the last run will be renamed to `.1`, up to `.3`. Sometimes when there is a problem, it is useful to have a look at the logs. Also when requesting support for VirtualBox, supplying the corresponding log file is mandatory.

For convenience, for each virtual machine, the VirtualBox main window can show these logs in a window. To access it, select a virtual machine from the list on the left and select "Show logs..." from the "Machine" window.

The release log file (VBox.log) contains a wealth of diagnostic information, such as Host OS type and version, VirtualBox version and build (32-bit or 64-bit), a complete dump of the guest's configuration (CFGM), detailed information about the host CPU type and supported features, whether hardware virtualization is enabled, information about VT-x/AMD-V setup, state transitions (creating, running, paused, stopping, etc.), guest BIOS messages, Guest Additions messages, device-specific log entries and, at the end of execution, final guest state and condensed statistics.

In case of crashes, it is very important to collect **crash dumps**. This is true for both host and guest crashes. For information about enabling core dumps on Linux, Solaris, and OS X systems, refer to the core dump article on the VirtualBox website.[1]

You can also use `VBoxManage debugvm` to create a dump of a complete virtual machine; see chapitre 8.31, *VBoxManage debugvm*, page 143.

For network related problems, it is often helpful to capture a trace of network traffic. If the traffic is routed through an adapter on the host, it is possible to use Wireshark or a similar tool to capture the traffic there. However, this often also includes a lot of traffic unrelated to the VM.

VirtualBox provides an ability to capture network traffic only on a specific VM's network adapter. Refer to the network tracing article on the VirtualBox website[2] for information on enabling this capture. The trace files created by VirtualBox are in `.pcap` format and can be easily analyzed with Wireshark.

12.1.3 The built-in VM debugger

VirtualBox includes a built-in VM debugger, which advanced users may find useful. This debugger allows for examining and, to some extent, controlling the VM state.

> **Avertissement:** Use the VM debugger at your own risk. There is no support for it, and the following documentation is only made available for advanced users with a very high level of familiarity with the x86/AMD64 machine instruction set, as well as detailed knowledge of the PC architecture. A degree of familiarity with the internals of the guest OS in question may also be very helpful.

The VM debugger is available in all regular production versions of VirtualBox, but it is disabled by default because the average user will have little use for it. There are two ways to access the debugger:

[1] http://www.virtualbox.org/wiki/Core_dump.
[2] http://www.virtualbox.org/wiki/Network_tips.

- A debugger console window displayed alongside the VM
- Via the `telnet` protocol at port 5000

The debugger can be enabled in three ways:

- Start the VM directly using `VirtualBox --startvm`, with an additional `--dbg`, `--debug`, or `--debug-command-line` argument. See the VirtualBox usage help for details.
- Set the `VBOX_GUI_DBG_ENABLED` or `VBOX_GUI_DBG_AUTO_SHOW` environment variable to `true` before launching the VirtualBox process. Setting these variables (only their presence is checked) is effective even when the first VirtualBox process is the VM selector window. VMs subsequently launched from the selector will have the debugger enabled.
- Set the `GUI/Dbg/Enabled` extra data item to `true` before launching the VM. This can be set globally or on a per VM basis.

A new 'Debug' menu entry will be added to the VirtualBox application. This menu allows the user to open the debugger console.

The VM debugger command syntax is loosely modeled on Microsoft and IBM debuggers used on DOS, OS/2 and Windows. Users familiar with symdeb, CodeView, or the OS/2 kernel debugger will find the VirtualBox VM debugger familiar.

The most important command is `help`. This will print brief usage help for all debugger commands. The set of commands supported by the VM debugger changes frequently and the `help` command is always up-to-date.

A brief summary of frequently used commands follows:

- `stop` – stops the VM execution and enables single stepping
- `g` – continue VM execution
- `t` – single step an instruction
- `rg/rh/r` – print the guest/hypervisor/current registers
- `kg/kh/k` – print the guest/hypervisor/current call stack
- `da/db/dw/dd/dq` – print memory contents as ASCII/bytes/words/dwords/qwords
- `u` – unassemble memory
- `dg` – print the guest's GDT
- `di` – print the guest's IDT
- `dl` – print the guest's LDT
- `dt` – print the guest's TSS
- `dp*` – print the guest's page table structures
- `bp/br` – set a normal/recompiler breakpoint
- `bl` – list breakpoints
- `bc` – clear a breakpoint
- `writecore` – writes a VM core file to disk, refer chapitre 12.1.4, *VM core format*, page 185

See the built-in `help` for other available commands.

The VM debugger supports symbolic debugging, although symbols for guest code are often not available. For Solaris guests, the `detect` command automatically determines the guest OS version and locates kernel symbols in guest's memory. Symbolic debugging is then available. For Linux guests, the `detect` commands also determines the guest OS version, but there are no symbols in the guest's memory. Kernel symbols are available in the file `/proc/kallsyms` on Linux guests. This file must be copied to the host, for example using `scp`. The `loadmap` debugger command can be used to make the symbol information available to the VM debugger. Note that the `kallsyms` file contains the symbols for the currently loaded modules; if the guest's configuration changes, the symbols will change as well and must be updated.

For all guests, a simple way to verify that the correct symbols are loaded is the `k` command. The guest is normally idling and it should be clear from the symbolic information that the guest operating system's idle loop is being executed.

Another group of debugger commands is the set of `info` commands. Running `info help` provides complete usage information. The information commands provide ad-hoc data pertinent to various emulated devices and aspects of the VMM. There is no general guideline for using the `info` commands, the right command to use depends entirely on the problem being investigated. Some of the info commands are:

- `cfgm` – print a branch of the configuration tree
- `cpuid` – display the guest CPUID leaves
- `ioport` – print registered I/O port ranges
- `mmio` – print registered MMIO ranges
- `mode` – print the current paging mode
- `pit` – print the i8254 PIT state
- `pic` – print the i8259A PIC state
- `ohci/ehci` – print a subset of the OHCI/EHCI USB controller state
- `pcnet0` – print the PCnet state
- `vgatext` – print the contents of the VGA framebuffer formatted as standard text mode
- `timers` – print all VM timers

The output of the `info` commands generally requires in-depth knowledge of the emulated device and/or VirtualBox VMM internals. However, when used properly, the information provided can be invaluable.

12.1.4 VM core format

VirtualBox uses the 64-bit ELF format for its VM core files created by `VBoxManage debugvm`; see chapitre 8.31, *VBoxManage debugvm*, page 143. The VM core file contain the memory and CPU dumps of the VM and can be useful for debugging your guest OS. The 64-bit ELF object format specficiation can be obtained here: `http://downloads.openwatcom.org/ftp/devel/docs/elf-64-gen.pdf`.

The overall layout of the VM core format is as follows:

```
[ ELF 64 Header]
[ Program Header, type PT_NOTE ]
  -> offset to COREDESCRIPTOR
[ Program Header, type PT_LOAD ] - one for each contiguous physical memory range
  -> Memory offset of range
```

```
  -> File offset
[ Note Header, type NT_VBOXCORE ]
[ COREDESCRIPTOR ]
  -> Magic
  -> VM core file version
  -> VBox version
  -> Number of vCPUs etc.
[ Note Header, type NT_VBOXCPU ] - one for each vCPU
[ vCPU 1 Note Header ]
  [ CPUMCTX - vCPU 1 dump ]
[ Additional Notes + Data ] - currently unused
[ Memory dump ]
```

The memory descriptors contain physical addresses relative to the guest and not virtual addresses. Regions of memory such as MMIO regions are not included in the core file.

The relevant data structures and definitions can be found in the VirtualBox sources under the following header files: `include/VBox/dbgfcorefmt.h`, `include/VBox/cpumctx.h` and `src/VBox/Runtime/include/internal/ldrELFCommon.h`.

The VM core file can be inspected using `elfdump` and GNU `readelf` or other similar utilities.

12.2 General

12.2.1 Guest shows IDE/SATA errors for file-based images on slow host file system

Occasionally, some host file systems provide very poor writing performance and as a consequence cause the guest to time out IDE/SATA commands. This is normal behavior and should normally cause no real problems, as the guest should repeat commands that have timed out. However some guests (e.g. some Linux versions) have severe problems if a write to an image file takes longer than about 15 seconds. Some file systems however require more than a minute to complete a single write, if the host cache contains a large amount of data that needs to be written.

The symptom for this problem is that the guest can no longer access its files during large write or copying operations, usually leading to an immediate hang of the guest.

In order to work around this problem (the true fix is to use a faster file system that doesn't exhibit such unacceptable write performance), it is possible to flush the image file after a certain amount of data has been written. This interval is normally infinite, but can be configured individually for each disk of a VM.

For IDE disks use the following command:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/piix3ide/0/LUN#[x]/Config/FlushInterval" [b]
```

For SATA disks use the following command:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/LUN#[x]/Config/FlushInterval" [b]
```

The value [x] that selects the disk for IDE is 0 for the master device on the first channel, 1 for the slave device on the first channel, 2 for the master device on the second channel or 3 for the master device on the second channel. For SATA use values between 0 and 29. Only disks support this configuration option; it must not be set for CD/DVD drives.

The unit of the interval [b] is the number of bytes written since the last flush. The value for it must be selected so that the occasional long write delays do not occur. Since the proper flush interval depends on the performance of the host and the host filesystem, finding the optimal value that makes the problem disappear requires some experimentation. Values between 1000000 and 10000000 (1 to 10 megabytes) are a good starting point. Decreasing the interval both decreases the probability of the problem and the write performance of the guest. Setting the value unnecessarily low will cost performance without providing any benefits. An interval of 1 will cause a

flush for each write operation and should solve the problem in any case, but has a severe write performance penalty.

Providing a value of 0 for [b] is treated as an infinite flush interval, effectively disabling this workaround. Removing the extra data key by specifying no value for [b] has the same effect.

12.2.2 Responding to guest IDE/SATA flush requests

If desired, the virtual disk images can be flushed when the guest issues the IDE FLUSH CACHE command. Normally these requests are ignored for improved performance. The parameters below are only accepted for disk drives. They must not be set for DVD drives.

To enable flushing for IDE disks, issue the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/Devices/piix3ide/0/LUN#[x]/Config/IgnoreFlush" 0
```

The value [x] that selects the disk is 0 for the master device on the first channel, 1 for the slave device on the first channel, 2 for the master device on the second channel or 3 for the master device on the second channel.

To enable flushing for SATA disks, issue the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/Devices/ahci/0/LUN#[x]/Config/IgnoreFlush" 0
```

The value [x] that selects the disk can be a value between 0 and 29.

Note that this doesn't affect the flushes performed according to the configuration described in chapitre 12.2.1, *Guest shows IDE/SATA errors for file-based images on slow host file system*, page 186. Restoring the default of ignoring flush commands is possible by setting the value to 1 or by removing the key.

12.2.3 Poor performance caused by host power management

On some hardware platforms and operating systems, virtualization performance is negatively affected by host CPU power management. The symptoms may be choppy audio in the guest or erratic guest clock behavior.

Some of the problems may be caused by firmware and/or host operating system bugs. Therefore, updating the firmware and applying operating systems fixes is recommended.

For optimal virtualization performance, the C1E power state support in the system's BIOS should be disabled, if such a setting is available (not all systems support the C1E power state). Disabling other power management settings may also improve performance. However, a balance between performance and power consumption must always be considered.

12.2.4 GUI: 2D Video Acceleration option is grayed out

To use 2D Video Acceleration within VirtualBox, your host's video card should support certain OpenGL extensions. On startup, VirtualBox checks for those extensions, and, if the test fails, this option is silently grayed out.

To find out why it has failed, you can manually execute the following command:

```
VBoxTestOGL --log "log_file_name" --test 2D
```

It will list the required OpenGL extensions one by one and will show you which one failed the test. This usually means that you are running an outdated or misconfigured OpenGL driver on your host. It can also mean that your video chip is lacking required functionality.

12.3 Windows guests

12.3.1 Windows bluescreens after changing VM configuration

Changing certain virtual machine settings can cause Windows guests to fail during start up with a bluescreen. This may happen if you change VM settings after installing Windows, or if you copy a disk image with an already installed Windows to a newly created VM which has settings that differ from the original machine.

This applies in particular to the following settings:

- The ACPI and I/O APIC settings should never be changed after installing Windows. Depending on the presence of these hardware features, the Windows installation program chooses special kernel and device driver versions and will fail to startup should these hardware features be removed. (Enabling them for a Windows VM which was installed without them does not cause any harm. However, Windows will not use these features in this case.)
- Changing the storage controller hardware will cause bootup failures as well. This might also apply to you if you copy a disk image from an older version of VirtualBox to a virtual machine created with a newer VirtualBox version; the default subtype of IDE controller hardware was changed from PIIX3 to PIIX4 with VirtualBox 2.2. Make sure these settings are identical.

12.3.2 Windows 0x101 bluescreens with SMP enabled (IPI timeout)

If a VM is configured to have more than one processor (symmetrical multiprocessing, SMP), some configurations of Windows guests crash with an 0x101 error message, indicating a timeout for inter-processor interrupts (IPIs). These interrupts synchronize memory management between processors.

According to Microsoft, this is due to a race condition in Windows. A hotfix is available.[3] If this does not help, please reduce the number of virtual processors to 1.

12.3.3 Windows 2000 installation failures

When installing Windows 2000 guests, you might run into one of the following issues:

- Installation reboots, usually during component registration.
- Installation fills the whole hard disk with empty log files.
- Installation complains about a failure installing msgina.dll.

These problems are all caused by a bug in the hard disk driver of Windows 2000. After issuing a hard disk request, there is a race condition in the Windows driver code which leads to corruption if the operation completes too fast, i.e. the hardware interrupt from the IDE controller arrives too soon. With physical hardware, there is a guaranteed delay in most systems so the problem is usually hidden there (however it should be possible to reproduce it on physical hardware as well). In a virtual environment, it is possible for the operation to be done immediately (especially on very fast systems with multiple CPUs) and the interrupt is signaled sooner than on a physical system. The solution is to introduce an artificial delay before delivering such interrupts. This delay can be configured for a VM using the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/Devices/piix3ide/0/Config/IRQDelay" 1
```

This sets the delay to one millisecond. In case this doesn't help, increase it to a value between 1 and 5 milliseconds. Please note that this slows down disk performance. After installation, you should be able to remove the key (or set it to 0).

[3] See http://support.microsoft.com/kb/955076.

12.3.4 How to record bluescreen information from Windows guests

When Windows guests run into a kernel crash, they display the infamous bluescreen. Depending on how Windows is configured, the information will remain on the screen until the machine is restarted or it will reboot automatically. During installation, Windows is usually configured to reboot automatically. With automatic reboots, there is no chance to record the bluescreen information which might be important for problem determination.

VirtualBox provides a method of halting a guest when it wants to perform a reset. In order to enable this feature, issue the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/PDM/HaltOnReset" 1
```

12.3.5 No networking in Windows Vista guests

With Windows Vista, Microsoft dropped support for the AMD PCNet card that VirtualBox used to provide as the default virtual network card before version 1.6.0. For Windows Vista guests, VirtualBox now uses an Intel E1000 card by default.

If, for some reason, you still want to use the AMD card, you need to download the PCNet driver from the AMD website (available for 32-bit Windows only). You can transfer it into the virtual machine using a shared folder, see (see chapitre 4.3, *Dossiers partagés*, page 68).

12.3.6 Windows guests may cause a high CPU load

Several background applications of Windows guests, especially virus scanners, are known to increases the CPU load notably even if the guest appears to be idle. We recommend to deactivate virus scanners within virtualized guests if possible.

12.3.7 Long delays when accessing shared folders

The performance for accesses to shared folders from a Windows guest might be decreased due to delays during the resolution of the VirtualBox shared folders name service. To fix these delays, add the following entries to the file `\windows\system32\drivers\etc\lmhosts` of the Windows guest:

```
255.255.255.255        VBOXSVR #PRE
255.255.255.255        VBOXSRV #PRE
```

After doing this change, a reboot of the guest is required.

12.3.8 USB tablet coordinates wrong in Windows 98 guests

If a Windows 98 VM is configured to use the emulated USB tablet (absolute pointing device), the coordinate translation may be incorrect and the pointer is restricted to the upper left quarter of the guest's screen.

The USB HID (Human Interface Device) drivers in Windows 98 are very old and do not handle tablets the same way all more recent operating systems do (Windows 2000 and later, Mac OS X, Solaris). To work around the problem, issue the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/USB/HidMouse/0/Config/CoordShift" 0
```

To restore the default behavior, remove the key or set its value to 1.

12.3.9 Windows guests are removed from an Active Directory domain after restoring a snapshot

If a Windows guest is a member of an Active Directory domain and the snapshot feature of VirtualBox is used, it could happen it loses this status after you restore an older snapshot.

The reason is the automatic machine password changing performed by Windows in regular intervals for security purposes. You can disable this feature by following the instruction of this http://support.microsoft.com/kb/154501 article from Microsoft.

12.4 Linux and X11 guests

12.4.1 Linux guests may cause a high CPU load

Some Linux guests may cause a high CPU load even if the guest system appears to be idle. This can be caused by a high timer frequency of the guest kernel. Some Linux distributions, for example Fedora, ship a Linux kernel configured for a timer frequency of **1000Hz**. We recommend to recompile the guest kernel and to select a timer frequency of 100Hz.

Linux kernels shipped with Red Hat Enterprise Linux (RHEL) as of release 4.7 and 5.1 as well as kernels of related Linux distributions (for instance CentOS and Oracle Enterprise Linux) support a kernel parameter *divider=N*. Hence, such kernels support a lower timer frequency without recompilation. We suggest to add the kernel parameter *divider=10* to select a guest kernel timer frequency of 100Hz.

12.4.2 AMD Barcelona CPUs

Most Linux-based guests will fail with AMD Phenoms or Barcelona-level Opterons due to a bug in the Linux kernel. Enable the I/O-APIC to work around the problem (see chapitre 3.4, *Paramètres système*, page 47).

12.4.3 Buggy Linux 2.6 kernel versions

The following bugs in Linux kernels prevent them from executing correctly in VirtualBox, causing VM boot crashes:

- The Linux kernel version 2.6.18 (and some 2.6.17 versions) introduced a race condition that can cause boot crashes in VirtualBox. Please use a kernel version 2.6.19 or later.
- With hardware virtualization and the I/O APIC enabled, kernels before 2.6.24-rc6 may panic on boot with the following message:

  ```
  Kernel panic - not syncing: IO-APIC + timer doesn't work!  Boot with
  apic=debug and send a report.  Then try booting with the 'noapic' option
  ```

  If you see this message, either disable hardware virtualization or the I/O APIC (see chapitre 3.4, *Paramètres système*, page 47), or upgrade the guest to a newer kernel.[4]

12.4.4 Shared clipboard, auto-resizing and seamless desktop in X11 guests

Guest desktop services in guests running the X11 window system (Solaris, Linux and others) are provided by a guest service called `VBoxClient`, which runs under the ID of the user who started the desktop session and is automatically started using the following command lines

[4] See http://www.mail-archive.com/git-commits-head@vger.kernel.org/msg30813.html for details about the kernel fix.

```
VBoxClient --clipboard
VBoxClient --display
VBoxClient --seamless
```

when your X11 user session is started if you are using a common desktop environment (Gnome, KDE and others). If a particular desktop service is not working correctly, it is worth checking whether the process which should provide it is running.

The `VBoxClient` processes create files in the user's home directory with names of the form `.vboxclient-*.pid` when they are running in order to prevent a given service from being started twice. It can happen due to misconfiguration that these files are created owned by root and not deleted when the services are stopped, which will prevent them from being started in future sessions. If the services cannot be started, you may wish to check whether these files still exist.

12.5 Windows hosts

12.5.1 VBoxSVC out-of-process COM server issues

VirtualBox makes use of the Microsoft Component Object Model (COM) for inter- and intra-process communication. This allows VirtualBox to share a common configuration among different virtual machine processes and provide several user interface options based on a common architecture. All global status information and configuration is maintained by the process `VBoxSVC.exe`, which is an out-of-process COM server. Whenever a VirtualBox process is started, it requests access to the COM server and Windows automatically starts the process. Note that it should never be started by the end user.

When the last process disconnects from the COM server, it will terminate itself after some seconds. The VirtualBox configuration (XML files) is maintained and owned by the COM server and the files are locked whenever the server runs.

In some cases - such as when a virtual machine is terminated unexpectedly - the COM server will not notice that the client is disconnected and stay active for a longer period (10 minutes or so) keeping the configuration files locked. In other rare cases the COM server might experience an internal error and subsequently other processes fail to initialize it. In these situations, it is recommended to use the Windows task manager to kill the process `VBoxSVC.exe`.

12.5.2 CD/DVD changes not recognized

In case you have assigned a physical CD/DVD drive to a guest and the guest does not notice when the medium changes, make sure that the Windows media change notification (MCN) feature is not turned off. This is represented by the following key in the Windows registry:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Cdrom\Autorun
```

Certain applications may disable this key against Microsoft's advice. If it is set to 0, change it to 1 and reboot your system. VirtualBox relies on Windows notifying it of media changes.

12.5.3 Sluggish response when using Microsoft RDP client

If connecting to a Virtual Machine via the Microsoft RDP client (called Remote Desktop Connection), there can be large delays between input (moving the mouse over a menu is the most obvious situation) and output. This is because this RDP client collects input for a certain time before sending it to the RDP server.

The interval can be decreased by setting a Windows registry key to smaller values than the default of 100. The key does not exist initially and must be of type DWORD. The unit for its values is milliseconds. Values around 20 are suitable for low-bandwidth connections between the

RDP client and server. Values around 4 can be used for a gigabit Ethernet connection. Generally values below 10 achieve a performance that is very close to that of the local input devices and screen of the host on which the Virtual Machine is running.

Depending whether the setting should be changed for an individual user or for the system, either

```
HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Min Send Interval
```

or

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Terminal Server Client\Min Send Interval
```

can be set appropriately.

12.5.4 Running an iSCSI initiator and target on a single system

Deadlocks can occur on a Windows host when attempting to access an iSCSI target running in a guest virtual machine with an iSCSI initiator (e.g. Microsoft iSCSI Initiator) that is running on the host. This is caused by a flaw in the Windows cache manager component, and causes sluggish host system response for several minutes, followed by a "Delayed Write Failed" error message in the system tray or in a separate message window. The guest is blocked during that period and may show error messages or become unstable.

Setting the environment variable `VBOX_DISABLE_HOST_DISK_CACHE` to 1 will enable a workaround for this problem until Microsoft addresses the issue. For example, open a command prompt window and start VirtualBox like this:

```
set VBOX_DISABLE_HOST_DISK_CACHE=1
VirtualBox
```

While this will decrease guest disk performance (especially writes), it does not affect the performance of other applications running on the host.

12.5.5 Bridged networking adapters missing

If no bridged adapters show up in the "Networking" section of the VM settings, this typically means that the bridged networking driver was not installed properly on your host. This could be due to the following reasons:

- The maximum allowed filter count was reached on the host. In this case, the MSI log would mention the `0x8004a029` error code returned on NetFlt network component install:

  ```
  VBoxNetCfgWinInstallComponent: Install failed, hr (0x8004a029)
  ```

  You can try to increase the maximum filter count in the Windows registry at the following key:

  ```
  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Network\MaxNumFilters
  ```

  The maximum number allowed is 14. After a reboot, try to re-install VirtualBox.

- The INF cache is corrupt. In this case, the install log (`%windir%\inf\setupapi.log` on XP or `%windir%\inf\setupapi.dev.log` on Vista or later) would typically mention the failure to find a suitable driver package for either the `sun_VBoxNetFlt` or `sun_VBoxNetFltmp` components. The solution then is to uninstall VirtualBox, remove the INF cache (`%windir%\inf\INFCACHE.1`), reboot and try to re-install VirtualBox

12.5.6 Host-only networking adapters cannot be created

If host-only adapter cannot be created (either via the Manager or VBoxManage), then the INF cache is probably corrupt. In this case, the install log (`%windir%\inf\setupapi.log` on XP or `%windir%\inf\setupapi.dev.log` on Vista or later) would typically mention the failure to find a suitable driver package for the `sun_VBoxNetAdp` component. Again, as with the bridged networking problem described above, the solution is to uninstall VirtualBox, remove the INF cache (`%windir%\inf\INFCACHE.1`), reboot and try to re-install VirtualBox.

12.6 Linux hosts

12.6.1 Linux kernel module refuses to load

If the VirtualBox kernel module (`vboxdrv`) refuses to load, i.e. you get an "Error inserting vboxdrv: Invalid argument", check (as root) the output of the `dmesg` command to find out why the load failed. Most probably the kernel disagrees with the version of the gcc used to compile the module. Make sure that you use the same compiler as used to build the kernel.

12.6.2 Linux host CD/DVD drive not found

If you have configured a virtual machine to use the host's CD/DVD drive, but this does not appear to work, make sure that the current user has permission to access the corresponding Linux device file (`/dev/hdc` or `/dev/scd0` or `/dev/cdrom` or similar). On most distributions, the user must be added to a corresponding group (usually called `cdrom` or `cdrw`).

12.6.3 Linux host CD/DVD drive not found (older distributions)

On older Linux distributions, if your CD/DVD device has a different name, VirtualBox may be unable to find it. On older Linux hosts, VirtualBox performs the following steps to locate your CD/DVD drives:

1. VirtualBox examines if the environment variable `VBOX_CDROM` is defined (see below). If so, VirtualBox omits all the following checks.
2. VirtualBox tests if `/dev/cdrom` works.
3. In addition, VirtualBox checks if any CD/DVD drives are currently mounted by checking `/etc/mtab`.
4. In addition, VirtualBox checks if any of the entries in `/etc/fstab` point to CD/DVD devices.

In other words, you can try to set VBOX_CDROM to contain a list of your CD/DVD devices, separated by colons, for example as follows:

```
export VBOX_CDROM='/dev/cdrom0:/dev/cdrom1'
```

On modern Linux distributions, VirtualBox uses the hardware abstraction layer (hal) to locate CD and DVD hardware.

12.6.4 Linux host floppy not found

The previous instructions (for CD and DVD drives) apply accordingly to floppy disks, except that on older distributions VirtualBox tests for `/dev/fd*` devices by default, and this can be overridden with the `VBOX_FLOPPY` environment variable.

12.6.5 Strange guest IDE error messages when writing to CD/DVD

If the experimental CD/DVD writer support is enabled with an incorrect VirtualBox, host or guest configuration, it is possible that any attempt to access the CD/DVD writer fails and simply results in guest kernel error messages (for Linux guests) or application error messages (for Windows guests). VirtualBox performs the usual consistency checks when a VM is powered up (in particular it aborts with an error message if the device for the CD/DVD writer is not writable by the user starting the VM), but it cannot detect all misconfigurations. The necessary host and guest OS configuration is not specific for VirtualBox, but a few frequent problems are listed here which occurred in connection with VirtualBox.

Special care must be taken to use the correct device. The configured host CD/DVD device file name (in most cases /dev/cdrom) must point to the device that allows writing to the CD/DVD unit. For CD/DVD writer units connected to a SCSI controller or to a IDE controller that interfaces to the Linux SCSI subsystem (common for some SATA controllers), this must refer to the SCSI device node (e.g. /dev/scd0). Even for IDE CD/DVD writer units this must refer to the appropriate SCSI CD-ROM device node (e.g. /dev/scd0) if the ide-scsi kernel module is loaded. This module is required for CD/DVD writer support with all Linux 2.4 kernels and some early 2.6 kernels. Many Linux distributions load this module whenever a CD/DVD writer is detected in the system, even if the kernel would support CD/DVD writers without the module. VirtualBox supports the use of IDE device files (e.g. /dev/hdc), provided the kernel supports this and the ide-scsi module is not loaded.

Similar rules (except that within the guest the CD/DVD writer is always an IDE device) apply to the guest configuration. Since this setup is very common, it is likely that the default configuration of the guest works as expected.

12.6.6 VBoxSVC IPC issues

On Linux, VirtualBox makes use of a custom version of Mozilla XPCOM (cross platform component object model) for inter- and intra-process communication (IPC). The process `VBoxSVC` serves as a communication hub between different VirtualBox processes and maintains the global configuration, i.e. the XML database. When starting a VirtualBox component, the processes `VBoxSVC` and `VirtualBoxXPCOMIPCD` are started automatically. They are only accessible from the user account they are running under. `VBoxSVC` owns the VirtualBox configuration database which normally resides in `~/.VirtualBox`. While it is running, the configuration files are locked. Communication between the various VirtualBox components and `VBoxSVC` is performed through a local domain socket residing in `/tmp/.vbox-<username>-ipc`. In case there are communication problems (i.e. a VirtualBox application cannot communicate with `VBoxSVC`), terminate the daemons and remove the local domain socket directory.

12.6.7 USB not working

If USB is not working on your Linux host, make sure that the current user is a member of the `vboxusers` group. On older hosts, you need to make sure that the user has permission to access the USB filesystem (`usbfs`), which VirtualBox relies on to retrieve valid information about your host's USB devices. The rest of this section only applies to those older systems.

As `usbfs` is a virtual filesystem, a `chmod` on `/proc/bus/usb` has no effect. The permissions for `usbfs` can therefore *only* be changed by editing the `/etc/fstab` file.

For example, most Linux distributions have a user group called `usb` or similar, of which the current user must be a member. To give all users of that group access to usbfs, make sure the following line is present:

```
# 85 is the USB group
none     /proc/bus/usb     usbfs      devgid=85,devmode=664    0    0
```

Replace 85 with the group ID that matches your system (search /etc/group for "usb" or similar). Alternatively, if you don't mind the security hole, give all users access to USB by changing "664" to "666".

The various distributions are very creative from which script the usbfs filesystem is mounted. Sometimes the command is hidden in unexpected places. For SuSE 10.0 the mount command is part of the udev configuration file /etc/udev/rules.d/50-udev.rules. As this distribution has no user group called usb, you may e.g. use the vboxusers group which was created by the VirtualBox installer. Since group numbers are allocated dynamically, the following example uses 85 as a placeholder. Modify the line containing (a linebreak has been inserted to improve readability)

```
DEVPATH="/module/usbcore", ACTION=="add",
    RUN+="/bin/mount -t usbfs usbfs /proc/bus/usb"
```

and add the necessary options (make sure that everything is in a single line):

```
DEVPATH="/module/usbcore", ACTION=="add",
    RUN+="/bin/mount -t usbfs usbfs /proc/bus/usb -o devgid=85,devmode=664"
```

Debian Etch has the mount command in /etc/init.d/mountkernfs.sh. Since that distribution has no group usb, it is also the easiest solution to allow all members of the group vboxusers to access the USB subsystem. Modify the line

```
domount usbfs usbdevfs /proc/bus/usb -onoexec,nosuid,nodev
```

so that it contains

```
domount usbfs usbdevfs /proc/bus/usb -onoexec,nosuid,nodev,devgid=85,devmode=664
```

As usual, replace the 85 with the actual group number which should get access to USB devices. Other distributions do similar operations in scripts stored in the /etc/init.d directory.

12.6.8 PAX/grsec kernels

Linux kernels including the grsec patch (see http://www.grsecurity.net/) and derivates have to disable PAX_MPROTECT for the VBox binaries to be able to start a VM. The reason is that VBox has to create executable code on anonymous memory.

12.6.9 Linux kernel vmalloc pool exhausted

When running a large number of VMs with a lot of RAM on a Linux system (say 20 VMs with 1GB of RAM each), additional VMs might fail to start with a kernel error saying that the vmalloc pool is exhausted and should be extended. The error message also tells you to specify vmalloc=256MB in your kernel parameter list. If adding this parameter to your GRUB or LILO configuration makes the kernel fail to boot (with a weird error message such as "failed to mount the root partition"), then you have probably run into a memory conflict of your kernel and initial RAM disk. This can be solved by adding the following parameter to your GRUB configuration:

```
uppermem 524288
```

12.7 Solaris hosts

12.7.1 Cannot start VM, not enough contiguous memory

The ZFS file system is known to use all available RAM as cache if the default system settings are not changed. This may lead to a heavy fragmentation of the host memory preventing VirtualBox VMs from being started. We recommend to limit the ZFS cache by adding a line

```
set zfs:zfs_arc_max = xxxx
```

to /etc/system where xxxx bytes is the amount of memory usable for the ZFS cache.

12.7.2 VM aborts with out of memory errors on Solaris 10 hosts

32-bit Solaris 10 hosts (bug 1225025) require swap space equal to, or greater than the host's physical memory size. For example, 8 GB physical memory would require at least 8 GB swap. This can be configured during a Solaris 10 install by choosing a 'custom install' and changing the default partitions.

Note: This restriction applies only to 32-bit Solaris hosts, 64-bit hosts are not affected!

For existing Solaris 10 installs, an additional swap image needs to be mounted and used as swap. Hence if you have 1 GB swap and 8 GB of physical memory, you require to add 7 GB more swap. This can be done as follows:

For ZFS (as root user):

```
zfs create -V 8gb /_<ZFS volume>_/swap
swap -a /dev/zvol/dsk/_<ZFS volume>_/swap
```

To mount if after reboot, add the following line to /etc/vfstab:

```
/dev/zvol/dsk/_<ZFS volume>_/swap - - swap - no -
```

Alternatively, you could grow the existing swap using:

```
zfs set volsize=8G rpool/swap
```

And reboot the system for the changes to take effect.

For UFS (as root user):

```
mkfile 7g /path/to/swapfile.img
swap -a /path/to/swapfile.img
```

To mount it after reboot, add the following line to /etc/vfstab:

```
/path/to/swap.img - - swap - no -
```

13 Security guide

13.1 Overview

13.1.1 General Security Principles

The following principles are fundamental to using any application securely.

Keep Software Up To Date One of the principles of good security practise is to keep all software versions and patches up to date. Activate the VirtualBox update notification to get notified when a new VirtualBox release is available. When updating VirtualBox, do not forget to update the Guest Additions. Keep the host operating system as well as the guest operating system up to date.

Restrict Network Access to Critical Services Use proper means, for instance a firewall, to protect your computer and your guest(s) from accesses from the outside. Choosing the proper networking mode for VMs helps to separate host networking from the guest and vice versa.

Follow the Principle of Least Privilege The principle of least privilege states that users should be given the least amount of privilege necessary to perform their jobs. Always execute VirtualBox as a regular user. We strongly discourage anyone from executing VirtualBox with system privileges.

Choose restrictive permissions when creating configuration files, for instance when creating /etc/default/virtualbox, see chapitre 2.3.4.7, *Options d'installation automatique*, page 40. Mode 0600 would be preferred.

Monitor System Activity System security builds on three pillars: good security protocols, proper system configuration and system monitoring. Auditing and reviewing audit records address the third requirement. Each component within a system has some degree of monitoring capability. Follow audit advice in this document and regularly monitor audit records.

Keep Up To Date on Latest Security Information Oracle continually improves its software and documentation. Check this note note yearly for revisions.

13.2 Secure Installation and Configuration

13.2.1 Installation Overview

The VirtualBox base package should be downloaded only from a trusted source, for instance the official website `http://www.virtualbox.org`. The integrity of the package should be verified with the provided SHA256 checksum which can be found on the official website.

General VirtualBox installation instructions for the supported hosts can be found in chapitre 2, *Détails d'installation*, page 32.

On Windows hosts, the installer allows for disabling USB support, support for bridged networking, support for host-only networking and the Python language bindings, see chapitre 2.1, *Installation sur des hôtes Windows*, page 32. All these features are enabled by default but disabling some of them could be appropriate if the corresponding functionality is not required by any virtual machine. The Python language bindings are only required if the VirtualBox API is

to be used by external Python applications. In particular USB support and support for the two networking modes require the installation of Windows kernel drivers on the host. Therefore disabling those selected features can not only be used to restrict the user to certain functionality but also to minimize the surface provided to a potential attacker.

The general case is to install the complete VirtualBox package. The installation must be done with system privileges. All VirtualBox binaries should be executed as a regular user and never as a privileged user.

The Oracle VM VirtualBox extension pack provides additional features and must be downloaded and installed separately, see chapitre 1.5, *Installer et lancer VirtualBox*, page 15. As for the base package, the SHA256 checksum of the extension pack should be verified. As the installation requires system privileges, VirtualBox will ask for the system password during the installation of the extension pack.

13.2.2 Post Installation Configuration

Normally there is no post installation configuration of VirtualBox components required. However, on Solaris and Linux hosts it is necessary to configure the proper permissions for users executing VMs and who should be able to access certain host resources. For instance, Linux users must be member of the *vboxusers* group to be able to pass USB devices to a guest. If a serial host interface should be accessed from a VM, the proper permissions must be granted to the user to be able to access that device. The same applies to other resources like raw partitions, DVD/CD drives and sound devices.

13.3 Security Features

This section outlines the specific security mechanisms offered by VirtualBox.

13.3.1 The Security Model

One property of virtual machine monitors (VMMs) like VirtualBox is to encapsulate a guest by executing it in a protected environment, a virtual machine, running as a user process on the host operating system. The guest cannot communicate directly with the hardware or other computers but only through the VMM. The VMM provides emulated physical resources and devices to the guest which are accessed by the guest operating system to perform the required tasks. The VM settings control the resources provided to the guest, for example the amount of guest memory or the number of guest processors, (see chapitre 3.3, *Paramètres généraux*, page 46) and the enabled features for that guest (for example remote control, certain screen settings and others).

13.3.2 Secure Configuration of Virtual Machines

Several aspects of a virtual machine configuration are subject to security considerations.

13.3.2.1 Networking

The default networking mode for VMs is NAT which means that the VM acts like a computer behind a router, see chapitre 6.3, *Network Address Translation (NAT)*, page 93. The guest is part of a private subnet belonging to this VM and the guest IP is not visible from the outside. This networking mode works without any additional setup and is sufficient for many purposes.

If bridged networking is used, the VM acts like a computer inside the same network as the host, see chapitre 6.4, *Réseau bridgé*, page 95. In this case, the guest has the same network access as the host and a firewall might be necessary to protect other computers on the subnet from a potential malicious guest as well as to protect the guest from a direct access from other

computers. In some cases it is worth considering using a forwarding rule for a specific port in NAT mode instead of using bridged networking.

Some setups do not require a VM to be connected to the public network at all. Internal networking (see chapitre 6.5, *Réseau interne*, page 96) or host-only networking (see chapitre 6.6, *Réseau privé avec l'hôte (Host-only)*, page 97) are often sufficient to connect VMs among each other or to connect VMs only with the host but not with the public network.

13.3.2.2 VRDP remote desktop authentication

When using the VirtualBox extension pack provided by Oracle for VRDP remote desktop support, you can optionally use various methods to configure RDP authentication. The "null" method is very insecure and should be avoided in a public network. See chapitre 7.1.5, *RDP authentication*, page 105 for details.

13.3.2.3 Clipboard

The shared clipboard allows users to share data between the host and the guest. Enabling the clipboard in "Bidirectional" mode allows the guest to read and write the host clipboard. The "Host to guest" mode and the "Guest to host" mode limit the access to one direction. If the guest is able to access the host clipboard it can also potentially access sensitive data from the host which is shared over the clipboard.

If the guest is able to read from and/or write to the host clipboard then a remote user connecting to the guest over the network will also gain this ability, which may not be desirable. As a consequence, the shared clipboard is disabled for new machines.

13.3.2.4 Shared folders

If any host folder is shared with the guest then a remote user connected to the guest over the network can access these files too as the folder sharing mechanism cannot be selectively disabled for remote users.

13.3.2.5 3D graphics acceleration

Enabling 3D graphics via the Guest Additions exposes the host to additional security risks; see chapitre 4.5.1, *Accélération 3D matérielle (OpenGL et Direct3D 8/9)*, page 72.

13.3.2.6 CD/DVD passthrough

Enabling CD/DVD passthrough allows the guest to perform advanced operations on the CD/DVD drive, see chapitre 5.9, *Opération sur le lecteur de CD/DVD*, page 89. This could induce a security risk as a guest could overwrite data on a CD/DVD medium.

13.3.2.7 USB passthrough

Passing USB devices to the guest provides the guest full access to these devices, see chapitre 3.10.1, *Paramètres USB*, page 54. For instance, in addition to reading and writing the content of the partitions of an external USB disk the guest will be also able to read and write the partition table and hardware data of that disk.

13.3.3 Configuring and Using Authentication

The following components of VirtualBox can use passwords for authentication:

- When using remote iSCSI storage and the storage server requires authentication, an initiator secret can optionally be supplied with the `VBoxManage storageattach` command. As long as no settings password is provided (command line option

  `--settingspwfile`

  , this secret is stored **unencrypted** in the machine configuration and is therefore potentially readable on the host. See chapitre 5.11, *Serveurs iSCSI*, page 90 and chapitre 8.17, *VBoxManage storageattach*, page 132.

- When using the VirtualBox web service to control a VirtualBox host remotely, connections to the web service are authenticated in various ways. This is described in detail in the VirtualBox Software Development Kit (SDK) reference; please see chapitre 11, *VirtualBox programming interfaces*, page 181.

13.3.4 Potentially insecure operations

The following features of VirtualBox can present security problems:

- Enabling 3D graphics via the Guest Additions exposes the host to additional security risks; see chapitre 4.5.1, *Accélération 3D matérielle (OpenGL et Direct3D 8/9)*, page 72.

- When teleporting a machine, the data stream through which the machine's memory contents are transferred from one host to another is not encrypted. A third party with access to the network through which the data is transferred could therefore intercept that data. An SSH tunnel could be used to secure the connection between the two hosts. But when considering teleporting a VM over an untrusted network the first question to answer is how both VMs can securely access the same virtual disk image(s) with a reasonable performance.

- When using the VirtualBox web service to control a VirtualBox host remotely, connections to the web service (through which the API calls are transferred via SOAP XML) are not encrypted, but use plain HTTP by default. This is a potential security risk! For details about the web service, please see chapitre 11, *VirtualBox programming interfaces*, page 181.

  The web services are not started by default. Please refer to chapitre 9.19, *Starting the VirtualBox web service automatically*, page 169 to find out how to start this service and how to enable SSL/TLS support. It has to be started as a regular user and only the VMs of that user can be controled. By default, the service binds to localhost preventing any remote connection.

- Traffic sent over a UDP Tunnel network attachment is not encrypted. You can either encrypt it on the host network level (with IPsec), or use encrypted protocols in the guest network (such as SSH). The security properties are similar to bridged Ethernet.

13.3.5 Encryption

The following components of VirtualBox use encryption to protect sensitive data:

- When using the VirtualBox extension pack provided by Oracle for VRDP remote desktop support, RDP data can optionally be encrypted. See chapitre 7.1.6, *RDP encryption*, page 106 for details. Only the Enhanced RDP Security method (RDP5.2) with TLS protocol provides a secure connection. Standard RDP Security (RDP4 and RDP5.1) is vulnerable to a man-in-the-middle attack.

14 Known limitations

This sections describes known problems with VirtualBox 4.2.12_RPMFusion. Unless marked otherwise, these issues are planned to be fixed in later releases.

- The following **Guest SMP (multiprocessor) limitations** exist:
  - **Poor performance** with 32-bit guests on AMD CPUs. This affects mainly Windows and Solaris guests, but possibly also some Linux kernel revisions. Partially solved in 3.0.6 for 32 bits Windows NT, 2000, XP and 2003 guests. Requires 3.0.6 or higher Guest Additions to be installed.
  - **Poor performance** with 32-bit guests on certain Intel CPU models that do not include virtual APIC hardware optimization support. This affects mainly Windows and Solaris guests, but possibly also some Linux kernel revisions. Partially solved in 3.0.12 for 32 bits Windows NT, 2000, XP and 2003 guests. Requires 3.0.12 or higher Guest Additions to be installed.
- **64-bit guests on some 32-bit host systems with VT-x** can cause instabilities to your system. If you experience this, do not attempt to execute 64-bit guests. Refer to the VirtualBox user forum for additional information.
- **Direct 3D support in Windows guests.** For this to work, the Guest Additions must be installed in Windows "safe mode". Press F8 when the Windows guest is booting and select "Safe mode", then install the Guest Additions. Otherwise Windows' file protection mechanism will interfere with the replacement DLLs installed by VirtualBox and keep restoring the original Windows system DLLs.
- **Guest control.** On Windows guests, a process lauched via the guest control execute support is only able to display a graphical user interface if the user account it is started under, is currently logged in and has a desktop session. Otherwise, the process will not be able to display its user interface.

  Also, for using accounts without or with an empty password specified, the group policy needs to be changed on the guest. To do so, open the group policy editor on the command line by typing `gpedit.msc`, open the key *Computer Configuration\Windows Settings\Security Settings\Local Policies\Security Options* and change the value of *Accounts: Limit local account use of blank passwords to console logon only* to *Disabled*.
- **Guest multi-monitor support.** This feature is currently only supported with Windows guests.
- **Deleting the only snapshot with a running VM is not implemented.** Trying to perform this operation will result in an error message. This feature will be added in one of the next maintenance releases. It is possible to delete the only snapshot when the VM is not running, e.g. in "poweroff" or "saved" state.
- **Disabled host I/O caches.** Disabling the host I/O cache (see chapitre 5.7, *Images de disque et mise en cache E/S*, page 87) will yield poor performance with VHD and sparse VMDK files as these do not currently support asynchronous I/O. This does not apply to VDI files and raw disk/partition VMDK files, which do support async I/O. This restriction will be lifted in a future maintenance update.

- **Compacting virtual disk images is limited to VDI files.** The `VBoxManage modifyhd --compact` command is currently only implemented for VDI files. At the moment the only way to optimize the size of a virtual disk images in other formats (VMDK, VHD) is to clone the image and then use the cloned image in the VM configuration.
- **OVF import/export:**
  - When importing an OVF that was previously exported by VirtualBox 3.2 or higher which contains a complete VirtualBox machine configuration in the <vbox:Machine> element, some of the import customizations that can be specified (in either the GUI or on the VBoxManage command line) are presently ignored. In particular, customizations of the imported storage configuration are ignored. This will be fixed in the next release.
  - OVF localization (multiple languages in one OVF file) is not yet supported.
  - Some OVF sections like StartupSection, DeploymentOptionSection and InstallSection are ignored.
  - OVF environment documents, including their property sections and appliance configuration with ISO images, are not yet supported.
  - OVA archives (TAR containers) are not yet supported.
  - Remote files via HTTP or other mechanisms are not yet supported.
- **Seamless mode** does not work correctly with Linux guests that have 3D effects enabled (such as with compiz-enabled window managers).
- **Mac OS X hosts.** The following restrictions apply (all of which will be resolved in future versions):
  - The numlock emulation has not yet been implemented.
  - The CPU frequency metric is not supported.
  - 3D OpenGL acceleration, in particular with Linux guests that have 3D effects enabled (such as with compiz-enabled window managers).
  - Memory ballooning is not supported.
- **Mac OS X Server guests.**
  - Mac OS X Server guests can only run on a certain host hardware. For details about license and host hardware limitations, please see chapitre 3.1.1, *Invités Mac OS X Server*, page 44.
  - VirtualBox does not provide Guest Additions for Mac OS X Server at this time.
  - The graphics resolution currently defaults to 1024x768 as Mac OS X Server falls back to the built-in EFI display support. See chapitre 3.12.1, *Modes graphiques en EFI*, page 57 for more information on how to change EFI video modes.
  - Even when idle, Mac OS X Server guests currently burn 100% CPU. This is a power management issue that will be addressed in a future release.
  - Mac OS X Server guests only work with one CPU assigned to the VM. Support for SMP will be provided in a future release.
  - Depending on your system and version of Mac OS X Server, you might experience guest hangs after some time. This can be fixed by turning off energy saving (set timeout to “Never”) in the system preferences.
  - By default, the VirtualBox EFI enables debug output of the Mac OS X Server kernel to help you diagnose boot problems. Note that there is a lot of output and not all errors are fatal (they would also show on your physical Mac). You can turn off these messages by issuing this command:

```
VBoxManage setextradata "VM name" "VBoxInternal2/EfiBootArgs" "  "
```

To revert to the previous behavior, use:

```
VBoxManage setextradata "VM name" "VBoxInternal2/EfiBootArgs" ""
```

- **Solaris hosts.** The following restrictions apply for OpenSolaris and Solaris 10:
  - There is no support for USB devices connected to Solaris 10 hosts.
  - USB support on OpenSolaris hosts requires version snv_124 or higher. Webcams and other isochronous devices are known to have poor performance.
  - No ACPI information (battery status, power source) is reported to the guest.
  - No support for using wireless adapters with bridged networking.
- **Guest Additions for OS/2.** Shared folders are not yet supported with OS/2 guests. In addition, seamless windows and automatic guest resizing will probably never be implemented due to inherent limitations of the OS/2 graphics system.

15 Change log

This section summarizes the changes between VirtualBox versions. Note that this change log is not exhaustive; not all changes are listed.

VirtualBox version numbers consist of three numbers separated by dots where the first and second number represent the major version and the 3rd number the minor version. Minor version numbers of official releases are always even. An odd minor version number represents an internal development or test build. In addition, each build contains a revision number.

15.1 Version 4.2.12 (2013-04-12)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed a Guru Meditation on putting Linux guest CPU online if nested paging is disabled
- VMM: invalidate TLB entries even for non-present pages
- GUI: Multi-screen support: fixed a crash on visual-mode change
- GUI: Multi-screen support: disabled guest-screens should now remain disabled on visual-mode change
- GUI: Multi-screen support: handle host/guest screen plugging/unplugging in different visual-modes
- GUI: Multi-screen support: seamless mode: fixed a bug when empty seamless screens were represented by fullscreen windows
- GUI: Multi-screen support: each machine window in multi-screen configuration should have correct menu-bar now (Mac OS X hosts)
- GUI: Multi-screen support: machine window View menu should have correct content in seamless/fullscreen mode now (Mac OS X hosts)
- GUI: VM manager: vertical scroll-bars should be now updated on content/window resize
- GUI: VM settings: fixed crash on machine state-change event
- GUI: don't show warnings about enabled or disabled mouse integration if the VM was restored from a saved state
- Virtio-net: properly announce that the guest has to handle partial TCP checksums (bug #9380)
- Storage: Fixed incorrect alignment of VDI images causing disk size changes when using snapshots (bug #11597)
- Audio: fixed broken ALSA & PulseAudio on some Linux hosts due to invalid symbol resolution (bug #11615)
- PS/2 keyboard: re-apply keyboard repeat delay and rate after a VM was restored from a saved state (bug #10933)

- BIOS: updated DMI processor information table (type 4): corrected L1 & L2 cache table handles
- Timekeeping: fix several issues which can lead to incorrect time, Solaris guests sporadically showed time going briefly back to Jan 1 1970
- Main/Metrics: disk metrics are collected properly when software RAID, symbolic links or rootfs are used on Linux hosts
- VBoxManage: don't stay paused after a snapshot was created and the VM was running before
- VBoxManage: introduced *controlvm nicpromisc* (bug #11423)
- VBoxManage: don't crash on *controlvm guestmemoryballoon* of the VM isn't running (bug #11639)
- VBoxHeadless: don't filter guest property events as this would affect all clients (bug #11644)
- Guest control: prevent double CR in the output generated by guest commands and do NLS conversion
- Linux hosts / guests: fixed build errors on Linux 3.5 and newer kernels if the *CONFIG_UIDGID_STRICT_TYPE_CHECKS* config option is enabled (bug #11664)
- Linux Additions: handle fall-back to VESA driver on RedHat-based guests if vboxvideo cannot be loaded
- Linux Additions: RHEL/OEL/CentOS 6.4 compile fix (bug #11586)
- Linux Additions: Debian Linux kernel 3.2.0-4 (3.2.39) compile fix (bug #11634)
- Linux Additions: added auto-logon support for Linux guests using LightDM as the display manager
- Windows Additions: Support for multimonitor. Dynamic enable/disable of secondary virtual monitors. Support for XPDM/WDDM based guests (bug #6118)
- X11 Additions: support X.Org Server 1.14 (bug #11609)

15.2 Version 4.2.10 (2013-03-05)

This is a maintenance release. The following items were fixed and/or added:

- GUI: fixed keyboard with XQuartz X server (bug #10664)
- Main/Display: fixed a crash with multi-monitors under certain conditions (Mac OS X hosts only)
- Main/Properties: fixed a crash under certain conditions, for example after wakeup from host hibernate (bug #11444)
- Settings: don't lose the internal network settings if they are not currently active
- Storage: another incompatibility fix for VHD differencing images with Hyper-V (bug #5990)
- VBoxManage: don't read beyond the end of the file specified with *export –eulafile (bug #11528)*

- Linux hosts / guests: Linux 3.9-rc0 compile fixes
- Linux Additions: fixed two warnings in the shared folders guest kernel module (bug #11390)
- Linux Additions: don't crash VBoxService if libdbus is not available (bug #8638)
- Windows Additions: fixed upgrading MultiMedia Redirection (MMR) support

15.3 Version 4.2.8 (2013-02-20)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed a guest crash with a huge amount of guest RAM on VT-x hosts (bug #11306)
- GUI: fixed a layout bug in the Mac OS X clone VM dialog (bug #10982)
- GUI: not all the translation tags were taken into account during the language switch (bug #11342)
- GUI: the *take guest screenshot* dialog sometimes had no keyboard input available on Windows host
- Main/Machine: fix the generation of a spurious event for inaccessible VMs which triggered an endless event generation loop in cooperation with the GUI which became unresponsive (4.2.6 regression; bug #11323)
- Main/Display: fix for an access violation under certain conditions in multi-monitor configurations (bug #10539)
- Main/Metrics: network metrics are now collected for active (up) interfaces only, the state of an interface being evaluated when the associated metric is enabled via *setupMetrics*
- Snapshots: reduce the time for merging snapshots under certain conditions
- Storage: fixed data corruption after resizing a VDI image under certain circumstances (bug #11344)
- Storage: fixed non working online merging of snapshots (4.2.6 regression, bug #11359)
- Storage: fixed crash when connecting to certain QNAP iSCSI targets
- Storage: fixed incompatibility of VHD differencing images with Hyper-V (bug #5990)
- Bridged Networking: fixed TCP pseudo header checksum computation for IPv6 (bug #9380)
- 3D support: fix Battlefield 1942 game crashes (bug #11369)
- Settings: really sanitise the name of VM folders and settings file, the code was disabled before (bug #10549)
- Settings: allow to change VRDE settings for saved VMs
- VBoxManage: don't crash during *screenshotpng* if there is no display (bug #11363)
- Linux hosts: work around gcc bug 55940 which might lead to wrong kernel module code if gcc 4.7 is used to compile the 32-bit Linux host kernel (bug #11035)
- Linux hosts: fixed inconsistent lock state and deadlock warnings on module load and VM startup when CONFIG_PROVE_LOCKING is enabled (bug #11318)

- Linux hosts: made "]" key work again on Japanese keyboards
- Mac OS X hosts: don't crash the kernel during dtrace if the VBox kernel extensions are loaded (10.6 hosts only; bug #11273)
- Solaris / Mac OS X hosts: machine CPU load metrics now report 100% when all cores are fully utilized (used to be a single core)
- Solaris 11 host installer: wait for any services left over from a previous installation to be terminated to avoid confusing SMF.
- Guest Additions: don't block signals for processes executed via guest control
- Guest Additions: fixed a small memory leak in VBoxService (bug #10970)
- Windows Additions: fixed shared folder issue with large reads/writes on 64 bit Windows guests (bug #11115)
- Linux Additions: Linux 3.8 compile fixes (bug #11036)
- X11 Additions: fixed blocked SIGALRM in 3D desktop sessions (bug #10987)
- X11 Additions: fixed an unresolved reference in vboxvideo_drv for X.org 6.8 guests and before (e.g. RHEL4; 4.2.0 regression)
- X11 Additions: fixed screen automatic resizing for guests with X.org 1.3 or older (4.2.0 regression)

15.4 Version 4.2.6 (2012-12-19)

This is a maintenance release. The following items were fixed and/or added:

- VMM: don't inject stale VT-x events to prevent crashes after VM reset (bug #11256)
- VMM: workaround for buggy BIOSes which enable *MONITOR* only for certain CPUs (bug #9460)
- GUI: fixed trimming of anti-aliased text in details-view element headers (4.2.0 regression)
- GUI: fixed create-settings-file-alias functionality on Mac hosts (4.2.0 regression)
- GUI: fixed take-guest-screenshot functionality on Windows hosts (bug #11095)
- GUI: several minor fixes, including palette fixes (bug #11191)
- GUI: fixed Windows 2012 OS type (bug #11206)
- GUI: allow to terminate the VM even if VBoxSVC crashed
- API: fixed cancelling of snapshots, previously this could lead to inconsistent VM configs (bug #6877)
- API: fixed identifying disk images by UUID (bug #11209)
- 3D Support: several fixes
- VRDP: fixed occasional crash with external authentication (bug #11156)
- VGA: fix for double scan text modes
- USB: fixed invalid pending request count at the time of service of *DEVICE POWER* requests (Windows hosts only; bugs #10021, #11056)

- USB keyboard: Korean keyboard workaround (bug #11150)
- Storage: fixed hang with QCOW images and asynchronous I/O enabled
- Storage: fixed hang with newer VHD images (bug #11279)
- Storage: actually write the non-rotational flag to the VM settings (4.2.0 regression)
- Virtio-net: fixed the problem with network statistics counters in Session Information dialog (GUI)
- Metrics: introduced *network rate* and *disk usage* metrics
- Metrics: fixed a crash under certain conditions on Solaris hosts
- BIOS: fix for El Torito
- Shared Folders: if the host folder of a shared folder mapping does not exist, keep it active but mark it as invalid to prevent inconsitent saved states (bug #11147)
- VBoxManage: fixed converting disks from raw images
- VBoxManage: show snapshot description in the VM or snapshot information
- VBoxManage: make implicit opening of media consistent in all places dealing with media
- VBoxManage: the iSCSI initiator name was not stored in the settings file when doing *storageattach* (bug #11212)
- VBoxManage: *metrics collect* now properly handles `'CPU/MHz'` and `'Net/*/LinkSpeed'` metrics
- VBoxManage: changing the image UUID or parent UUID as part of *storageattach* works now in all safe cases
- VBoxManage: introduced *storageattach –medium additions* as a shortcut to mount the additions image (bug #11165)
- OVF: fixed importing OVF files cerated by recent VMware products (bug #10905)
- Linux hosts / Bridged Networking: fixed the problem with leaking connections in conntrack (bug #11178)
- Linux Additions: added support for ConsoleKit sessions in the vminfo service of VBoxService
- Linux Additions: don't crash during remount under certain conditions (bug #11291)
- Linux/Solaris Additions: fixed guest memory metrics collection
- Solaris hosts: added a dependency to ensure that the user directories are reachable when starting VBox services
- Windows host installer: integrated user-contributed translations, thanks to all contributors!
- Windows Additions: fixed auto-logon installation for Windows 8
- Windows Additions: don't fail if the shared folders host service is not available
- Windows Additions: fixed Guest Additions startup on Windows 2000 guests (bug #11253)
- Windows Additions: auto-resize fixes for Windows 8 guests

15.5 Version 4.2.4 (2012-10-26)

This is a maintenance release. The following items were fixed and/or added:

- GUI: fixed validation warning on global settings / proxy page (4.2.2 regression, bug #11089)
- GUI: fixed crash with multiple guest screens on certain conditions (OS X hosts only)
- VBoxBalloonCtrl: fixed command line argument handling of ballooning module
- VRDP: fixed occasional crash during a video playback in the guest (bug #11082)
- BIOS: fixed broken DMI information (4.2 regression)
- BIOS: workaround for booting from Windows 2000 floppy disks
- EFI: fixed video mode selection loss on VM reboot (#10983)
- Parallel: fixed parallel port printing failure/ paper queue empty error (Windows hosts only)
- NAT: fixed crash on alias-less DNS responses when host-resolver is used
- Storage: fixed hang under rare circumstances

15.6 Version 4.2.2 (2012-10-18)

This is a maintenance release. The following items were fixed and/or added:

- VMM: adapted to changes in Mac OS X 10.8.2 (bug #10965)
- GUI: restored VM item tool-tip functionality (4.2 regression)
- GUI: added group item tool-tip functionality
- GUI: fixed handling of the .ova/.ovf file name association (4.2 regression)
- GUI: it was not possible to change any setting before the first VM was created (bug #10928)
- GUI: disable grouping action if all the selected items are full children list of the same group already
- GUI: added menu for runtime drag-and-drop option change
- GUI: cleanup shared-clipboard menu on visual-mode change
- GUI: make sure VM receives keyboard focus on entering fullscreen-mode on Win host (bug #11051)
- GUI: disable proxy authentication for security reasons
- 3D Support: *DrawIndexedPrimitiveUP* implementation fixes for the Windows WDDM video driver (bug #10929)
- Storage: fixed a release assertion in the AHCI emulation when requests where cancelled with asynchronous I/O disabled
- Storage: fixed a hang during VM reset under certain circumstances (bug #10898)
- NAT: fixed a crash under rare circumstances (Windows hosts only; bug #10128)

- NAT: automatically use the host resolver if the host name server is set to some unusual loopback value (bug #10864)
- E1000: fixed a VirtualBox crash during intensive network transfers (4.2 regression; bugs #10936, #10969, #10980)
- ICH9: fixed PCI bridge initialization
- USB mouse: ensure that the last mouse event doesn't get lost if no URBs are available
- BIOS: certain legacy guests (e.g. Windows 95) didn't find the boot device after a warm reboot
- BIOS: don't trash the palette in text modes when setting the border color
- EFI: fixed OS X guest autoboot (4.2 regression)
- VBoxManage: fixed output of *showvminfo –machinereadable* (bug #10973)
- VBoxManage: fixed parsing of *storageattach –discard* (bug #11023)
- VBoxManage: fixed wrong output of the HPET setting in *showvminfo* (bug #11065)
- VBoxManage: fixed closing the guest session after executing a guest process via guest control
- VBoxShell: adaptions to interface name changes
- Guest Additions device: fixed a Guest Additions hang when a machine was reset after a crash
- Linux hosts / guests: Linux 3.7-rc1 fixes
- Linux Additions: support X.Org Server 1.13
- Linux Additions: fixed a hang when the X server was restarted with old guest kernels
- Linux Additions: fixed a VBoxService crash during CPU hot remove (bug #10964)
- Windows Additions: fixed automatic screen resize issue for NT4 guests
- OS/2 Additions: fixed shutdown hang
- OS/2 Additions: fixed mouse driver panic
- Solaris hosts: fixed autostart service going into maintenance mode after all VMs started
- Solaris hosts: fixed linking the host driver with the dtrace module

15.7 Version 4.2.0 (2012-09-13)

This is a major update. The following major new features were added:

- Improved Windows 8 support, in particular many 3D-related fixes
- GUI: VM groups (bug #288)
- GUI: expert mode for wizards
- GUI: allow to alter some settings during runtime

- Support for up to 36 network cards, in combination with an ICH9 chipset configuration (bug #8805)
- Resource control: added support for limiting network IO bandwidth; see chapitre 6.9, *Limiting bandwidth for network I/O*, page 99 (bug #3653)
- Added possibility to start VMs during system boot on Linux, OS X and Solaris; see chapitre 9.21, *Starting virtual machines during system boot*, page 169 (bug #950)
- Added experimental support for Drag'n'drop from the host to Linux guests. Support for more guests and for guest-to-host is planned. (bug #81)
- Added support for parallel port passthrough on Windows hosts
- Enhanced API for controlling the guest; please see the SDK reference and API documentation for more information

In addition, the following items were fixed and/or added:

- Mac OS X hosts: sign application and installer to avoid warnings on Mountain Lion
- VMM: fixed a potential host crash triggered by shutting down a VM when another VM was running (only affected 32-bit hosts and 64-bit OS X hosts, 4.1 regression, bug #9897)
- VMM: fixed a potential host crash under a high guest memory pressure (seen with Windows 8 guests)
- VMM: improved VM context switch performance for Intel CPUs using nested paging
- VMM: added support for *FlushByASID* features of AMD CPUs (Bulldozer and newer)
- VMM: fixed unreal mode handling on older CPUs with VT-x (gPXE, Solaris 7/8/9; bug #9941)
- VMM: fixed MP tables fixes for I/O APIC interrupt routing relevant for ancient SMP guests (e.g. old OS/2 releases)
- VMM: support recent VIA CPUs (bug #10005)
- VMM: fixed handling of task gates if VT-x/AMD-V is disabled
- VMM: page fusion fixes
- GUI: network operations manager
- GUI: allow taking screenshots of the current VM window content (bug #5561)
- GUI: allow automatically sorting of the VM list
- GUI: allow starting of headless VMs from the GUI
- GUI: allow reset, shutdown and poweroff from the Manager window
- GUI: allow to globally limit the maximum screen resolution for guests
- GUI: show the full medium part on hovering the list of recently used ISO images
- GUI: do not create additional folders when a new machine has a separator character in its name (bug #6541)
- GUI: don't crash on terminate if the settings dialog is still open (bug #9973)
- GUI: consider scaled DPI values when display fonts on Windows hosts (bug #9864)

- GUI: if a bridged network interface cannot be found, don't refuse to start the VM but allow the user to change the setting immediately
- Snapshots: fixed a crash when restoring an old snapshot when powering off a VM (bugs #9364, #9604, #10491)
- Clipboard: disable the clipboard by default for new VMs (see chapitre 13.3.2.3, *Clipboard*, page 199). It can be enabled at any time using the VM menu.
- Settings: sanitise the name of VM folders and settings file (bug #10549)
- Settings: allow to store the iSCSI initiator secret encrypted
- NAT: improvements for the built-in TFTP server (bugs #7385, #10286)
- NAT: fixed memory leak when disabling the NAT engine (bug #10801)
- E1000: 802.1q VLAN support (bug #10429)
- Storage: implemented burning of audio CDs in passthrough mode
- Storage: fixed audio CD passthrough for certain media players
- Storage: implemented support for discarding unused image blocks through TRIM for SATA and IDE and UNMAP for SCSI when using VDI images
- Storage: added support for QED images
- Storage: added support for QCOW (full support for v1 and readonly support for v2 images)
- Storage: added readonly support for VHDX images
- USB: don't crash if a USB device is plugged or unplugged when saving or loading the VM state (SMP guests only)
- Solaris additions: added support for X.org Server 1.11 and 1.12
- Solaris additions: switched to using an in-kernel mouse driver
- Windows hosts: no need to recreate host-only adapters after a VirtualBox update
- Windows hosts: updated toolchain; make the source code compatible to VC 2010 and enable some security-related compiler options
- Windows Additions: fixed memory leak in VBoxTray (bug #10808)

15.8 Version 4.1.18 (2012-06-06)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed *VERR_REM_VIRTUAL_CPU_ERROR* under rare conditions after the guest has been reset (bug #5164 and others)
- VMM: fixed host freezes with 64-bit guests on 32-bit Linux hosts (bug #10528)
- VRDP: added a workaround for rdesktop clients not properly updating the screen size when minimized
- AHCI: fixed a rare bug which can cause a guest memory corruption after the guest storage controler has been reset

- NAT: another attempt to fix crashes under rare conditions (Windows hosts only; bug #10513)
- Mac OS X hosts: addressed issues running Leopard / Snow Leopard (bug #10631)
- Linux hosts / Bridged Networking: fixed the problem with device driver unloading on kernels 3.2.18 and newer due to an invalid reference counter (bug #10624)
- Linux hosts / guests: Linux 3.5-rc1 fixes
- Linux Additions: the guest content was sometimes not properly updated (bug #9887)
- Solaris Additions: installer fix for X.org Server 1.11 and 1.12

15.9 Version 4.1.16 (2012-05-22)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed a Guru Meditation *VERR_ACCESS_DENIED* with certain guests (bugs #7589, #8247)
- VMM: fixed a Guru Meditation *VERR_PAGE_TABLE_NOT_PRESENT* with Ubuntu 32-bit guests with nested paging enabled on AMD CPUs (bug #10183)
- VMM: preserve segment limits and attributes when switching to unreal mode required for some legacy guests to work properly (VT-x without unrestricted guest execution only; bug #9941)
- VMM: fixed a VM hang after a resume from pause / savestate with SMP guests in rare cases
- 3D Support: several fixes for the Windows WDDM video driver crash
- NAT: fixed a crash on attempt to process ICMP datagram under some circumstances (bug #10527)
- Host-only Networking: lifted the maximal number of interfaces to 128 on Linux and Mac OS X hosts (there is no such limitation for Solaris and Windows hosts)
- EFI: fixed wrong SEC/PEI Core entry point calculation (bug #10531)
- VRDP: fixed a display update problem (bug #10503)
- Main: set the proper VM state if savestate failed for some reason (bug #6189)
- Main: more useful error message if a medium is inaccessible
- VBoxManage: fixed *controlvm savestate* if the VM is already paused
- Mac OS X hosts: addressed issues running on Mountain Lion Preview 3 (bug #10267)
- Linux hosts: Linux 3.4 compile fixes
- Linux hosts: fixed wrong help path in some rpm-based packages (bug #10418)
- Guest Additions: fixed handling of custom environment variables during `VBoxManage guestcontrol execute` (bug #10581)
- Windows Additions: fixed guest driver crash of VBoxSF in certain cases (4.1.10 regression, bug #10408)
- Windows Additions: don't load the WDDM driver if 3D support is not available for Windows 8 guests to keep the guest maintainable in that case (still better to miss some features than providing a blank screen)
- Solaris Additions: added support for X.org Server 1.11 and 1.12

15.10 Version 4.1.14 (2012-04-13)

This is a maintenance release. The following items were fixed and/or added:

- Network: fixed the problem with packets larger than MTU-4 when PCnet or PRO/1000 was bridged to certain types of adapters on OS X hosts (bug #3783)
- NAT: fixed a segfault under rare circumstances
- 3D Support: fixed Windows WDDM video driver crash for SMP guests (bugs #10200, #10331)
- Windows Additions, VRDP: fixed occasional corruption of vertical text

15.11 Version 4.1.12 (2012-04-03)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed *VERR_NOT_SUPPORTED* and *VERR_RAW_MODE_INVALID_SMP* guru meditation due to an invalid reschedule to raw mode (bug #10370)
- VMM: fixed *PDMCritSectLeave* guru meditation under rare circumstances with SMP guests
- VMM: proper *Math Fault* handling with certain legacy guests (bug #9042)
- NAT: fixed a socket leak under certain conditions
- Storage: better sanity check against reading beyond end-of-file
- Audio: fixed a crash in the NUL audio backend (bug #10374; 4.1.10 regression)
- HGCM: fixed a crash during savestate under rare circumstances
- Metrics: fixed an occasional crash during VM shutdown if host RAM/VMM metrics are enabled
- VBoxSVC: several locking fixes
- VBoxManage: return the correct error code if *controlvm savestate* failed (bug #10134)
- Guest Additions: VBoxService should not crash on exit (bug #10326; 4.1.10 regression)
- Windows Additions: set the correct time stamp when a file is closed (bug #6473)
- Windows Additions: better help if the DirectX backups are not properly installed
- Linux Additions: Linux 3.4-rc1 compile fixes

15.12 Version 4.1.10 (2012-03-13)

This is a maintenance release. The following items were fixed and/or added:

- GUI: if 3D support on the host is not available for some reason, do not only disable this VM setting but also uncheck the checkbox
- VMM: fixed a potential problem causing to schedule interrupts during *SYSEXIT* right after *STI*
- VMM: fixed a potential guest memory corruption issue with page fusion

- VMM: adjusted the module matching algorithm for page fusion a little, generally resulting in slightly more shared pages
- Main: host interfaces no longer have “defaults” for IP address and network mask attributes
- Main: don’t depend on a password for certain guest control operations (bug #10030)
- 3D Support: fixed Windows XP hosts support (4.1.8 regression; bugs #10071 and #10088
- 3D Support: rendering fixes for Linux hosts with NVIDIA graphics
- 3D Support: fixed saved state issues (4.1.8 regression; bug #10126)
- 3D Support: WDDM driver: fixed powershell_ise crashes (bug #10167), make WPF-based apps work with Aero enabled, fixed additional possible WinSAT crashes
- VRDP: fixed remote clipboard compatibility issues with some clients
- Storage: fixed a possible data corruption when compacting VDI or VHD images with snapshots (32-bit hosts only)
- iSCSI: fixed crash when using incorrect credentials when authenticating with a LIO target (bug #10173)
- Serial: don’t abort in host mode under rare error conditions (non-Windows hosts only)
- SDK: actually ship current C bindings
- SDK: fixed the Java glue code for JDK 1.7.0 (bug #9848)
- SDK: added Python example
- Metrics: make metrics collection more robust regarding blocked VMs
- Web service: added SSL/TLS support
- VBoxShell: fixed Guest Additions information lookup
- Solaris installer: fixed dependency checking while installing VirtualBox in a zone
- Linux hosts/guests: Linux 3.3-rc1 compile fixes
- Solaris hosts: fixed debug kernel panics while opening module CTF data (bug #9651)
- Mac OS X hosts: fixed Python support on Lion
- Linux Additions: make 3D passthrough work on OL/RHEL 6.2 (bug #10010)
- Linux Additions: fixed missing implementation when copying shared folder data through kernel high memory (bug #9878)
- Linux Additions: make sure all data is written when closing a memory mapped file on a shared folder
- Linux Additions: added support for X.Org Server 1.12
- Solaris Additions: fixed guest kernel driver to load properly on guest reboot (4.1.8 regression; bug #10113)
- Solaris Additions: fixed missing 64-bit OpenGL library VBoxOGL.so (bug #10151)
- Solaris Additions: fixed VBoxService import and start for Solaris 11 guests.
- Windows Additions: some Windows 8 adaptions
- Windows Additions: several fixes for shared folders (bug #9753)
- Guest control: miscellaneous bugfixes

15.13 Version 4.1.8 (2011-12-19)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed *VERR_MAP_FAILED* during savestate under certain circumstances (bug #7929)
- GUI: stop updating the VM status icons when the VM is paused (bug #8368)
- VBoxManage: fixed wrong return code after *startvm* (bug #9642)
- BIOS: fixed hang at launch of DOS applications generated by Clipper 5.3 (note that hardware virtualization may be required)
- USB: fixed OS/2 boot hang when using recent USB drivers
- NAT: increase maximum number of parallel connections making connections with port forwarding more robust (#8471)
- Metrics: fixed potential problem with invalid access in guest metrics collection upon VM destruction
- Main: don't crash if a medium is ejected twice (bug #9922)
- VBoxSVC: fixed crash under rare circumstances (e.g. client crash)
- VRDP: fixed screen freeze (bug #9620)
- OVF/OVA: fixed broken disk images on import under rare circumstances
- OVF/OVA: better error message when importing corrupted appliances
- VMDK/VHD: fixed a possible corruption with host cache disabled when using snapshots under rare circumstances (bug #9749)
- 3D Support: fixed full screen mode issues for ATI graphics (bug #9775), Windows Media Player rendering for XPDM-based Direct3D support (bug# 8341). Multiple fixes to XPDM and WDDM - based 3D support for Windows Guests and for 3D support in general
- Linux hosts: fixes for Fedoras Linux 2.6.41 (bug #9948)
- Linux hosts/guests: fixes for Linux 3.2 (bug #9743)
- Solaris Additions: various shared folder fixes (bugs #9856, #9862, #9917)
- Windows Additions: various fixes for Direct3D support (un)installation, added detection of missing or wrong Direct3D system files

15.14 Version 4.1.6 (2011-11-04)

This is a maintenance release. The following items were fixed and/or added:

- VRDP: fixed screen corruption
- NAT: the interface stopped working after a lot of failed ICMP requests (bug #9371)
- E1000: fixed rare Windows 7 guest hangs, either at boot time or when reconfiguring the network card in unusual setups (bug #6223)
- ATA: fixed a possible crash during ATAPI passthrough with certain guests
- ATA: improved compatibility with ancient Linux kernels

- Main: fixed incorrect framebuffer information after leaving the full screen mode with X11 guests, this lead to a scrambled preview window in the GUI for example
- Mac OS X hosts: fixed the problem with duplicate packets when bridged to a wireless interface (bug #9648)
- Linux hosts: fix for 3D support on Fedora 15 (bug #9799)
- Linux hosts: don't call *del_timer_sync* from an interrupt context (Fedora bug report 746331)
- Windows Vista and 7 guests: more WDDM fixes
- Linux Additions: fixed kernel module compilation failure on Redhat Enterprise Linux 4 (bug #9709)
- Linux Additions: install the DRI driver correctly on Ubuntu 11.10 guests
- Solaris Additions: added read-only mmap support for shared folders
- Solaris Additions: added directory, file mode and mask options for shared folders
- Windows Additions: implemented faster detection of logged-in guest users and stale sessions
- X.Org Additions: fixed graphical corruption when switching to a virtual terminal (bug #9490)

15.15 Version 4.1.4 (2011-10-03)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed PAE guests running on 32-bit hosts (4.0 regression; bug #9458)
- VMM: fixed *INVALID_PARAMETER* guru meditation caused by insufficient memory conditions (4.1.0 regression; bug #9240)
- VMM: fixed clobbered CPU registers during *stos/lods/ins/outs* emulation under rare circumstances
- VMM: another fix for 64-bit guests on recent AMD CPUs
- GUI: warn the user if he uses an outdated extension pack and allow to download and upgrade in that case (bug #8025)
- GUI: fixed spurious LCtrl key events on some Windows hosts (4.0.6 regression; bug #373)
- VBoxManage: another fix for *extpack install –replace* (bug #9308)
- VBoxManage: allow to specify the UUID of the target image in *convertfromraw*
- VRDP: fixed a rare crash when two or more clients connect to the server (bug #9503)
- VRDP: fixed a case when a client was not resized correctly (bug #9242)
- USB: make device capturing work on Windows hosts with usbhub class filter drivers installed (bug #9298); bugfixes for Windows host USB support
- VHD: fixed reads crossing block boundaries (bug #8532)
- VMDK: fixed progress indicator when creating split VMDK images

- Floppy: fixed medium size detection for physical mediums (Windows hosts only)
- Main: fixed VM initialization if a shared folder does not exist (bug #7941)
- Main: fixed possible deadlock between hard disk enumeration and resetting of immutable disks (bug #9549)
- Main: fixed not detaching of virtual storage after a snapshot restore operation (bug #8296)
- Main: always reset modified state after restoring a snapshot (bug #9387)
- Main: fixed writing of changed VM settings to disk when using snapshots (bug #8932)
- Main: fixed taking screen shots (e.g. the preview window) for headless VMs if the Guest Additions are active
- Virtio-net: fixed the problem with UDP packets exceeding MTU (bug #9370)
- 3D Support: fixed incorrect rendering and possible crashes when switching to/from full screen with enabled 2D acceleration
- 3D Support: fixed compiz under ubuntu 9.10
- Guest control: miscellaneous small bugfixes
- Linux / Solaris hosts: don't use hard links in packages (4.1.2 regression; bug #9441)
- Mac OS X hosts: fix installation issues when installed as root (bug #1578)
- Mac OS X hosts: fixed packet capture issues with vboxnetX host-only interfaces (bug #8076)
- Solaris hosts: fixed incoming multicast packets for Crossbow based bridged networking (bug #9532)
- Solaris hosts: fixed starting EFI guests due to missing EFI ROM files (bug #9535)
- Windows hosts installer: fixed *ADDLOCAL* usage on command line (4.1.0 regression; bug #9488)
- Windows hosts installer: fixed dangling shortcut to the .chm help file (bug #9646)
- Windows hosts installer: try to fix installation error 2869 which appeared under certain circumstances (bug #9664)
- Windows hosts: VBoxHeadless should never open a separate console window (bug #3549)
- Guest Additions: fixed hang while waiting for guest execution output (4.1.2 regression; bug #9446)
- Linux Additions: add support for X.Org Server 1.11 (bug #9519)
- Linux Additions: suppress an incorrect error message and allow a shared folder to be mounted in multiple places (bug #9627)
- Linux Additions: start VBoxService correctly on openSUSE 12.1 (bug #6229)
- Linux Additions: properly finish the installation even if the compilation of the DRM module fails (which is not fatal)
- Solaris Additions: fixed memory leaks and panics while mounting (from console) and unmounting shared folders

- Windows Additions: fixed swapped mouse cursor hotspot coordinates (4.1.0 regression; bug #9447)
- Windows Additions: fixed *PAGE_FAULT_IN_NONPAGED_AREA* BSOD in VBoxDisp.dll when running SnippingTool (bug #9508)
- Windows Additions: make image viewer work correctly with Aero enabled (Vista and Windows 7)
- Windows Additions: fixed incorrect window border blurs under Aero with ATI cards (Vista and Windows 7)
- Windows Additions: fixed incorrect rendering when moving 3D applications outside guest desktop and back under Aero (Vista and Windows 7)
- Windows Additions: fixed guest hangs when switching back from full screen text mode applications (bug #9636)

15.16 Version 4.1.2 (2011-08-15)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed 64-bit guests on AMD Fusion CPUs (bug #8824)
- VMM: fixed handling of the *sysenter/sysexit* machine instructions in 64-bit guests on Intel hosts
- GUI: added linked clone support
- GUI: fixed error check when changing the VM controller settings (4.1.0 regression; bug #9246)
- GUI: fixed the inclusion of additional information (license, vendor, ...) on OVF export
- GUI: when taking a snapshot from the VM selector, don't do a live snapshot
- GUI: fixed immediate crash during start on certain environments due to a library version conflict
- Guest control execution: implemented copying single files from guest to the host (directories and filters not supported yet)
- Guest control execution: limited to serve up to 5 guest processes at a time by default
- Main: return the correct RDP default port fixing settings warnings in the GUI if the remote display was enabled at runtime (bug #9368)
- VBoxSVC: made the path comparison OS independent when images are registered (bug #9303)
- VBoxSVC: several fixes/performance improvements for cloning of VMs
- Snapshots: fixed "Solid-State drive" flag survival on snapshot creation (bug #9379)
- VBoxManage: implemented *copyfrom* and *stat* guest control tools to copy files from guest to host and to check for existence of files on the guest
- VBoxManage: fixed *extpack install –replace* (bug #9308)
- VBoxManage: allow to start multiple VMs with *startvm*

- VBoxManage: experimental support for SATA CD/DVD hotplugging
- Mouse support: fixed an off-by-one pointer position error affecting Windows guests (bugs #7566, #9321)
- VRDP: fixed a case when the screen was not redrawn correctly (bug #9242)
- NAT: changed TCP connection establishment according to Winsock API requirements (Windows hosts only; bug #8919)
- USB: fixed device recognition issues (bugs #9299) and fixed several device capturing issues (Windows hosts; bug #9299)
- USB: fixed issues with remote filters being applied to local USB devices
- Guest Additions: keep logged in user count in out-of-memory situations and warn about it
- OVA: fixed import of renamed OVA files (bug #9033)
- Windows host: fixed icon for .VDI files (bug #9393)
- Solaris hosts: fixed a bug preventing parallel usage of multiple host NICs with Crossbow based bridged networking
- Linux hosts: fixed random kernel panics on host suspend / shutdown (4.1.0 regression; bug #9305)
- Solaris Additions: fixed shared folders due to missing symbols in the vboxfs module (4.0.12 / 4.1.0 regression; bug #9264)
- Windows Additions: fixed file truncation on a shared folder with some applications (bugs #9276, #9315)
- Windows Additions: fixed shared folder issues with antivirus software and 64 bit Windows guests (bug #9318)
- Windows Vista and 7 guests: WDDM driver fixes and performance enhancements, fixed WinSAT crashes (#9267)
- Linux Additions: fixed *llseek* for Linux kernels 2.6.37 and above

15.17 Version 4.1.0 (2011-07-19)

This version is a major update. The following major new features were added:

- Support for cloning of VMs (bug #5853, see chapitre 1.11, *Cloning virtual machines*, page 28): full clones can be created through the GUI and VBoxManage, linked clones only through VBoxManage
- GUI: enhanced wizard for creating new virtual disks
- GUI: new wizard for copying virtual disks
- GUI: keep the aspect ratio in scale mode (Windows and OSX hosts only; bug #7822)
- VMM: raised the memory limit for 64-bit hosts to 1TB
- Experimental support for PCI passthrough for Linux hosts, see chapitre 9.6, *PCI passthrough*, page 155

- Windows guests: Experimental WDDM graphics driver, supporting Windows Aero (bug #4607) and providing Direct3D support using a cleaner approach (no need to install the guest drivers in Safe Mode anymore)
- Guest Additions: status of modules and features can now be queried separately by the frontends
- Networking: new network attachment mode "Generic Driver", which offers an open plugin architecture for arbitrary and separately distributable virtual network implementations
- Host-only Networking: fixed host crash in kernels prior to 2.6.29
- New Networking Mode *UDP Tunnel*: allows to interconnect VMs running on different hosts easily and transparently, see chapitre 6.2, *Introduction aux modes réseaux*, page 92
- Experimental support for SATA hard disk hotplugging available with VBoxManage
- Solaris hosts: New Crossbow based bridged networking driver for Solaris 11 build 159 and above

In addition, the following items were fixed and/or added:

- VMM: more SMP timer fixes
- VMM: fixed sporadic recompiler crashes with SMP guests
- VMM: many small fixes
- GUI: when restoring a snapshot, ask for taking a snapshot of the current state
- GUI: added a *View* menu
- GUI: added a setting for the promiscuous mode policy for internal networks, bridged networks and host-only networks
- GUI: added slider for setting the CPU execution cap allowing to limit the amount of CPU time spent for the execution of the guest, see chapitre 3.4.2, *Onglet processeur*, page 49
- GUI: the VM description is editable during the runtime of a VM (bug #1551)
- GUI: added proxy settings (bug #2870)
- GUI: made the number of SATA ports configurable
- GUI: decrease time before showing the VM configuration dialog
- VBoxManage: more convenient configuration of storage controller attachments by automatically determining the port or device parameter when a storage controller has only one port or device per port
- VBoxManage: changed syntax of the *guestcontrol* command group, fixed various bugs, removed obsolete options
- VBoxBalloonCtrl: new service for automatic dynamic adjustment of the balloon size for running VMs
- Settings: machine names and snapshot names are not allowed to be a valid UUID
- Settings: provide better diagnostics if a single medium is used twice in a VM configuration
- Settings: provide better diagnostics for errors in medium create/merge/clone operations, and fix memory leaks in error cases

- Storage: ATA/SATA drives can be marked as non-rotational, i.e. the guest OS will detect them as a SSD if supported, which can improve performance
- Storage: virtual CD/DVD images will be detached if the guest ejects the medium, unless the drive is marked to handle ejects only on a temporary basis
- Storage: the medium UUID can be changed again when attaching a medium for the first time, which allows using images which are exact duplicates including the UUID
- Storage: fixed possible data corruption under certain circumstances with VHD and Parallels images (bug #9150)
- Storage: fixed unnecessary expansion when cloning differential images in VDI format
- Storage: fixed detection code to handle empty files for VDI and VMDK format
- Storage: fixed access to CD/DVD images beyond 4GB when using the SATA controller (bug #8592)
- Floppy: several bugs have been fixed
- Floppy: make it possible to unmount a host floppy disk (bug #6651)
- BIOS: disk-related structures are now checksummed correctly (bug #8739)
- USB: many fixes for the Windows USB host driver
- NAT: reduced memory footprint
- Networking: fixed the problem with segmentation offloading when several VMs are transmitting in parallel
- Networking: workaround for a bug in wireshark when operating directly on a capture file created by VirtualBox
- Serial: announce the serial devices in the ACPI tables to make Windows guests find the virtual hardware (bug #7411)
- VRDP: support for TLS connections (see chapitre 7.1.6, *RDP encryption*, page 106)
- VRDP: support for multimonitor client configurations with MS RDP clients
- VRDP: fixed a rare screen corruption
- 3D support: fixed *GL_VERSION* string for different locales (bug #8916)
- Web service: fixed timeout handling with HTTP 1.1 keepalive, and be more robust when connections fail
- VBoxSVC: fixed regression when several clients trigger autostart simultaneously
- Main: fixed incorrect handling of the medium location for media which are not file based (e.g. iSCSI), which resulted in confusing location values in many places
- JAX-WS client bindings: fixed resource leak
- Sources: fixed USB 2.0 support using extension packs for non-official builds
- Mac OS X hosts: fixed non-VT-x mode on Lion hosts
- Windows hosts: fixed copy'n'paste in the GUI and for the VM window (bug #4491)
- Windows hosts (64-bit only): enabled removing of all mediums when removing a VM

- Windows hosts (64-bit only): enabled live snapshot deletion
- Windows hosts: use native controls in the installer (bug #5520)
- Solaris hosts: fixed preemption issue with Solaris 11 hosts (builds 166 and above)
- Solaris hosts: better control of USB device access on Solaris 11 hosts
- Guest Additions: improved driver installation on Windows guests
- Guest Additions: fixed high CPU usage while executing guest programs from the host
- Solaris Additions: fixed automounting of shared folders (bug #8014)

15.18 Version 4.0.14 (2011-10-13)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed 64bit guests on AMD Fusion CPUs (bug #8824)
- VMM: fixed handling of the *sysenter/sysexit* machine instructions in 64-bit guests on Intel hosts
- GUI: fixed the inclusion of additional information (license, vendor, ...) on OVF export
- GUI: when taking a snapshot from the VM selector, don't do a live snapshot
- GUI: fixed spurious LCtrl key events on some Windows hosts (4.0.6 regression; bug #5908)
- VBoxManage: fixed *extpack install –replace*
- VRDP: fixed a rare screen corruption
- E1000: fixed rare Windows 7 guest hangs, either at boot time or when reconfiguring the network card in unusual setups (bug #6223)
- Mouse support: fixed an off-by-one pointer position error affecting Windows guests (bugs #7566, #9321)
- NAT: changed TCP connection establishment according to Winsock API requirements (Windows hosts only; bug #8919)
- VHD: fixed reads crossing block boundaries (bug #8532)
- VMDK: fixed progress indicator when creating split VMDK images
- Floppy: fixed medium size detection for physical mediums (Windows hosts only)
- VBoxSVC: made the path comparison OS independent when images are registered (bug #9303)
- Main: return the correct RDP default port fixing settings warnings in the GUI if the remote display was enabled at runtime (bug #9368)
- Main: fix VM initialization if a shared folder does not exist (bug #7941)
- Main: fixed taking screen shots (e.g. the preview window) for headless VMs if the Guest Additions are active
- Main: fixed possible deadlock between hard disk enumeration and the resetting of immutable disks (bug #9549)

- OVA: fixed import of renamed OVA files (bug #9033)
- Virtio-net: fixed the problem with UDP packets exceeding MTU (bug #9370)
- Solaris hosts: fixed starting EFI guests due to missing EFI ROM files (bug #9535)
- Mac OS X hosts: fix installation issues when installed as root (bug #1578)
- Windows Additions: fixed file truncation on a shared folder with some applications (bugs #9276, #9315)
- Linux Additions: suppress an incorrect error message and allow a shared folder to be mounted in multiple places (bug #9627)
- Linux Additions: add support for X.Org Server 1.11 (bug #9519)
- Linux Additions: fixed *llseek* for Linux kernels 2.6.37 and above
- Linux Additions: start VBoxService correctly on openSUSE 12.1 (bug #6229)
- Linux Additions: properly finish the installation even if the compilation of the DRM module fails (which is not fatal)

15.19 Version 4.0.12 (2011-07-15)

This is a maintenance release. The following items were fixed and/or added:

- Mac OS X hosts: Lion fixes
- Solaris hosts: fixed preemption issue with Solaris 11 hosts (builds 166 and above)
- VBoxManage: more convenient configuration of storage controller attachments by automatically determining the port or device parameter when a storage controller has only one port or device per port (bug #9188)
- Storage: fixed possible data corruption under certain circumstances with VHD and Parallels images (bug #9150)
- Storage: fixed access to CD/DVD images beyond 4GB when using the SATA controller (bug #8592)
- Floppy: make it possible to unmount a host floppy disk (bug #6651)
- Networking: fixed the problem with segmentation offloading when several VMs were transmitting at once
- 3D support: fixed *GL_VERSION* string for different locales (bug #8916)
- Sources: fixed USB 2.0 support using extension packs for non-official builds
- Solaris Additions: fixed automounting of shared folders (bug #8014)

15.20 Version 4.0.10 (2011-06-22)

This is a maintenance release. The following items were fixed and/or added:

- GUI: fixed disappearing settings widgets on KDE hosts (bug #6809)
- Storage: fixed hang under rare circumstances with flat VMDK images
- Storage: a saved VM could not be restored under certain circumstances after the host kernel was updated (bug #8983)
- Storage: refuse to create a medium with an invalid variant (for example Split2G with VDI; bug #7227)
- iSCSI: pause the VM if a request times out
- Snapshots: none of the hard disk attachments must be attached to another VM in normal mode when creating a snapshot
- USB: fixed occasional VM hangs with SMP guests (bug #4580)
- USB: proper device detection on RHEL/OEL/CentOS 5 guests (partial fix for bug #8978)
- ACPI: force the ACPI timer to return monotonic values for improve behavior with SMP Linux guests (bug #8511 and others)
- VRDP: fixed screen corruption under rare circumstances (bug #8977)
- rdesktop-vrdp: updated to version 1.7.0
- OVF: under rare circumstances some data at the end of a VMDK file was not written during export
- Mac OS X hosts: Lion fixes
- Mac OS X hosts: GNOME 3 fix
- Linux hosts: fixed VT-x detection on Linux 3.0 hosts (bug #9071)
- Linux hosts: fixed Python 2.7 bindings in the universal Linux binaries
- Windows hosts: fixed leak of thread and process handles
- Windows Additions: fixed bug when determining the extended version of the Guest Additions (4.0.8 regression; bug #8948)
- Solaris Additions: fixed installation to 64-bit Solaris 10u9 guests (4.0.8 regression)
- Linux Additions: RHEL6.1/OL6.1 compile fix
- Linux Additions: fixed a memory leak during `VBoxManage guestcontrol execute` (bug #9068)

15.21 Version 4.0.8 (2011-05-16)

This is a maintenance release. The following items were fixed and/or added:

- Mac OS X hosts: fixed incompatibility with recent Mac OS X versions in 64-bit mode (bug #8474)
- Mac OS X hosts: fixed incompatibility with hosts with more than 16 cores (bug #8389)
- Mac OS X hosts: fixed painting corruptions on a second monitor in 64-bit mode (bug #7606)
- GUI: restored functionality to set an empty host key to disallow any host key combination (4.0.6 regression; bug #8793)
- GUI: more expressive error messages for USB proxy permission problems (mainly Linux hosts; bug #8823)
- VBoxManage: added `controlvm screenshotpng` subcommand for saving the screenshot of a running VM in PNG format
- VBoxHeadless: fixed potential crash during shutdown (Windows hosts only)
- NAT: built-in services use the correct Ethernet addresses in Ethernet header and in ARP requests
- Host-only networking: fixed adapter reference counting
- E1000: fixed rare guest crashes with Linux SMP guests (bug #8755)
- SATA: fixed guest disk corruption under rare circumstances (only relevant for guests with more than 2GB RAM; bug #8826)
- Storage: fixed data corruption after a snapshot was taken with asynchronous I/O enabled (bug #8498)
- Floppy: several improvement
- HPET: another fix for time jumps (bug #8707)
- USB: use correct permissions when creating /dev/vboxusb (Linux hosts only)
- USB: removed assumption that string descriptors are null-terminated (Windows hosts only)
- 3D support: fixed a potential crash when resizing the guest window
- 3D support: fixed GNOME 3 rendering under Ubuntu 11.04 and Fedora 15
- Snapshots: fixed another bug which could lose entries in the media registry when restoring a snapshot (bug #8363)
- Shared Folders: don't stop mounting the other valid folders if one host folder is inaccessible (4.0.6 regression)
- Linux Additions: check whether gcc and make are installed before building kernel modules (bug #8795)
- Solaris Additions: added support for X.Org Server 1.10
- Guest Additions: fixed inappropriate Guest Additions update notification when using vendor-specific version suffixes (bug #8844)

15.22 Version 4.0.6 (2011-04-21)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed incorrect handling of ballooned pages when restoring a VMM from a saved state
- VMM: don't crash on hosts with more than 64 cores / hyperthreads; implemented support for up to 256 host cores (except Windows hosts; bug #8489)
- VMM: fixed guru meditation for PAE guests running on hosts without PAE (bug #8006)
- VMM: fixed slow Linux guests with raw mode and recent guest kernels (bug #8726)
- GUI: support host key combinations (bug #979)
- GUI: fixed progress indicator (bug #7814)
- GUI: show the mouse pointer while the VM is paused if the USB tablet mouse emulation is used (bug #6799)
- GUI: adapt the snapshot folder as well when renaming a VM (bug #8469)
- GUI: persistently remember the last folders of the disk/DVD/floppy selectors
- GUI: never allow to start a VM with USB-2.0 activated if the proper extension pack is missing (bug #8182)
- GUI: fixed hang/crash when opening a file dialog in an non-existing folder (bug #8673)
- Snapshots: fixed a bug which could lose entries in the media registry when restoring a snapshot (bug #8363)
- Snapshots: allow snapshots to be stored in the VM directory
- 3D support: fixed a crash if a VM was forced to terminate (Windows hosts only; bug #7133)
- Storage: fixed memory leak (4.0 regression; bug #7966)
- Storage: fixed access to iSCSI targets over internal network
- Storage: fixed reading from disks with more than one snapshot for VHD and VMDK images with disabled host cache (bug #8408)
- Storage: fixed a possible hang during VM suspend after an I/O error occurred
- Storage: fixed a possible hang during VM suspend / reset (bug #8276, #8294)
- Storage: automatically create a diff image when attaching a streamOptimized VMDK image to a VM
- ATA/SATA: fixed automounting of virtual CD/DVD mediums with recent Linux distributions by correctly reporting the current profile as 'none' if no medium is present
- Buslogic: fixed emulation for certain guests (e.g. jRockit VE)
- Host-Only Networking: fixed interface creation failure on Windows hosts (4.0.4 regression; bug #8362)
- Host-Only & Bridged & Internal Networking: fix for processing promiscuous mode requests by VMs, defaulting to switch behaviour

- Host-Only Networking: fixed connectivity issue after resuming the host from sleep (bug #3625)
- Bridged Networking: support for interface bonding on Mac OS X hosts (bug #8731)
- NAT: fixed processing of ARP announcements for guests with static assigned IPs (bug #8609)
- VRDP: backward compatibility with VRDPAuth external authentication library (bug #8063)
- Shared Folders: don't fail to start a VM if a path is not absolute, for example when importing an OVF from a different host (bug #7941)
- Audio: fixed crash under certain conditions (bug #8527)
- USB: fixed a crash when plugging certain USB devices (bug #8699)
- HPET: fixed time jumps when reading the counter (bug #8707)
- OVF/OVA: automatically adjust disk paths if the VM name is changed on import
- OVF/OVA: fix export to slow medias
- OVF/OVA: automatically repair inconsistent appliances with multiple disks (bug #8253)
- rdesktop-vrdp: fixed an assertion triggered under certain conditions (bug #8593)
- Windows hosts: fixed occasional hangs during VM shutdown because sometimes COM was not properly uninitialized
- Mac OS X hosts: prevent the mouse from leaving the VM window while captured
- Mac OS X hosts: keep aspect ratio while resizing in scale mode (shift for old behaviour) (part of bug #7822)
- X11 hosts: fixed Yen key support (bug #8438)
- X11 hosts: fixed a regression which caused Host+F1 to pop up help instead of sending Ctrl+Alt+F1
- Linux hosts / Linux Additions: mangle IPRT symbols to allow installing VirtualBox inside a VM while the Guest Additions are active (bug #5686)
- Linux hosts / Linux guests: workaround for a bug in GLIBC older than version 1.11 leading to crashes under certain conditions (signed/unsigned problem with memchr on 64-bit machines)
- Solaris hosts: fixed a deadlock in event semaphores that could lead to unkillable VM processes
- Windows Additions: fixed Sysprep parameter handling
- Windows Additions: fixed spontaneous guest reboots under certain circumstances (4.0.2 regression; bugs #8406, #8429)
- Windows Additions: added auto logon support for locked workstations on legacy Windows versions
- Windows Additions: fixed driver bugcheck error when handling PnP messages (4.0 regression; bug #8367)
- Windows Additions: fixed memory leak in VBoxVideo

- X11 Additions: added support for X.Org Server 1.10 final
- Linux Additions: Linux kernel 2.6.39-rc1 fixes
- Linux Additions: improved auto-run support (bug #5509)
- Linux Additions: fix mouse support on SUSE 11 SP 1 guests (bug #7946)
- Solaris Additions: added support for X.Org Server 1.9
- Guest Additions: various bugfixes for guest control execution
- Web service: use own log file, with log rotation to limit size

15.23 Version 4.0.4 (2011-02-17)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed recompiler crashes under certain conditions (bugs #8255, #8319 and further)
- VMM: fixed running 64-bit guests on 32-bit host with nested paging enabled on AMD CPUs (4.0 regression; bug #7938)
- VMM: fixed timing issues / hangs for certain guests using the programmable interval timer (bugs #8033 and #8062)
- VMM: large page and monitoring fixes for live snapshots (bugs #7910, #8059, #8125)
- GUI: fixed error message when trying to exceed the maximum number of host network interfaces
- GUI: fixed saving of changes to the metadata of an existing snapshot (bug #8145)
- GUI: fixed rare crash on X11 hosts (bug #8131)
- GUI: when selecting a shared folder, start the file dialog in the users home directory (bug #8017)
- ExtPack: enforce the correct permissions which might be restricted by umask when creating directories (non-Windows hosts only; bug #7878)
- VBoxSDL: fixed crash when starting by specifying the VM UUID (4.0 regression; bug #8342)
- VBoxManage: allow savestate even if the VM is already paused
- VBoxManage: fixed *modifyvm –synthcpu* (bug #6577)
- VBoxManage: fixed hang when doing *guestcontrol execute –wait-for exit* and displaying process status on exit (bug #8235)
- VBoxManage: decreased CPU load during *guestcontrol execute –wait-for exit/stdout* while waiting for the guest process to terminate (bug #7872)
- VBoxManage: fixed *list hostdvds/hostfloppies*
- VBoxManage: fixed *storageattach* for host DVD drives and host floppy drives
- Metrics: introduced *RAM/VMM* base metric
- Main: improved sanity check when taking a VM screen shot (bug #7966)

- Main: fixed a crash under rare circumstances if a VM failed to start
- Main: fixed attaching of immutable disk images (bug #8105)
- Main: fixed a crash at VM shutdown (bug #6443)
- Main: fixed incorrect handling of cross-referenced medium attachments (bug #8129)
- Settings: fixed truncating of big integer values (4.0 regression)
- Settings: properly store the ICH9 chipset type (bug #8123)
- Host-Only & Bridged Networking: fixed VBox DHCP server startup issue for Windows hosts (4.0 regression; bug #7905)
- Host-Only Networking: re-create vboxnetX interfaces after vboxnetadp.ko module reload on Linux and Darwin (bugs #5934, #6341)
- NAT: fixed an mbuf leak under rare circumstances (bug #7459)
- ACPI: don't allow the guest to enter S4 by default and don't announce S1 and S4 in the ACPI tables if disabled (bug #8008)
- Graphics card: made re-enabling disabled screens work correctly to prevent problems when X11 guests enter screen saving mode (bug #8122)
- Storage: fixed write errors with snapshots if the host cache is disabled (4.0 regression; bug #8221)
- ATA/SATA: fixed reset handling after ACPI suspend/resume
- BusLogic: fixed hang with SMP VMs
- Serial: another attempt to prevent lost characters during transmission (bug #1548)
- Linux hosts/guests: Linux 2.6.38-rc1 compile fixes
- Mac OS X hosts: fixed VBoxSVC crash when listing host interfaces without default gateway (64-bit hosts only; bug #7955)
- Solaris/Darwin hosts: fixed VM CPU execution cap
- X.Org guests: fixed a crash on X server restart (bug #8231)
- X.Org guests: support X.Org Server 1.10 pre-release and Ubuntu 11.04 Alpha
- X.Org guests: Add EDID emulation in the graphics driver to prevent GNOME settings daemon changing the mode on login
- X.Org guests: never send graphics modes to the host that older VirtualBox versions can't handle
- Linux Additions: fixed a memory leak in the shared folders code if a host link is not readable (bug #8185)
- Windows Additions: fixed handling of Security Attention Sequence (SAS) with VBoxGINA

15.24 Version 4.0.2 (2011-01-18)

This is a maintenance release. The following items were fixed and/or added:

- GUI: don't crash if a removable host drive referenced from the VM settings vanished
- GUI: fixed a crash when using the KDE4 Oxygen theme and clicked on the settings button (4.0 regression; bug #7875)
- GUI: properly warn if the machine folder cannot be created (bug #8031)
- GUI: several fixes for multimonitor X11 guests
- ExtPack: don't make the installer helper application suid root (Linux .deb/.rpm packages only)
- ExtPack: improved user experience on Vista / Windows 7 when installing an extension pack
- ExtPack: fixed issue with non-ascii characters in the path name during installing an extension pack (bug #9717)
- ExtPack: fixed SELinux issues on 32-bit Linux hosts
- VBoxManage: Host-only interface creation and removal is now supported for all platforms except Solaris (bug #7741)
- VBoxManage: fixed segmentation fault when removing non-existent host-only interface
- Storage: fixed possible crashes with VMDK/VHD images with snapshots and asynchronous I/O (4.0 regression)
- Storage: don't eject the physical medium if a DVD/CDROM/floppy drive is detached from a VM (bug #5825)
- Storage: be more robust when a faulty guest sends ATA commands to an ATAPI device (bug #6597)
- Parallels: fixed deletion of the image during suspend, pause or power off (4.0 regression)
- Bridged networking: fixed host kernel panic when bridging to devices with no TX queue (4.0 regression; Linux hosts only; bug #7908)
- NAT: port-forwarding rule registration respects protocol parameter (bug #8094)
- E1000: fixed PXE boot issues with WDS (bug #6330)
- Virtio-net: fixed the issue with TX performance in some Linux guests
- ICH9: fixed VM crash (software virtualization only; bug #7885)
- VGA: fixed VESA screen issue (4.0 regression; bug #7986)
- Shared Folders: fixed parameter parsing when creating symbolic links, fixes 32-bit/64-bit bitness issue (bug #818)
- Main: fixed crash under rare circumstances due to an invalid logging string (4.0 regression)
- Main: improve error information propagation for errors preventing a VM start
- Main: fixed problems with snapshots and non-ASCII characters in machine paths (bug #8024)

- Web service: now listens to localhost by default as documented (bug #6067)
- Settings: do not fail loading machine settings if removeable drive attachment (host drive or image) cannot be found; with 4.0 this is much more likely when machines are moved from one host to another
- Settings: fixed issue that changing a snapshot name or description was not saved to machine XML
- OVF/OVA: fixed import of files created by other OVF tools (bug #7983)
- rdesktop-vrdp: fix a crash during USB device enumeration (bug #7981)
- Linux hosts: fixed a crash during USB device enumeration
- Linux hosts: try a bit harder to allocate memory (bug #8035; 4.0 regression)
- Guest Additions: fixed parsing of parameters for guest control in VBoxService (4.0 regression; bug #8010)
- Windows Guest Additions: automatic logon on Windows Vista/Windows 7 now supports unlocking previously locked workstations

15.25 Version 4.0.0 (2010-12-22)

This version is a major update. The following major new features were added:

- Reorganization of VirtualBox into a base package and Extension Packs; see chapitre 1.5, *Installer et lancer VirtualBox*, page 15
- New settings/disk file layout for VM portability; see chapitre 10.1, *Where VirtualBox stores its files*, page 171
- Major rework of the GUI (now called “VirtualBox Manager”):
  - Redesigned user interface with guest window preview (also for screenshots)
  - New “scale” display mode with scaled guest display; see chapitre 1.7.3, *Resizing the machine’s window*, page 23
  - Support for creating and starting `.vbox` desktop shortcuts (bug #1889)
  - The VM list is now sortable
  - Machines can now be deleted easily without a trace including snapshots and saved states, and optionally including attached disk images (bug #5511; also, `VBoxManage unregistervm --delete` can do the same now)
  - Built-in creation of desktop file shortcuts to start VMs on double click (bug #2322)
- VMM: support more than 1.5/2 GB guest RAM on 32-bit hosts
- New virtual hardware:
  - Intel ICH9 chipset with three PCI buses, PCI Express and Message Signaled Interrupts (MSI); see chapitre 3.4.1, *Onglet Carte mère*, page 47
  - Intel HD Audio, for better support of modern guest operating systems (e.g. 64-bit Windows; bug #2785)
- Improvements to OVF support (see chapitre 1.12, *Importer et exporter des machines virtuelles*, page 29):

  - Open Virtualization Format Archive (OVA) support
  - Significant performance improvements during export and import
  - Creation of the manifest file on export is optional now
  - Imported disks can have formats other than VMDK

- Resource control: added support for limiting a VM's CPU time and IO bandwidth; see chapitre 5.8, *Limiting bandwidth for disk images*, page 89
- Storage: support asynchronous I/O for iSCSI, VMDK, VHD and Parallels images
- Storage: support for resizing VDI and VHD images; see chapitre 8.22, *VBoxManage modifyhd*, page 136
- Guest Additions: support for multiple virtual screens in Linux and Solaris guests using X.Org server 1.3 and later
- Language bindings: uniform Java bindings for both local (COM/XPCOM) and remote (SOAP) invocation APIs

In addition, the following items were fixed and/or added:

- VMM: Enable large page support by default on 64-bit hosts (applies to nested paging only)
- VMM: fixed guru meditation when running Minix (VT-x only; bug #6557)
- VMM: fixed crash under certain circumstances (Linux hosts only, non VT-x/AMD-V mode only; bugs #4529 and #7819)
- GUI: add configuration dialog for port forwarding in NAT mode (bug #1657)
- GUI: show the guest window content on save and restore
- GUI: certain GUI warnings don't stop the VM output anymore
- GUI: fixed black full screen minitoolbar on KDE4 hosts (Linux hosts only; bug #5449)
- BIOS: implemented multi-sector reading to speed up booting of certain guests (e.g. Solaris)
- Bridged networking: improved throughput by filtering out outgoing packets intended for the host before they reach the physical network (Linux hosts only; bug #7792)
- 3D support: allow use of *CR_SYSTEM_GL_PATH* again (bug #6864)
- 3D support: fixed various clipping/visibility issues (bugs #5659, #5794, #5848, #6018, #6187, #6570)
- 3D support: guest application stack corruption when using glGetVertexAttrib[ifd]v (bug #7395)
- 3D support: fixed OpenGL support for libMesa 7.9
- 3D support: fixed Unity/Compiz crashes on natty
- 2D Video acceleration: multimonitor support
- VRDP: fixed rare crash in multimonitor configuration
- VRDP: support for upstream audio
- Display: fixed occasional guest resize crash

- NAT: port forwarding rules can be applied at runtime
- SATA: allow to attach CD/DVD-ROM drives including passthrough (bug #7058)
- Floppy: support readonly image files, taking this as the criteria for making the medium readonly (bug #5651)
- Audio: fixed memory corruption during playback under rare circumstances
- Audio: the DirectSound backend now allows VMs to be audible when another DirectSound application is active, including another VM (bug #5578)
- EFI: support for SATA disks and CDROMs
- BIOS: reduce the stack usage of the VESA BIOS function *#4F01* (Quake fix)
- OVF/OVA: fixed export of VMs with iSCSI disks
- Storage: Apple DMG image support for the virtual CD/DVD (bug #6760)
- Linux host USB support: introduced a less invasive way of accessing raw USB devices (bugs #1093, #5345, #7759)
- Linux hosts: support recent Linux kernels with *CONFIG_DEBUG_SET_MODULE_RONX* set
- Guest Additions: Shared Folders now can be marked as being auto-mounted on Windows, Linux and Solaris guests
- Linux Additions: Shared Folders now support symbolic links (bug #818)
- Linux Additions: combined 32-bit and 64-bit additions into one file
- Windows Additions: automatic logon on Windows Vista/Windows 7 is now able to handle renamed user accounts; added various bugfixes

15.26 Version 3.2.12 (2010-11-30)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed rare host crash when running 64-bit guests on 32-bit hosts (bug #7577)
- VMM: fixed host reboots under rare circumstances due to NMIs triggered by active performance counters (Linux hosts in non-VT-x/AMD-V mode only; bug #4529)
- VMM: fixed out of memory guru meditation for large memory guests (bug #7586)
- VMM: fixed a guru meditation related to large pages
- VMM: use new VT-x feature to keep the guest from hogging the CPU
- Snapshots: implemented deleting the last remaining snapshot while the VM is running
- GUI: perform the checks for exceeding the size limit of the host file system and for broken asynchronous I/O on older Linux kernels with ext4 / xfs file systems not only when starting the VM from scratch but also when starting from a saved state
- NAT: fixed memory leak (3.2.0 regression; bugs #6918, #7353)
- NAT: fixed Linux NFS root issue (bugs #7299)
- Networking: fixed VM reset handling in e1000

- VRDP: fixed rare crash in multimonitor configuration
- Display: fixed occasional guest resize crash
- Mouse: don't send relative mouse events together with absolute mouse events (3.2.10 regression; bug #7571)
- Keyboard: fixes for the USB keyboard emulation; fixes for Korean keyboards
- Serial: don't hang if the host device would block during open (bugs #5756, #5380)
- Serial: fixed modem status lines (Linux hosts only; bug #812)
- Graphics: Horizontal resolutions are no longer restricted to a multiple of 8 pixels (bug #2047; requires Guest Additions update).
- USB: fixed a crash with older Linux kernels and non-ASCII characters in device strings (Linux hosts only; bug #6983, #7158, #7733; version 3.2.8 contained an incomplete fix)
- USB: fixed a crash under rare circumstances (bug #7409; Windows hosts only)
- "iSCSI: respond to NOP-In requests from the target immediately to avoid being disconnected if the guest is idle
- 3D support: fixed a crash under certain circumstances (bug #7659)
- 3D support: fixed crashes for GLUT based apps (bug #6848)
- 3D support: added missing GLX 1.3 functionality (bugs #7652, #7195)
- 2D Video acceleration: fixed potential deadlock when saving the VM state (bug #4124)
- Windows hosts: another fix for BSODs under certain circumstances in VBoxNetFlt.sys (bug #7601)
- Solaris hosts: fixed host USB DVD drive detection
- Mac OS X hosts: fixed swapped keys for certain ISO keyboard types (bug #2996)
- Linux hosts: added link state handling for TAP devices needed for proper operation with bridged networking on kernels 2.6.36 and above (bug #7649)
- Linux hosts/guests: Linux 2.6.37 fixes
- Linux Additions: properly compile the vboxvideo module if DKMS is not installed (bug #7572)
- Linux Additions: fixed a memory leak when accessing non-existing files on a Shared Folders (bug #7705)
- Windows Additions: skip none-mapped user accounts when enumerating user accounts for VM information

15.27 Version 3.2.10 (2010-10-08)

This is a maintenance release. The following items were fixed and/or added:

- VMM: V8086 mode fix for legacy DOS/Windows guests with EMM386 (3.2.8 regression)
- VMM: stability fix (bug #7342)
- VMM: fixed a Guru meditation related to large pages (bug #7300)
- VMM: fixed support for large pages on Linux hosts
- VMM: fixed a Guru meditation for large memory 64-bit guests on 32-bit hosts with nested paging (bug #7544)
- VMM: performance improvements for VMs with more than 2GB RAM (bug #6928)
- GUI: fixed host key handling if the host key is set to Left Alt (Linux/Solaris hosts only; 3.2.0 regression; bug #6758)
- GUI: the VM can be minimized from the mini toolbar (bug #4952)
- GUI: handle Ctrl+Break properly on X11 hosts (3.2.0 regression; bug #6122)
- GUI: fixed the case where the user aborted the media selector for selecting the boot hard disk from the VM wizard
- GUI: added a check for Linux kernels 2.6.36 or later which are known to have the asynchronous I/O bug on ext4 / xfs file systems fixed (Linux hosts only)
- OpenSolaris guests: use SATA controller by default
- Storage: fixed I/O errors in the guest after compacting VDI images (3.2.6 regression; bug #7294)
- Storage: automatically repair base disk images with non-zero parent UUID which made them inaccessible (bug #7289)
- Storage: fixed corrupted images if a merge operation was canceled
- IDE: added ATAPI passthrough support for audio CDs
- SATA: fixed a potential hang during boot of recent Solaris guests
- SATA: handle out of disk space and similar conditions better
- iSCSI: fixed sporadic hangs when closing the connection
- VGA: fixed missing redraw with multiple screens under certain circumstances (bug #7291)
- VGA: serveral small fixes for legacy VGA graphics modes
- Bridged networking: fixed occasional host freeze during VM shutdown (Linux hosts only)
- NAT: don't check for the existence of the TFTP prefix when delivering a file via bootp (bug #7384)
- NAT: fixed resolving of names at the host resolver (bug #7138)
- NAT: under rare conditions the NAT engine consumed 100% CPU load (non-Windows hosts only)
- VRDP: fixed memory leak under certain circumstances (bug #5966)

- VRDP: fixed missing redraws with Windows guests under certain circumstances
- USB: properly discard blocking outstanding bulk URBs, fixes some printers
- USB: Blackberry fix (bug #6465)
- VBoxHeadless: fixed event queue processing problems which led to hangs if the VM could not be started successfully
- VBoxManage: don't crash if parameters with invalid characters are passed (bug #7388)
- VBoxManage: `clonehd`: fixed a bug where the command aborted with an error message under rare circumstances
- VBoxManage `metrics`: made it work for directly started VMs again (3.2.8 regression; bug #7482)
- 3D support: report *GLX_ARB_get_proc_address* as supported extension
- 3D support: guest application stack corruption when using *glGetVertexAttrib[ifd]v* (bug #7395)
- 3D support: fixed broken 3D support when switching to full screen / seamless modes (bug #7314)
- 3D support: fixed 32bit OpenGL apps under 64bit Windows XP/Vista (bug #7066)
- OVF: fixed bug when exporting a VM with multiple attached disks (bug #7366)
- OVF: fixed slow export for certain filesystems (bug #3719)
- OVF: disabled manifest (.mf file) support; manifests are no longer verified on import nor written on export
- Shared clipboard/Windows: improved the reliability of the shared clipboard on Windows hosts and guest (partial fix to bug #5266)
- Shared Folders: don't show an empty directory if filenames with an invalid encoding exist on the host (bug #7349)
- Shared Folders: return the proper error code when trying to list files for a non-existing wildcard (bug #7004)
- Audio: fixed guest memory corruption when capturing from the NULL audio backend (bug #6911)
- Audio: improved playback quality (less choppy)
- Web service: avoid unnecessary creation of idle threads
- Additions: fixed bug in the guest execution feature when passing more than one environment variable
- Additions: refresh all guest properties written by VBoxService after the VM was restored from a saved state
- Additions: fixed a *division by zero* crash of VBoxService under certain circumstances
- Additions: immediately resynchronize the guest time with the host time after the VM was restored from a saved state (bug #4018)
- Additions/Windows: fixed *LsaEnumerate* error when enumerating logged in users

- Additions/X.Org: support X.Org Server 1.9 (bug #7306)
- Additions/X.Org: don't crash VBoxClient during reboot
- Solaris hosts: fixed host DVD drive enumeration on Solaris 10
- Solaris hosts: added a custom core dumper to procure more data in the event of a VM crash
- Solaris guests: fixed user idle detection
- Solaris guests: fixed a possible panic in Shared Folders when using the wrong user or group IDs (bug #7295)
- Solaris guests: fixed Shared Folders from truncating files to 2GB on 32-bit guests (bug #7324)
- Windows hosts: fixed a BSOD under certain circumstances in VBoxNetFlt.sys (bug #7448)
- Linux hosts/guests: Linux 2.6.36 fixes
- Linux hosts/guests: DKMS fixes (bug #5817)
- Mac OS X hosts: fixed missing dock menu entries (bug #7392)

15.28 Version 3.2.8 (2010-08-05)

This is a maintenance release. The following items were fixed and/or added:

- VMM: properly terminate the VM with an error if the guest is trying to switch to the PAE mode but PAE is disabled in the VM settings
- GUI: switch to native file dialogs (Windows hosts only; bug #5459)
- GUI: don't use native file dialogs on KDE hosts (Linux hosts only; bug #6809)
- 3D support: fixed *GL_EXT_texture_sRGB* support
- PXE: fixed ZENworks PXE boot regression
- OVF: fixed slower export and larger images under certain circumstances (3.2.6 regression; bug #7073)
- USB: properly signal an interrupt if the port suspend status changes
- USB: respect the remote-only filter
- USB: avoid VM hang when changing the configuration of certain devices (Windows hosts only)
- USB: fix a crash with older Linux kernels and non-ASCII characters in device strings (Linux hosts only; bug #6983)
- PageFusion: fixed conflict with the guest execution feature
- PageFusion: fixed stability issues with a large number of VMs
- PageFusion: fixed host crashes with guest SMP and Win64 guests
- Memory ballooning: fixed problems restoring VMs with pre-allocation enabled
- Bridged networking: fixed performance issue with GRO enabled on bridged device (bug #7059)

- Hostonly networking: fixed performance issue (3.2.6 regression; bug #7081)
- Hard disks: fix auto-reset of immutable disk at VM startup (bug #6832)
- BusLogic: several fixes for Windows NT/2000 and SCO OpenServer guests
- LsiLogic: fixed I/O errors under rare circumstances
- Sharing disks: support for attaching one disk to several VMs without external tools and tricks
- Shared Folders: several fixes and performance enhancements for Solaris guests (bugs #4154 and #6512)
- Solaris Installer: added support for remote installations
- Guest Properties API: correctly support enumerating the properties of a running VM with an empty "patterns" field (bug #7171)
- Guest properties: properly delete transient properties on shutdown
- VRDP video redirection performance improvements and stability fixes
- Settings: silently fix host audio driver when reading machine XML settings files or OVF written by VirtualBox on a different host OS, for example convert DirectSound to PulseAudio (bug #7209)
- Settings: properly store the NAT network setting in XML settings file version 1.10 and later (bug #6176)
- VBoxManage: handle differencing images with parent UUID correctly in subcommand *openmedium disk* (bug #6751)
- Web service: enabled HTTP keepalive for much better performance
- Web service: added timestamps to logging output
- Web service: treat 8-bit strings as UTF-8 not ASCII
- X11 Additions: fix for Xorg 6.8 guests (e.g. RHEL4)

15.29 Version 3.2.6 (2010-06-25)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed host crash when running 64-bit guests on 32-bit hosts with certain Intel CPUs (VT-x only; bug #6166)
- VMM: allow 64-bit SMP guests on 32-bit hosts (VT-x and AMD-V only; does not apply to Mac OS X, which already supports it)
- VMM: fixed Guru mediation if guests with more than 2GB are booted with VT-x/AMD-V disabled (bug #5740)
- VMM: fixed TR limit trashing (VT-x and 64-bit host only; bug #7052)
- Page Fusion: several bug fixes for SMP guests (including bug #6964)
- Teleportation: several fixes and improvements
- Mac OS X server guests: compatibility fix

- EFI: fixed memory detection for guests with 2GB or more RAM assigned
- GUI: added a workaround for a Linux kernel bug which affecting asynchronous I/O on ext4 / xfs file systems (Linux hosts only)
- GUI: added setting for multiple VRDP connections; useful if multiple screens are enabled
- GUI: another fix for the keyboard capturing bug under metacity (bug #6727)
- GUI: fixed quit dialog when used in seamless or full screen mode (Mac OS X hosts only; bug #6938)
- GUI: handle the extra key on the Brazilian keyboard on X11 hosts again (bug #7022).
- 2D Video acceleration: fixed crashes when leaving the full screen mode (bug #6768)
- VBoxManage: fixed *storageattach* error handling (bug #6927)
- VBoxManage: fixed *dhcpserver add* (3.2.0 regression; bug #7031)
- Storage: fixed hang with images located on filesystems which don't support asynchronous I/O (bug #6905)
- Storage: fixed raw disks on Windows hosts (3.2.0 regression; bug #6987)
- LsiLogic: fixed hang with older Linux guests
- BusLogic: fixed hang during I/O
- SATA: set initial number of ports to 1 as some guests can't handle 30 ports (e.g. CentOS 4 and FreeBSD; bug #6984)
- SATA: performance improvement
- SCSI: fixed error when using the full format option during Windows installation (bug #5101)
- iSCSI: fixed authentication (bug #4031)
- Host-only/bridged networking: fixed excessive host kernel warnings under certain circumstances (Linux hosts only; 3.2.0 regression; bug #6872)
- NAT: fixed potential memory leaks
- NAT: increased the size of the memory pool for 16K Jumbo frames (performance tweak)
- NAT: allow to link/unlink the network cable even if the VM is currently paused
- E1000: disconnect cable was not properly handled if the NIC was not yet initialized by the guest
- OVF: export performance optimization
- OVF: upgraded OS type definitions to CIM 2.25.0 so that Windows 7 and other OSes are now tagged correctly on export
- Settings: the setting for disabling the host I/O cache was sometimes not properly saved
- Settings: save machine state into XML correctly even when snapshot folder has been changed to a non-default location (bug #5656)
- USB: allow the guest to disable an EHCI port

- USB: find a valid language ID before querying strings (bug #7034)
- POSIX hosts: fixed several memory leaks (3.2.0 regression)
- Solaris hosts: fixed VDI access problem under certain circumstances (IDE/SATA; 3.2.0 regression)
- Solaris hosts: fixed VM fails to start on 32-bit hosts (3.2.0 regression; bug #6899)
- Windows hosts (32-bit): increase guest RAM limit if the host kernel allows for more virtual address space
- Linux Additions: re-read a directory after a file was removed (bug #5251)
- Linux Additions: install the DRI driver in the right location on ArchLinux guests (bug #6937)
- X11 Additions: fixed spurious mouse movement events (bug #4260)
- Solaris Additions: fixed guest execution feature
- Windows Additions: automatic logon on Windows Vista/Windows 7 is now able to handle renamed and principal user accounts; added various bugfixes
- Windows Additions: improved command line parsing of the installer
- Windows Additions: fixed driver verifier bugcheck in VBoxMouse (bug #6453)
- 3D support: fixed OpenGL support for 32bit applications under 64bit Windows guests

15.30 Version 3.2.4 (2010-06-07)

This is a maintenance release. The following items were fixed and/or added:

- GUI: fixed superfluous resize-event on powering-on VM for X11 (improvement for the 3.2.2 fix)
- GUI: fixed keyboard capturing bug under metacity (bug #6727)
- Host-only/bridged networking: fixed guest-to-guest communication over wireless (3.2.0 regression; bug #6855)
- Storage: fixed a potential guest disk corruption with growing images (3.2.0 regression)
- Page Fusion: fixed shared module detection for Win64 guests
- 3D support: allow use of *CR_SYSTEM_GL_PATH* again (bug #6864)
- 3D support: fixed a host assertion for some multi-threaded guest applications (bug #5236)
- 3D support: fixed host crashes with nVIDIA drivers on WDDM startup
- OVF: fixed import of OVFs with a VM description (annotation) (3.2.2 regression; bug #6914)
- VRDP: fixed issues with secondary monitors (bug #6759)

15.31 Version 3.2.2 (2010-06-02)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed rare invalid guest state guru meditation (VT-x only)
- VMM: fixed poor performance with nested paging and unrestricted guest execution (VT-x only; bug #6716)
- VMM: fixed occasional guru meditation during Windows 7 bootup (bug #6728)
- GUI: keep the status for remote control in sync with the actual state
- GUI: don't exit after a successful refresh of an invalid VM configuration
- GUI: fixed keyboard capturing bug under metacity (bug #6727)
- GUI: fixed crash during VM termination if a modal dialog is open
- GUI: default controllers names of New VM Wizard are synchronized with VM settings
- GUI: fixed superfluous resize-event on powering-on VM for X11
- GUI: fixed regression - missed USB item's tool-tip of USB devices menu
- GUI: Activate VM window on mouse-hovering for multi-monitor VMs
- VBoxSDL/Linux hosts: automated keyboard type detection (bug #5764)
- SATA: fixed crash during VM suspend under rare circumstances
- SATA: fixed crash during VM reset after a snapshot was taken
- Storage: fixed sporadic hang of SMP guests using SATA or LSI Logic SCSI and asynchronous I/O
- Virtio-net: fix for guests with more than about 4GB RAM (bug #6784)
- Page Fusion: fixed VBoxService crash with enabled Page Fusion on Win64 guests
- Page Fusion: added kernel module sharing
- HGCM: fixed memory leak which showed up if the Guest Additions were accessing a non-existing HGCM service
- Teleportation: several fixes
- Floppy: don't disable the host I/O cache by default
- USB: fixed 3.1 regression with certain devices (e.g. iPhone); Windows host only
- Serial: updated the guest device emulation to 16550A and reduced the probability for losing bytes during transmission (bug #1548)
- NAT: re-fetch the name server parameters from the host on guest DHCP requests to handle host network switches more gracefully (bug #3847)
- NAT: fixed parsing of IPv4 addresses in CIDR notation (bug #6797)
- NAT: limit the number of name servers passed to the guest to four (non-Windows hosts only; bug #4098)
- NAT: fixed DNS transaction ID mismatch (bug #6833)

- VDE: fixed changing the attachment during runtime
- Bridged networking: fixed memory leak in the Bridged Networking driver for Windows hosts (bug #6824)
- Windows Additions: fix for NT4 guests (bug #6748)
- Windows Additions: re-introduced system preparation feature
- Linux guests: enable PAE for RedHat guests by default
- Linux guests: fix support for disabling mouse integration (bug #6714)
- Web service: fixed a rare crash when calling IGuest methods from the web service
- OVF: fixed wrong hard disk UUIDs on export (bug #6802)
- OVF: fixed 3.2.0 regression importing legacy OVF 0.9 files
- 3D support: fixed OpenGL support for 64bit applications on windows guests
- 3D support: fixed various host crashes (bugs #2954, #5713, #6443)

15.32 Version 3.2.0 (2010-05-18)

This version is a major update. The following major new features were added:

- Following the acquisition of Sun Microsystems by Oracle Corporation, the product is now called “Oracle VM VirtualBox” and all references were changed without impacting compatibility
- Experimental support for Mac OS X Server guests (see chapitre 3.1.1, *Invités Mac OS X Server*, page 44)
- Memory ballooning to dynamically in- or decrease the amount of RAM used by a VM (64-bit hosts only) (see chapitre 4.8, *Faire de la montgolfière avec la mémoire*, page 75)
- Page Fusion automatically de-duplicates RAM when running similar VMs thereby increasing capacity. Currently supported for Windows guests on 64-bit hosts (see chapitre 4.9, *Fusion de page*, page 76)
- CPU hot-plugging for Linux (hot-add and hot-remove) and certain Windows guests (hot-add only) (see chapitre 9.5, *CPU hot-plugging*, page 154)
- New Hypervisor features: with both VT-x/AMD-V on 64-bit hosts, using large pages can improve performance (see chapitre 10.6, *Nested paging and VPIDs*, page 179); also, on VT-x, unrestricted guest execution is now supported (if nested paging is enabled with VT-x, real mode and protected mode without paging code runs faster, which mainly speeds up guest OS booting)
- Support for deleting snapshots while the VM is running
- Support for multi-monitor guest setups in the GUI for Windows guests (see chapitre 3.5, *Paramètres d’affichage*, page 50)
- USB tablet/keyboard emulation for improved user experience if no Guest Additions are available (see chapitre 3.4.1, *Onglet Carte mère*, page 47)
- LsiLogic SAS controller emulation (see chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78)

- VRDP video acceleration (see chapitre 7.1.9, *VRDP video redirection*, page 108)
- NAT engine configuration via API and VBoxManage
- Use of host I/O cache is now configurable (see chapitre 5.7, *Images de disque et mise en cache E/S*, page 87)
- Guest Additions: added support for executing guest applications from the host system (replaces the automatic system preparation feature; see chapitre 4.7, *Contrôle invité*, page 75)
- OVF: enhanced OVF support with custom namespace to preserve settings that are not part of the base OVF standard

In addition, the following items were fixed and/or added:

- VMM: fixed Windows 2000 guest crash when configured with a large amount of RAM (bug #5800)
- Linux/Solaris guests: PAM module for automatic logons added
- GUI: guess the OS type from the OS name when creating a new VM
- GUI: added VM setting for passing the time in UTC instead of passing the local host time to the guest (bug #1310)
- GUI: fixed seamless mode on secondary monitors (bugs #1322 and #1669)
- GUI: offer to download the user manual in the OSE version (bug #6442)
- GUI: allow to set an empty host key to disallow any host key combination (bug #684)
- GUI: allow to restrict the possible actions when shutting down the VM from the GUI
- Main: allow to start a VM even if a virtual DVD or floppy medium is not accessible
- Settings: be more robust when saving the XML settings files
- Mac OS X: rewrite of the CoreAudio driver and added support for audio input (bug #5869)
- Mac OS X: external VRDP authentication module support (bug #3106)
- Mac OS X: moved the realtime dock preview settings to the VM settings (no global option anymore). Use the dock menu to configure it
- Mac OS X: added the VM menu to the dock menu
- 3D support: fixed corrupted surface rendering (bug #5695)
- 3D support: fixed VM crashes when using *ARB_IMAGING* (bug #6014)
- 3D support: fixed assertion when guest applications uses several windows with single OpenGL context (bug #4598)
- 3D support: added GL_ARB_pixel_buffer_object support
- 3D support: added OpenGL 2.1 support
- 3D support: fixed Final frame of Compiz animation not updated to the screen (Mac OS X only) (bug #4653)
- 3D support: fixed blank screen after loading snapshot of VM with enabled Compiz

- Added support for *Virtual Distributed Ethernet* (VDE) (Linux hosts only; see chapitre 6.2, *Introduction aux modes réseaux*, page 92)
- Added support for virtual high precision event timer (HPET)
- OVF: fixed mapping between two IDE channels in OVF and the one IDE controller in VirtualBox
- OVF: fix VMDK format string identifiers and sort XML elements from rasd: namespace alphabetically as prescribed by standard
- VBoxShell: interactive Python shell extended to be fully functional TUI for VirtualBox
- Linux Additions: support Fedora 13 (bug #6370)
- VBoxManage: fixed overly strict checks when creating a raw partition VMDK (bugs #688, #4438)

15.33 Version 3.1.8 (2010-05-10)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed crash with the OpenSUSE 11.3 milestone kernel during early boot (software virtualization only)
- VMM: fixed invalid state during teleportation
- VMM: fixed OS/2 guest crash with nested paging enabled
- VMM: fixed massive display performance loss (AMD-V with nested paging only)
- GUI: fixed off-by-one bug when passing absolute mouse coordinates to the guest (3.1.6 regression)
- GUI: show the real version of the Guest Additions, not the interface version
- GUI: when adding a DVD or floppy slot in the VM mass storage settings dialog, don't attach a random medium but just leave the slot empty
- GUI: added `--seamless` and `--fullscreen` command line switches (bug #4220)
- GUI: fixed a SEGFAULT under rare circumstances
- 2D Video acceleration: fixed display issues when working with non 32-bit modes (bugs #6094 & #6208)
- LsiLogic: fixed detection of hard disks attached to port 0 when using the drivers from LSI
- ATA: fixed sporadic crash with Linux guests when having a hard disk and DVD drive on the same channel (bug #6079)
- Network: allow to start a VM even if not all network adapters are attached
- Network: promiscuous mode support for e1000 and paravirtualized adapters (bug #6519)
- NAT: fixed ICMP latency (non-Windows hosts only; bug #6427)
- SCSI: fixed guest crashes under certain circumstances when booting from SCSI devices
- VBoxManage: fixed several bugs in cloning of images (one of them is bug #6408)

- VBoxManage: fixed *modifyvm –natnet default*
- Solaris hosts: fixed a kernel panic when bridged networking might fail to initialize
- Solaris hosts: fixed priority tagged VLAN packets in bridged networking
- Shared Folders: fixed issue with copying read-only files (Linux guests only; bug #4890)
- Shared Folders: renamed the guest kernel module from *vboxvfs* to *vboxsf* to make it load on demand by the Linux kernel. Fixes mounting from /etc/fstab in Ubuntu 10.04
- Shared Folders: fixed setuid file permissions (Solaris guests only)
- Shared Folders: fixed deleting directories recursively (Solaris guests only; bug #6513)
- Guest Additions: support seamless and dynamic resizing on certain older X11 guests (bug #5840)
- Solaris Additions: fixed OpenGL library dependencies (bug #6435)
- Keyboard/Mouse emulation: fixed handling of simultaneous mouse/keyboard events under certain circumstances (bug #5375)
- Mouse emulation: never switch straight back from Explorer to IntelliMouse mode as it confuses the FreeBSD mouse driver (bug #6488)
- SDK: fixed memory leak in *IDisplay::takeScreenShotSlow()* (bug #6549)
- 3D support: fixed Final frame of Compiz animation not updated to the screen (Mac OS X only) (bug #4653)
- VRDP: allow to bind to localhost only on Mac OS X (bug #5227)
- Linux hosts: add host USB support for Ubuntu 10.04 and other hosts without the hal daemon or usbfs (bug #6343)
- Web service: more structs and array fixes in PHP bindings
- Windows hosts: make the bridged networking driver notify dll be correctly unregistred on uninstall (bug #5780)

15.34 Version 3.1.6 (2010-03-25)

This is a maintenance release. The following items were fixed and/or added:

- Linux hosts: fixed timing issue on hosts with Linux kernels 2.6.31 or later with certain CPUs (asynchronous timer mode; bug #6250)
- Linux hosts: properly handle host suspend/resume events on Linux kernels 2.6.30 or later (bug #5562)
- Mac OS X hosts: fixed VBoxSVC crash while enumerating the host network interfaces under certain circumstances
- Snapshots: fixed image corruption after snapshot merge under certain circumstances (bug #6023)
- Snapshots: fixed crash with VBoxHeadless / OSE
- VMM: fixed reference counting guru meditation (bug #4940)

- VMM: improved guest SMP stability
- VMM: fixed VT-x hardware debug issues (bugs #477 & #5792)
- VMM: fixed *PGMDynMapHCPage* guru meditation (Mac OS X; VT-x only; bug #6095)
- VMM: fixed *pgmPoolTrackFlushGCPhysPTInt* guru meditations (Mac OS X; VT-x only; bugs #6095 & #6125)
- VMM: fixed host crash when running PAE guests in VT-X mode (Mac OS X only; bug #5771)
- GUI: fix displaying of error message (bug #4345)
- GUI: fix inability to enter seamless mode (bugs #6185, #6188)
- 3D support: fixed assertion and flickering when guest application uses several windows with a single OpenGL context (bug #4598)
- 3D support: fixed host crashes when using *GL_EXT_compiled_vertex_array* and array element calls (bug #6165)
- 3D support: fixed runtime linker errors with OpenGL guest libs (bug #5297)
- 3D support: fixed OpenGL extension viewer crash on startup (bug #4962)
- NAT: fixed a 3.1.4 regression on Windows hosts where graceful connection termination was broken (bug #6237)
- NAT: alternative network setting was not stored persistent (bug #6176)
- NAT: fixed memory corruption during ICMP traffic under certain circumstances
- Network: allow to switch the host interface or the internal network while a VM is running (bug #5781)
- VHD: fix for images with a block size different than 2MB
- USB: fixed filtered device attach regression (bug #6251)
- USB: fixed crash in OHCI under rare circumstances (bug #3571)
- VRDP: fixed hang under rare circumstances when attaching USB devices
- ACPI: prevent guest freezes when accessing /proc/acpi for determining the state of the host battery and the AC adapter (Linux hosts only; bug #2836)
- PulseAudio: fixed guest freezes under certain conditions (3.1.4 regression; bug #6224)
- BIOS: increased space for DMI strings
- BIOS: fixed interrupt routing problem for certain configurations (I/O-APIC enabled, ACPI not used; bug #6098)
- iSCSI: be more robust when handling the *INQUIRY* response
- iSCSI: be more robust when handling sense data
- BusLogic: fixed FreeBSD guests
- Web service: vboxwebsrv is now multithreaded
- Web service: fixed handling of structs and arrays in PHP bindings

- Solaris Installer: fixed netmask to stay persistent across reboots for Host-only interface (bug #4590)
- Linux installer: removed external dependency to libpng12.so (bug #6243)
- Solaris Additions: fixed superfluous kernel logging (bug #6181)
- Linux Additions: fixed hang when starting the X server in Fedora12 guests and in guests with Linux 2.6.33 or later (bug #6198)
- Linux Additions: support Mandriva speedboot runlevel (bug #5484)
- Linux Additions: fixed SELinux security context of mount.vboxsf (bug #6362)
- Linux Additions: support Ubuntu 10.04 (bug #5737)
- Web service: update PHP bindings to fix problems with enums and collections

15.35 Version 3.1.4 (2010-02-12)

This is a maintenance release. The following items were fixed and/or added:

- VMM: SMP stability fixes
- VMM: fixed guru meditation in certain rare cases (bug #5968)
- VMM: activate NXE for PAE enabled guests (VT-x and AMD-V on 32 bits hosts only; bug #3578)
- VMM: added workaround for broken BIOSes that make VirtualBox think AMD-V is in use (for details see bug #5639)
- VMM: fixed rare host reboot when restoring a saved state (bug #3945)
- VMM: fixed incompatibility with 2.6.32 Linux kernels (software virtualization only; bug #6100)
- VMM: turn on nested paging by default for new VMs (if available; VT-x and AMD-V only)
- VMM: turn on VPID by default for new VMs (if available; VT-x only)
- VMM: perform strict CPUID compatibility checks when teleporting; to get the old behavior set “VBoxInternal/CPUM/StrictCpuIdChecks” to 0
- VMM: fixed VM crash with certain 16 bits Windows applications (software virtualization only; bug #5399)
- Snapshots: fixed a 3.1 regression that broke deletion of snapshots when a machine had immutable or writethrough storage attached (bug #5727)
- Saved state: fixed *VERR_SSM_LOADED_TOO_MUCH* error when loading *DisplayScreenshot*(bug #6162)
- VBoxManage: add *restorecurrent* operation to snapshots command
- VBoxManage: fixed broken snapshot lookup by name (bug #6070
- GUI: fixed the broken “Reload” button that reloads the machine XML when a machine is inaccessible
- GUI: fixed guest full screen mode after reboot (bug #5372)

- GUI: handle Ctrl+Break properly on X11 hosts (bug #6122)
- GUI: fixed status LEDs for storage devices
- GUI: workaround for disabling the seamless mode on KDE hosts (KWin bug)
- 3D support: fixed SELinux warning saying VBoxOGL.so requires text relocation (bug #5690)
- 3D support: fixed Corrupted surface rendering (bug #5695)
- 3D support: free textures on guest application termination (bug #5206)
- 3D support: fixed *ubigraph_server* crashes (bug #4674)
- 3D support: fixes for 64-bit Solaris guests
- Seamless: disable seamless mode when guest changes screen resolution (bug #5655)
- NAT: fixed high CPU load under certain circumstances (Windows hosts only; bug #5787)
- NAT: fixed handling of the *broadcast* flag in DHCP requests
- NAT: fixed rare crash due to an assertion in the ICMP code (bug #3217)
- Virtio-net: don't crash when ports accessed beyond the valid range (bug #5923)
- LsiLogic: fix for Windows 7 guests
- ATA: fix for guru meditation when installing Solaris 8 guests (bug #5972)
- VHD: fixed an incompatibility with Virtual PC (bug #5990)
- VHD: update the footer backup after setting a new UUID (bug #5004)
- Host DVD: really fixed loading "passthrough" setting from configuration file (bug #5681)
- Shared Folders: fixed resolving of symlink target on Linux (3.1.2 regression)
- VRDP: fixed *VERR_NET_ADDRESS_IN_USE* error when restarting a VM (3.1 regression; bug #5902)
- VRDP: fixed crash on Mac OS X when 3D is enabled (3.1 regression)
- PulseAudio: fixed recording (bug #4302)
- USB: fixed a shutdown blue screen (Windows hosts only; bug #5885)
- BIOS: fixed attribute during text scroll (bug #3407)
- OVF: fix strange error messages on disk import errors
- OVF: do not require write access to the .ovf file during import (3.1 regression; bug #5762)
- iSCSI: fix taking snapshots of a running VM (bug #5849)
- Solaris hosts: several USB fixes (including support for Apple iPod; bug #5873)
- Solaris installer: fixed USB module removal and Solaris 10 "id" binary incompatibility
- Guest Additions: fixed wrong guest time adjustment if the guest clock is ahead (3.1 regression; non-Windows guests only)
- Linux Additions: fixed shared folders for Linux 2.6.32 guests (bug #5891)

- Linux Additions: make the mouse driver work on Debian 5.0.3 guests again (3.1.2 regression, bug #5832)
- Windows Additions: fixed malfunctioning VBoxService that broke time-sync (bug #5872)
- Windows Additions: fixed uninstallation issues on 64-bit guests
- Windows Additions: fixed some sysprep execution issues
- X.Org Additions: never reject the saved video mode as invalid (bug #5731)
- XFree86 Additions: accept video mode hints for the initial mode again

15.36 Version 3.1.2 (2009-12-17)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed SMP stability regression
- USB: fixed USB related host crashes on 64 bits Windows hosts (bug #5237)
- Main: wrong default HWVirtExExclusive value for new VMs (bug #5664)
- Main: DVD passthrough setting was lost (bug #5681)
- VBoxManage: iSCSI disks do not support adding a comment (bug #4460)
- VBoxManage: added missing *–cpus* and *–memory* options to OVF *–import*
- GUI: fixed VBox URL in update dialog for German and Dutch languages
- GUI: NLS updates
- OVF: fixed export of non standard storage controller names (bug #5643)
- Solaris hosts: several USB fixes (including support for Apple iPhone)
- Mac OS X hosts: several fixes for the 3D support
- Mac OS X hosts: re-enabled CMD+Key combinations, even if the Host-Key isn't CMD (bug #5684)
- Mac OS X hosts: fixed to fast scrolling if the mouse wheel is used inside the guest (bug #5672)
- Mac OS X hosts: dock & menubar don't disappear in full screen when the VM is not running on the primary display (bug #1762)
- Mac OS X hosts: added an option for enabling "Auto show Dock & Menubar in full screen" (bug #5636)
- Windows host installer: fixed starting VBox with wrong privileges right after installation (bug #4162)
- Host interface and host-only networking: prevent driver from unloading while a VM is still active (Windows host only)
- Host-only networking: fixed host-only interface creation (Windows host only) (bug #5708)
- Virtio-net: don't crash without an attached network
- Virtio-net: fixed the issue with intermittent network in VM with several virtual CPU cores

- NAT: fixed port-forwarding regressions (bug #5666)
- NAT: fixed crash under certain conditions (bug #5427)
- NAT: fixed resolving of names containing a slash or underscore when using the host resolver DNS proxy (bug #5698)
- ATA: fixed sporadic crash when resuming after a VM was forcefully paused (e.g. due to iSCSI target being unavailable)
- SATA: fixed raw vmdk disks (bug #5724)
- Linux guests: increased the default memory for Redhat and Fedora guests
- Linux Additions: fixed installation on RHEL 3.9 guests and on some 64bit guests
- Linux Additions: prevent SELinux warnings concerning text relocations in VBoxOGL.so (bug #5690)
- X11 guests: fixed mouse support for some Xorg 1.4 guests (openSUSE 11.0)
- X11 guests: fixed xorg.conf modification for some older Xorg releases (openSUSE 11.1)
- Windows guests: fixed some VBoxService shutdown issues
- Windows guests: fixed VBoxVideo spinlock issues on NT4
- Windows Guest Additions: fixed uninstallation issues of NT4
- Shared Folders: fixed resolving of symlink target (bug #5631)
- 2D Video acceleration: delay loading of OpenGL dlls for Windows hosts to avoid GUI crashes on misconfigured systems
- 2D Video acceleration: fixed issues with video picture not displayed on playback

15.37 Version 3.1.0 (2009-11-30)

This version is a major update. The following major new features were added:

- Teleportation (aka live migration); migrate a live VM session from one host to another (see chapitre 7.2, *Teleporting*, page 108)
- VM states can now be restored from arbitrary snapshots instead of only the last one, and new snapshots can be taken from other snapshots as well (“branched snapshots”; see chapitre 1.8, *Instantanés*, page 24)
- 2D video acceleration for Windows guests; use the host video hardware for overlay stretching and color conversion (see chapitre 4.5.2, *Accélération vidéo 2D pour Windows*, page 73)
- More flexible storage attachments: CD/DVD drives can be attached to arbitrary storage controllers, and there can be more than one such drive (chapitre 5, *Stockage virtuel*, page 78)
- The network attachment type can be changed while a VM is running
- Complete rewrite of experimental USB support for OpenSolaris hosts making use of the latest USB enhancements in Solaris Nevada 124 and higher

- Significant performance improvements for PAE and AMD64 guests (VT-x and AMD-V only; normal (non-nested) paging)
- Experimental support for EFI (Extensible Firmware Interface; see chapitre 3.12, *Firmware alternatif (EFI)*, page 56)
- Support for paravirtualized network adapters (virtio-net; see chapitre 6.1, *Matériel de réseau virtuel*, page 91)

In addition, the following items were fixed and/or added:

- VMM: guest SMP fixes for certain rare cases
- GUI: snapshots include a screenshot
- GUI: locked storage media can be unmounted by force
- GUI: the log window grabbed all key events from other GUI windows (bug #5291)
- GUI: allow to disable USB filters (bug #5426)
- GUI: improved memory slider in the VM settings
- 3D support: major performance improvement in VBO processing
- 3D support: added *GL_EXT_framebuffer_object*, *GL_EXT_compiled_vertex_array* support
- 3D support: fixed crashes in FarCry, SecondLife, Call of Duty, Unreal Tournament, Eve Online (bugs #2801, #2791)
- 3D support: fixed graphics corruption in World of Warcraft (bug #2816)
- 3D support: fixed Final frame of Compiz animation not updated to the screen (bug #4653)
- 3D support: fixed incorrect rendering of non ARGB textures under compiz
- iSCSI: support iSCSI targets with more than 2TiB capacity
- VRDP: fixed occasional VRDP server crash (bug #5424)
- Network: fixed the E1000 emulation for QNX (and probably other) guests (bug #3206)
- NAT: added host resolver DNS proxy (see chapitre 9.11.6, *Using the host's resolver as a DNS proxy in NAT mode*, page 163)
- VMDK: fixed incorrectly rejected big images split into 2G pieces (bug #5523, #2787)
- VMDK: fixed compatibility issue with fixed or raw disk VMDK files (bug #2723)
- VHD: fixed incompatibility with Hyper-V
- Support for Parallels version 2 disk image (HDD) files; see chapitre 5.2, *Fichiers images de disque (VDI, VMDK, VHD, HDD)*, page 81
- OVF: create manifest files on export and verify the content of an optional manifest file on import
- OVF: fixed memory setting during import (bug #4188)
- Mouse device: now five buttons are passed to the guest (bug #3773)
- VBoxHeadless: fixed loss of saved state when VM fails to start

- VBoxSDL: fixed crash during shutdown (Windows hosts only)
- X11 based hosts: allow the user to specify their own scan code layout (bug #2302)
- Mac OS X hosts: don't auto show the menu and dock in full screen (bug #4866)
- Mac OS X hosts (64 bit): don't interpret mouse wheel events as left click (bug #5049)
- Mac OS X hosts: fixed a VM abort during shutdown under certain conditions
- Solaris hosts: combined the kernel interface package into the VirtualBox main package
- Solaris hosts: support for OpenSolaris Boomer architecture (with OSS audio backend)
- Shared Folders: VBOXSVR is visible in Network folder (Windows guests, bug #4842)
- Shared Folders: performance improvements (Windows guests, bug #1728)
- Windows, Linux and Solaris Additions: added balloon tip notifier if VirtualBox host version was updated and Additions are out of date
- Solaris guests: fixed keyboard emulation (bug #1589)
- Solaris Additions: fixed *as_pagelock()* failed errors affecting guest properties (bug #5337)
- Windows Additions: added automatic logon support for Windows Vista and Windows 7
- Windows Additions: improved file version lookup for guest OS information
- Windows Additions: fixed runtime OS detection on Windows 7 for session information
- Windows Additions: fixed crash in seamless mode (contributed by Huihong Luo)
- Linux Additions: added support for uninstalling the Linux Guest Additions (bug #4039)
- Linux guest shared folders: allow mounting a shared folder if a file of the same name as the folder exists in the current directory (bug #928)
- SDK: added object-oriented web service bindings for PHP5

15.38 Version 3.0.12 (2009-11-10)

This is a maintenance release. The following items were fixed and/or added:

- VMM: reduced IO-APIC overhead for 32 bits Windows NT/2000/XP/2003 guests; requires 64 bits support (VT-x only; bug #4392)
- VMM: fixed double timer interrupt delivery on old Linux kernels using IO-APIC (caused guest time to run at double speed; bug #3135)
- VMM: re-initizalize VT-x and AMD-V after host suspend or hibernate; some BIOSes forget this (Windows hosts only; bug #5421)
- VMM: fixed loading of saved state when RAM preallocation is enabled
- BIOS: ignore unknown shutdown codes instead of causing a guru meditation (bug #5389)
- GUI: never start a VM on a single click into the selector window (bug #2676)
- Serial: reduce the probability of lost bytes if the host end is connected to a raw file
- VMDK: fixed handling of split image variants and fix a 3.0.10 regression (bug #5355)

- VRDP: fixed occasional VRDP server crash
- Network: even if the virtual network cable was disconnected, some guests were able to send / receive packets (E1000; bug #5366)
- Network: even if the virtual network cable was disconnected, the PCNet card received some spurious packets which might confuse the guest (bug #4496)
- Shared Folders: fixed changing case of file names (bug #2520)
- Windows Additions: fixed crash in seamless mode (contributed by Huihong Luo)
- Linux Additions: fixed writing to files opened in *O_APPEND* mode (bug #3805)
- Solaris Additions: fixed regression in Guest Additions driver which among other things caused lost guest property updates and periodic error messages being written to the system log

15.39 Version 3.0.10 (2009-10-29)

This is a maintenance release. The following items were fixed and/or added:

- VMM: guest SMP stability fixes
- VMM: fixed guru meditation with nested paging and SMP guests (bug #5222)
- VMM: changed VT-x/AMD-V usage to detect other active hypervisors; necessary for e.g. Windows 7 XP compatibility mode (Windows & Mac OS X hosts only; bug #4239)
- VMM: guru meditation during SCO OpenServer installation and reboot (VT-x only; bug #5164)
- VMM: fixed accessed bit handling in certain cases (bug #5248)
- VMM: fixed VPID flushing (VT-x only)
- VMM: fixed broken nested paging for 64 bits guests on 32 bits hosts (AMD-V only; bug #5285)
- VMM: fixed loading of old saved states/snapshots (bug #3984)
- Mac OS X hosts: fixed memory leaks (bug #5084)
- Mac OS X hosts (Snow Leopard): fixed redraw problem in a dual screen setup (bug #4942)
- Windows hosts: installer updates for Windows 7
- Solaris hosts: out of memory handled incorrectly (bug #5241)
- Solaris hosts: the previous fix for #5077 broke the DVD host support on Solaris 10 (VBox 3.0.8 regression)
- Linux hosts: fixed module compilation against Linux 2.6.32rc4 and later
- Guest Additions: fixed possible guest OS kernel memory exhaustion
- Guest Additions: fixed stability issues with SMP guests
- Windows Additions: fixed color depth issue with low resolution hosts, netbooks, etc. (bug #4935)

- Windows Additions: fixed NO_MORE_FILES error when saving to shared folders (bug #4106)
- Windows Additions: fixed subdirectory creation on shared folders (bug #4299)
- Linux Additions: *sendfile()* returned *-EOVERFLOW* when executed on a shared folder (bug #2921)
- Linux Additions: fixed incorrect disk usage value (non-Windows hosts only)
- Linux installer: register the module sources at DKMS even if the package provides proper modules for the current running kernel
- 3D support: removed invalid OpenGL assertion (bug #5158)
- Network: fixed the Am79C973 PCNet emulation for QNX (and probably other) guests (bug #3206)
- VMDK: fix handling of split image variants
- VHD: do not delay updating the footer when expanding the image to prevent image inconsistency
- USB: stability fix for some USB 2.0 devices
- GUI: added a search index to the .chm help file
- GUI/Windows hosts: fixed CapsLock handling on French keyboards (bug #2025)
- Shared clipboard/X11 hosts: fixed a crash when clipboard initialisation failed (bug #4987)

15.40 Version 3.0.8 (2009-10-02)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed 64 bits guest on 32 bits host regression in 3.0.6 (VT-x only; bug #4947)
- VMM: fixed a recompiler triple fault guru meditation (VT-x & AMD-V only; bug #5058)
- VMM: fixed hang after guest state restore (AMD-V, 32 bits Windows guest and IO-APIC enabled only; bug #5059)
- VMM: fixed paging issue with OS/2 guests
- VMM: fixed guru meditation in rare cases (2.0 regression; software virtualization only)
- VMM: fixed release assertion during state restore when using the Sound Blaster 16 emulation (bug #5042)
- Security: fixed vulnerability that allowed to execute commands with root privileges
- Linux hosts: fixed runtime assertion in semaphore implementation which was triggered under certain conditions (bug #616)
- Linux hosts: change the default USB access mode on certain distributions (bugs #3394 and #4291)
- Linux hosts: on hardened Gentoo, the VBoxSVC daemon crashed by opening the VM network settings (bug #3732)

- Linux hosts, Solaris hosts: pass the XAUTHORITY variable along the DISPLAY variable when starting a VM from VBoxManage or from the VM selector (bug #5063)
- Linux hosts: use sysfs to enumerate host drives if hal is not available
- Solaris hosts: fixed a bug which would hang the host sporadically as interrupts were not re-enabled every time
- Solaris hosts: fixed a kernel panic with bridged and host-only networking (bug #4775)
- Solaris hosts: fixed incorrectly persistent CD/DVD-ROMs when changing them (bug #5077)
- X11-based hosts: support additional function keys on Sun keyboards (bug #4907)
- Mac OS X hosts (Snow Leopard): fixed problem starting headless VMs without a graphical session (bug #5002)
- Mac OS X hosts: fixed problem listing host-only adapter names with trailing garbage (attached VMs won't start)
- Windows Additions: now work with Vista 64-bit Home editions (bug #3865)
- Windows Additions: fixed screen corruption with ZoomText Magnifier
- Windows Additions: fixed NPGetUniversalName failure (bug #4853)
- Windows Additions: fixed Windows NT regression (bug #4946)
- Windows Additions: fixed VBoxService not running if no Shared Folders are installed
- Linux Additions: implemented *ftrunctate* (bug #4771)
- VRDP: start VM even if configured VRDP port is in use
- Networking: the PCnet network device stopped receiving under rare conditions (bug #4870)
- VBoxManage: implemented `controlvm vrdpport` command
- iSCSI: fixed issue with NetApp targets (bug #5072)
- SCSI: add support for virtual disks larger than 2TB
- USB: fixed potential crash when unplugging USB2 devices (bug #5089)
- NAT: IPSEC did not properly work with Linux guests (bug #4801)

15.41 Version 3.0.6 (2009-09-09)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed IO-APIC overhead for 32 bits Windows NT, 2000, XP and 2003 guests (AMD-V only; bug #4392)
- VMM: fixed a Guru meditation under certain circumstances when enabling a disabled device (bug #4510)
- VMM: fixed a Guru meditation when booting certain Arch Linux guests (software virtualization only; bug #2149)

- VMM: fixed hangs with 64 bits Solaris & OpenSolaris guests (bug #2258)
- VMM: fixed decreasing *rdtsc* values (AMD-V & VT-x only; bug #2869)
- VMM: small Solaris/OpenSolaris performance improvements (VT-x only)
- VMM: *cpuid* change to correct reported virtual CPU ID in Linux
- VMM: NetBSD 5.0.1 CD hangs during boot (VT-x only; bug #3947)
- Solaris hosts: worked around an issue that caused the host to hang (bug #4486)
- Solaris hosts: fixed a rare host system deadlock when using bridged networking
- Solaris hosts: fixed a potential host system deadlock when CPUs were onlined or offlined
- Solaris hosts installer: added missing dependency for UTF-8 package (bug #4899)
- Linux hosts: don't crash on Linux PAE kernels < 2.6.11 (in particular RHEL/CentOS 4); disable VT-x on Linux kernels < 2.6.13 (bug #1842)
- Linux/Solaris hosts: correctly detect keyboards with fewer keys than usual (bug #4799)
- Mac OS X hosts: prevent password dialogs in 32 bits Snow Leopard
- Python WS: fixed issue with certain enumerations constants having wrong values in Python web services bindings
- Python API: several threading and platform issues fixed
- Python shell: added *exportVM* command
- Python shell: various improvements and bugfixes
- Python shell: corrected detection of home directory in remote case
- OVF: fixed XML comment handling that could lead to parser errors
- Main: fixed a rare parsing problem with port numbers of USB device filters in machine settings XML
- Main: restrict guest RAM size to 1.5 GB (32 bits Windows hosts only)
- Main: fixed possible hang during guest reboot (bug #3792)
- GUI: fixed rare crash when removing the last disk from the media manager (bug #4795)
- VBoxManage: fixed `guestproperty` for Mac OS X hosts (bug #3806)
- VBoxManage: fixed setting guest properties with *–flags* or *-flags*
- Web service: fixed a severe memory leak, at least on platforms using XPCOM
- Serial: fixed host mode (Solaris, Linux and Mac OS X hosts; bug #4672)
- VRDP: Remote USB Protocol version 3
- SATA: fixed hangs and BSODs introduced with 3.0.4 (bugs #4695, #4739, #4710)
- SATA: fixed a bug which prevented Windows 7 from detecting more than one hard disk
- SATA/SCSI: fixed rare random guest crashes and hangs
- SCSI: fixed problem with Fedora 11 refusing to boot after kernel update

- iSCSI: fix logging out when the target has dropped the connection, fix negotiation of parameters, fix command resend when the connection was dropped, fix processing SCSI status for targets which do not use phase collapse
- BIOS: fixed a bug that caused the OS/2 boot manager to fail (2.1.0 regression, bug #3911)
- PulseAudio: don't hang during VM termination if the connection to the server was unexpectedly terminated (bug #3100)
- Mouse: fixed weird mouse behaviour with SMP (Solaris) guests (bug #4538)
- HostOnly Network: fixed failure in *CreateHostOnlyNetworkInterface()* on Linux (no GUID)
- HostOnly Network: fixed wrong DHCP server startup while hostonly interface bringup on Linux
- HostOnly Network: fixed incorrect factory and default MAC address on Solaris
- HostOnly Network: fixed the problem with listing host-only interfaces on Mac OS X when all physical interfaces are down (bugs #4698, #4790)
- DHCP: fixed a bug in the DHCP server where it allocated one IP address less than the configured range
- E1000: fixed receiving of multicast packets
- E1000: fixed up/down link notification after resuming a VM
- NAT: fixed Ethernet address corruptions (bug #4839)
- NAT: fixed hangs, dropped packets and retransmission problems (bug #4343)
- Bridged networking: fixed packet queue issue which might cause DRIVER_POWER_STATE_FAILURE BSOD for Windows hosts (bug #4821)
- Windows Additions: fixed a bug in VBoxGINA which prevented selecting the right domain when logging in the first time
- Windows host installer: should now also work on unicode systems (like Korean, bug #3707)
- Windows host installer: check for sufficient disk space
- Shared clipboard: do not send zero-terminated text to X11 guests and hosts (bug #4712)
- Shared clipboard: use a less CPU intensive way of checking for new data on X11 guests and hosts (bug #4092)
- Guest Additions: do not hide the host mouse cursor when restoring a saved state (bug #4700)
- Windows guests: fixed issues with the display of the mouse cursor image (bugs #2603, #2660 and #4817)
- SUSE 11 guests: fixed Guest Additions installation (bug #4506)
- Guest Additions: support Fedora 12 Alpha guests (bugs #4731, #4733 and #4734)

15.42 Version 3.0.4 (2009-08-04)

This is a maintenance release. The following items were fixed and/or added:

- VMM: 64 bits guest stability fixes (AMD-V only; bugs #3923 & #3666)
- VMM: SMP stability fixes (AMD-V only)
- VMM: SMP performance improvement (esp. for Solaris guests)
- VMM: eliminated several bugs which could lead to a host reboot
- VMM: fixed OS/2 ACP2 boot floppy hang (VT-x only)
- VMM: small performance improvement for OpenSolaris guests (AMD-V only)
- VMM: fixed CentOS/Xen reboot (software virtualization only; bug #4509)
- SATA: fixed hangs / BSOD during Windows XP installation (bug #4342)
- SATA: mark the ports as non hotpluggable (bug #3920)
- 3D support: fix deadlocks and context/window tracking for multithreaded applications (bug #3922)
- 3D support: fix memory leaks when terminating OpenGL guest applications
- 3D support: fix crash in Call of Duty
- NAT: using two or more NAT adapters in one VM was broken (3.0.0 regression)
- NAT: fixed network communication corruptions (bugs #4499, #4540, #4591, #4604)
- NAT: fixed passive ftp access to host server (bug #4427)
- iSCSI: fixed cloning to/from iSCSI disks
- GUI: fixed path separator handling for the OVF export on Windows (bug #4354)
- GUI: the mini toolbar was only shown on the first host display (bug #4654)
- GUI: added a VM option to display the mini toolbar on top
- GUI: don't crash when adding plus configuring host-only network interfaces
- Shared Folders: fixed selection of a drive root directory as a shared folder host path in VirtualBox (Windows host only)
- USB: fixed a bug that may have rendered USB device filter settings inactive (3.0.2 regression, bug #4668)
- Guest Additions: report the Guest Additions version to the guest properties (bug #3415)
- Mac OS X hosts: fix creation of VMDK files giving raw partition access (bug #1461)
- Mac OS X hosts: improved support for Snow Leopard
- Linux hosts: fixed problems leading to wrong colors or transparency in host windows with some graphics drivers (bug #3095)
- Linux hosts: hardware detection fallbacks if the hal service fails to find any DVD drives
- Linux and Solaris hosts: Work around color handling problems in Qt (bug #4353)

- Solaris hosts: fixed memory leaks in host-only networking
- Solaris Installer: fixed incorrect netmask for Host-only interface (bug #4590)
- Solaris Installer: added package dependency for Python and Python-devel (bug #4570)
- X11 guests: prevent windows from being skipped in seamless mode KDE guests (bugs #1681 and #3574)
- X11 guests: fixed screen corruption in X11 guests when large amounts of video RAM were allocated (bug #4430)
- X11 guests: some fixes when switching between host and guest-drawn mouse pointers
- X11 guests: fixed an issue which caused seamless mode to stop working as it should (the main issue listed in bug #2238)

15.43 Version 3.0.2 (2009-07-10)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed network regressions (guest hangs during network IO) (bug #4343)
- VMM: guest SMP performance improvements
- VMM: fixed hangs and poor performance with Kaspersky Internet Security (VT-x/AMD-V only; bug #1778)
- VMM: fixed crashes when executing certain Linux guests (software virtualization only; bugs #2696 & #3868)
- ACPI: fixed Windows 2000 kernel hangs with IO-APIC enabled (bug #4348)
- APIC: fixed high idle load for certain Linux guests (3.0 regression)
- BIOS: properly handle Ctrl-Alt-Del in real mode
- iSCSI: fixed configuration parsing (bug #4236)
- OVF: fix potential confusion when exporting networks
- OVF: compatibility fix (bug #4452)
- OVF: accept ovf:/disk/ specifiers with a single slash in addition to ovf://disk/ (bug #4452)
- NAT: fixed crashes under certain circumstances (bug #4330)
- 3D support: fixed dynamic linking on Solaris/OpenSolaris guests (bug #4399)
- 3D support: fixed incorrect context/window tracking for multithreaded apps
- Shared Folders: fixed loading from saved state (bug #1595)
- Shared Folders: host file permissions set to 0400 with Windows guest (bug #4381)
- X11 host and guest clipboard: fixed a number of issues, including bug #4380 and #4344
- X11 Additions: fixed some issues with seamless windows in X11 guests (bug #3727)
- Windows Additions: added VBoxServiceNT for NT4 guests (for time synchronization and guest properties)

- Windows Additions: fixed version lookup
- Linux Installer: support Pardus Linux
- Linux hosts: workaround for buggy graphics drivers showing a black VM window on recent distributions (bug #4335)
- Linux hosts: fixed typo in kernel module startup script (bug #4388)
- Solaris hosts: several installer fixes
- Solaris hosts: fixed a preemption issue causing VMs to never start on Solaris 10 (bug #4328)
- Solaris guests: fixed mouse integration for OpenSolaris 2009.06 (bug #4365)
- Windows hosts: fixed high CPU usage after resuming the host (bug #2978)
- Fixed a settings file conversion bug which sometimes caused hardware acceleration to be enabled for virtual machines that had no explicit configuration in the XML

15.44 Version 3.0.0 (2009-06-30)

This version is a major update. The following major new features were added:

- Guest SMP with up to 32 virtual CPUs (VT-x and AMD-V only; see chapitre 3.4.2, *Onglet processeur*, page 49)
- Windows guests: ability to use Direct3D 8/9 applications / games (experimental; see chapitre 4.5.1, *Accélération 3D matérielle (OpenGL et Direct3D 8/9)*, page 72)
- Support for OpenGL 2.0 for Windows, Linux and Solaris guests

In addition, the following items were fixed and/or added:

- Solaris hosts: allow suspend/resume on the host when a VM is running (bug #3826)
- Solaris hosts: loosen the restriction for contiguous physical memory under certain conditions
- Mac OS X hosts: fixed guest PAE
- Linux hosts: kernel module compile fixes for 2.6.31 (bug #4264)
- VMM: fixed occasional guru meditation when loading a saved state (VT-x only)
- VMM: eliminated IO-APIC overhead with 32 bits guests (VT-x only, some Intel CPUs don't support this feature (most do); bug #638)
- VMM: fixed 64 bits CentOS guest hangs during early boot (AMD-V only; bug #3927)
- VMM: performance improvements for certain PAE guests (e.g. Linux 2.6.29+ kernels)
- VMM: some Windows guests detected a completely wrong CPU frequency (bug #2227)
- VMM: fixed hanging and unkillable VM processes (bug #4040)
- VMM: fixed random infrequent guest crashes due XMM state corruption (Win64 hosts only)
- VMM: performance improvements for network I/O (VT-x/AMD-V only)

- GUI: added mini toolbar for full screen and seamless mode (Thanks to Huihong Luo)
- GUI: redesigned settings dialogs
- GUI: allow to create/remove more than one host-only network adapters (non Windows hosts)
- GUI: display estimated time for long running operations (e.g. OVF import/export)
- GUI: fixed rare hangs when open the OVF import/export wizards (bug #4157)
- 3D support: fixed VM crashes for client applications using incorrect OpenGL states
- 3D support: fixed memory corruption when querying for supported texture compression formats
- 3D support: fixed incorrect rendering of glDrawRangeElements
- 3D support: fixed memory leak when using VBOs
- 3D support: fixed glew library detection
- 3D support: fixed random textures corruption
- VRDP: support Windows 7 RDP client
- Networking: fixed another problem with TX checksum offloading with Linux kernels up to version 2.6.18
- NAT: fixed “open ports on virtual router 10.0.2.2 - 513, 514” (forum)
- NAT: allow to configure socket and internal parameters
- NAT: allow to bind sockets to specific interface
- PXE boot: significant performance increase (VT-x/AMD-V only)
- VHD: properly write empty sectors when cloning of VHD images (bug #4080)
- VHD: fixed crash when discarding snapshots of a VHD image
- VHD: fixed access beyond the block bitmap which could lead to arbitrary crashes
- VBoxManage: fixed incorrect partition table processing when creating VMDK files giving raw partition access (bug #3510)
- VBoxManage: support cloning to existing image file
- OVF: several OVF 1.0 compatibility fixes
- OVF: fixed exporting of disk images when multiple virtual machines are exported at once
- Virtual mouse device: eliminated micro-movements of the virtual mouse which were confusing some applications (bug #3782)
- Shared Folders: sometimes a file was created using the wrong permissions (2.2.0 regression; bug #3785)
- Shared Folders: allow to change file attributes from Linux guests and use the correct file mode when creating files
- Shared Folders: some content was incorrectly written under certain conditions (bug #1187)

- Shared Folders: fixed incorrect file timestamps, when using Windows guest on a Linux host (bug #3404)
- X11 clipboard: fix duplicate end of lines (bug #4270)
- X11 guests: a number of shared clipboard fixes
- Linux guests: Guest Additions support for SUSE Linux Enterprise Desktop 11
- Linux guests: new daemon vboxadd-service to handle time synchronization and guest property lookup
- Linux guests: implemented guest properties (OS info, logged in users, basic network information)
- Windows host installer: VirtualBox Python API can now be installed automatically (requires Python and Win32 Extensions installed)
- USB: Support for high-speed isochronous endpoints has been added. In addition, read-ahead buffering is performed for input endpoints (currently Linux hosts only). This should allow additional devices to work, notably webcams (bug #242)
- USB: fixed error handling for some USB dongles
- Web service: fixed inability to handle NULL pointers for object arguments, which are valid values for a lot of APIs, in both the raw and the object-oriented web service
- Web service: object-oriented bindings for JAX-WS did not exhibit interface inheritance correctly, fixed
- Web service: added support for IDisplay and IGuest interfaces, which were previously unavailable
- Registration dialog uses Sun Online accounts now

15.45 Version 2.2.4 (2009-05-29)

This is a maintenance release. The following items were fixed and/or added:

- Windows Installer: fixed a potential hang during installation
- Windows Installer: fixed several problems (bug #3892)
- Solaris hosts: make it work with Solaris build 114 or later (bug #3981)
- Solaris hosts: fixed a bug serial port character handling found during loopback (bug #3120)
- Linux hosts: adapted vboxdrv.sh to the latest changes in `VBoxManage list runningvms` (bug #4034)
- Windows hosts: fixed a crash caused by host-only/bridged networking
- Mac OS X hosts: fixed access to host DVD with passthrough disabled (bug #4077)
- Guest Additions: fixed problems with KDE 4 not recognizing mouse clicks
- Windows Additions: fixed incorrect 8-bit guest color depth in Windows 7 guests
- GUI: warn if VT-x/AMD-V could not be enabled for guests that require this setting (bug #4055)

- VMM: fixed occasional crash due to insufficient memory
- VMM: fixed hanging 64 bits Solaris guests
- VMM: restore from a saved state occasionally failed (bugs #3984 and #2742)
- Clipboard: fixed a deadlock while shutting down the shared clipboard on X11 hosts (bug #4020)
- OVF: fixed potential hang during import
- OVF: fixed potential crashes during import/export on Win64 hosts
- VBoxManage `modifyhd --compact`: fixed bug which could lead to crashes and image corruption (bug #3864)
- VBoxManage `metrics collect`: now flushes the output stream
- VHD: made `VBoxManage internalcommands sethduuid` work for .vhd files (bug #3443)
- VHD: some .vhd files could not be cloned (bug #4080)
- NAT: improvement of TCP connection establishment (bug #2987)
- NAT: fixed order of DNS servers in DHCP lease (bug #4091)
- NAT: fixed DHCP lease for multiple name servers (bug #3692)
- NAT: fixed a potential segfault if the host lost its connectivity (bug #3964)
- Shared Folders: deny access to parent directories on Windows hosts (bug #4090)
- Shared Folders: make rm/rmdir work with Solaris guests on Windows hosts
- Networking: fixed the problem with blocked receiving thread when a broadcast packet arrives too early to be handled by uninitialized e1000 adapter
- Networking: fixed the problem that caused host freezes/crashes when using bridged mode with host's interface having RX checksum offloading on (bug #3926 and related). Fixes problems with TX offloading as well (bug #3870)
- PXE boot: Added support for PRO/1000 MT Server adapter
- Python bindings: fixed keyword conflict
- SCSI: fixed occasional crashes on Win64
- Serial: allow to redirect the serial port to a raw file (bug #1023)
- VRDP: fixed a rare incorrect screen update
- VMDK: fixed creating snapshots

15.46 Version 2.2.2 (2009-04-27)

This is a maintenance release. The following items were fixed and/or added:

- Host and guest clipboard: fixed a number of issues affecting hosts and guests running the X window system
- Guest Additions: make sure the virtual mouse autodetection works on first reboot after installing the Additions on X.Org server 1.5 and later
- Guest Additions: properly report process identity number of running services
- Guest Additions: clean up properly if the X Window server terminates
- Linux Additions: fixed installation path for OpenGL libraries in some 64-bit guests (bug #3693)
- Solaris Additions: fixed installation to work when X.Org is not installed on the guest
- Solaris Additions: fixed a bug that could panic the guest when unmounting a busy shared folder
- Windows Additions: fixed mouse pointer integration of some Windows guests (2.2.0 regression, bug #3734)
- Windows Additions: fixed installation on Windows Server 2008 Core (bug #2628)
- Main: do not try to use older versions of D-Bus (Linux hosts only, bug #3732)
- VMM: fixed out-of-memory conditions on Windows hosts (bug #3657)
- VMM: fixed occasional hangs when attaching USB devices during VM startup (2.2.0 regression; bugs #3787)
- VMM: fixed guru meditation related to memory management (software virtualization only)
- Virtual disks: fix possible data corruption when writing to diff images, incorrect detection of redundant writes
- GUI: reworked network settings dialog
- GUI: properly show the detailed settings dialog of NAT networks (bug #3702)
- GUI: HostKey could not be changed (2.2.0 regression, bug #3689)
- GUI: fixed memory textfield size (Windows hosts only; bug #3679)
- GUI: fixed crash when selecting a shared folder path (Windows hosts only; bugs #3694, #3751, #3756)
- VBoxManage `modifyhd --compact`: implemented again for VDI files, and now supports relative paths (bug #2180, #2833)
- VBoxManage `snapshot discard`: made it work again (2.1.0 regression; bug #3714)
- NAT: on some Windows hosts, the guest didn't receive a DHCP lease (bug #3655)
- NAT: fixed release assertion during *poll()* (bug #3667)
- Networking: fixed a deadlock caused by the PCnet network device emulation (2.2.0 regression, bug #3676)

- Clipboard: fixed random crashes (X11 hosts only, bug #3723)
- Shared Folders: fixed incorrect permissions for Solaris guests
- Shared Folders: fixed wrong file sizes with Solaris guests
- CBindings: fixed possible memory leak while releasing the IVirtualBox and ISession Objects
- Solaris hosts: fixed host-only network interface incompatibility with nwam/dhcpagent (bug #3754)
- Windows installer: fixed several install and uninstall issues (bugs #3659, #3686, #1730, #3711, #3373, #3382, #3701, #3685, #3710)
- Mac OS X hosts: preliminary support for Snow Leopard

15.47 Version 2.2.0 (2009-04-08)

This version is a major update. The following major new features were added:

- OVF (Open Virtualization Format) appliance import and export (see chapitre 1.12, *Importer et exporter des machines virtuelles*, page 29)
- Host-only networking mode (see chapitre 6.6, *Réseau privé avec l'hôte (Host-only)*, page 97)
- Hypervisor optimizations with significant performance gains for high context switching rates
- Raised the memory limit for VMs on 64-bit hosts to 16GB
- VT-x/AMD-V are enabled by default for newly created virtual machines
- USB (OHCI & EHCI) is enabled by default for newly created virtual machines (Qt GUI only)
- Experimental USB support for OpenSolaris hosts
- Shared Folders for Solaris and OpenSolaris guests
- OpenGL 3D acceleration for Linux and Solaris guests (see chapitre 4.5.1, *Accélération 3D matérielle (OpenGL et Direct3D 8/9)*, page 72)
- Added C API in addition to C++, Java, Python and Web Services

In addition, the following items were fixed and/or added:

- VMM: FreeBSD guest related fix for V86 flags (bug #2342)
- VMM: fixed guru meditation when booting an AsteriskNow Linux VM (bug #2342)
- VMM: fixed PGMPOOLKIND_FREE guru meditation (bugs #3356, #3431)
- VMM: fixed Windows XP boot hang (guest PAE + nested paging only)
- VMM: allow mixing of VT-x/AMD-V and software virtualization
- VMM: fixed extremely slow safe mode booting in e.g. Windows 2008 (VT-x/AMD-V only)
- VMM: significant speedup of certain GRUB boot loaders (e.g. Solaris) (VT-x/AMD-V only)
- VMM: real-mode IOPL fix for DOS guests (VT-x only)

- VMM: fixed VT-x detection with certain BIOSes that enable VT-x, but don't set the lock bit in MSR_IA32_FEATURE_CONTROL
- VMM: fixed hibernation issues on Windows XP hosts (VT-x only; bug #1794)
- VMM: properly emulate *RDMSR* from the TSC MSR, should fix some NetBSD guests
- VMM: emulate *RDPMC*; fixes Windows guests crashes when using the Kaspersky virus scanner (bug #1778)
- NAT: fixed truncated downloads (FTP) (bug #3257)
- NAT: blocked UDP packets caused a crash (bug #3426)
- NAT: allow to configure the *next server* and the *boot file* via VBoxManage (bug #2759)
- IDE: fixed hard disk upgrade from XML-1.2 settings (bug #1518)
- Hard disk: support more VMDK file variants (including fixed-size ESX server images)
- Hard disks: refuse to start the VM if a disk image is not writable
- USB: further reduced host CPU utilization for OHCI and EHCI; the "VBoxInternal/Devices/usb-ohci/0/Config/FrameRate" CFG key is no longer necessary and no longer supported
- USB: fixed BSOD on the host with certain USB devices (Windows hosts only; bug #1654)
- E1000: properly handle cable disconnects (bug #3421)
- VRDP: fixed hangs when VRDP server is enabled or disabled in runtime
- Shared Folders: respect umask settings on Linux, OSX and Solaris hosts when creating files
- X11 guests: prevented setting the locale in vboxmouse, as this caused problems with Turkish locales (bug #3563)
- X11 guests: show the guest mouse pointer at the right position if the virtual desktop is larger than the guest resolution (bug #2306)
- Linux Additions: fixed typo when detecting Xorg 1.6 (bug #3555)
- Solaris guests: added xpg4/xcu4 dependency to the Guest Additions installer (bug #3524)
- Windows guests: bind the VBoxMouse.sys filter driver to the correct guest pointing device (bug #1324)
- Windows hosts: fixed BSOD when starting a VM with enabled host interface (bug #3414)
- Linux hosts: do proper reference counting to prevent unloading the vboxnetflt module as long as this code is in use (bug #3104)
- Linux hosts: do not leave zombies of VBoxSysInfo.sh (bug #3586)
- Linux installers: fixes for Slackware, Arch Linux and Linux from Scratch systems
- Windows installers: combined installer executable which contains both (32- and 64-bit) architectures
- VBoxManage: less cryptic command-line error messages
- VBoxManage `list vms` commands now default to compact format
- VBoxManage `controlvm dvdattach` did not work if the image was attached before

- VBoxManage: allow creation of all supported disk image variants
- VBoxManage `showvminfo`: don't spam the release log if the Guest Additions don't support statistics information (bug #3457)
- VBoxManage: big command line processing cleanup, the legacy single-dash options are deprecated and will be removed in the next major release, so switch to the new options now
- Hard disks: improved immutable disk support to auto-reset diff file at VM startup (related to bug #2772)
- GUI: enable the audio adapter by default for new VMs
- GUI: warn if VT-x/AMD-V is not operational when starting a 64-bit guest
- GUI: deactivate 64-bit guest support when the host CPU does not support VT-x/AMD-V
- GUI: removed floppy icon from the status bar
- GUI: show build revision in about dialog
- GUI: fixed sticky status bar text
- GUI: improved error dialogs
- GUI: fail with an appropriate error message when trying to boot a read-only disk image (bug #1745)
- GUI/Mac OS X: fixed disabled close button
- GUI/Windows: re-enabled support for copy and paste (Windows hosts 2.0 regression; bug #2065)
- 3D support: added OpenGL select/feedback support (bug #2920)
- 3D support: close OpenGL subsystem for terminated guest applications (bug #3243)
- 3D support: fixed VM hangs when starting guests with 3D acceleration enabled (bug #3437)
- PXE: fixed boot hangs when hardware virtualization is used (bug #2536)
- LsiLogic: fixed problems with Solaris guests
- Main API: close machine settings XML file when unregistering machine (bug #3548)

15.48 Version 2.1.4 (2009-02-16)

This is a maintenance release. The following items were fixed and/or added:

- Windows hosts: fixed host crashes/hangs on certain 32 bits Windows systems when running Linux guests (bugs #1606, #2269, #2763)
- Windows hosts: fixed network component BSOD issue (bugs #3168, #2916)
- Windows hosts: fixed installation issues (bugs #2517, #1730, #3130)
- Linux hosts: fixed occasional kernel oopses (bug #2556)
- Linux hosts: fixed module dependency for shipped modules (bug #3115)

- Linux hosts: moved the udev rules for USB forward so that they don't override existing system rules (bug #3143)
- Linux hosts: fixed the issue with guest not being able to communicate with each other when attached via TAP interfaces (bug #3215)
- Linux hosts: give up probing for USB gracefully if DBus or hal are not available (bug #3136)
- Linux hosts: fixed warnings in installer when SELinux was disabled (bug #3098)
- Linux hosts: VirtualBox sometimes failed to start if it had been started using sudo previously (bug #3270)
- Solaris hosts: fixed high CPU load while running many guests in parallel
- Solaris hosts: fixed inability to start more than 128 VMs
- VMM: fixed performance regression for Windows guests (bug #3172)
- VMM: ignore CPU stepping when restoring a saved state/snapshot
- REM: fixed inability to use gdb to debug programs in Linux guests with software virtualization (bug #3245)
- GUI: fixed dead key handling on Solaris hosts (bug #3256)
- GUI: in the shutdown dialog, disable the action *send the shutdown signal* if the guest is currently not using ACPI
- GUI: suppress additional key release events sent by X11 hosts when keys are auto-repeated (bug #1296)
- API: restore case insensitive OS type name lookup (bug #3087)
- VBoxHeadless: really don't start X11 services (clipboard service, 3D acceleration; Solaris & Darwin hosts only; bug #3199)
- NAT: fixed occasional crashes when the guest is doing traceroute (non-Windows hosts; bug #3200)
- NAT: fixed crashes under high load (bug #3110)
- NAT: fixed truncated downloads (Windows hosts only, bug #3257)
- NAT: don't intercept TFTP packages with a destination address different from the builtin TFTP server (bug #3112)
- USB: several fixes for USB passthrough on Linux hosts
- USB: reduced host CPU utilization if EHCI is active
- VRDP: fixed VRDP server black screen after a client reconnect (bug #1989)
- VRDP: modified rdesktop client (rdesktop-vrdp) now uses NumLock state synchronization (bug #3253)
- LsiLogic: make FreeBSD guests work (bug #3174)
- ATA: fixed deadlock when pausing VM due to problems with the virtual disk (e.g. disk full, iSCSI target unavailable)
- iSCSI: fixed possible crash when pausing the VM

- 3D support: added missing GL_MAX_TEXTURE_COORDS_ARB (bug #3246)
- Windows Additions: fixed *ERROR (e0000101)* error during installation (bug #1923)
- Windows Additions: fixed Windows Explorer hang when browsing shared folders with 64 bit guests (bug #2225)
- Windows Additions: fixed guest screen distortions during a video mode change
- Windows Additions: fixed the *Network drive not connected* message for mapped shared folders drives after the guest startup (bug #3157)
- Linux Additions: fixed occasional file corruption when writing files in *O_APPEND* mode to a shared folder (bug #2844)
- Linux Additions: the mouse driver was not properly set up on X.Org release candidates (bug #3212)
- Linux Additions: fixed installer to work with openSUSE 11.1 (bug #3213)
- Linux Additions: disable dynamic resizing if the X server is configured for fixed resolutions
- Linux/Solaris Additions: handle virtual resolutions properly which are larger than the actual guest resolution (bug #3096)

15.49 Version 2.1.2 (2009-01-21)

This is a maintenance release. The following items were fixed and/or added:

- USB: Linux host support fixes (bug #3136)
- VMM: fixed guru meditation for PAE guests on non-PAE hosts (AMD-V)
- VMM: fixed guru meditation on Mac OS X hosts when using VT-x
- VMM: allow running up to 1023 VMs on 64-bit hosts (used to be 127)
- VMM: several FreeBSD guest related fixes (bugs #2342, #2341, #2761)
- VMM: fixed guru meditation when installing Suse Enterprise Server 10U2 (VT-x only; bug #3039)
- VMM: fixed guru meditation when booting Novell Netware 4.11 (VT-x only; bug #2898)
- VMM: fixed VERR_ADDRESS_TOO_BIG error on some Mac OS X systems when starting a VM
- VMM: clear MSR_K6_EFER_SVME after probing for AMD-V (bug #3058)
- VMM: fixed guru meditation during Windows 7 boot with more than 2 GB guest RAM (VT-x, nested paging only)
- VMM: fixed hang during OS/2 MCP2 boot (AMD-V and VT-x only)
- VMM: fixed loop during OpenBSD 4.0 boot (VT-x only)
- VMM: fixed random crashes related to FPU/XMM with 64 bits guests on 32 bits hosts
- VMM: fixed occasional XMM state corruption with 64 bits guests
- GUI: raised the RAM limit for new VMs to 75% of the host memory

- GUI: added Windows 7 as operating system type
- VBoxSDL: fixed `-fixed fixedmode` parameter (bug #3067)
- Clipboard: stability fixes (Linux and Solaris hosts only, bug #2675 and #3003)
- 3D support: fixed VM crashes for certain guest applications (bugs #2781, #2797, #2972, #3089)
- LsiLogic: improved support for Windows guests (still experimental)
- VGA: fixed a 2.1.0 regression where guest screen resize events were not properly handled (bug #2783)
- VGA: significant performance improvements when using VT-x/AMD-V on Mac OS X hosts
- VGA: better handling for VRAM offset changes (fixes GRUB2 and Dos DOOM display issues)
- VGA: custom VESA modes with invalid widths are now rounded up to correct ones (bug #2895)
- IDE: fixed ATAPI passthrough support (Linux hosts only; bug #2795)
- Networking: fixed kernel panics due to NULL pointer dereference in Linux kernels < 2.6.20 (Linux hosts only; bug #2827)
- Networking: fixed intermittent BSODs when using the new host interface (Windows hosts only; bugs #2832, #2937, #2929)
- Networking: fixed several issues with displaying hostif NICs in the GUI (Windows hosts only; bugs 2814, #2842)
- Networking: fixed the issue with displaying hostif NICs without assigned IP addresses (Linux hosts only; bug #2780)
- Networking: fixed the issue with sent packets coming back to internal network when using hostif (Linux hosts only; bug #3056).
- NAT: fixed port forwarding (Windows hosts only; bug #2808)
- NAT: fixed booting from the builtin TFTP server (bug #1959)
- NAT: fixed occasional crashes (bug #2709)
- SATA: vendor product data (VPD) is now configurable
- SATA: raw disk partitions were not recognized (2.1.0 regression, Windows host only, bug #2778)
- SATA: fixed timeouts in the guest when using raw VMDK files (Linux host only, bug #2796)
- SATA: huge speed up during certain I/O operations like formatting a drive
- SATA/IDE: fixed possible crash/errors during VM shutdown
- VRDP: fixed loading of `libpam.so.1` from the host (Solaris hosts only)
- VRDP: fixed RDP client disconnects
- VRDP: fixed VRDP server misbehavior after a broken client connection
- VBoxManage `showvminfo`: fixed assertion for running VMs (bug #2773)

- VBoxManage `convertfromraw`: added parameter checking and made it default to creating VDI files; fixed and documented format parameter (bug #2776)
- VBoxManage `clonehd`: fixed garbled output image when creating VDI files (bug #2813)
- VBoxManage `guestproperty`: fixed property enumeration (incorrect parameters/exception)
- VHD: fixed error when attaching certain container files (bug #2768)
- Solaris hosts: added support for serial ports (bug #1849)
- Solaris hosts: fix for Japanese keyboards (bug #2847)
- Solaris hosts: 32-bit and 64-bit versions now available as a single, unified package
- Linux hosts: don't depend on `libcap1` anymore (bug #2859)
- Linux hosts: kernel module compile fixes for 2.6.29-rc1
- Linux hosts: don't drop any capability if the VM was started by root (2.1.0 regression)
- Mac OS X hosts: save the state of running or paused VMs when the host machine's battery reaches critical level
- Mac OS X hosts: improved window resizing of the VM window
- Mac OS X hosts: added GUI option to disable the dock icon realtime preview in the GUI to decrease the host CPU load when the guest is doing 3D
- Mac OS X hosts: polished realtime preview dock icon
- Windows Additions: fixed guest property and logging OS type detection for Windows 2008 and Windows 7 Beta
- Windows Additions: added support for Windows 7 Beta (bugs #2995, #3015)
- Windows Additions: fixed Windows 2000 guest freeze when accessing files on shared folders (bug #2764)
- Windows Additions: fixed CTRL-ALT-DEL handling when using VBoxGINA
- Windows Additions Installer: added /extract switch to only extract (not install) the files to a directory (can be specified with /D=path)
- Linux installer and Additions: added support for the Linux From Scratch distribution (bug #1587) and recent Gentoo versions (bug #2938)
- Additions: added experimental support for X.Org Server 1.6 RC on Linux guests
- Linux Additions: fixed bug which prevented to properly set `fmode` on mapped shared folders (bug #1776)
- Linux Additions: fixed appending of files on shared folders (bug #1612)
- Linux Additions: ignore `noauto` option when mounting a shared folder (bug #2498)
- Linux Additions: fixed a driver issue preventing X11 from compiling keymaps (bug #2793 and #2905)
- X11 Additions: workaround in the mouse driver for a server crash when the driver is loaded manually (bug #2397)

15.50 Version 2.1.0 (2008-12-17)

This version is a major update. The following major new features were added:

- Support for hardware virtualization (VT-x and AMD-V) on Mac OS X hosts
- Support for 64-bit guests on 32-bit host operating systems (experimental; see chapitre 3.1.2, *Invités 64 bits*, page 45)
- Added support for Intel Nehalem virtualization enhancements (EPT and VPID; see chapitre 10.3, *Hardware vs. software virtualization*, page 175)
- Experimental 3D acceleration via OpenGL (see chapitre 4.5.1, *Accélération 3D matérielle (OpenGL et Direct3D 8/9)*, page 72)
- Experimental LsiLogic and BusLogic SCSI controllers (see chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78)
- Full VMDK/VHD support including snapshots (see chapitre 5.2, *Fichiers images de disque (VDI, VMDK, VHD, HDD)*, page 81)
- New NAT engine with significantly better performance, reliability and ICMP echo (ping) support (bugs #1046, #2438, #2223, #1247)
- New Host Interface Networking implementations for Windows and Linux hosts with easier setup (replaces TUN/TAP on Linux and manual bridging on Windows)

In addition, the following items were fixed and/or added:

- VMM: significant performance improvements for VT-x (real mode execution)
- VMM: support for hardware breakpoints (VT-x and AMD-V only; bug #477)
- VMM: VGA performance improvements for VT-x and AMD-V
- VMM: Solaris and OpenSolaris guest performance improvements for AMD-V (Barcelona family CPUs only)
- VMM: fixed guru meditation while running the Dr. Web virus scanner (software virtualization only; bug #1439)
- VMM: deactivate VT-x and AMD-V when the host machine goes into suspend mode; reactivate when the host machine resumes (Windows, Mac OS X & Linux hosts; bug #1660)
- VMM: fixed guest hangs when restoring VT-x or AMD-V saved states/snapshots
- VMM: fixed guru meditation when executing a one byte debug instruction (VT-x only; bug #2617)
- VMM: fixed guru meditation for PAE guests on non-PAE hosts (VT-x)
- VMM: disallow mixing of software and hardware virtualization execution in general (bug #2404)
- VMM: fixed black screen when booting OS/2 1.x (AMD-V only)
- GUI: pause running VMs when the host machine goes into suspend mode (Windows & Mac OS X hosts)
- GUI: resume previously paused VMs when the host machine resumes after suspend (Windows & Mac OS X hosts)

- GUI: save the state of running or paused VMs when the host machine's battery reaches critical level (Windows hosts)
- GUI: properly restore the position of the selector window when running on the compiz window manager
- GUI: properly restore the VM in seamless mode (2.0 regression)
- GUI: warn user about non optimal memory settings
- GUI: structure operating system list according to family and version for improved usability
- GUI: predefined settings for QNX guests
- IDE: improved ATAPI passthrough support
- Networking: added support for up to 8 Ethernet adapters per VM
- Networking: fixed issue where a VM could lose connectivity after a reboot
- iSCSI: allow snapshot/diff creation using local VDI file
- iSCSI: improved interoperability with iSCSI targets
- Graphics: fixed handling of a guest video memory which is not a power of two (bug #2724)
- VBoxManage: fixed bug which prevented setting up the serial port for direct device access
- VBoxManage: added support for VMDK and VHD image creation
- VBoxManage: added support for image conversion (VDI/VMDK/VHD/RAW)
- Solaris hosts: added IPv6 support between host and guest when using host interface networking
- Mac OS X hosts: added ACPI host power status reporting
- API: redesigned storage model with better generalization
- API: allow attaching a hard disk to more than one VM at a time
- API: added methods to return network configuration information of the host system
- Shared Folders: performance and stability fixes for Windows guests (Microsoft Office Applications)

15.51 Version 2.0.8 (2009-03-10)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed guest hangs when restoring VT-x or AMD-V saved states/snapshots
- VMM: fixed memory allocation issues which can cause VM start failures with VERR_PGM_MAPPING_CONFLICT error
- VMM: fixed host crashes/hangs on certain 32 bits Windows systems when running Linux guests (bugs #1606, #2269, #2763)
- XPCOM/Main: fixed synchronization bug caused by SYSV semaphore key collisions
- ATA: fixed deadlock when pausing VM due to problems with the virtual disk (e.g. disk full, iSCSI target unavailable)

- iSCSI: fixed possible crash when pausing the VM
- iSCSI: fix PDU validity checking and detect final PDU reliably
- VBoxHeadless: really don't start X11 services (clipboard service, 3D acceleration; Solaris & Darwin hosts only; bug #3199)
- Networking: fixed issue where a VM could lose connectivity after a reboot
- Linux hosts: fixed occasional kernel oopses (bug #2556)
- Solaris hosts: fixed high CPU load while running many guests in parallel
- Solaris hosts: fixed inability to start more than 128 VMs
- Solaris/Web services: fixed SMF script to set home directory correctly
- Linux Additions: fixed occasional file corruption when writing files in *O_APPEND* mode to a shared folder (bug #2844)

15.52 Version 2.0.6 (2008-11-21)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed Guru meditation when running 64 bits Windows guests (bug #2220)
- VMM: fixed Solaris 10U6 boot hangs (VT-x and AMD-V) bug #2565)
- VMM: fixed Solaris 10U6 reboot hangs (AMD-V only; bug #2565)
- GUI: the host key was sometimes not properly displayed (Windows hosts only, bug #1996)
- GUI: the keyboard focus was lost after minimizing and restoring the VM window via the Windows taskbar (bugs #784)
- VBoxManage: properly show SATA disks when showing the VM information (bug #2624)
- SATA: fixed access if the buffer size is not sector-aligned (bug #2024)
- SATA: improved performance
- SATA: fixed snapshot function with ports>1 (bug #2510)
- E1000: fixed crash under rare circumstances
- USB: fixed support for iPhone and Nokia devices (Linux host: bugs #470 & #491)
- Windows host installer: added proper handling of open VirtualBox applications when updating the installation
- Windows host installer: fixed default installation directory on 64-bit on new installations (bug #2501)
- Linux/Solaris/Darwin hosts: verify permissions in `/tmp/vbox-$USER-ipc`
- Linux hosts: fixed assertion on high network load (AMD64 hosts, fix for Linux distributions with glibc 2.6 and newer (bug #616)
- Linux hosts: don't crash during shutdown with serial ports connected to a host device
- Solaris hosts: fixed incompatibility between IPSEC and host interface networking

- Solaris hosts: fixed a rare race condition while powering off VMs with host interface networking
- Solaris hosts: fixed VBoxSDL on Solaris 10 by shipping the required SDL library (bug #2475)
- Windows Additions: fixed logged in users reporting via guest properties when using native RDP connections
- Windows Additions: fixed Vista crashes when accessing shared folders under certain circumstances (bug #2461)
- Windows Additions: fixed shared folders access with MS-Office (bug #2591)
- Linux Additions: fixed compilation of vboxvfs.ko for 64-bit guests (bug #2550)
- SDK: added JAX-WS port caching to speedup connections

15.53 Version 2.0.4 (2008-10-24)

This is a maintenance release. The following items were fixed and/or added:

- VMM: better error reporting for VT-x failures
- VMM: don't overflow the release log with PATM messages (bug #1775)
- VMM: fixed save state restore in real mode (software virtualization only)
- GUI: work around a Qt bug on Mac OS X (bug #2321)
- GUI: properly install the Qt4 accessible plugin (bug #629)
- SATA: error message when starting a VM with a VMDK connected to a SATA port (bug #2182)
- SATA: fixed Guru mediation when booting OpenSolaris/64; most likely applies to other guests as well (bug #2292)
- Network: don't crash when changing the adapter link state if no host driver is attached (bug #2333)
- VHD: fixed bug which prevents booting from VHD images bigger than 4GB (bug #2085)
- VRDP: fixed a repaint problem when the guest resolution was not equal to the client resolution
- Clipboard: don't crash when host service initialization takes longer than expected (Linux hosts only; bug #2001)
- Windows hosts: VBoxSVC.exe crash (bug #2212)
- Windows hosts: VBoxSVC.exe memory leak due to a Windows WMI memory leak (Vista only) (bug #2242)
- Windows hosts: VBoxSVC.exe delays GUI startup
- Linux hosts: handle jiffies counter overflow (VM stuck after 300 seconds of host uptime; bug #2247)
- Solaris hosts: fixed host or guest side networking going stale while using host interface networking (bug #2474)

- Solaris hosts: added support for using unplumbed network interfaces and Crossbow Virtual Network Interfaces (VNICs) with host interface networking
- Solaris hosts: reworked threading model improves performance for host interface networking
- Windows Additions: fixed crash when accessing deep directory structures in a shared folder
- Windows Additions: improved shared folder name resolving (bug #1728)
- Windows Additions: fixed Windows 2000 shutdown crash (bug #2254)
- Windows Additions: fixed error code for `MoveFile()` if the target exists (bug #2350)
- Linux Additions: fixed `seek()` for files bigger than 2GB (bug #2379)
- Linux Additions: support Ubuntu 8.10
- Linux Additions: clipboard fixes (bug #2015)
- Web services: improved documentation and fixed example (bug #1642)

15.54 Version 2.0.2 (2008-09-12)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed inability to run more than one VM in parallel (AMD-V on CPUs with erratum 170 only; bug #2167)
- VMM: VT-x stability fixes (bug #2179 and others)
- VMM: fixed Linux 2.6.26+ kernel crashes (used by Ubuntu 8.10 Alpha, Fedora 10 Alpha; bug #1875)
- VMM: fixed 64 bits Linux 2.6.26 kernel crashes (Debian)
- VMM: fixed Vista (32 bits) guest crash during boot when PAE and NX are enabled (applied to 64 bits hosts with VT-x enabled only)
- VMM: fixed OS/2 guest crashes during boot (AMD-V; bug #2132)
- GUI: fixed crash when trying to release an inaccessible image in the virtual disk manager
- GUI: fixed invalid error message for a changed snapshot path even if that path wasn't changed (bug #2064)
- GUI: fixed crash when creating a new hard disk image (bug #2060)
- GUI: fixed crash when adding a hard disk in the VM settings (bug #2081)
- GUI: fixed a bug where VirtualBox isn't working with the new QGtkStyle plugin (bug #2066)
- GUI: fixed VM close dialog in seamless mode (Mac OS X hosts only; bug #2067)
- GUI: fixed standard menu entries for NLS versions (Mac OS X hosts only)
- GUI: disable the VT-x/AMD-V setting when it's not supported by the CPU (or on Mac OS X hosts)
- VBoxManage: fixed crash during `internalcommands createrawvmdk` (bug #2184)

- VBoxManage: fixed output of `snapshot showvminfo` (bug #698)
- Guest properties: added information about guest network interfaces (Windows guests only)
- Shared Folders: fixed regression that caused Windows guest crashes
- API: fixed number of installed CPUs (Solaris hosts only)
- VRDP: allow a client to reconnect to an existing session on the VRDP server by dropping the existing connection (configurable and disabled by default; only relevant when multi-connection mode is disabled)
- VRDP: fixed an image repaint problem
- Linux hosts: fixed bug in vboxdrv.ko that could corrupt kernel memory and panic the kernel (bug #2078)
- Linux hosts: compile fixes for kernel module on Linux 2.6.27
- Mac OS X hosts: added Python support
- Additions: fixed a possible hang in HGCM communication after a VM reboot
- Windows Additions: added support for Windows XP 64 bits (bug #2117)
- Linux Additions: deactivate dynamic resizing on Linux guests with buggy X servers
- Linux Additions: support Ubuntu 8.10 guests and Fedora 9 guests (dynamic resizing disabled for the latter)
- Linux Additions: added installer check for the system architecture
- Linux Additions: fixed Xorg modules path for some Linux distributions (bug #2128)
- VMDK: be more liberal with ambiguous parts of the format specification and accept more format variants (bug #2062)
- VHD: fixed a bug in the VHD backend which resulted in reading the wrong data (bug #2085)
- Solaris hosts: fixed kernel panic on certain machines when starting VMs with host interface networking (bug #2183)
- Solaris hosts: fixed inability to access NFS shares on the host when host interface networking was enabled
- Solaris hosts: installer now detects and reports when installing under the wrong architecture
- Solaris hosts: fixed security hardening that prevented starting VMs from non-global zones even as root (bug #1948)
- Solaris Additions: combined the 32 bit and 64 bit Additions installer into a single package
- Mac OS X hosts: experimental support for attaching a real serial port to the guest

15.55 Version 2.0.0 (2008-09-04)

This version is a major update. The following major new features were added:

- 64 bits guest support (64 bits host only)
- New native Leopard user interface on Mac OS X hosts
- The GUI was converted from Qt3 to Qt4 with many visual improvements
- New-version notifier
- Guest property information interface
- Host Interface Networking on Mac OS X hosts
- New Host Interface Networking on Solaris hosts
- Support for Nested Paging on modern AMD CPUs (major performance gain)
- Framework for collecting performance and resource usage data (metrics)
- Added SATA asynchronous IO (NCQ: Native Command Queuing) when accessing raw disks/partitions (major performance gain)
- Clipboard integration for OS/2 Guests
- Created separate SDK component featuring a new Python programming interface on Linux and Solaris hosts
- Support for VHD disk images

In addition, the following items were fixed and/or added:

- VMM: VT-x fixes
- AHCI: improved performance
- GUI: keyboard fixes
- Linux installer: properly uninstall the package even if unregistering the DKMS module fails
- Linux Additions: the guest screen resolution is properly restored
- Network: added support for jumbo frames (> 1536 bytes)
- Shared Folders: fixed guest crash with Windows Media Player 11
- Mac OS X: Ctrl+Left mouse click doesn't simulate a right mouse click in the guest anymore. Use Hostkey+Left for a right mouse click emulation. (bug #1766)

With VirtualBox 3.2, changelog information for versions before 2.0 was removed in order to save space. To access this information, please consult the User Manual of VirtualBox version 3.1 or earlier.

16 Third-party materials and licenses

VirtualBox incorporates materials from several Open Source software projects. Therefore the use of these materials by VirtualBox is governed by different Open Source licenses. This document reproduces these licenses and provides a list of the materials used and their respective licensing conditions. Section 1 contains a list of the materials used. Section 2 reproduces the applicable Open Source licenses. For each material, a reference to its license is provided.

The source code for the materials listed below as well as the rest of the VirtualBox code which is released as open source are available at http://www.virtualbox.org, both as tarballs for particular releases and as a live SVN repository.

16.1 Materials

- VirtualBox contains portions of QEMU which is governed by the licenses in chapitre 16.2.5, *X Consortium License (X11)*, page 297 and chapitre 16.2.2, *GNU Lesser General Public License (LGPL)*, page 286 and

  (C) 2003-2005 Fabrice Bellard; Copyright (C) 2004-2005 Vassili Karpov (malc); Copyright (c) 2004 Antony T Curtis; Copyright (C) 2003 Jocelyn Mayer

- VirtualBox contains code which is governed by the license in chapitre 16.2.5, *X Consortium License (X11)*, page 297 and

  Copyright 2004 by the Massachusetts Institute of Technology.

- VirtualBox contains code of the BOCHS VGA BIOS which is governed by the license in chapitre 16.2.2, *GNU Lesser General Public License (LGPL)*, page 286 and

  Copyright (C) 2001, 2002 the LGPL VGABios developers Team.

- VirtualBox contains code of the BOCHS ROM BIOS which is governed by the license in chapitre 16.2.2, *GNU Lesser General Public License (LGPL)*, page 286 and

  Copyright (C) 2002 MandrakeSoft S.A.; Copyright (C) 2004 Fabrice Bellard; Copyright (C) 2005 Struan Bartlett.

- VirtualBox contains the zlib library which is governed by the license in chapitre 16.2.6, *zlib license*, page 297 and

  Copyright (C) 1995-2003 Jean-loup Gailly and Mark Adler.

- VirtualBox may contain OpenSSL which is governed by the license in chapitre 16.2.7, *OpenSSL license*, page 297 and

  Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

- VirtualBox may contain NSPR and XPCOM which is governed by the license in chapitre 16.2.3, *Mozilla Public License (MPL)*, page 291 and

  Copyright (C) The Authors.

- VirtualBox contains Slirp which is governed by the license in chapitre 16.2.8, *Slirp license*, page 298 and was written by Danny Gasparovski.

  Copyright (C) 1995, 1996 All Rights Reserved.

- VirtualBox contains liblzf which is governed by the license in chapitre 16.2.9, *liblzf license*, page 299 and

  Copyright (C) 2000-2005 Marc Alexander Lehmann <schmorp@schmorp.de>

- VirtualBox may ship with a modified copy of rdesktop which is governed by the license in chapitre 16.2.1, *GNU General Public License (GPL)*, page 282 and

  Copyright (C) Matthew Chapman and others.

- VirtualBox may ship with a copy of kchmviewer which is governed by the license in chapitre 16.2.1, *GNU General Public License (GPL)*, page 282 and

  Copyright (C) George Yunaev and others.

- VirtualBox may contain Etherboot which is governed by the license in chapitre 16.2.1, *GNU General Public License (GPL)*, page 282 with the exception that aggregating Etherboot with another work does not require the other work to be released under the same license (see `http://etherboot.sourceforge.net/clinks.html`). Etherboot is

  Copyright (C) Etherboot team.

- VirtualBox contains code from Wine which is governed by the license in chapitre 16.2.2, *GNU Lesser General Public License (LGPL)*, page 286 and

  Copyright 1993 Bob Amstadt, Copyright 1996 Albrecht Kleine, Copyright 1997 David Faure, Copyright 1998 Morten Welinder, Copyright 1998 Ulrich Weigand, Copyright 1999 Ove Koven

- VirtualBox contains code from lwIP which is governed by the license in chapitre 16.2.11, *lwIP license*, page 299 and

  Copyright (C) 2001, 2002 Swedish Institute of Computer Science.

- VirtualBox contains libxml which is governed by the license in chapitre 16.2.12, *libxml license*, page 300 and

  Copyright (C) 1998-2003 Daniel Veillard.

- VirtualBox contains libxslt which is governed by the license in chapitre 16.2.13, *libxslt licenses*, page 300 and

  Copyright (C) 2001-2002 Daniel Veillard and Copyright (C) 2001-2002 Thomas Broyer, Charlie Bozeman and Daniel Veillard.

- VirtualBox contains code from the gSOAP XML web services tools, which are licensed under the license in chapitre 16.2.14, *gSOAP Public License Version 1.3a*, page 301 and

  Copyright (C) 2000-2007, Robert van Engelen, Genivia Inc., and others.

- VirtualBox ships with the application tunctl (shipped as VBoxTunctl) from the User-mode Linux suite which is governed by the license in chapitre 16.2.1, *GNU General Public License (GPL)*, page 282 and

  Copyright (C) 2002 Jeff Dike.

- VirtualBox contains code from Chromium, an OpenGL implementation, which is goverened by the licenses in chapitre 16.2.15, *Chromium licenses*, page 306 and

  Copyright (C) Stanford University, The Regents of the University of California, Red Hat, and others.

- VirtualBox contains libcurl which is governed by the license in chapitre 16.2.16, *curl license*, page 308 and

  Copyright (C) 1996-2009, Daniel Stenberg.

- VirtualBox contains dnsproxy which is governed by the license in chapitre 16.2.4, *MIT License*, page 297 and

  Copyright (c) 2003, 2004, 2005 Armin Wolfermann.

- VirtualBox may contain iniparser which is governed by the license in chapitre 16.2.4, *MIT License*, page 297 and

  Copyright (c) 2000-2008 by Nicolas Devillard.

- VirtualBox contains some code from libgd which is governed by the license in chapitre 16.2.17, *libgd license*, page 308 and

  Copyright 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007 Pierre-Alain Joye (pierre@libgd.org).

- VirtualBox contains code from the EFI Development Kit II which is governed by the license in chapitre 16.2.18, *BSD license from Intel*, page 309 and

  Copyright (c) 2004-2008, Intel Corporation.

- VirtualBox contains libjpeg which is governed by the license in chapitre 16.2.19, *libjpeg License*, page 309 and

  Copyright (C) 1991-2010, Thomas G. Lane, Guido Vollbeding.

- VirtualBox may contain x86 SIMD extension for IJG JPEG library which is governed by the license in chapitre 16.2.20, *x86 SIMD extension for IJG JPEG library license*, page 310 and

  Copyright 2009 Pierre Ossman <ossman@cendio.se> for Cendio AB; Copyright 2010 D. R. Commander; Copyright (C) 1999-2006, MIYASAKA Masaru.

16.2 Licenses

16.2.1 GNU General Public License (GPL)

GNU GENERAL PUBLIC LICENSE Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.
When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.
To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.
For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this

License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously

your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PRO-

GRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

16.2.2 GNU Lesser General Public License (LGPL)

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages–typically libraries–of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the “Lesser” General Public License because it does Less to protect the user’s freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users’ freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a “work based on the library” and a “work that uses the library”. The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called “this License”). Each licensee is addressed as “you”.

A “library” means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The “Library”, below, refers to any such software library or work which has been distributed under these terms. A “work based on the Library” means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term “modification”.)

“Source code” for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library’s complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a “work that uses the Library”.

Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a “work that uses the Library” with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a “work that uses the library”. The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a “work that uses the Library” uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.) Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a “work that uses the Library” with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer’s own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable “work that uses the Library”, as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user’s computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the “work that uses the Library” must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries,

so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

16.2.3 Mozilla Public License (MPL)

MOZILLA PUBLIC LICENSE Version 1.1

1. Definitions.

1.0.1. "Commercial Use" means distribution or otherwise making the Covered Code available to a third party.

1.1. "Contributor" means each entity that creates or contributes to the creation of Modifications.

1.2. "Contributor Version" means the combination of the Original Code, prior Modifications used by a Contributor, and the Modifications made by that particular Contributor.

1.3. "Covered Code" means the Original Code or Modifications or the combination of the Original Code and Modifications, in each case including portions thereof.

1.4. "Electronic Distribution Mechanism" means a mechanism generally accepted in the software development community for the electronic transfer of data.

1.5. "Executable" means Covered Code in any form other than Source Code.

1.6. "Initial Developer" means the individual or entity identified as the Initial Developer in the Source Code notice required by Exhibit A.

1.7. “Larger Work” means a work which combines Covered Code or portions thereof with code not governed by the terms of this License.

1.8. “License” means this document.

1.8.1. “Licensable” means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein.

1.9. “Modifications” means any addition to or deletion from the substance or structure of either the Original Code or any previous Modifications. When Covered Code is released as a series of files, a Modification is:

A. Any addition to or deletion from the contents of a file containing Original Code or previous Modifications.

B. Any new file that contains any part of the Original Code or previous Modifications.

1.10. “Original Code” means Source Code of computer software code which is described in the Source Code notice required by Exhibit A as Original Code, and which, at the time of its release under this License is not already Covered Code governed by this License.

1.10.1. “Patent Claims” means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent Licensable by grantor.

1.11. “Source Code” means the preferred form of the Covered Code for making modifications to it, including all modules it contains, plus any associated interface definition files, scripts used to control compilation and installation of an Executable, or source code differential comparisons against either the Original Code or another well known, available Covered Code of the Contributor’s choice. The Source Code can be in a compressed or archival form, provided the appropriate decompression or de-archiving software is widely available for no charge.

1.12. “You” (or “Your”) means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License or a future version of this License issued under Section 6.1. For legal entities, “You” includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, “control” means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2. Source Code License.

2.1. The Initial Developer Grant. The Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims:

(a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer to use, reproduce, modify, display, perform, sublicense and distribute the Original Code (or portions thereof) with or without Modifications, and/or as part of a Larger Work; and

(b) under Patents Claims infringed by the making, using or selling of Original Code, to make, have made, use, practice, sell, and offer for sale, and/or otherwise dispose of the Original Code (or portions thereof).

(c) the licenses granted in this Section 2.1(a) and (b) are effective on the date Initial Developer first distributes Original Code under the terms of this License.

(d) Notwithstanding Section 2.1(b) above, no patent license is granted: 1) for code that You delete from the Original Code; 2) separate from the Original Code; or 3) for infringements caused by: i) the modification of the Original Code or ii) the combination of the Original Code with other software or devices.

2.2. Contributor Grant. Subject to third party intellectual property claims, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license

(a) under intellectual property rights (other than patent or trademark) Licensable by Contributor, to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof) either on an unmodified basis, with other Modifications, as Covered Code and/or as part of a Larger Work; and

(b) under Patent Claims infringed by the making, using, or selling of Modifications made by that Contributor either alone and/or in combination with its Contributor Version (or portions

of such combination), to make, use, sell, offer for sale, have made, and/or otherwise dispose of: 1) Modifications made by that Contributor (or portions thereof); and 2) the combination of Modifications made by that Contributor with its Contributor Version (or portions of such combination).

(c) the licenses granted in Sections 2.2(a) and 2.2(b) are effective on the date Contributor first makes Commercial Use of the Covered Code.

(d) Notwithstanding Section 2.2(b) above, no patent license is granted: 1) for any code that Contributor has deleted from the Contributor Version; 2) separate from the Contributor Version; 3) for infringements caused by: i) third party modifications of Contributor Version or ii) the combination of Modifications made by that Contributor with other software (except as part of the Contributor Version) or other devices; or 4) under Patent Claims infringed by Covered Code in the absence of Modifications made by that Contributor.

3. Distribution Obligations.

3.1. Application of License. The Modifications which You create or to which You contribute are governed by the terms of this License, including without limitation Section 2.2. The Source Code version of Covered Code may be distributed only under the terms of this License or a future version of this License released under Section 6.1, and You must include a copy of this License with every copy of the Source Code You distribute. You may not offer or impose any terms on any Source Code version that alters or restricts the applicable version of this License or the recipients' rights hereunder. However, You may include an additional document offering the additional rights described in Section 3.5.

3.2. Availability of Source Code. Any Modification which You create or to which You contribute must be made available in Source Code form under the terms of this License either on the same media as an Executable version or via an accepted Electronic Distribution Mechanism to anyone to whom you made an Executable version available; and if made available via Electronic Distribution Mechanism, must remain available for at least twelve (12) months after the date it initially became available, or at least six (6) months after a subsequent version of that particular Modification has been made available to such recipients. You are responsible for ensuring that the Source Code version remains available even if the Electronic Distribution Mechanism is maintained by a third party.

3.3. Description of Modifications. You must cause all Covered Code to which You contribute to contain a file documenting the changes You made to create that Covered Code and the date of any change. You must include a prominent statement that the Modification is derived, directly or indirectly, from Original Code provided by the Initial Developer and including the name of the Initial Developer in (a) the Source Code, and (b) in any notice in an Executable version or related documentation in which You describe the origin or ownership of the Covered Code.

3.4. Intellectual Property Matters

(a) Third Party Claims. If Contributor has knowledge that a license under a third party's intellectual property rights is required to exercise the rights granted by such Contributor under Sections 2.1 or 2.2, Contributor must include a text file with the Source Code distribution titled "LEGAL" which describes the claim and the party making the claim in sufficient detail that a recipient will know whom to contact. If Contributor obtains such knowledge after the Modification is made available as described in Section 3.2, Contributor shall promptly modify the LEGAL file in all copies Contributor makes available thereafter and shall take other steps (such as notifying appropriate mailing lists or newsgroups) reasonably calculated to inform those who received the Covered Code that new knowledge has been obtained.

(b) Contributor APIs. If Contributor's Modifications include an application programming interface and Contributor has knowledge of patent licenses which are reasonably necessary to implement that API, Contributor must also include this information in the LEGAL file.

3.5. Required Notices. You must duplicate the notice in Exhibit A in each file of the Source Code. If it is not possible to put such notice in a particular Source Code file due to its structure, then You must include such notice in a location (such as a relevant directory) where a user would be likely to look for such a notice. If You created one or more Modification(s) You may add your

name as a Contributor to the notice described in Exhibit A. You must also duplicate this License in any documentation for the Source Code where You describe recipients' rights or ownership rights relating to Covered Code. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Code. However, You may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor. You must make it absolutely clear than any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of warranty, support, indemnity or liability terms You offer.

3.6. Distribution of Executable Versions. You may distribute Covered Code in Executable form only if the requirements of Section 3.1-3.5 have been met for that Covered Code, and if You include a notice stating that the Source Code version of the Covered Code is available under the terms of this License, including a description of how and where You have fulfilled the obligations of Section 3.2. The notice must be conspicuously included in any notice in an Executable version, related documentation or collateral in which You describe recipients' rights relating to the Covered Code. You may distribute the Executable version of Covered Code or ownership rights under a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable version does not attempt to limit or alter the recipient's rights in the Source Code version from the rights set forth in this License. If You distribute the Executable version under a different license You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or any Contributor. You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of any such terms You offer.

3.7. Larger Works. You may create a Larger Work by combining Covered Code with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Code.

4. Inability to Comply Due to Statute or Regulation.If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Code due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be included in the LEGAL file described in Section 3.4 and must be included with all distributions of the Source Code. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5. Application of this License. This License applies to code to which the Initial Developer has attached the notice in Exhibit A and to related Covered Code.

6. Versions of the License.

6.1. New Versions. Netscape Communications Corporation ("Netscape") may publish revised and/or new versions of the License from time to time. Each version will be given a distinguishing version number.

6.2. Effect of New Versions. Once Covered Code has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Covered Code under the terms of any subsequent version of the License published by Netscape. No one other than Netscape has the right to modify the terms applicable to Covered Code created under this License.

6.3. Derivative Works. If You create or use a modified version of this License (which you may only do in order to apply it to code which is not already Covered Code governed by this License), You must (a) rename Your license so that the phrases "Mozilla", "MOZILLAPL", "MOZPL", "Netscape", "MPL", "NPL" or any confusingly similar phrase do not appear in your license (except to note that your license differs from this License) and (b) otherwise make it clear that Your version of the license contains terms which differ from the Mozilla Public License and Netscape

Public License. (Filling in the name of the Initial Developer, Original Code or Contributor in the notice described in Exhibit A shall not of themselves be deemed to be modifications of this License.)

7. DISCLAIMER OF WARRANTY.

COVERED CODE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED CODE IS FREE OF DEFECTS, MERCHANTABLE, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGING. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED CODE IS WITH YOU. SHOULD ANY COVERED CODE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED CODE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

8. TERMINATION.

8.1. This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. All sublicenses to the Covered Code which are properly granted shall survive any termination of this License. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

8.2. If You initiate litigation by asserting a patent infringement claim (excluding declaratory judgment actions) against Initial Developer or a Contributor (the Initial Developer or Contributor against whom You file such action is referred to as "Participant") alleging that:

(a) such Participant's Contributor Version directly or indirectly infringes any patent, then any and all rights granted by such Participant to You under Sections 2.1 and/or 2.2 of this License shall, upon 60 days notice from Participant terminate prospectively, unless if within 60 days after receipt of notice You either: (i) agree in writing to pay Participant a mutually agreeable reasonable royalty for Your past and future use of Modifications made by such Participant, or (ii) withdraw Your litigation claim with respect to the Contributor Version against such Participant. If within 60 days of notice, a reasonable royalty and payment arrangement are not mutually agreed upon in writing by the parties or the litigation claim is not withdrawn, the rights granted by Participant to You under Sections 2.1 and/or 2.2 automatically terminate at the expiration of the 60 day notice period specified above.

(b) any software, hardware, or device, other than such Participant's Contributor Version, directly or indirectly infringes any patent, then any rights granted to You by such Participant under Sections 2.1(b) and 2.2(b) are revoked effective as of the date You first made, used, sold, distributed, or had made, Modifications made by that Participant.

8.3. If You assert a patent infringement claim against Participant alleging that such Participant's Contributor Version directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by such Participant under Sections 2.1 or 2.2 shall be taken into account in determining the amount or value of any payment or license.

8.4. In the event of termination under Sections 8.1 or 8.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or any distributor hereunder prior to termination shall survive termination.

9. LIMITATION OF LIABILITY. UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED CODE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR

DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

10. U.S. GOVERNMENT END USERS. The Covered Code is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" and "commercial computer software documentation," as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire Covered Code with only those rights set forth herein.

11. MISCELLANEOUS. This License represents the complete agreement concerning subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. This License shall be governed by California law provisions (except to the extent applicable law, if any, provides otherwise), excluding its conflict-of-law provisions. With respect to disputes in which at least one party is a citizen of, or an entity chartered or registered to do business in the United States of America, any litigation relating to this License shall be subject to the jurisdiction of the Federal Courts of the Northern District of California, with venue lying in Santa Clara County, California, with the losing party responsible for costs, including without limitation, court costs and reasonable attorneys' fees and expenses. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not apply to this License.

12. RESPONSIBILITY FOR CLAIMS. As between Initial Developer and the Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability.

13. MULTIPLE-LICENSED CODE. Initial Developer may designate portions of the Covered Code as "Multiple-Licensed". "Multiple-Licensed" means that the Initial Developer permits you to utilize portions of the Covered Code under Your choice of the NPL or the alternative licenses, if any, specified by the Initial Developer in the file described in Exhibit A.

EXHIBIT A -Mozilla Public License.

"The contents of this file are subject to the Mozilla Public License Version 1.1 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.mozilla.org/MPL/

Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code is _____.

The Initial Developer of the Original Code is _____. Portions created by _____ are Copyright (C) _____ _____. All Rights Reserved.

Contributor(s): _____.

Alternatively, the contents of this file may be used under the terms of the _____ license (the "[___] License"), in which case the provisions of [_____] License are applicable instead of those above. If you wish to allow use of your version of this file only under the terms of the [____] License and not to allow others to use your version of this file under the MPL, indicate your decision by deleting the provisions above and replace them with the notice and other provisions required by the [___] License. If you do not delete the provisions above, a recipient may use your version of this file under either the MPL or the [___] License."

[NOTE: The text of this Exhibit A may differ slightly from the text of the notices in the Source Code files of the Original Code. You should use the text of this Exhibit A rather than the text found in the Original Code Source Code for Your Modifications.]

16.2.4 MIT License

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

16.2.5 X Consortium License (X11)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

16.2.6 zlib license

This software is provided ’as-is’, without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

```
Jean-loup Gailly        Mark Adler
jloup@gzip.org          madler@alumni.caltech.edu
```

16.2.7 OpenSSL license

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscape’s SSL.

This library is free for commercial and non-commercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)" The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related :-).

4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publicly available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

16.2.8 Slirp license

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: This product includes software developed by Danny Gasparovski.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL DANNY GASPAROVSKI OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,

PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

16.2.9 liblzf license

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

16.2.10 libpng license

The PNG Reference Library is supplied "AS IS". The Contributing Authors and Group 42, Inc. disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The Contributing Authors and Group 42, Inc. assume no liability for direct, indirect, incidental, special, exemplary, or consequential damages, which may result from the use of the PNG Reference Library, even if advised of the possibility of such damage.

Permission is hereby granted to use, copy, modify, and distribute this source code, or portions hereof, for any purpose, without fee, subject to the following restrictions:

1. The origin of this source code must not be misrepresented.

2. Altered versions must be plainly marked as such and must not be misrepresented as being the original source.

3. This Copyright notice may not be removed or altered from any source or altered source distribution.

The Contributing Authors and Group 42, Inc. specifically permit, without fee, and encourage the use of this source code as a component to supporting the PNG file format in commercial products. If you use this source code in a product, acknowledgment is not required but would be appreciated.

16.2.11 lwIP license

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

16.2.12 libxml license

Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar licence but with different Copyright notices) all the files are:

Copyright (C) 1998-2003 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

16.2.13 libxslt licenses

Licence for libxslt except libexslt:

Copyright (C) 2001-2002 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE

DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

Licence for libexslt:

Copyright (C) 2001-2002 Thomas Broyer, Charlie Bozeman and Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of the authors shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

16.2.14 gSOAP Public License Version 1.3a

The gSOAP public license is derived from the Mozilla Public License (MPL1.1). The sections that were deleted from the original MPL1.1 text are 1.0.1, 2.1.(c),(d), 2.2.(c),(d), 8.2.(b), 10, and 11. Section 3.8 was added. The modified sections are 2.1.(b), 2.2.(b), 3.2 (simplified), 3.5 (deleted the last sentence), and 3.6 (simplified).

1 DEFINITIONS

1.1. "Contributor" means each entity that creates or contributes to the creation of Modifications.

1.2. "Contributor Version" means the combination of the Original Code, prior Modifications used by a Contributor, and the Modifications made by that particular Contributor.

1.3. "Covered Code" means the Original Code, or Modifications or the combination of the Original Code, and Modifications, in each case including portions thereof.

1.4. "Electronic Distribution Mechanism" means a mechanism generally accepted in the software development community for the electronic transfer of data.

1.5. "Executable" means Covered Code in any form other than Source Code.

1.6. "Initial Developer" means the individual or entity identified as the Initial Developer in the Source Code notice required by Exhibit A.

1.7. "Larger Work" means a work which combines Covered Code or portions thereof with code not governed by the terms of this License.

1.8. "License" means this document.

1.8.1. "Licensable" means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein.

1.9. "Modifications" means any addition to or deletion from the substance or structure of either the Original Code or any previous Modifications. When Covered Code is released as a series of files, a Modification is:

A. Any addition to or deletion from the contents of a file containing Original Code or previous Modifications.

B. Any new file that contains any part of the Original Code, or previous Modifications.

1.10. “Original Code” means Source Code of computer software code which is described in the Source Code notice required by Exhibit A as Original Code, and which, at the time of its release under this License is not already Covered Code governed by this License.

1.10.1. “Patent Claims” means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent Licensable by grantor.

1.11. “Source Code” means the preferred form of the Covered Code for making modifications to it, including all modules it contains, plus any associated interface definition files, scripts used to control compilation and installation of an Executable, or source code differential comparisons against either the Original Code or another well known, available Covered Code of the Contributor’s choice. The Source Code can be in a compressed or archival form, provided the appropriate decompression or de-archiving software is widely available for no charge.

1.12. “You” (or “Your”) means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License or a future version of this License issued under Section 6.1. For legal entities, “You” includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, “control” means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2 SOURCE CODE LICENSE.

2.1. The Initial Developer Grant.

The Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims:

(a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer to use, reproduce, modify, display, perform, sublicense and distribute the Original Code (or portions thereof) with or without Modifications, and/or as part of a Larger Work; and

(b) under patents now or hereafter owned or controlled by Initial Developer, to make, have made, use and sell (“offer to sell and import”) the Original Code, Modifications, or portions thereof, but solely to the extent that any such patent is reasonably necessary to enable You to utilize, alone or in combination with other software, the Original Code, Modifications, or any combination or portions thereof.

(c)

(d)

2.2. Contributor Grant.

Subject to third party intellectual property claims, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license

(a) under intellectual property rights (other than patent or trademark) Licensable by Contributor, to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof) either on an unmodified basis, with other Modifications, as Covered Code and/or as part of a Larger Work; and

(b) under patents now or hereafter owned or controlled by Contributor, to make, have made, use and sell (“offer to sell and import”) the Contributor Version (or portions thereof), but solely to the extent that any such patent is reasonably necessary to enable You to utilize, alone or in combination with other software, the Contributor Version (or portions thereof).

(c)

(d)

3 DISTRIBUTION OBLIGATIONS.

3.1. Application of License.

The Modifications which You create or to which You contribute are governed by the terms of this License, including without limitation Section 2.2. The Source Code version of Covered

Code may be distributed only under the terms of this License or a future version of this License released under Section 6.1, and You must include a copy of this License with every copy of the Source Code You distribute. You may not offer or impose any terms on any Source Code version that alters or restricts the applicable version of this License or the recipients' rights hereunder. However, You may include an additional document offering the additional rights described in Section 3.5.

3.2. Availability of Source Code.

Any Modification created by You will be provided to the Initial Developer in Source Code form and are subject to the terms of the License. 3.3. Description of Modifications.

You must cause all Covered Code to which You contribute to contain a file documenting the changes You made to create that Covered Code and the date of any change. You must include a prominent statement that the Modification is derived, directly or indirectly, from Original Code provided by the Initial Developer and including the name of the Initial Developer in (a) the Source Code, and (b) in any notice in an Executable version or related documentation in which You describe the origin or ownership of the Covered Code.

3.4. Intellectual Property Matters.

(a) Third Party Claims. If Contributor has knowledge that a license under a third party's intellectual property rights is required to exercise the rights granted by such Contributor under Sections 2.1 or 2.2, Contributor must include a text file with the Source Code distribution titled "LEGAL" which describes the claim and the party making the claim in sufficient detail that a recipient will know whom to contact. If Contributor obtains such knowledge after the Modification is made available as described in Section 3.2, Contributor shall promptly modify the LEGAL file in all copies Contributor makes available thereafter and shall take other steps (such as notifying appropriate mailing lists or newsgroups) reasonably calculated to inform those who received the Covered Code that new knowledge has been obtained.

(b) Contributor APIs. If Contributor's Modifications include an application programming interface and Contributor has knowledge of patent licenses which are reasonably necessary to implement that API, Contributor must also include this information in the LEGAL file.

(c) Representations. Contributor represents that, except as disclosed pursuant to Section 3.4(a) above, Contributor believes that Contributor's Modifications are Contributor's original creation(s) and/or Contributor has sufficient rights to grant the rights conveyed by this License.

3.5. Required Notices. You must duplicate the notice in Exhibit A in each file of the Source Code. If it is not possible to put such notice in a particular Source Code file due to its structure, then You must include such notice in a location (such as a relevant directory) where a user would be likely to look for such a notice. If You created one or more Modification(s) You may add your name as a Contributor to the notice described in Exhibit A. You must also duplicate this License in any documentation for the Source Code where You describe recipients' rights or ownership rights relating to Covered Code. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Code. However, You may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor.

3.6. Distribution of Executable Versions. You may distribute Covered Code in Executable form only if the requirements of Section 3.1-3.5 have been met for that Covered Code. You may distribute the Executable version of Covered Code or ownership rights under a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable version does not attempt to limit or alter the recipient's rights in the Source Code version from the rights set forth in this License. If You distribute the Executable version under a different license You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or any Contributor. If you distribute executable versions containing Covered Code as part of a product, you must reproduce the notice in Exhibit B in the documentation and/or other materials provided with the product.

3.7. Larger Works. You may create a Larger Work by combining Covered Code with other code not governed by the terms of this License and distribute the Larger Work as a single product. In

such a case, You must make sure the requirements of this License are fulfilled for the Covered Code.

3.8. Restrictions. You may not remove any product identification, copyright, proprietary notices or labels from gSOAP.

4 INABILITY TO COMPLY DUE TO STATUTE OR REGULATION.

If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Code due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be included in the LEGAL file described in Section 3.4 and must be included with all distributions of the Source Code. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5 APPLICATION OF THIS LICENSE.

This License applies to code to which the Initial Developer has attached the notice in Exhibit A and to related Covered Code.

6 VERSIONS OF THE LICENSE.

6.1. New Versions.

Grantor may publish revised and/or new versions of the License from time to time. Each version will be given a distinguishing version number.

6.2. Effect of New Versions.

Once Covered Code has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Covered Code under the terms of any subsequent version of the License.

6.3. Derivative Works.

If You create or use a modified version of this License (which you may only do in order to apply it to code which is not already Covered Code governed by this License), You must (a) rename Your license so that the phrase “gSOAP” or any confusingly similar phrase do not appear in your license (except to note that your license differs from this License) and (b) otherwise make it clear that Your version of the license contains terms which differ from the gSOAP Public License. (Filling in the name of the Initial Developer, Original Code or Contributor in the notice described in Exhibit A shall not of themselves be deemed to be modifications of this License.)

7 DISCLAIMER OF WARRANTY.

COVERED CODE IS PROVIDED UNDER THIS LICENSE ON AN “AS IS” BASIS, WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, OF FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS, AND ANY WARRANTY THAT MAY ARISE BY REASON OF TRADE USAGE, CUSTOM, OR COURSE OF DEALING. WITHOUT LIMITING THE FOREGOING, YOU ACKNOWLEDGE THAT THE SOFTWARE IS PROVIDED “AS IS” AND THAT THE AUTHORS DO NOT WARRANT THE SOFTWARE WILL RUN UNINTERRUPTED OR ERROR FREE. LIMITED LIABILITY THE ENTIRE RISK AS TO RESULTS AND PERFORMANCE OF THE SOFTWARE IS ASSUMED BY YOU. UNDER NO CIRCUMSTANCES WILL THE AUTHORS BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL, EXEMPLARY OR CONSEQUENTIAL DAMAGES OF ANY KIND OR NATURE WHATSOEVER, WHETHER BASED ON CONTRACT, WARRANTY, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY OR OTHERWISE, ARISING OUT OF OR IN ANY WAY RELATED TO THE SOFTWARE, EVEN IF THE AUTHORS HAVE BEEN ADVISED ON THE POSSIBILITY OF SUCH DAMAGE OR IF SUCH DAMAGE COULD HAVE BEEN REASONABLY FORESEEN, AND NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY EXCLUSIVE REMEDY PROVIDED. SUCH LIMITATION ON DAMAGES INCLUDES, BUT IS NOT LIMITED TO, DAMAGES FOR LOSS OF GOODWILL, LOST PROFITS, LOSS OF DATA OR SOFTWARE, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION OR IMPAIRMENT OF OTHER GOODS. IN NO EVENT WILL THE AUTHORS BE LIABLE FOR THE COSTS OF PROCUREMENT OF SUBSTITUTE SOFTWARE OR SERVICES. YOU ACKNOWLEDGE THAT THIS SOFTWARE IS NOT DESIGNED FOR USE IN ON-LINE EQUIP-

MENT IN HAZARDOUS ENVIRONMENTS SUCH AS OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION OR CONTROL, OR LIFE-CRITICAL APPLICATIONS. THE AUTHORS EXPRESSLY DISCLAIM ANY LIABILITY RESULTING FROM USE OF THE SOFTWARE IN ANY SUCH ON-LINE EQUIPMENT IN HAZARDOUS ENVIRONMENTS AND ACCEPTS NO LIABILITY IN RESPECT OF ANY ACTIONS OR CLAIMS BASED ON THE USE OF THE SOFTWARE IN ANY SUCH ON-LINE EQUIPMENT IN HAZARDOUS ENVIRONMENTS BY YOU. FOR PURPOSES OF THIS PARAGRAPH, THE TERM "LIFE-CRITICAL APPLICATION" MEANS AN APPLICATION IN WHICH THE FUNCTIONING OR MALFUNCTIONING OF THE SOFTWARE MAY RESULT DIRECTLY OR INDIRECTLY IN PHYSICAL INJURY OR LOSS OF HUMAN LIFE. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED CODE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

8 TERMINATION.

8.1.

This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. All sublicenses to the Covered Code which are properly granted shall survive any termination of this License. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

8.2.

8.3.

If You assert a patent infringement claim against Participant alleging that such Participant's Contributor Version directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by such Participant under Sections 2.1 or 2.2 shall be taken into account in determining the amount or value of any payment or license.

8.4. In the event of termination under Sections 8.1 or 8.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or any distributor hereunder prior to termination shall survive termination.

9 LIMITATION OF LIABILITY.

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED CODE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

10 U.S. GOVERNMENT END USERS.

11 MISCELLANEOUS.

12 RESPONSIBILITY FOR CLAIMS.

As between Initial Developer and the Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability.

EXHIBIT A.

"The contents of this file are subject to the gSOAP Public License Version 1.3 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the Li-

cense at http://www.cs.fsu.edu/~engelen/soaplicense.html. Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code of the gSOAP Software is: stdsoap.h, stdsoap2.h, stdsoap.c, stdsoap2.c, stdsoap.cpp, stdsoap2.cpp, soapcpp2.h, soapcpp2.c, soapcpp2_lex.l, soapcpp2_yacc.y, error2.h, error2.c, symbol2.c, init2.c, soapdoc2.html, and soapdoc2.pdf, httpget.h, httpget.c, stl.h, stldeque.h, stllist.h, stlvector.h, stlset.h.

The Initial Developer of the Original Code is Robert A. van Engelen. Portions created by Robert A. van Engelen are Copyright (C) 2001-2004 Robert A. van Engelen, Genivia inc. All Rights Reserved.

Contributor(s): "_____." [Note: The text of this Exhibit A may differ slightly form the text of the notices in the Source Code files of the Original code. You should use the text of this Exhibit A rather than the text found in the Original Code Source Code for Your Modifications.]

EXHIBIT B.

"Part of the software embedded in this product is gSOAP software. Portions created by gSOAP are Copyright (C) 2001-2004 Robert A. van Engelen, Genivia inc. All Rights Reserved. THE SOFTWARE IN THIS PRODUCT WAS IN PART PROVIDED BY GENIVIA INC AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE."

16.2.15 Chromium licenses

16.2.15.1 Main license

Copyright (c) 2002, Stanford University All rights reserved.

Some portions of Chromium are copyrighted by individiual organizations. Please see the files COPYRIGHT.LLNL and COPYRIGHT.REDHAT for more information.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Stanford University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL

DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

16.2.15.2 COPYRIGHT.LLNL file

This Chromium distribution contains information and code which is covered under the following notice:

Copyright (c) 2002, The Regents of the University of California. Produced at the Lawrence Livermore National Laboratory For details, contact: Randall Frank (rjfrank@llnl.gov). UCRL-CODE-2002-058 All rights reserved.

This file is part of Chromium. For details, see accompanying documentation.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the disclaimer below.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the disclaimer (as noted below) in the documentation and/or other materials provided with the distribution.

Neither the name of the UC/LLNL nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OF THE UNIVERSITY OF CALIFORNIA, THE U.S. DEPARTMENT OF ENERGY OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Additional BSD Notice

1. This notice is required to be provided under our contract with the U.S. Department of Energy (DOE). This work was produced at the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-ENG-48 with the DOE.

2. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately-owned rights.

3. Also, reference herein to any specific commercial products, process, or services by trade name, trademark, manufacturer or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

16.2.15.3 COPYRIGHT.REDHAT file

This Chromium distribution contains information and code which is covered under the following notice:

Copyright 2001,2002 Red Hat Inc., Durham, North Carolina.

All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation on the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice (including the next paragraph) shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. IN NO EVENT SHALL RED HAT AND/OR THEIR SUPPLIERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

16.2.16 curl license

COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1996 - 2009, Daniel Stenberg, daniel@haxx.se.

All rights reserved.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

16.2.17 libgd license

Portions copyright 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002 by Cold Spring Harbor Laboratory. Funded under Grant P41-RR02188 by the National Institutes of Health.

Portions copyright 1996, 1997, 1998, 1999, 2000, 2001, 2002 by Boutell.Com, Inc.

Portions relating to GD2 format copyright 1999, 2000, 2001, 2002 Philip Warner.

Portions relating to PNG copyright 1999, 2000, 2001, 2002 Greg Roelofs.

Portions relating to gdttf.c copyright 1999, 2000, 2001, 2002 John Ellson (ellson@lucent.com).

Portions relating to gdft.c copyright 2001, 2002 John Ellson (ellson@lucent.com).

Portions copyright 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007 Pierre-Alain Joye (pierre@libgd.org).

Portions relating to JPEG and to color quantization copyright 2000, 2001, 2002, Doug Becker and copyright (C) 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, Thomas G. Lane. This software is based in part on the work of the Independent JPEG Group. See the file README-JPEG.TXT for more information.

Portions relating to WBMP copyright 2000, 2001, 2002 Maurice Szmurlo and Johan Van den Brande.

Permission has been granted to copy, distribute and modify gd in any context without fee, including a commercial application, provided that this notice is present in user-accessible supporting documentation.

This does not affect your ownership of the derived work itself, and the intent is to assure proper credit for the authors of gd, not to interfere with your productive use of gd. If you have questions, ask. "Derived works" includes all programs that utilize the library. Credit must be given in user-accessible documentation.

This software is provided "AS IS." The copyright holders disclaim all warranties, either express or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose, with respect to this code and accompanying documentation.

Although their code does not appear in gd, the authors wish to thank David Koblas, David Rowley, and Hutchison Avenue Software Corporation for their prior contributions.

16.2.18 BSD license from Intel

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the Intel Corporation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

16.2.19 libjpeg License

The authors make NO WARRANTY or representation, either express or implied, with respect to this software, its quality, accuracy, merchantability, or fitness for a particular purpose. This software is provided "AS IS", and you, its user, assume the entire risk as to its quality and accuracy.

This software is copyright (C) 1991-2010, Thomas G. Lane, Guido Vollbeding. All Rights Reserved except as specified below.

Permission is hereby granted to use, copy, modify, and distribute this software (or portions thereof) for any purpose, without fee, subject to these conditions:

(1) If any part of the source code for this software is distributed, then this README file must be included, with this copyright and no-warranty notice unaltered; and any additions, deletions, or changes to the original files must be clearly indicated in accompanying documentation.

(2) If only executable code is distributed, then the accompanying documentation must state that "this software is based in part on the work of the Independent JPEG Group".

(3) Permission for use of this software is granted only if the user accepts full responsibility for any undesirable consequences; the authors accept NO LIABILITY for damages of any kind.

These conditions apply to any software derived from or based on the IJG code, not just to the unmodified library. If you use our work, you ought to acknowledge us.

Permission is NOT granted for the use of any IJG author's name or company name in advertising or publicity relating to this software or products derived from it. This software may be referred to only as "the Independent JPEG Group's software".

We specifically permit and encourage the use of this software as the basis of commercial products, provided that all warranty or liability claims are assumed by the product vendor.

ansi2knr.c is included in this distribution by permission of L. Peter Deutsch, sole proprietor of its copyright holder, Aladdin Enterprises of Menlo Park, CA. ansi2knr.c is NOT covered by the above copyright and conditions, but instead by the usual distribution terms of the Free Software Foundation; principally, that you must include source code if you redistribute it. (See the file ansi2knr.c for full details.) However, since ansi2knr.c is not needed as part of any program generated from the IJG code, this does not limit you more than the foregoing paragraphs do.

The Unix configuration script "configure" was produced with GNU Autoconf. It is copyright by the Free Software Foundation but is freely distributable. The same holds for its supporting scripts (config.guess, config.sub, ltmain.sh). Another support script, install-sh, is copyright by X Consortium but is also freely distributable.

The IJG distribution formerly included code to read and write GIF files. To avoid entanglement with the Unisys LZW patent, GIF reading support has been removed altogether, and the GIF writer has been simplified to produce "uncompressed GIFs". This technique does not use the LZW algorithm; the resulting GIF files are larger than usual, but are readable by all standard GIF decoders.

We are required to state that

"The Graphics Interchange Format(c) is the Copyright property of CompuServe Incorporated. GIF(sm) is a Service Mark property of CompuServe Incorporated."

16.2.20 x86 SIMD extension for IJG JPEG library license

Copyright 2009 Pierre Ossman <ossman@cendio.se> for Cendio AB

Copyright 2010 D. R. Commander

Based on

x86 SIMD extension for IJG JPEG library - version 1.02

Copyright (C) 1999-2006, MIYASAKA Masaru.

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

17 VirtualBox privacy policy

Policy version 4, Apr 22, 2010

This privacy policy sets out how Oracle Corporation ("Oracle") treats personal information related to the virtualbox.org website and the VirtualBox application.

§ 1 virtualbox.org. The "virtualbox.org" website logs anonymous usage information such as your IP address, geographical location, browser type, referral source, length of visit and number of page views while you visit (collectively, "anonymous data"). In addition, but only if you choose to register, the website's bug tracking and forum services store the data you choose to reveal upon registration, such as your user name and contact information.

§ 2 Cookies. The virtualbox.org website, the bug tracker and the forum services use cookies to identify and track the visiting web browser and, if you have registered, to facilitate login. Most browsers allow you to refuse to accept cookies. While you can still visit the website with cookies disabled, logging into the bug tracker and forum services will most likely not work without them.

§ 3 VirtualBox registration process. The VirtualBox application may ask that the user optinoally register with Oracle. in der If you choose to register, your name, e-mail address, country and company will be submitted to Oracle and stored together with the IP address of the submitter as well as product version and platform being used. The standard Oracle Privacy Policies as posted on http://www.oracle.com/html/privacy.html apply to this data.

§ 4 Update notifications. The VirtualBox application may contact Oracle to find out whether a new version of VirtualBox has been released and notify the user if that is the case. In the process, anonymous data such as your IP address and a non-identifying counter, together with the product version and the platform being used, is sent so that the server can find out whether an update is available. By default, this check is performed once a day. You change this interval or disable these checks altogether in the VirtualBox preferences.

§ 5 Usage of personal information. Oracle may use anonymous and personal data collected by the means above for statistical purposes as well as to automatically inform you about new notices related to your posts on the bug tracker and forum services, to administer the website and to contact you due to technical issues. Oracle may also inform you about new product releases related to VirtualBox.

In no event will personal data without your express consent be provided to any third parties, unless Oracle may be required to do so by law or in connection with legal proceedings.

§ 6 Updates. Oracle may update this privacy policy by posting a new version on the virtualbox.org website. You should check this page occasionally to ensure you are happy with any changes.

Glossary

A

ACPI Advanced Configuration and Power Interface, an industry specification for BIOS and hardware extensions to configure PC hardware and perform power management. Windows 2000 and higher as well as Linux 2.4 and higher support ACPI. Windows can only enable or disable ACPI support at installation time.

AHCI Advanced Host Controller Interface, the interface that supports SATA devices such as hard disks. See chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78.

AMD-V The hardware virtualization features built into modern AMD processors. See chapitre 10.3, *Hardware vs. software virtualization*, page 175.

API Application Programming Interface.

APIC Advanced Programmable Interrupt Controller, a newer version of the original PC PIC (programmable interrupt controller). Most modern CPUs contain an on-chip APIC ("local APIC"). Many systems also contain an I/O APIC (input output APIC) as a separate chip which provides more than 16 IRQs. Windows 2000 and higher use a different kernel if they detect an I/O APIC during installation. Therefore an I/O APIC must not be removed after installation.

ATA Advanced Technology Attachment, an industry standard for hard disk interfaces (synonymous with IDE). See chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78.

B

BIOS Basic Input/Output System, the firmware built into most personal computers which is responsible of initializing the hardware after the computer has been turned on and then booting an operating system. VirtualBox ships with its own virtual BIOS that runs when a virtual machine is started.

C

COM Microsoft Component Object Model, a programming infrastructure for modular software. COM allows applications to provide application programming interfaces which can be accessed from various other programming languages and applications. VirtualBox makes use of COM both internally and externally to provide a comprehensive API to 3rd party developers.

D

DHCP Dynamic Host Configuration Protocol. This allows a networking device in a network to acquire its IP address (and other networking details) automatically, in order to avoid having to configure all devices in a network with fixed IP addresses. VirtualBox has a built-in DHCP server that delivers an IP addresses to a virtual machine when networking is configured to NAT; see chapitre 6, *Réseau virtuel*, page 91.

DKMS Dynamic Kernel Module Support. A framework that simplifies installing and updating external kernel modules on Linux machines; see chapitre 2.3.2, *Le module noyau de VirtualBox*, page 35.

E

EFI Extensible Firmware Interface, a firmware built into computers which is designed to replace the aging BIOS. Originally designed by Intel, most modern operating systems can now boot on computers which have EFI instead of a BIOS built into them; see chapitre 3.12, *Firmware alternatif (EFI)*, page 56.

EHCI Enhanced Host Controller Interface, the interface that implements the USB 2.0 standard.

G

GUI Graphical User Interface. Commonly used as an antonym to a "command line interface", in the context of VirtualBox, we sometimes refer to the main graphical `VirtualBox` program as the "GUI", to differentiate it from the `VBoxManage` interface.

GUID See UUID.

I

IDE Integrated Drive Electronics, an industry standard for hard disk interfaces. See chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78.

I/O APIC See APIC.

iSCSI Internet SCSI; see chapitre 5.11, *Serveurs iSCSI*, page 90.

M

MAC Media Access Control, a part of an Ethernet network card. A MAC address is a 6-byte number which identifies a network card. It is typically written in hexadecimal notation where the bytes are separated by colons, such as `00:17:3A:5E:CB:08`.

MSI Message Signalled Interrupts, as supported by modern chipsets such as the ICH9; see chapitre 3.4.1, *Onglet Carte mère*, page 47. As opposed to traditional pin-based interrupts, with MSI, a small amount of data can accompany the actual interrupt message. This reduces the amount of hardware pins required, allows for more interrupts and better performance.

N

NAT Network Address Translation. A technique to share networking interfaces by which an interface modifies the source and/or target IP addresses of network packets according to specific rules. Commonly employed by routers and firewalls to shield an internal network from the Internet, VirtualBox can use NAT to easily share a host's physical networking hardware with its virtual machines. See chapitre 6.3, *Network Address Translation (NAT)*, page 93.

O

OVF Open Virtualization Format, a cross-platform industry standard to exchange virtual appliances between virtualization products; see chapitre 1.12, *Importer et exporter des machines virtuelles*, page 29.

P

PAE Physical Address Extension. This allows accessing more than 4 GB of RAM even in 32-bit environments; see chapitre 3.3.2, *Onglet Avancé*, page 46.

PIC See APIC.

PXE Preboot Execution Environment, an industry standard for booting PC systems from remote network locations. It includes DHCP for IP configuration and TFTP for file transfer. Using UNDI, a hardware independent driver stack for accessing the network card from bootstrap code is available.

R

RDP Remote Desktop Protocol, a protocol developed by Microsoft as an extension to the ITU T.128 and T.124 video conferencing protocol. With RDP, a PC system can be controlled from a remote location using a network connection over which data is transferred in both directions. Typically graphics updates and audio are sent from the remote machine and keyboard and mouse input events are sent from the client. A VirtualBox extension package by Oracle provides VRDP, an enhanced implementation of the relevant standards which is largely compatible with Microsoft's RDP implementation. See chapitre 7.1, *Remote display (VRDP support)*, page 101 for details.

S

SAS Serial Attached SCSI, an industry standard for hard disk interfaces. See chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78.

SATA Serial ATA, an industry standard for hard disk interfaces. See chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78.

SCSI Small Computer System Interface. An industry standard for data transfer between devices, especially for storage. See chapitre 5.1, *Contrôleurs de disques durs : IDE, SATA (AHCI), SCSI, SAS*, page 78.

SMP Symmetrical Multiprocessing, meaning that the resources of a computer are shared between several processors. These can either be several processor chips or, as is more common with modern hardware, multiple CPU cores in one processor.

SSD Solid-state drive, uses microships for storing data in a computer system. Compared to classical hard-disks they are having no mechanical components like spinning disks.

T

TAR A widely used file format for archiving. Originally, this stood for "Tape ARchive" and was already supported by very early Unix versions for backing up data on tape. The file format is still widely used today, for example, with OVF archives (with an `.ova` file extension); see chapitre 1.12, *Importer et exporter des machines virtuelles*, page 29.

U

UUID A Universally Unique Identifier – often also called GUID (Globally Unique Identifier) – is a string of numbers and letters which can be computed dynamically and is guaranteed to be unique. Generally, it is used as a global handle to identify entities. VirtualBox makes use of UUIDs to identify VMs, Virtual Disk Images (VDI files) and other entities.

V

VM Virtual Machine – a virtual computer that VirtualBox allows you to run on top of your actual hardware. See chapitre 1.2, *Un peu de terminologie*, page 11 for details.

VMM Virtual Machine Manager – the component of VirtualBox that controls VM execution. See chapitre 10.2, *VirtualBox executables and components*, page 173 for a list of VirtualBox components.

VRDE VirtualBox Remote Desktop Extension. This interface is built into VirtualBox to allow VirtualBox extension packages to supply remote access to virtual machines. A VirtualBox extension package by Oracle provides VRDP support; see chapitre 7.1, *Remote display (VRDP support)*, page 101 for details.

VRDP See RDP.

VT-x The hardware virtualization features built into modern Intel processors. See chapitre 10.3, *Hardware vs. software virtualization*, page 175.

X

XML The eXtensible Markup Language, a metastandard for all kinds of textual information. XML only specifies how data in the document is organized generally and does not prescribe how to semantically organize content.

XPCOM Mozilla Cross Platform Component Object Model, a programming infrastructure developed by the Mozilla browser project which is similar to Microsoft COM and allows applications to provide a modular programming interface. VirtualBox makes use of XPCOM on Linux both internally and externally to provide a comprehensive API to third-party developers.