grub2

multiboot bootloader

Brièvement, un boot loader est le premier logiciel qui est lancé au démarrage de l'ordinateur. Il est responsable du chargement et de transférer le contrôle à un kernel. Le kernel, en retour, initialise le reste du système d'exploitation.

GRUB est un boot loader très puissant, qui peut charger une grande variété de systèmes d'exploitation. GRUB est conçu pour adresser la complexité du boot d'un ordinateur personnel.

Une des fonctionnalité les plus importantes est sa flexibilité ; GRUB comprend les systèmes de fichier et les formats exécutable de kernel, ce qui permet de charger un kernel sans enregistrer sa position physique sur le disque.

En bootant avec GRUB, vous pouvez utiliser soit une interface en ligne de commande, ou une interface. En utilisant la ligne de commande, il suffit de spécifier l'emplacement du dique et le nom de fichier du kernel manuellement. Dans l'interface par menu, il suffit de sélectionner un OS en utilisant les touches. Le menu est basé sur un fichier de configuration qui peut être préparé avant. Il est possible de basculer d'un mode à l'autre, et même éditer les entrées du menu avant de les utiliser.

Fonctionnalités

- Reconnaissante de plusieurs formats d'exécutable
- Supporte de nombreuses variantes de a.out, plus ELF. Les tables de symbole sont également chargés.
- Support de kernels non-multiboot
- Supporte de nombreux kernels 32bits qui ne sont pas multiboot compliant (FreeBSD, NetBSD, OpenBSD, et Linux).
- Charger plusieurs modules
- Support complet du chargement de modules.
- Charger un fichier de configuration
- Support de fichiers de configuration au format texte avec des commandes de boot. Il est également possible de charger un autre fichier de configuration dynamiquement et embarque un fichier de configuration preset dans un fichier image grub. La liste des commandes sont un superset de ces lignes de commandes supportées.
- Fournis une interface par menu
- Une interface à menu avec un timeout programmable est disponible. Il n'y a pas de limite fixée sur le nombre d'entrées de boot, et l'implémentation actuelle a l'espace pour des centaines.
- Interface en ligne de commande flexible : Une interface flexible, accessible depuis le menu, est disponible pour éditer les commandes, ou écrire une nouvelle commande. Si aucun fichier de configuration n'est présent, GRUB supprime la ligne de commande.
- Suporte plusieurs systèmes de fichier : AFFS, AtheOs Fs, BeFS, BtrFS, cpio, ext2/3/4, DOS, ISO9660, JFS, Minix Fs, NTFS, ReiserFS, ROMFS, SFS, Squash4, tar, UDF, BSD UFS, XFS et ZFS.
- Décompression automatique : Peut décompresser des fichiers compressés avec gzip ou xz. Cette fonction est automatique et transparente à l'utilisateur.
- Accède aux données sur un périphérique installé
- Supporte la lecture des données depuis des disquettes ou disques dur reconnus par le BIOS
- Indépendant des translation de géometrie du disque. À la différence de nombreux boot loader, GRUB gère les géometries de disque de manière transparente.
- Detecte toute la RAM installée. GRUB peut généralement trouver toute la RAM installée sur une machine compatible PC. Il utiliser des techniques de requête du BIOS pour trouver toutes les régions mémoire.

- Support du mode LBA. GRUB détecte automatiquement si le mode LBA est accessible et l'utilise si disponible.
- Support du boot réseau
- GRUB est un bootloader basé sur le disque mais supporte également le réseau, en utilisant TFTP
- Support de terminaux distants
- Pour supporter des machines sans console, GRUB fournis un terminal distant. Seul le terminal série est implémenté actuellement.

Convention de nommage

La syntaxe de périphérique utilisée dans GRUB est un peu différent de ce qui est vu dans un OS, et il est nécessaire de spécifier un couple disque/partition. l'exemple :

(fd0)

Avant tout, GRUB exige que le nom du périphérique soit entre parenthèses. fd signifie une disquette. Le numéro 0 est le numéro du disque, qui est compté en partant de 0. Cette expression indique toute la disquette.

(hd0,msdos2)

Ici, hd signifie un disque dur. 0 indique le numéro du disque. msdos indique le type de partition, et 2 indique e numéro de partition. Les numéros de partitions sont comptés à partir de 1, et non 0. Cette expression spécifie la 2ème partition du premier disque. Dans ce cas, GRUB utilise une partition du disque, au lieu du disque entier.

(hd0,msdos5)

Cela spécifie la première partition étendue du disque dur. Noter que les numéros de partition pour les partitions étendues sont comptés à partir de 5, sans regarder le nombre de partition primaire sur le disque.

(hd1,msdos1,bsd1)

Signifie la partition a de BSD sur le premier slice du second disque.

Noter que GRUB ne fait pas de distinction entre IDE et SCSI. Il compte simplement les numéros de disque de 0, sans regarder leur type. Normalement, un disque IDE est inférieur au numéro d'un disque SCSI, bien que ce n'est pas vrai si l'on change la séquence de boot en inversant les disque IDE et SCSI dans le BIOS.

Maintenant, pour spécifier un fichier on utilisera la syntaxe :

(hd0,msdos1)/vmlinuz

Cela spécifie le fichier nommé 'vmlinuz', trouvé sur la première partition du premier disque dur.

Installation

Pour installer GRUB comme boot loader, il faut d'abord installer GRUB et les utilitaires sous un OS type UNIX. Une fois installé, la commande grub-install install le bootloader sur le disque.

GRUB est fournis avec des images de boot, qui sont généralement dans /usr/lib/grub/<cpu>-<plateform> ; c'est le répertoire d'image, et le répertoire où le boot loader doit les trouver (généralement /boot) est le répertoire de boot.

Pour installer GRUB sous un OS type Unix, invoquer le programme grub-install en root. L'utilisation est très simple. Vous devez simplement spécifier un argument au programme, où installer le boot loader. L'argument doit être un fichier de périphérique (par ex : /dev/hda). l'option –boot-directory spécifie le répertoire de boot (/boot par défaut).

Certains bios ont un bug qui exposent la première partition d'un disque usb comme disquette au lieu de l'exposer comme disque usb, il est nécessaire dans ce cas d'utiliser :

losetup /dev/loop0 /dev/sdb1
mount /dev/loop0 /mnt/usb
grub-install –boot-directory=/mnt/usb/bugbios –force –allow-floppy /dev/loop0

Noter que grub-install est actuellement un simple script shell et le vrai travail est fait pas grub-mkimage et grub-setup.

Créer un CDROM bootable

GRUB supporte le mode sans émulation dans la spécification El Torito. Celà signifie qu'on peut utiliser tout le CDROM depuis GRUB sans avoir à créer un fichier image disquette ou disque dur, qui peut créer des problèmes de compatibilité.

Pour booter depuis un CD-ROM, GRUB utilise une image spéciale appelée cdboot.img, qui est concaténé avec core.img. Le core.img utilisé pour cela devrait être construit avec au moins les modules iso9660 et biosdisk. Le CDROM bootable va généralement nécessiter également un fichier de configuration grub.cfg et quelques autres modules.

Pour créer un CD de récupération GRUB simple, utiliser la commande grub-mkrescue :
grub-mkrescue -o grub.iso
Pour inclure d'autres fichiers à l'image, créer un répertoire
mkdir iso
créer un répertoire pour GRUB :
makd -p iso/boot/grub
Si souhaité, créer le fichier de configuration grub.cfg et copier les fichier et répertoires pour le disque dans iso/. Enfin, créer l'image :
grub-mkrescue -o grub.iso iso
Le périphérique root sera définis de manière approprié en entrant le fichier de configuration, pour référrer aux noms de fichier dans le cd sans avoir à utiliser de nom de périphérique expicit. Celà simplifie la production d'images qui fonctionnent sur les disques optiques et les périphériques de stockage USB.

Mappage entre les disques BIOS et les périphériques OS

Si le fichier de map de périphérique existe, les utilitaires GRUB (grub-probe, grub-setup, etc) le lise pour mapper les périphériques BIOS aux périphériques OS. Ce fichier consiste de ligne sous la forme : **(device) file**

device est un disque spécifié dans la syntaxe GRUB, et file est un fichier OS, qui est normalement un fichier de périphérique.

Historiquement, le fichier de mappage de périphérique a été utilisé parce que les noms de périphérique GRUB devaient être utilisés dans le fichier de configuration, et ils étaient dérivés des numéro de lecteur du BIOS. Le mappage entre les disque BIOS et OS ne peut pas toujours être deviné correctement : par exemple GRUB obtiendra le mauvais ordre si on échange la séquence de boot entre IDE et SCSI dans le BIOS.

Malheureusement, même les noms de périphérique OS ne sont pas toujours stable. Les versions moderne de Linux peuvent sonder les disques dans un ordre différent d'un boot à l'autre, et le préfixe (/dev/hd* vs /dev/sd*) peut changer en fonction du pilote utilisé. Cela impose d'éditer fréquemment le fichier de map.

GRUB évite ce problème en utilisant les UUID ou les labels de système de fichier en générant grub.cfg, et il est recommandé de faire de même pour toute entrée ajoutée manuellement. Si le fichier de map n'existe pas, les utilitaires GRUB assument un map temporaire à la volée. C'est suffisant, en particulier pour les système à un seul disque.

Cependant, le fichier de map de périphérique n'est pas entièrement obsolète, et il est utilisé pour écraser un environnement courant différent de celui du boot. Un cas courant est l'utilisation d'une partition ou d'un volume logique comme disque pour une machine virtuelle.

Installation BIOS

Le format de la table de partition traditionnellement utilisé pour les plateformes PC BIOS est appelée le format MBR ; c'est le format qui autorise jusqu'à 4 partitions primaire et des partitions logiques additionnelles. Avec ce format de table de partition il y a 2 manières d'installer GRUB : Il peut être embarqué dans la zone entre le MBR et la première partition, (généralement 31KiB, appelée la zone embarquée), ou l'image peut être installé dans un système de fichier et une liste de blocks le forme peuvent être stockés dans le premier secteur de cette partition.

Chaque méthode a ses inconvénients. Il n'y a pas de manière de réserver de l'espace dans la zone embarquée de manière sûre, et certains logiciels propriétaire sont connus pour l'utiliser ; et les systèmes sont parfois partitionnés sans laisser suffisamment d'espace avant la première partition. D'un autre coté, installer dans un système de fichier signifie que GRUB est vulnérable si ses blocks sont déplacés dans le système de fichiers tel que fsck, donc cette approche est fragile, et peut seulement être utilisée si le système /boot est sur le même disque que le BIOS boot.

Les systèmes récents utilisent le format GPT. C'est spécifié dans EFI, mais peut également être utilisé sur les plateformes BIOS si le système le supporte. Par exemple, GRUB et GNU/Linux peuvent être utilisé dans cette configuration. Avec ce format, il est possible de réserver une partition entière pour GRUB, appelée la partition de Boot. GRUB peut ainsi être embarquée dans cette partition sant rique d'être écrasé par un autre logiciel et dans être contenu dans un système de fichier.

En créant un BIOS Boot Partition dans un système GPT, on doit s'assurer qu'il y a au moins 31KiB d'espace disponible. Il faut également s'assurer qu'il a le type de partition correcte. En utilisant parted :

Boot

Grub peut charger les kernels conforme au Multiboot de manière consistante mais pour certains OS, il faut utiliser une méthode spécifique.

booter un OS

GRUB a 2 méthodes de boot distinct. Une des 2 est de charger un OS directement, et l'autre est de chaîner le loader qui va ainsi charger l'OS.

Multiboot est le format natif supporté par GRUB. Il supporte également Linux. Cependant, DOS et Windows ont certains problèmes.

Pour les OS qui ne supportent pas Multiboot et n'ont pas de support spécifique dans GRUB, le chain-loader doit être utilisé, ce qui implique qu'un autre loader est chargé.

La commande chainloader est utilisée pour définir ce mode. Il est normalement nécessaire de charger certains modules GRUB et définir le périphérique root de manière approprié. Pour un système Windows, une entrée peut ressembler à :

```
menuentry "Windows" {
 insmod chain
 insmod ntfs
 set root=(hd0,1)
 chainloader +1
}
```

Sur les système avec plusieurs disques dur, une travail supplémentaire doit être fournis. Le chain-Loading est seulement supporté sur les PC BIOS et les plateformes EFI.

Boot Loopback

Grub est capable de lire une image (CD ou HDD) stocké dans un stockage accessible. Cependant l'OS lui-même devrait être capable de trouver son root. Cela implique généralement de lancer un programme dans l'espace utilisateur avant que le vrai root soit trouvé. GRUB peut charger une petite image faite spécialement en ram en la passant au kernel, avec les commandes kfreebsd_module, knetbsd_module_elf, kopenbsd_ramdisk, initrd, initrd16, multiboot_module, multiboot2_module, ou xnu_ramdisk, en fonction du chargeur.

Problèmes spécifiques à l'OS

Vu que GNU/Hurd est conforme à Multiboot, il est facile à booter. Il ne faut cependant pas oublier de spécifier la partition root du kernel.

1. Spécifier le périphérique root de GRUB sur le même disque que celui de GNU/Hurd. La commande search –set=root –file /boot/gnumach.gz ou similaire peut aider.
2. Charger le kernel et les modules :

multiboot /boot/gnumach.gz root=device :hd0s1

module /hurd/ext2fs.static ext2fs –readonly –multiboot-command-line='${kernel-command-line}' –host-priv-port='${host-port}' –de

module /lib/ld.so.1 exec /hurd/exec '$(exec-task=task-create)'

3. Finalement, lancer la commande boot.

GNU/Linux

Il est relativement simple à booter, parce qu'il ressemble au boot des OS conformes Multiboot.

1. Définir le périphérique root de grub au même disque que le root de Linux.
2. Charger le kernel avec la commande : linux /vmlinuz root=/dev/sda1. Si nécessaire ajouter des paramètres au kernel : linux /vmlinuz root=/dev/sda1 acpi=off
3. Si besoin d'un initrd, utiliser la commande initrd : initrd /initrd
4. Finalement lancer la commande boot.

Fichier de configuration

GRUB est configuré en utilisant grub.cfg, généralement dans /boot/grub. Ce fichier est flexible, mais beaucoups d'utilisateurs n'ont pas besoin d'écrire tout à la main.

Configuration simple

Le programme grub-mkconfig génère grub.cfg correctement dans la plupart des cas. Il est prévu pour mettre à jours une distribution, et découvre les kernels disponible et tente de générer des entrées.

grub-mkconfig a quelques limitations. En ajoutant des entrées manuellement à la fin de la liste en éditant /etc/grub.d/40_custom, ou en créant /boot/grub/custom.cfg, changer l'ordre des entrées du menu ou changer les scripts dans /etc/grub.d ou changer leur nom peut rendre les changements plus complexe. Cela peut être améliorer dans le future. Il est préférable dans ce cas d'éditer directement grub.cfg et de désactiver l'utilisation automatique de grub-mkconfig du système.

Le fichier /etc/default/grub contrôle les opération de grub-mkconfig. Il est lu par un script shell, et doit être une entrée de shell POSIX valide ; normalement c'est simplement une séquence de lignes KEY=value ; mais si la valeur contient des espace ou des caractères spéciaux, ils doivent être entre guillemets.

GRUB_DEFAULT Entrée du menu par défaut. Peut être un nombre (la nième entrée) dans le menu, en commançant à 0, ou le titre d'une entrée du menu, ou la chaîne spéciale "saved", qui est l'entrée sauvée par GRUB_SAVEDEFAULT, grub-set-default, ou grub-reboot. Défaut : 0

GRUB_TIMEOUT Boot sur le défaut au boot de n secondes. Défaut : 5. 0 boot immédiatement sans afficher le menu, et -1 attend indéfiniment

GRUB_HIDDEN_TIMEOUT Attend n secondes pour une touche avant d'afficher le menu. Avec GRUB_TIMEOUT à 0, le menu n'est jamais affiché sauf si une touche est pressée. Défaut : non-définis

GRUB_HIDDEN_TIMEOUT_QUIET Supprime le décompte de GRUB_HIDDEN_TIMEOUT. non définis par défaut.

GRUB_DEFAULT_BUTTON

GRUB_TIMEOUT_BUTTON

GRUB_HIDDEN_TIMEOUT_BUTTON

GRUB_BUTTON_CMOS_ADDRESS Variantes des variables correspondantes sans le suffix _BUTTON, utilisés pour supporter les boutons de mise sous tension spécifique au vendeur.

GRUB_DISTRIBUTOR Définis par les distributeurs de GRUB pour identifier leur nom. Utilisé pour générer des titre d'entrée de menu plus informatifs

GRUB_TERMINAL_INPUT Sélectionne le périphérique d'entrée du terminal. les noms dépendent de la plateform, mais peut inclure console (PC Bios et consoles EFI), serial, ofconsole (OpenFirmware Console), at_keyboard (PC AT keyboard utilisant le protocole de boot HID), ou usb_keyboard (clavier USB utilisant le protocole de boot HID). Défaut : utilise l'entrée de terminal natif de la plateforme.

GRUB_TERMINAL_OUTPUT Sélectionne le périphérique de sortie du terminal. console, serial, gfxterm, ofconsole, ou vga_text. Défaut : utilise la sortie de terminal native de la plateforme.

GRUB_TERMINAL Si définis, remplace GRUB_TERMINAL_INPUT et GRUB_TERMINAL_OUTPUT.

GRUB_SERIAL_COMMAND Une commande pour configurer le port série pour utiliser la console série. Défaut : serial

GRUB_CMDLINE_LINUX Arguments à ajouter aux entrées de menu pour le kernel Linux

GRUB_CMDLINE_LINUX_DEFAULT Sauf si GRUB_DISABLE_RECOVERY est définis, 2 entrées de menu sont générés pour chaque kernel linux : une entrée par défaut, et une entrée pour le mode récupération. Cette option liste les arguments à ajouter seulement à l'entrée par défaut, après ceux listés dans GRUB_CMDLINE_LINUX.

GRUB_DISABLE_LINUX_UUID Normalement, grub-mkconfig génère des entrées de menu qui utilisent des UUID pour identifier le système de fichie racine pour le kernel Linux en utilisant "root=UUID=...". Pour désactiver l'utilisation des UUID, définir cette option à true.

GRUB_DISABLE_RECOVERY À true, désactive la génération des entrées de menu pour le mode récupération

GRUB_VIDEO_BACKEND Si le support vidéo graphique est requis, soit parce que le terminal graphique gfxterm est utilisé ou parce que GRUB_GFXPAYLOAD_LINUX est définis, grub-mkconfig va normalement charger tous les pilotes vidéo GRUB et utiliser le plus approprié pour votre hardware. Cette option permet de forcer des paramètres spécifique. Une fois grub-install lancé, les pilotes disponible sont listés dans /boot/grub/video.lst.

GRUB_GFXMODE Définis la résolution utilisée dans le terminal graphique gfxterm. Noter que vous pouvez seulement utiliser les modes que votre carte supporte via les extensions VESA BIOS (VBE), donc pour les résolution d'écran LCD peuvent ne pas être disponible. Le défaut est auto, qui tente de sélectionner la résolution préférrée.

GRUB_BACKGROUND Définis l'image à utiliser avec le terminal graphique gfxterm. Ce fichie doit être accéssible au moment du boot et doit se terminer avec .png, .tga, .jpg, ou .jpeg. L'image sera redimensionné si besoin.

GRUB_THEME Définis un thème à utiliser avec le terminal graphique gfxterm

GRUB_GFGPAYLOAD_LINUX à text, force Linux à booter en mode text normal. à keep, préserve le mode graphique définis avec GRUB_GFXMODE.

GRUB_DISABLE_OS_PROBER Normalement, grub-mkconfig tente d'utiliser le programme os-prober, si installé, pour découvrir les autres OS installés sur le système et générer les entrées de menu appropriés pour eux. à true, désactive cette recherche.

GRUB_INIT_TUNE Joue un son dans les haut-parleurs quand GRUB démarre. Utile pour les utilisateurs ne pouvant pas voir l'écran. La valeur de cette option est passée directement à play.

GRUB_BARAM Si mis, GRUB émet une commande badram pour filtrer des régions mémoire.

GRUB_PRELOAD_MODULES Cette option peut être mis à une liste de modules GRUB séparés par des espaces. Chaque module sera chargé le plus tôt possible, au début de grub.cfg.

Pour une personnalisation plus détaillée de la sortie de /etc/grub.d/40_custom est particulièrement utile pour ajouter des entrées particulièrement utiles pour ajouter des entréess

Ecrire des fichiers de configuration directement

grub.cfg est écrit dans la langage de scripting intégré à GRUB, qui a une syntaxe similaire à BASH et d'autres shells.

MOTS Un mot est une séquence de caractères considérés comme une unité simple. Les mots sont séparés par des métacaractères, qui sont les suivane, plus espace, tabulation et nouvelle ligne.

{ } | & $; > < les guillemets peuvent être utilisés pour inclure les métacaractères dans les mots.

Mots réservés

Les mots réservés ont une signification spéciale pour GRUB. Les mots suivants sont reconus :

```
! [[ ]] { }
case do done elif else esac fi for function
if in menuentry select then time until while
```

guillemets

Les guillemets sont utilisés pour supprimés la signification spéciale de certains caractères ou mots. Il peut être utilisé pour traiter les métacaractères comme partie d'un mot. Il y a 3 mécanismes de guilemets : le caractère d'échappement, guillemets simple et guillemets double.

expansion de variables

Le caractère '$' introduit l'expansion de variable. Le nom de la variable à étendre peut être entre crochets, qui servent optionnellement à protéger la variable à étendre des caractères qui suivent immédiatement.

Les noms de variables normales commencent avec un caractère alphanumérique, suivi par 0 ou plusieurs caractères alphanumérique. Ces noms réfèrent aux entrées dans l'environnement GRUB.

Les noms de variables positionnels consistent en 1 ou plusieurs chiffres. Ils représentent les paramètres passés aux appels de fonction, avec $1 représentant le premier paramètre.

Le nom de variable spéciale ? s'étend au code de sortie de la dernière commande exécutée. Quand les noms de variable positionnels sont actifs, d'autres variables @, *, et # sont définis et s'étendent à tous les paramètres positionnels avec des guillemets nécessaire, sans guillemets, et comme compteur de paramètres, respectivement.

Commandes simple

Une commande simple est une séquence de mots séparés par des espaces ou des tabulation et terminés par un ' ;' ou un newline. le premier mot spécifie la commande à exécuter. Les autres mots sont passés en argument à la commande.

La valeur de retour d'une commande simple est sont status de sortie. Si le mot réservé ! précède la commande, la valeur de retour est un NOT du status de sortie de la commande.

Commandes composés

Une commande composée est une des suivantes :

```
for name in word ...; do list; done
if list; then list; [elif list; then list;] ... [else list;]fi
while cond; do list; done
until cond; do list; done
function name { command; ...}
menuentry title [-class=class ...][-users=users][-unrestricted][-hotkey=key]{command;...}
```

Commande embarquées

Certaines commande sont également fournis par GRUB pour effectuer des action qui ne seraient pas possible autrement.

```
break[n]
continue[n]
return[n]
shift[n]
```

Configuration manuelle Multiboot

L'autogénération des fichiers de configuration pour les environnements multi-bot dépendent de os-prober. Il est possible de créer soi-même la configuration

Tout d'abord, créer une partition séparée pour GRUB. Certaines options ci-dessous montrent comment charger des images d'installeur d'OS depuis cette partition. Mounter cette partition dans /mnt/boot et désactiver GRUB dans tous les OS et installer manuellement le dernier GRUB compilé avec :

grub-install –boot-directory=/mnt/boot /dev/sda

Dans tous les OS installer les outils GRUB mais désactiver l'installation de GRUB dans le secteur de boot, pour avoir le menu.lst et grub.cfg disponible. Désactiver également l'utilisation d'os-prober en définissant :

GRUB_DISABLE_OS_PROBER=true

Puis écrire un grub.cfg :

```
menuentry "OS using grub2" {
 insmod xfs
 search -set=root -label OS1 -hint hd0,msdos8
 configfile /boot/grub/grub.cfg
}

menuentry "OS using grub2-legacy" {
 insmod ext2
 search -set=root -label OS2 -hint hd0,msdos6
 legacy_configfile /boot/grub/menu.lst
}

menuentry "Windows XP" {
 insmod ntfs
 search -set=root -label WINDOWS_XP -hint hd0,msdos1
 ntldr /ntldr
}

menuentry "Windows 7" {
 insmod ntfs
```

```
 search -set=root -label WINDOWS_7 -hint hd0,msdos2
 ntldr /bootmgr
}

menuentry "FreeBSD" {
 insmod zfs
 search -set=root -label freepool -hint hd0,msdos7
 kfreebsd /freebsd@/boot/kernel/kernel
 kfreebsd_module_elf /freebsd@/boot/kernel/opensolaris.ko
 kfreebsd_module_elf /freebsd@/boot/kernel/zfs.ko
 kfreebsd_module /freebsd@/boot/zfs/zpool.cache type=/boot/zfs/zpool.cache
 set kFreeBSD.vfs.root.mountfrom=zfs:freepool/freebsd
 set kFreeBSD.hw.psm.synaptics_support=1
}

menuentry "experimental GRUB" {
 search -set=root -label GRUB -hint hd0,msdos5
 multiboot /experimental/grub/i386-pc/core.img
}

menuentry "Fedora 16 installer" {
 search -set=root -label GRUB -hint hd0,msdos5
 linux /fedora/vmlinuz lang=en_US keymap=sg resolution=1280x800
 initrd /fedora/initrd.img
}

menuentry "Fedora rawhide installer" {
 search -set=root -label GRUB -hint hd0,msdos5
 linux /fedora/vmlinuz repo=ftp://mirror.switch.ch/mirror/fedora/linux/development/rawhide/x86_64 lang=en_US
keymap=sg resolution=1280x800
 initrd /fedora/initrd.img
}

menuentry "Debian sid installer" {
 search -set=root -label GRUB -hint hd0,msdos5
 linux /debian/dists/sid/main/installer-amd64/current/images/hd-media/vmlinuz
 initrd /debian/dists/sid/main/installer-amd64/current/images/hd-media/initrd.gz
}
```

Embarquer un fichier de configuration dans GRUB

GRUB supporte un fichier de configuration embarqué directement dans l'image core, pour qu'il puisse être chargé avant d'entrée dans le mode normal. C'est utile pour par exemple quand est difficile de trouver le vrai fichier de configuration ou pour débugger un problème avec le chargement de ce fichier. grub-install utilise cette fonctionnalité quand il n'utilise pas de fonction disque du bios ou en installant sur un disque différent de celui contenant /boot/grub, auquel cas il doit utiliser la commande search pour trouver /boot/grub.

Pour embarquer un fichier de configuration, utiliser l'option -c de grub-mkimage. Le fichier est copié dans l'image core.

Une fois embarqué, le fichier de configuration est exécuté, GRUB va charger le module 'normal', qui va lire le fichier de configuration de $prefix/grub.cfg. À ce point, la variable root aura également été définis au nom du périphérique root. Par exemple, prefix peut être définis à '(hd0,1)/boot/grub' et et root peut être définis à 'hd0,1'. Donc, dans la plupart des cas, le fichier de configuration embarqué a seulement besoin des variables root et prefix, et ainsi supprimer le traitement normale de GRUB. Un exemple typique est :

```
search.fs_uuid 01234567-89ab-cdef-0123-456789abcdef root
set prefix=($root)/boot/grub
```

Le module search_fs_uuid doit être inclus dans l'image core pour que cet exemple fonctionne.

Dans des cas plus complexe, il peut être utile pour lire d'autres fichiers de configuration directement depuis le fichier de configuration embarquée. Cela permet de lire des fichiers qui ne s'appellent pas grub.cfg ou lire des fichiers depuis un répertoire autre que celui où les modules GRUB sont installés. Pour faire cela, inclure les modules configfile et normal dans l'image core, et embarquer un fichier de configuration qui utilise la commande configfile pour charger un autre fichier. L'exemple suivant nécessite également les modules echo, search_label, et test à inclure dans l'image core :

```
search.fs_label grub root
if [ -e /boot/grub/example/test1.cfg ]; then
 set prefix=($root)/boot/grub
 configfile /boot/grub/example/test1.cfg
else
 if [ -e /boot/grub/example/test2.cfg ]; then
  set prefix=($root)/boot/grub
  configfile /boot/grub/example/test2.cfg
 else
  echo "Could not find an example configuration file!"
 fi
fi
```

Le fichier de configuration embarqué peut ne pas contenir d'entrées de menu directement.

format du fichier de thème

Le menu graphique GRUB supporte les thèmes qui peuvent être personnalisés. Le thème est configuré via un fichier texte qui spécifie les composant de l'interface graphique (incluant le menu de boot, la barre de progression de timeout, et les messages) ainsi que l'apparence en utilisant les couleurs, fonts, et images. Des exemples sont disponible dans docs/example_theme.txt.

Couleurs

Les couleurs peuvent être spécifié de nombreuses manières :

```
Style HTML #RRGGBB ou #RGB
au format RGB décimal: "128,128,255".
Avec les noms de couleur SVG 1.0 qui doivent être spécifiés en minuscule.
```

Fonts

Les fonts GRUB utilisent le format de fonts bitmap PFF2. Les fonts sont spécifié avec le nom des font. Actuellement il n'y a pas de provision pour une liste de font préférentielle, ou de dériver une font d'une autre. Les fonts sont chargés avec la commande loadfont dans GRUB. Pour voir la liste des fonts chargés, exécuter lsfonts. S'il y a trop de font pour tenir dans l'écran, définir set pager=1 avant lsfonts.

Progress Bar

Les barres de progression sont utilisées pour afficher le temps restant avant que GRUB boot sur l'entrée par défaut. Pour créer une barre de progression qui va afficher le temps restant, simplement créer un composant progress_bar avec l'id __timeout__. Celà indique à GRUB

que la progress bar devrait être mis à jours si le compteur du boot automatique est intérrompu par l'utilisateur.

Les barres de progression peuvent optionnellement avoir un texte affiché. Ce texte est contrôlé par la variable text qui contient un template printf avec le seul argument %d qui est le nombre de secondes restants. Additionnellement les valeurs spéciales "@TIMEOUT_NOTIFICATION_SHORT@", "@TIMEOUT_NOTIFICATION_MIDDLE@", "@TIMEOUT_NOTIFICATION_LONG@" sont remplacées avec les templates standard et traduits.

Indicateur de progression circulaire

L'indicateur de progression circulaire fonctionne de manière similaire à la progress bar. en donnant un id de __timeout__, GRUB met à jours l'indicateur de progression circulaire pour indiquer le temps restant. pour cet indicateur, il y a 2 images utilisées : l'image au centre, et l'image de tick. L'image du centre est rendue au centre du composant, et l'image tick est utilisée pour rendre chaque marque à la circomférence de l'indicateur.

Labels

Les labels texte peuvent être placés sur l'écran de boot. La font, couleur et alignement horizontale peuvent être spécifiés pour les labels. Si un label a l'id __timeout__, le text pour ce label est également utilisé avec un message informant l'utilisateur du nombre de secondes restant jusqu'au boot automatique. C'est utile dans le acs où vous souhaitez que le texte soit affiché au lieu d'une progress bar.

Boot Menu

Le menu de boot où GRUB affiche les entrées de menu. C'est une liste d'éléments, où chaque élément a un titre, et un icône optionnel. L'icône est sélectionné basésur les classes spécifiée pour l'entrée du menu. Si il y a un fichier png nommé myclass.png dans grub/themes/icons, il sera affiché pour les éléments ayant cette classe. Le menu de boot peut être personnalisé de différentes manières, tels que la font et la couleur utilisé pour le titre de l'entrée de menu, et en spécifiant de boites stylisés pour le menu lui-même, et pour l'élement sélectionné.

Boites stylisées

Une de fonctionnalité les plus importantes pour personnalisés la couche est d'utiliser des boites stylisés, composés de 9 régions rectangulaires, qui sont utilisés pour dessiner les boites.

```
Northwest (nw) North (n) Northeast (ne)
West (w) Center (c) East (e)
Southwest (sw) South (s) Southeast (se)
```

Fichier de thème

Le fichier de thème est un fichier texte, qui contient 2 types de déclarations, Les propriétés globales, et la construction de composant.

Les propriétés globales sont spécifiées avec le format simple :

```
name1: value1
name2: "valeur qui peut contenir des espaces"
```

```
name3: #88F
```

Liste de propriétés globles

title-text Spécifie le texte à afficher en haut de l'écran

title-font Font utilisée pour le titre

title-color Couleur du titre

message-font font à utiliser pour les messages

message-color Couleurs pour les messages

message-bg-color couleur de fond

desktop-image Image à utiliser comme fond

desktop-color Couleur pour le fond si desktop n'est pas spécifié

terminal-box Spécifie le motif de nom de fichier pour la boite stylisée.

Construction

Une plus grande personnalisation est fournie par des composants. Une arborescence de composants forment l'interface utilisateur. Les conteneurs sont des composants qui peuvent contenir d'autres composants, et il y a toujours un composant root qui est une instance d'un conteneur caneva.

Les composant sont créés dans le fichier thème en préfixant le type de composant avec le signe + :

```
+ label { text="GRUB" font="aqui 11" color="#8FF" }
```

Les propriétés d'un composant sont spécifiés avec "name=value", où value peut être un simple mot, un chaîne entre guillemets, ou un tuple "(ex : preferred_size=(120,80)"

Liste de composants

label Un label affiche une ligne de texte. Propriétés :

- **id** À "__timeout__" pour afficher le temps passé pour booter sur l'entrée par défaut.
- **text** Le texte à afficher
- **font** La font à utiliser
- **color** La couleur du texte
- **align** left, center et right
- **visible** À false permet de cacher le label.

image Affiche une image. Propriétés :

- **file** Chenin de l'image à charger

progress_bar Affiche une barre horizontale de progression. Propriétés

- **id** À "__timeout__" pour afficher le temps passé pour booter sur l'entrée par défaut.
- **fg_color** couleur de devant pour le rendu
- **bg_color** couleur de fond pour le rendu
- **border_color** La couleur de bordure pour le rendu

text_color Couleur du texte

bar_style La spécification de la boite stylisée pour la barre de progression.

highlight_style La spécification de boite stylisée pour la région survolée de la barre de progression.

highlight_overlay À true, la highlight box est affichée par dessus.

text Le texte à afficher sur la barre de progression.

font font à utiliser

circular_progress Affiche un indicateur de progression circulaire. L'apparence de ce composant est déterminé par 2 images. Propriétés :

id À "__timeout__" pour afficher le temps passé pour booter sur l'entrée par défaut.

center_bitmap Nom de l'image au centre

tick_bitmap Nom de l'image de tick

num_ticks Nombre de ticks qui sont créés dans un cercle complet

ticks_disappear booléen indiquant si les ticks apparaîssent ou disparaissent progressivement.

start_angle Position du premier tick, mesuré en "parrot" (1 parrot = 1/256 du cercle complot.)

boot_menu Affiche le menu de boot GRUB. il permet de sélectionner les éléments et de les exécuter. propriétés :

item_font font à utiliser pour les items de menu

selected_item_font Font à utiliser pour l'élément sélectionné

item_color Couleur du texte

selected_item_color Couleur du texte de l'élément séléctionné

icon_width Largeur de l'icône

icon_height Hauteur de l'icône

item_height Hauteur de chaque élément du menu

item_padding Espace en pixels à laisser de chaque côté du contenu de l'élément du menu

item_icon_space L'espace entre un icône et le texte, en pixels

item_spacing Espace à laisser entre les éléments du menu, en pixel

menu_pixmap_style Motif de fichier pour la boite stylisée.

selected_item_pixmap_style Boite stylisée pour l'élement sélectionné

scrollbar booléen indiquant si la barre de défilement est affichée

scrollbar_frame Motif de fichier pour le fond de la barre de défilement.

scrollbar_thumb Motif de fichier pour la barre de défilement.

scrollbar_thumb_overlay À true, le scrollbar thumb des bords s'affichent par dessus la scrollbar frame.

scrollbar_slice Emplatement de la barre de défilement (west, center, east)

scrollbar_left_pad padding à gauche, en pixel

scrollbar_right_pad padding à droite, en pixel

scrollbar_top_pad Padding en haut, en pixel

scrollbar_bottom_pad padding en bas, en pixel

visible À false permet de cacher la barre de progression

booter grub depuis le réseau

Les instructions suivantes fonctionnent seulement sur les systèmes PC BIOS où l'environnement PXE est disponible.
Pour générer une image PXE, lancer :
grub-mkimage –format=i386-pc-pxe –output=grub.pxe –prefix='(pxe)/boot/grub' pxe pxecmd
copier grub.pxe, /boot/grub/*.mod et /boot/grub/*.lst sur le serveur TFTP, s'assurant que *.mod et *.lst sont accessibles via le chemin /boot/grub depuis les serveur TFTP. définir la configuration du serveur DHCP pour offrir grub.pxe comme fichier de boot.

On peut également utiliser l'utilitaire grub-mknetdir pour générer une image et un répertoire GRUB, au lier de copier les fichiers manuellement.

Une fois GRUB démarré, les fichiers sur le serveur TFTP sera accessible via le périphérique pxe.

L'adresse ip du serveur et de la passerelle peuvent être contrôlés en changeant le périphérique (pxe) en (pxe :server-ip) ou (pxe :server-ip :gateway-ip).

GRUB fournis de nombreuses variables d'environnements qui peuvent être utilisée pour inspecter ou changer le comportement du périphérique PXE :

net_<interface>_ip L'adresse IP de cette machine. Lecture seule

net_<interface>_mac L'adresse MAC de l'interface réseau. Lecture seule

net_<interface>_hostname Le nom d'hôte du client fournis par DHCP. Lecture seule.

net_<interface>_domain Nom de domaine du client fournis par DHCP. Lecture seule.

net_<interface>_rootpath Le chemin du disque root du client, fournis par DHCP. Lecture seule.

net_<interface>_extensionspath Le chemin d'extensions de vendeur DHCP additionnels fournis par DHCP. Lecture seule

net_<interface>_boot_file Nom du fichier de boot fournis pas DHCP. Lecture seule.

net_<interface>_dhcp_server_name Le nom du serveur DHCP responsable pour ces paramètres de boot. Lecture seule.

net_default_ip L'adresse IP de l'interface par défaut. Lecture seule. alias de net_${net_default_interface}_ip

net_default_mac L'adresse mac de l'interface par défaut. Lecture seule. alias de net_${net_default_interface}_mac

net_default_server Le serveur par défaut. Lecture écriture

Utilitser GRUB via une ligne série

Pour les machines sans écran/clavier, il peut être utile de contrôler via des communications série. Pour connecter une machine avec un autre via une ligne série, il faut préparer un cable série, et avoir une carte série multiport. De plus, un émulateur de terminal est également requis, tels que minicom.

Comme pour GRUB, l'instruction pour définir un terminal série et simple. Exemple :
serial –unit=0 –speed=9600
terminal_input serial ; terminal_output serial
La commande serial initialise la série, COM1. Cette commande accepte de nombreuses options. Les commandes terminal_input et terminal_output choisissent quel type de terminal utiliser.

Utiliser GRUB avec des touches de mise en route

Certains vendeurs de laptop fournissent un bouton power-on additionnel qui boot un autre OS. GRUB supporte de tels boutons avec les variables GRUB_TIMEOUT_BUTTON, GRUB_DEFAULT_BUTTON, GRUB_HIDDEN_TIMEOUT_BUTTON and GRUB_BUTTON_CMOS_ADDRESS dans default/grub. Les valeurs connue sont :

```
Dell XPS M1530
85:3
Asus EeePC 1005PE
84:1
```

Fichiers image GRUB

GRUB consiste de nombreuses images : une variété d'image de boot pour démarrer GRUB de différentes manières, une image kernel, et un jeu de modules qui sont combinés avec l'image kernel pour former l'image core.

boot.img Sur les systèmes PC BIOS, cette image est la première partie de GRUB à démarrer. Elle est écrite dans un MBR ou sur le secteur de boot d'une partition. Parce que le secteur de boot PC fait 512 octets, la taille de cette image fait exactement 512 octets. La seule fonction de boot.img est de lire le premier secteur de l'image core depuis le disque local, et de sauter dessus. À cause de la restriction de la taille, boot.img ne peut pas comprendre la structure d'un système de fichier, donc grub-setup code en dur l'emplacement du premier secteur de l'image core dans boot.img en installant GRUB.

diskboot.img Cette image est utilisée comme premier secteur de l'image core en bootant depuis un disque dur. Elle lit le reste de l'image core en mémoire et démarre le kernel. Vu que la gestion du système de fichier n'est pas encore disponible, il encode l'emplacement de l'image core en utilisant un format de liste de blockes

cdboot.img Cette image est utilisée comme premier secteur de l'image core en bootant depuis un CD-ROM.

pxeboot.img Cette image est utilisée comme début de l'image core en bootant depuis le réseau en utilisant PXE.

lnxboot.img Cette images peut être placée comme début de l'image core pour que GRUB ressemble à un kernel liste qui peut être booté par LILO.

kernel.img Cette image contient les facilités basiques de GRUB : frameworks pour les périphériques et gestion des fichiers, variables d'environnement, mode récupération, etc. Elle est rarement utilisée directement, mais intégré dans toutes les images core.

core.img L'image core de GRUB. Elle est construite dynamiquement depuis d'image kernel et une liste arbitraire de modules par le programme grub-mkimage. Généralement, elle contient suffisamment de modules pour accéder à /boot/grub, et charger tout le reste. Le design modulaire permet à l'image core d'être conservée petite, vu que la zone disque où elle doit résider fait souvent moins de 32Ko.

***.mod** Tout le reste de GRUB réside dans des modules. Ils sont souvent chargés automatiquement, ou intégrés dans l'image core s'ils sont essentiels, mais peuvent également être chargés manuellement en utilisant insmod.

Syntaxe et sémantiques de système de fichiers

GRUB utilise une syntaxe spéciale pour spécifier les disques qui peuvent être accédés par le BIOS. À cause de la limitation des BIOS, GRUB ne peut pas distinguer les disques IDE, ESDI, SCSI, et d'autres. Vous devez connaître quel périphérique BIOS est équivalent à quel périphérique OS. Normalement, cela devrait être clair si les fichiers sont visible dans un périphérique ou en utilisant la commande search.

Comment spécifier les périphériques

La syntaxe de périphérique est la suivante :

```
(device[,partmap-name1part-num1[,partmap-name2part-num2[,...]]])
```

Les disques BIOS et EFI utilisent soit fd ou hd suivis par un chiffre, comme fd0 ou cd. AHCI, PATA, crypto, USB utilisent le nom du pilote suivi par un nombre. Memdisque et host sont limités à un disque et donc son référrés juste par le nom du pilote. RAID(md), ofdisk(ieee1275 et nand), LVM(lv), LDM et arcdisq (arc) utilisent un nom de disque intrinsèque du disque aliasé comme nand. Les conflit sont résolus en suffixant un nombre si nécessaire. Les virgules doivent être échappés :

```
(fd0)
(hd0)
(cd)
(ahci0)
(ata0)
(crypto0)
(usb0)
(cryptouuid/123456789abcdef0123456789abcdef0)
(mduuid/123456789abcdef0123456789abcdef0)
(lvm/system-root)
```

```
(lvmid/F1ikgD-2RES-306G-il9M-7iwa-4NKW-EbV1NV/eLGuCQ-L4Ka-XUgR-sjtJ-ffch-bajr-fCNfz5)
(md/myraid)
(md/0)
(ieee1275/disk2)
(ieee1275//pci@1f\,0/ide@d/disk@2)
(nand)
(memdisk)
(host)
(myloop)
(hostdisk//dev/sda)
```

part-num représente le numéro de partition de device, commençant à 1. partname est optionnel mais est recommandé vu que le disque peut avoir de nombreux partmaps top-level. En spécifiant le 3ème et dernier composant on peut accéder aux sous-partitions.

La syntaxe (hd0) représente utilisant tout le disque, alors que la syntaxe (hd0,1) représente l'utilisation de la première partition du disque.

```
(hd0,msdos1)
(hd0,msdos1,msdos5)
(hd0,msdos1,bsd3)
(hd0,netbsd1)
(hd0,gpt1)
(hd0,1,3)
```

En activant le support réseau (tftp) , (http) et autre sont également disponibles. Avant d'utiliser le disque réseau, il faut initialiser le réseaux. En bootant sur un cdrom, (cd) est disponible.

Comment spécifier des fichiers

Il y a 2 manières de spécifier des fichiers, par nom de fichier absolue, et par liste de block.

Un nom de fichier absolue a la forme (hd0,1)/boot/grub/grub.cfg. Si le nom du périphérique est omis, GRUB utilise le périphérique root implicitement.

Une liste de blocks est utilisée pour spécifier un fichie qui n'apparaît pas dans le système de fichier, comme un chainloader. La syntaxe est [offset]+length[,[offset]+length]... par exemple :
0+100,200+1,300+300
Cela signifie que GRUB devrait lire les blocks 0 à 99, le block 200, et les blocks 300 à 599. Si l'offset est absent, GRUB assume que l'offset est 0. De même, si le nom du périphérique est absent, le périphérique root est assumé.

Interface utilisateur de GRUB

GRUB a une interface de menu simple pour choisir une entrée depuis un fichier de configuration, et une ligne de commande. GRUB lit son fichier de configuration dès qu'il l'a chargé. S'il en trouve un, l'interface à menu est activée en utilisant les entrées trouvées dans le fichier. Avec la ligne de commande, ou si le fichier de configuration n'a pas été trouvé, GRUB passe sur l'interface en ligne de commande.

L'interface en ligne de commande fournis un prompt similaire à un shell Unix. Chaque commande est immédiatement exécutée après qu'elle soit entrée. Les commandes sont un sous-jeu de celles disponible dans le fichier de configuration, utilisée avec la même syntaxe.

Le mouvement du curseur et l'édition du text sur la ligne peut être fait via un sous-jeu de fonctions disponible dans le shell BASH :

C-f (flèche droite) Déplacer d'un caractère en arrière

C-b (flèche gauche) Déplace d'un caractère en avant

C-a (HOME) Déplacer au début de la ligne

C-e (END) Déplace à la fin de la ligne

C-d (DEL) Supprimer le caractère sous le curseur

C-h (BS) Supprimer le caractère à gauche du curseur

C-k Supprimer le texte de la position courant à la fin de la ligne

C-u Supprimer le texte du début de la ligne à la position du curseur

C-y Placer le texte supprimé dans le tampon au curseur

C-p, C-n (flèche du haut/bas) De déplacer dans l'historique

En tapant des commande interactivement, si le curseur et dans ou avant le premier mot dans la ligne de commande, la touche TAB affiche un listing des commandes disponible, ou la completion des disques, partitions, et noms de fichiers.

Noter qu'on ne peut pas utiliser la fonctionnalité de completion dans le système de fichier TFTP.

L'interface à menu est facile à utiliser. ses commande sont raisonnablement intuitives et décrites à l'écran. Basiquement, l'interface à menu fournis une liste d'entrées de boot. Utiliser les touches fléchées pour sélectionner l'entrée, puis appuyer sur entrer pour valider le choix

Les commandes sont disponible pour entrer une ligne de commande en appuyant sur 'c' qui opèrent exactement comme la version sans fichier de configuration de GRUB, mais permet de retourner au menu si désiré en appuyant sur échappe.

En protégeant l'interface avec un mot de passe, tout ce que l'on peut faire est de choisir une entrée avec entrer ou appuyer sur p pour entrer un mot de passe.

Éditer une entrée de menu

L'éditeur d'entrée de menu ressemble à l'interface à menu principale, mais les lignes dans le menu sont des commandes individuelles dans l'entrée sélectionnée au lieu des noms de l'entrée.

Si échappe est pressé dans l'éditeur, il annule tous les changement faits sur l'entrée et retourne dans le menu principal.

Chaque ligne dans l'entrée de menu peut être édité librement, et on peut également ajouter des lignes. Pour booter sur l'entrée éditée, utiliser Ctrl+x.

Variables d'environnement GRUB

GRUB supporte les variables d'environnement qui sont similaires aux système UNIX

biosnum En chaînant un autre chargeur de boot, GRUB a besoin de connaître quel numéro de disque BIOS correspond au périphérique root pour qu'il puisse être enregistré correctement. Si cette variable n'est pas définie, GRUB le devine.

check_signatures Contrôle si GRUB force la signature numérique au chargement des fichiers.

chosen En exécutant une entrée du menu, GRUB définis cette variable au titre de cette entrée.

cmdpath L'emplacement depuis lequel core.img a été chargé, en chemin absolue. Définis par GRUB au démarrage.

color_highlight Cette variable contient les couleurs de terminal foreground et background, séparé par un '/'. Définir cette variable change ces couleurs. Défaut : 'black/white'

color_normal Cette variable contient les couleurs normales foreground et background, séparés par un '/'. Chaque couleur doit être un nom parmis black, blue, green, cyan, red, magenta, brown, light-gray, dark-gray, light-blue, light-green, light-cyan, light-red, light-magenta, yellow, white. défaut : white/black

debug Cette variable peut être mis pour activer le debuggage de différents composants de GRUB. Contient une liste de facilités, ou all.

default Si cette variable est définie, elle identifie une entrée du menu qui devrait être par défaut. Peut être identifier par un numéro ou un titre.

fallback Si définie, identifie une entrée de menu qui devrait être sélectionnée si l'entrée par défaut échoue.

gfxmode Si définie, indique la résolution utilisée dans le terminal graphique gfxterm. Noter qu'il n'est possible d'utiliser que les modes VESA BIOS Extensions (VBE). défaut : auto.

gfxpayload contrôle le mode vidéo dans lequel Linux démarre. Remplace l'option de boot vga=. peut être 'text' pour forcer linux à booter en mode texte normal, 'keep' pour préserver le mode graphique définis avec gfxmode, ou une des valeurs permises pour gfxmode.

gfxterm_font nom d'une font à utiliser pour le texte dans le terminal graphique gfxterm.

grub_cpu En mode normal, GRUB définis cette variable au type de CPU pour lequel GRUB a été construit.

grub_platform En mode normal, GRUB définis cette variable à la platform pour laquelle GRUB a été construit (ex : pc ou efi)

icondir Répertoire dans lequel le menu graphique GRUB devrait rechercher les icônes, après avoir recherché dans le répertoire 'icons'

lang code de langue que la commande gettext utilise pour traduire les chaînes.

locale_dir Répertoire où trouver les fichiers de traduction, généralement /boot/grub/locale. sinon, l'internationnalisation est désactivée.

menu_color_highlight Contient les couleurs foreground et background à utiliser pour les entrées de menu sélectionnés, séparés par un '/'.

menu_color_normal Contient les couleurs à utiliser pour les entrée non-sélectionnés

net_* Voir la partie réseaux

pager À 1, pause la sortie après chaque écran plein et attend une entrée clavier.

prefix L'emplacement du répertoire /boot/grub. Normalement définis par GRUB

root Nom du périphérique root

superusers liste de noms de superutilisateurs pour activer le support de l'authentification

theme Répertoire contenant un theme graphique

timeout Spécifie le temps en seconde à attendre une entrée clavier avant de booter l'entrée par défaut. 0 indique de booter immédiatement, et -1, attend indéfiniment.

timeout_style Peut être menu, countdown, ou hidden.

Block d'environnement GRUB

Il est souvent utile de se rappeler d'un boot à l'autre. Par exemple, l'entrée par défaut peut être celle sélectionnée la dernière fois. GRUB n'implémente pas délibérément le support pour écrire des fichiers afin de minimiser les risques de corruption des systèmes de fichier. Cependant, GRUB fournis un block d'environnement qui peut être utilisé pour sauver une petite quantité d'états.

Le block d'environnement est un fichie 1024 octets préalloués, qui vis dans /boot/grub/grubenv. Au boot, load_env charge les variables depuis ce fichier, et save_env sauve les variables dans ce fichier. grub-editenv peut être utilisé pour éditer le block d'environnement.

Pour des raisons de sécurité, ce stockage est seulement disponible si installé sur un simple disque (ni LVM ou RAID), en utilisant un système de fihier sans checksum (pas de ZFS), et en utilisant des fonction BIOS ou EFI (pas de ATA, USB ou IEEE1275).

grub-mkconfig utilise cette facilité pour implémenté GRUB_SAVEDEFAULT

Liste de commandes disponible

Les commandes appartiennent à différents groupes. Quelques commandes peuvent être utilisées dans la section globale du fichier de configuration (ou menu). La plupart d'entre-elles peuvent être entrée sur la ligne de commande et peut être utilisée soit dans le menu ou spécifiquement dans les entrées du menu.

En mode récupération, seul insmod, ls, set, et unset sont normalement disponibles.

Commandes pour le menu uniquement

Les sémantiques utilisées pour parser le fichier de configuration sont les suivantes :

- Les fichiers doivent être au format plain-text
- Les options sont séparés par des espaces
- Tous les nombres peuvent être en décimal ou en hexadécimal.

Ces commandes peuvent être seulement être utilisés dans le menu :

menuentry title [–class=class ...] [–users=users] [–unrestricted] [–hotkey=key] {command ;...} Définie une entrée du menu nommée title. –class peut être spécifié plusieurs fois, et est utilisée par les thèmes pour afficher différentes classes en utilisant différents styles. –users donne aux utilisateurs spécifiques l'accès à l'entrée. –unrestricted donne accès à tous les utilisateurs. –hotkey associe une touche avec l'entrée, ou 'backspace, tab ou delete.

submenu title [–class=class ...] [–users=users] [–unrestricted] [–hotkey=key] {command ;...} Définis un sous-menu.

Liste des commandes générales, utilisée dans le menu et sur la ligne de commande :

serial [–unit=unit] [–port=port] [–speed=speed] [–word=word] [–parity=parity] [–stop=stop] Initialise un périphérique série. unit est un nombre de 0 à 3. Le port série n'est pas utilisé sauf si les commandes terminal_input et terminal_output sont utilisés.

terminal_input [–append|–remove] [terminal1] [terminal2] ... Sélectionne un terminal d'entrée. Sans argument, liste les terminaux actifs et disponibles. –append ajoute les terminaux nommés à la liste de terminaux actifs. –remove supprime les terminaux nommés de la liste active.

terminal_output [–append|–remove] [terminal1] [terminal2] ... Identique à terminal_output, mais pour les terminaux se sortie

terminfo [-a|-u|-v] [term] Définis les capacités de votre terminal en donnant le nom d'une entrée dans la base terminfo, qui devrait correspondre à une variable d'environnement TERM dans Unix. "vt100", "vt100-color", "ieee1275", et "dumb". Les options -a (–ascii), -u (–utf8), et -v (–visual-utf8) contrôlent comment le text non-ASCII est affiché.

Liste des commande utilisable sur la ligne de commande et dans les entrées de menu

[Alias de test

acpi [-1|-2] [–exclude=table1,...|–load-only=table1,...] [–oemid=id] [–oemtable=table] [–oemtablerev=rev] [–oemtablecreator=
Les systèmes BIOS modern implémentent ACPI, et définissent diverses tables qui décrivent l'interface entre un OS conforme ACPI et le firmware. Dans certains cas, les tables fournies par défaut ne fonctionnent qu'avec certains OS, et il peut être nécessaire de les remplacer. Cette commande va remplacer le Root System Description Pointer (RSDP) dans le Extended BIOS Data Area pour pointer vers de nouvelles tables. Si –no-ebda est utilisé, les nouvelles tables seront connues uniquement de GRUB, mais peut être utilisé par l'émulation EFI de GRUB.

authenticate [userlist] Vérifie l'utilisateur dans userlist ou listé dans la variable superusers.

background_color color Définis la couleurs de fond pour le terminal actif. Ne peut être changé qu'en utilisant gfxterm

background_image [[–mode 'stretch'|'normal'] file] Charge l'image de fond pour le terminal actif depuis le fichier spécifié. Uniquement en utilisant gfxterm

badram addr,mask [,addr,mask...] Filtrer la mémoire endommagée. La syntaxe est la même que fournie par l'utilitaire memtest86+

blocklist file Affiche une liste de block pour le fichier donné.

boot La commande boot démarre un OS ou un chain-loader qui a été chargé.

cat [–dos] file Affiche le contenu d'un fichier. Si –dos est utilisé, les CR/NL sont affiché comme simple nouvelle ligne.

chainloader [–force] file Charge le fichier comme chain-loader. –force charge le fichier même si la signature est incorrect.

clear Éffface l'écran

cmosclean byte :bit Efface la varleur d'un bit dans le CMOS.

cmosdump contents Dump le contenu du CMOS en valeurs hexadécimal.

cmostest byte :bit Test la valeur d'un bit dans le CMOS.

cmp file1 file2 Compare 2 fichiers.

configfile file Charge le fichier de configuration spécifié, puis affiche le menu.

cpuid [-l] Vérifie les fonctionnalités CPU. Cette commande est seulement disponible sur les systèmes x86. -l retourne vrai si le CPU supporte le mode long (64-bits)

crc file Affiche le checksum CRC32 du fichier

cryptomount device|-u uuid|-a|-b Définis l'accès à un périphérique chiffré. -a monte tous les périphérique chiffrés détectés. GRUB supporte les disques chiffrés en utilisant LUKS et geli.

date [[year-]month-day] [hour :minute[:second]] Sans argument, affiche l'heure et la date courante. Sinon prend la date et l'heure courante, change un élément spécifié et définis le résultat comme nouvelle date et heure.

devicetree file Charge le dtb depuis un système de fichier, pour le passer à Linux.

distrust pubkey_id Supprime la clé publique donnée du jeu de clé de GRUB. pubkey_id est les 4 derniers octets en hexa de l'id de clé GPG v4, qui est également la sortie de list_trusted. L'id de clé peut être obtenu avec gpg –fingerprint. Ces clés sont utilisées pour valider les signatures.

drivemap -l|-r|[-s] from_drive to_drive Sans option, map le lecteur from_drive au lecteur to_drive. C'est nécessaire pour chain-load certains OS, comme DOS. -s inverse le mappage, inversant 2 disques. -l liste les mappages courant. -r réinitialise tous les mappages

echo [-n] [-e] string ... Affiche le texte demandé et, sauf si l'option -n est utilisée, une nouvelle ligne. -e autorise l'interprétation des caractères échappé.

eval string ... Concatène les arguments ensemble en utilisant un simple espace comme séparateur et évalue de résultat comme séquence de commandes GRUB.

export envvar Exporte la variable d'environnement envvar. Les variables exportée sont visible aux fichiers de configurations chargés avec configfile

false Ne fait rien, se termine avec un code de sortie false

gettext string Traduit la chaîne donnée dans la langue courante. La langue courant est stockée dans la variable lang.

gptsync device [partition[+/-[type]]]... Les disques utilisant une table de partition GUID ont également une table de partition MBR pour compatibilité avec le BIOS et avec d'anciens OS. Les MBR peut seulement représenter un sous-jeu limité d'entrées de partitions GPT. Cette commande popule le MBR avec les entrées de partition spécifiés sur le périphérique. Jusqu'à 3 partitions peuvent être utilisée. 'type' est un code de type de partition MBR, [+/-] rend la partition active ou non.

halt –no-apm arrête l'ordinateur. Si l'option –no-apm est spécifiée, aucun appel APM n'est effectué. Sinon, l'ordinateur est éteint en utilisant APM.

hashsum –hash hash –keep-going –uncompress –check file [–prefix dir]|file ... Calcule ou vérifie les hashes des fichier. (adler32, crc64, crc32, crc32rfc1510, crc24rfc2440, md4, md5, ripemd160, sha1, sha224, sha256, sha512, sha384, tiger192, tiger, tiger2, whirlpool.

help [pattern ...] Affiche des informations sur les commandes intégrées.

initrd file Charge le ramdisk initial pour une image kernel Linux et définis les paramètres appropriés dans la zone de configuration de Linux. Peut seulement être spécifiée après la commande linux.

initrd16 file Identique, mais pour un ramdisk initiale en mode 16bits.

insmod module Charge un module GRUB

keystatus [–shift] [–ctrl] [–alt] Retourne true si les touches Shift, Control, ou Alt sont enfoncée, comme requis par les options. Utile dans les scripts pour permettre un certain contrôle utilisateur sur le comportement sans avoir à attendre une touche.

linux file Charge l'image Linux. Le reste de la ligne de commande est passée telle quelle au kernel.

linux16 file Idem mais charge un kernel en mode 16bits.

list_env -f file Liste toutes les variables dans le fichier block d'environnement. -f écrase l'emplacement par défaut du block d'environnement.

load_env -f file Charge toutes les variables depuis le fichier block d'environnement dans l'environnement.

loadfont file Charge les fonts spécifiées.

loopback -d device file Charge le contenu d'un fichier comme périphérique. -d pour le supprimer. ex loopback loop0 /path/to/image ; ls (loop0)/

ls [arg ...] Liste des périphériques ou fichiers.

lsfonts Liste les fonts chargées

lsmod Liste les modules chargés

md5sum alias de hashsum –hash md5 arg . . .

module [–nounzip] file [arguments] Charge un module pour l'image kernel multiboot. Le reste de la ligne est passé comme ligne de commande du module.

multiboot [–quirk-bad-kludge] [–quirk-modules-after-kernel] file . . . Charge l'image kernel multiboot depuis le fichier spécifié. Le reste de la ligne est passé au kernel comme ligne de commande.

nativedisk switch d'un pilote de disque firmware à un natif.

normal [file] Entre en mode normal et affiche le menu GRUB. En mode normal, les commandes, modules de système de fichier, et modules cryptographique sont automatiquement chargés, et le parser de script GRUB est disponible. Si un fichier est donné, les commandes seront lues depuis ce fichier, sinon elles sont lues depuis $prefix/grub.cfg, s'il existe.

normal_exit sort du mode normal. Si cette instance du mode normal n'a pas été imbriqué dans un autre, retourne en mode rescue.

parttool partition commands Effectue diverses modifications des entrées de la table de partition. Chaque command est soit un booléen, auquel cas il doit être suivi avec + ou -, pour activer ou désactiver cette option, ou peut prendre une valeur sous la forme command=value.

password user clear-password Définir un utilisateur nommé user avec le mot de passe donné.

password_pbkdf2 user hashed-password Définis un utilisateur nommé user avec le mot de passe hashé. Utiliser grub-mkpasswd-pbkdf2 pour générer des hashs de mot de passe.

play file | tempo [pitch1 duration1] [pitch2 duration2]... Jouer un son. Si l'argument est un nom de fichier, joue le fichier.

probe [–set var] –driver|–partmap|–fs|–fs-uuid|–label device Récupère des informations de périphérique. –set assigne le résultat à la variable var, sinon affiche l'information à l'écran.

pxe_unload Décharge l'environnement PXE.

read [var] Lit une ligne d'entrée de l'utilisateur. Si une variable var est donnée, définis cette variable d'environnement.

reboot Redémarre la machine

regexp [–set [number :] var] regexp string Test si l'expression régulière matche la chaîne.

rmmod module Supprime un module chargé

save_env [-f file] var ... Sauve les variables nommée depuis l'environnement dans le fichier block d'environnement.

search [–file|–label|–fs-uuid] [–set [var]] [–no-floppy] name Recherche des périphérique par fichier (-f, –file) label de système de fichier (-l, –label), ou UUID de système de fichier (-u, –fs-uuid). Avec –set, la variable var est définis avec le premier périphérique trouvé. La variable par défaut est root. –no-floppy empêche de recherche les périphérique disquette. les commandes search.file, search.fs_label, et search.fs_uuid sont des alias de search –file, search –label, et search –fs-uuid, respectivement.

sendkey [–num|–caps|–scroll|–insert|–pause|–left-shift|–right-shift|–sysrq|–numkey|–capskey|–scrollkey|–insertkey|–left-alt|–right-alt|
Insert des touches dans le tampon clavier en bootant. certains OS ou chain-loader nécessitent que des touches particulières soient pressées pour entrer en mode safe. Jusqu'à 16 touches peuvent être fournis. Les noms des touches peuvent être :

Name Key

```
escape   Escape
exclam   !
at   @
numbersign   #
dollar   $
percent   %
caret   ^
ampersand   &
asterisk   *
parenleft   (
parenright   )
minus   -
```

```
underscore   _
equal   =
plus   +
backspace   Backspace
tab   Tab
bracketleft   [
braceleft   {
bracketright   ]
braceright   }
enter   Enter
control   press and release Control
semicolon   ;
colon   :
quote   ’
doublequote   "
backquote   ‘
tilde   ~
shift   press and release left Shift
backslash   \
bar   |
comma   ,
less   <
period   .
greater   >
slash   /
question   ?
rshift   press and release right Shift
alt   press and release Alt
space   space bar
capslock   Caps Lock
F1   F1
F2   F2
F3   F3
F4   F4
F5   F5
F6   F6
F7   F7
F8   F8
F9   F9
F10   F10
F11   F11
F12   F12
num1   1 (numeric keypad)
num2   2 (numeric keypad)
num3   3 (numeric keypad)
num4   4 (numeric keypad)
num5   5 (numeric keypad)
num6   6 (numeric keypad)
num7   7 (numeric keypad)
num8   8 (numeric keypad)
num9   9 (numeric keypad)
num0   0 (numeric keypad)
numperiod   . (numeric keypad)
numend   End (numeric keypad)
numdown   Down (numeric keypad)
numpgdown   Page Down (numeric keypad)
numleft   Left (numeric keypad)
numcenter   5 with Num Lock inactive (numeric keypad)
numright   Right (numeric keypad)
```

```
numhome    Home (numeric keypad)
numup    Up (numeric keypad)
numpgup    Page Up (numeric keypad)
numinsert    Insert (numeric keypad)
numdelete    Delete (numeric keypad)
numasterisk    * (numeric keypad)
numminus    - (numeric keypad)
numplus    + (numeric keypad)
numslash    / (numeric keypad)
numenter    Enter (numeric keypad)
delete    Delete
insert    Insert
home    Home
end    End
pgdown    Page Down
pgup    Page Up
down    Down
up    Up
left    Left
right    Right
```

set [envvar=value] Définis la variable envvar à la valeur spécifiée. Sans argument, affiche les variables d'environnement

sha1sum Alias de hashsum –hash sha1 arg . . .

sha256sum Alias de hashsum –hash sha256 arg . . .

sha512sum Alias de hashsum –hash sha512 arg . . .

sleep [–verbose] [–interruptible] count Attend count seconde. –interruptible permet à ESC d'intérompre. –verbose affiche un décompte.

source file Lit le fichier comme fichier de configuration, et son conteneu est incorporé directement dans le fichier source. À la différence de configfile, exécute le contenu du fichier sans changer le contexte.

test expression Évalue l'expression et retourne un code de sortie. Similaire à la commande test de bash.

true Ne fait rien, retourne true.

trust [–skip-sig] pubkey_file Lit la clé publique et l'ajoute au jeu de clé de GRUB.

unset envvar Indéfinis la variable d'environnement donnée

uppermem Cette commande n'est pas encore implémentée

verify_detached [–skip-sig] file signature_file [pubkey_file] Vérifie une signature détaché style GPG du fichier donné.

videoinfo [[WxH]xD] Liste les modes vidéo disponibles.

net_add_addr interface card address Ajoute une adresse réseau

net_add_dns server Ajoute un serveur DNS

net_add_route shortname ip[/prefix] [interface | 'gw' gateway] Ajoute une route

net_bootp [card] Effectue une autoconfiguration bootp

net_del_addr interface Supprime une adresse IP de l'interface

net_del_dns address Supprime un serveur DNS

net_del_route shortname Supprime une route

net_get_dhcp_option var interface number type Récupère les options DHCP

net_ipv6_autoconf [card] Effectue une autoconfiguration IPv6

net_ls_addr Liste les interfaces

net_ls_cards Liste les cartes réseaux

net_ls_dns Liste les serveurs DNS

net_ls_routes Liste les routes

net_nslookup name [server] Effectue des recherches DNS

Jeux de caractères

GRUB utilise UTF-8 en interne. Tous les fichiers texte, incluant les configuration, sont supposés être encodés en UTF-8.

GRUB supporte la traduction. Pour cela il faut avoir des fichiers *.mo dans $prefix/locale, charger le module gettext et définir la variable lang.

Authentification et autorisation

Par défaut, l'interface est accessible à tout le monde avec un accès physique à la console : tout le monde peut sélectionner et éditer une entrée du menu, et tout le monde a un accès directe à un prompt shell. Pour la plupart des systèmes, c'est raisonnable vu qu'avec un accès physique il existe d'autres manière d'avoir un accès complet.

Cependant, dans certains environnements, tels que les kiosques, il peut être approprié de bloquer le boot loader pour exiger une authentification avant d'effectuer certaines opérations.

Les commandes password et password_pbkdf2 peuvent être utilisées pour définir les utilisateurs, chacun ayant un mot de passe associé. password définis le mot de passe en clair, ce qui exige que grub.cfg soit sécurisé ; password_pbkdf2 définis le password hashé en utilisant la rfc2898, qui nécessite l'utilisation de grub-mkpasswd-pbkdf2 pour générer les hashs.

Pour activer l'authentification, la variable superusers doit être définis à la liste des nom d'utilisateurs séparés par des espaces, ',', ' ;' ou 'l'. Les superusers sont autorisés à utiliser la ligne de commande, éditer les entrées du menu, et exécuter une entrée du menu.

Les autres utilisateur peuvent avoir accès aux entrée de menu spécifique en donnant une liste d'utilisateurs en utilisant l'option –users de la commande menuentry. Si l'option –unrestricted est utilisée pour une entrée de menu, cette entrée n'est pas restreinte. Si l'option –users n'est pas utilisée pour une entrée de menu, seul les superusers y ont accès. Exemple :

```
set superusers="root"
password_pbkdf2 root grub.pbkdf2.sha512.10000.biglongstring
password user1 insecure

menuentry "May be run by any user" –unrestricted {
 set root=(hd0,1)
 linux /vmlinuz
}

menuentry "Superusers only" –users "" {
 set root=(hd0,1)
 linux /vmlinuz single
}

menuentry "May be run by user1 or a superuser" –users user1 {
 set root=(hd0,2)
 chainloader +1
}
```

Le programme grub-mkconfig n'a pas encore le support pour générer des fichiers de configuration avec l'authentification. Vous pouvez utiliser /etc/grub.d/40_custom pour ajouter une simple authentification superusers, en ajoutant set superusers= et password ou password_pbkdf2.

Utiliser des signatures numériques dans GRUB

core.img peut optionnellement fournir la vérification de tous les fichiers lus. Ce document ne couvre pas commen s'assurer que le firmware de la plateform (ex : coreboot) valide core.img.

Si la variable check_signatures est à 'enforce', toute tentative de core.img de charger un autre fichier implique d'invoquer verify_detached foo foo.sig. foo.sig doit contenir une signature valide pour le fichier foo.

GRUB utilise les signatures détachée type GPG, et supporte actuellement DSA et RSA. Une clé de signature peut être générée en utilisant
gpg –gen-key
Un fichier peut être signé avec
gpg –detach-sign /path/to/file
Pour une validation complète de tous les sous-composants et le kernel chargé, ils doivent tous être chargés. Cela peut être fait pas grub-mkconfig :

```
# Edit /dev/shm/passphrase.txt to contain your signing key's passphrase
for i in `find /boot -name "*.cfg" -or -name "*.lst" -or \
 -name "*.mod" -or -name "vmlinuz*" -or -name "initrd*" -or \
 -name "grubenv"`;
do
 gpg -batch -detach-sign -passphrase-fd 0 $i < /dev/shm/passphrase.txt
done
shred /dev/shm/passphrase.txt
```