Slackware Live Edition

Voorwoord

Welkom bij de Slackware Live Edition! Dit is een versie van Slackware-current (binnenkort uit te brengen als versie 14.2), die kan worden gestart vanaf een DVD of een USB-stick. Het is een ISO-image bedoeld als een showcase van wat je zo allemaal met Slackware kunt doen. Slackware Live Edition bevat een standaard installatie, zonder extra of aangepaste pakketten of een andere kernel, maar wel met alle sterke punten van deze Linux distributie. De ISO wordt “from scratch” dus uit het niets gemaakt uit een lokale kopie van alle Slackware pakketen (die je tevoren moet downloaden), met hulp van de “liveslak” scripts.

Slackware Live Edition hoeft niet te worden geïnstalleerd op de harde schijf van een computer (*maar je hebt die mogelijkheid wel als je dat wilt: middels het* `setup2hd` *script*). Je kunt de Live versie op USB-stick met je meenemen in in je broekzak. Je hebt daarmee een voorgeconfigureerd Slackware OS dat in een minuut operationeel is op elke computer met een USB-poort waar je je handen op kunt leggen.

De USB-versie is “persistent” - wat betekent dat het Live Operating System je wijzigingen op de USB-stick opslaat. De CD/DVD versies (en de USB stick als je de persistentie niet aanzet) werken zonder persistentie, wat betekent dat alle wijzigingen die je doet tijdens het werken met het Live Operating System verloren gaan wanneer je de computer opnieuw start.

Om je gevoelige persoonlijke gegevens te beschermen in het geval dat je je USB-stick verliest (of in het geval de USB stick wordt gestolen) kun je je persistente USB Live Operating System voorzien van een versleutelde home directory en/of een versleuteld persistentie-bestand. De versleutelde data komt beschikbaar voor het Live Operating System wanneer je tijdens het opstarten vanaf de USB stick het wachtwoord intypt dat alleen jij kent.

Waarom nog een Live variant van Slackware

Dit zijn de redenen die ik had om het Slackware Live Edition project te starten:

1. Maak een “live” versie van de echte originele Slackware distributie; daarmee kan iedereen met Slackware kennis maken zonder het eerst te moeten installeren. De kernel berichten tijdens het opstarten scrollen over het scherm net als bij een geïnstalleerd Slackware systeem; er zijn geen aangepaste wallpapers, etcetera. Bedoeld voor educatieve, evaluatie en demonstratie doeleinden.
2. Het project moet zich richten op slackware-current, de “bleeding edge”. Veel mensen willen weten hoe de ontwikkelversie van Slackware er uit ziet, maar zijn huiverig om Slackware-current op hun computer te installeren uit angst dat niet alles stabiel is en goed werkt en zodoende zorgt voor productiviteitsverlies.
3. Zorg voor een manier om een Live ISO te genereren vanuit niet meer dan een lokale mirror van Slackware pakketten. Dat moet volledig gescript en deterministisch kunnen gebeuren.
4. Maar nog steeds moet het project in staat zijn om de inhoud en functionaliteit van het Live OS aan te passen. Bijvoorbeeld door uitgeklede of minimalistische versies van Slackware aan te bieden, maar ook door het opnemen van 3rd party pakketten in een aparte variant.

5. Bied de mogelijkheid om via een script en een ISO bestand een beschrijfbare bootable USB-stick te creëren met Slackware Live er op (dit verschilt van de manier om middels 'dd' of 'cp' de hybride ISO naar een USB-stick te kopieren - de USB stick zal op deze manier niet verder te beschrijven zijn omdat het zich gedraagt als een DVD medium).
6. KISS: Keep It Simple Stupid! Ofwel: houd de implementatie eenvoudig en begrijpelijk.

ISO varianten

De “liveslak” scripts kunnen een verscheidenheid aan Slackware varianten genereren:

1. Een complete 64bit Slackware-current Live Edition (in een 2,6 GB ISO);
2. Een afgeslankte XFCE ISO (700 MB) met XDM als grafische login manager. Het past op een CDROM medium of een 1 GB USB-stick;
3. Een ISO-image (3,1 GB) van Slackware64-current met Plasma 5 in plaats van KDE 4, met een toevoeging van een aantal andere pakketten uit de alienBOB repositories: vlc, libreoffice, calibre, qbittorrent, ffmpeg, chromium, openjdk, veracrypt.
4. Een Mate variant (1,7 GB), waar KDE 4 is vervangen door Mate (een Gnome 2 afsplitsing);
5. Een Cinnamon variant (een afsplitsing van de Gnome 3 Shell die Slackware's KDE 4 vervangt).
6. Een *Custom* variant waarmee je het Live OS je eigen naam en een eigen pakket lijst kunt meegeven evenals een eigen post-installatie configuratie.

Downloaden ISO images

De meest bezochte downloadlocaties zijn:

- Primaire website: http://bear.alienbase.nl/mirrors/slackware-live/ (rsync://bear.alienbase.nl/mirrors/slackware-live/)
- Darren's http://slackware.uk/people/alien-slacklive/ (rsync://slackware.uk/people/alien-slacklive/)
- Willy's http://repo.ukdw.ac.id/slackware-live/
- Ryan's https://seattleslack.ryanpcmcquen.org/mirrors/slackware-live/
- Shasta http://ftp.slackware.pl/pub/slackware-live/ (rsync://ftp.slackware.pl/slackware-live/)

Gebruikers Documentatie

Het ISO-image gebruiken

De ISO images zijn hybride, wat betekent dat je ze ofwel kunt branden op DVD, ofwel met de 'dd' of 'cp' commando's de inhoud van de ISO kan kopiëren naar een USB-stick. Bij beide methoden krijg je een live-omgeving die je zal toestaan om veranderingen aan te brengen en schijnbare “schrijfacties” naar de harde schijf“ kunt doen. De veranderingen die je aanbrengt in het filesysteem zullen in werkelijkheid in een RAM-schijf worden bewaard, zodat een reboot het Live besturingssysteem in zijn oorspronkelijke staat zal herstellen. Met andere woorden, de gegevens worden niet persistent bewaard.

Slackware Live Edition kent twee gebruikersaccounts: “root” en “live”. Ze hebben wachtwoorden en standaard zijn die wachtwoorden … je raadt het al: “root” en “live”. Standaard zal de Live ISO starten

in runlevel 4, dat wil zeggen dat je een grafische login te zien zult krijgen. De bootloader stelt je in staat om een niet-Amerikaanse taal en/of toetsenbordindeling en (bij het opstarten van een UEFI-systeem) een aangepaste tijdzone te kiezen. De commandoregel van de bootloader kan natuurlijk ook nog handmatig verder aangepast worden om je eigen tweaks aan te brengen.

Slackware Live Edition wijkt zo weinig mogelijk af van de manier waarop een reguliere Slackware opstart. Nadat je de Live boot fase afrondt en het eigenlijke Operating System gestart is log je in als gebruiker “live”. Vanaf dat moment ben je in een gewone Slackware omgeving.

Opstarten van het Live OS

BIOS boot

Slackware Live Edition maakt gebruik van syslinux om de Linux kernel op BIOS computers op te starten. Beter gezegd, wordt de “isolinux” variant geïnstalleerd in het ISO image en de “extlinux” variant wordt geïnstalleerd in de Linux-partitie van de USB Live versie.

Syslinux toont een grafisch boot menu met een mooie Slackware achtergrond en verschillende keuzemogelijkheden:

- Start (SLACKWARE | PLASMA5 | XFCE | MATE) Live (afhankelijk van welke van de ISO's je boot)
- Niet-Amerikaanse Keyboard selectie
- Niet-Amerikaanse Taal selectie
- Geheugentest met memtest86+

Je kunt een toetsenbord layout kiezen die past bij je computer. Ook kun je Slackware opstarten in een andere taal dan het Engels. Als je de US-Engels interfacetaal aanhoudt zul je waarschijnlijk nog steeds de tijdzone willen veranderen, omdat die standaard op UTC staat ingesteld. Je moet een aangepaste tijdzone handmatig opgeven door het toevoegen van “tz=JouwGeografie/JouwLokatie”, omdat het syslinux bootmenu je niet een selectie van tijdzones biedt. In syslinux kun je de boot commandoregel bewerken door op <TAB> te drukken. Druk dan op <ENTER> om de computer op te starten na het doen van je wijzigingen of <ESC> om je bewerking ongedaan maken en terug te keren naar het menu.

UEFI boot

Op UEFI computers verzorgt Grub2 de boot en het zal een soortgelijk menu tonen (en met hetzelfde thema) als het Syslinux menu:

- Start (SLACKWARE | PLASMA5 | XFCE | MATE) Live (afhankelijk van welke van de ISO's je boot)
- Niet-Amerikaanse Keyboard selectie
- Niet-Amerikaanse Taal selectie
- Niet-Amerikaanse Tijdzone selectie
- Geheugentest met memtest86+
- Hulp bij de opstart parameters

Een Grub menu regel kan worden bewerkt door op de “e” toets te drukken. Na het maken van je wijzigingen op de boot commandline, druk je op <F10> om de computer op te starten. Om je wijzigingen te negeren, druk je op <ESC>.

Een ander verschil tussen Syslinux en Grub2 menu's: in Grub2 kun je een niet-VS-toetsenbord, taal en/of tijdzone selecteren en daarna keer je telkens terug naar het hoofdmenu. Je moet nog steeds “Start SLACKWARE Live” kiezen om de computer op te starten. In het Syslinux menu brengt alleen het toetsenbord keuzemenu je terug naar het hoofdmenu. Echter, elke niet-Amerikaanse *taal* selectie resulteert onmiddellijk in het opstarten van Slackware Live; zonder terug te keren naar het hoofdmenu. Dit is een beperking van syslinux. Het zou exponentieel meer menu bestanden vereisen om een menu te construeren met meer keuzemogelijkheden. Grub2 ondersteunt variabelen die het gemakkelijk maken om de eigenschappen van een menu-item dynamisch te wijzigen.

Overdracht van ISO inhoud naar USB-stick

Een script is beschikbaar waarmee je de inhoud van een ISO-image kunt overbrengen naar een USB-stick, waarbij een aantal functionele wijzigingen kunnen worden gedaan aan het Live OS, afhankelijk van de parameters van het script.

De USB-stick zal worden gewist en opnieuw geformatteerd bij het uitvoeren van dit script! Voordat eventueel dataverlies plaatsvindt, zal het script je daarvoor waarschuwen. Op dat moment kun je oordelen of het veilig is om door te gaan.

Dit script, genaamd 'iso2usb.sh', accepteert de volgende parameters:

```
  -c | --crypt grootte|perc  Voeg een LUKS-versleutelde /home directory toe;
                             waarde van de parameter is de gewenste grootte
van
                             het containerbestand in kB, MB, GB, of als een
                             percentage van de vrije ruimte.
                             Voorbeelden: '-c 125M', '-c 1.3G', '-c 20%'.
  -f | --force               Negeer de meeste waarschuwingen (met
uitzondering
                             van de back-out vraag).
  -h | --help                Deze hulptekst.
  -i | --infile <filename>   Het volledige pad naar het ISO image-bestand.
  -o | --outdev <filename>   De devicenaam van de USB drive.
  -p | --persistence <name>  Niet-standaard naam van de 'persistente'
directory
                             of bestand (standaard naam is 'persistence').
  -r|--refresh               Ververs de USB stick met de inhoud van de ISO.
                             Stick wordt niet geformatteerd,
gebruikersgegevens worden niet geraakt.
  -u | --unattended          Script zal aflopen zonder enige interactie.
  -v | --verbose             Toon uitgebreide voortgangsberichten.
  -w | --wait <nummer>       Stel <nummer> seconden wachttijd in waarin de
                             kernel het USB systeem kan initialiseren.
  -C | --cryptpersistfile grootte|perc
                             Gebruik een LUKS versleuteld 'persistence'
bestand
                             in plaats van een directory (voor gebruik op
                             FAT-bestandssysteem).
  P | --persistfile          Gebruik een niet-versleuteld 'persistence'
bestand
```

```
                                in plaats van een directory (voor gebruik op
                                FAT-bestandssysteem).
```

Voorbeelden:

- Maak een USB-versie van Slackware Live, waar de USB-stick bij het systeem bekend is als '/dev/sdX'. NB - de waarde voor de output parameter is de devicenaam van de stick zelf en niet van een van de partities!

./iso2usb.sh -i ~/download/slackware64-live-14.2.iso -o /dev/sdX

- Maak een USB Live als hierboven, maar deze keer voegen we een versleuteld /home bestandssysteem toe met 750 MB aan ruimte, en tegelijkertijd verhogen we de wachttijd bij het opstarten tot 15 seconden (handig voor langzame USB media waarbij het Live OS anders niet wil starten):

./iso2usb.sh -i slackware64-live-14.2.iso -o /dev/sdX -c 750M -w 15

- Maak een USB Live met een versleutelde /home (waarbij 30% van de vrije ruimte van de stick wordt gereserveerd voor /home), de persistente gegevens zullen bovendien worden opgeslagen in een eigen container bestand in plaats van in een directory:

./iso2usb.sh -i slackware64-live-current.iso -o /dev/sdX -c 30% -P

- Maak een USB Live waar zowel /home als de persistente data versleuteld zijn (het persistente bestandssysteem is 300 MB):

./iso2usb.sh -i slackware64-live-current.iso -o /dev/sdX -c 30% -C 300M

Misschien heb je gemerkt dat de parameter ”-P“ geen waarde meekrijgt voor de grootte van het bestandssysteem. Dit komt omdat het onversleutelde container bestand wordt gemaakt als een “sparse” bestand dat begint bij een grootte “nul” en dynamisch mag groeien tot maximaal 90% van de aanvankelijke vrije ruimte op de Linux partitie van de USB-stick.

PXE opstarten van het Live OS

Slackware Live Edition kan vanaf het netwerk opstarten met behulp van het PXE-protocol waarbij de bestanden op de PXE server via NFS ge-exporteerd worden. Doe een loop-mount van het ISO bestand en kopieer de inhoud van de ISO naar (bijvoorbeeld) een nieuwe map genaamd `slackware-live` onder de directory van je TFTP-server's top directory `/tftproot`. Exporteer die map via NFS. Vervolgens moeten regels zoals hieronder worden toegevoegd aan het bestand `pxelinux.cfg/default` dat de configuratie van je PXE server bevat (hierbij wordt er van uit gegaan dat je NFS-server het IP-adres `192.168.0.1` heeft):

```
label liveslak
kernel slackware-live/boot/generic
append initrd=slackware-live/boot/initrd.img load_ramdisk=1 prompt_ramdisk=0
rw printk.time=0 kbd=us tz=Europe/Amsterdam locale=us_EN.utf8
nfsroot=192.168.0.1:/tftpboot/slackware-live hostname=pxelive
```

Zoals in het bovenstaande voorbeeld te zien is, wordt een bootparameter `nfsroot` gebruikt voor

netwerk boot. De parameter waarde definieert het IP-adres van de NFS server en het pad van de NFS export waar de inhoud van de Slackware Live ISO is uitpepakt. Hint: om een lijst van de ge-exporteerde shares van je NFS server te zien, gebruik je het commando `showmount -e localhost` op de NFS-server.

Eigenlijk zijn er twee boot parameters beschikbaar om netwerk boot goed te ondersteunen. De tweede boot parameter `nic` is meestal niet nodig. Deze kan worden gebruikt om de eigenschappen van de netwerkconfiguratie in de Live omgeving te definiëren, zoals de naam van de netwerkinterface, een statisch IP-adres en dergelijke. Indien Slackware Live opstart in een netwerk waar een DHCP-server actief is, dan zal de parameter `nic` niet nodig zijn omdat Slackware Live Edition alle details zelf kan uitzoeken.

Syntax van deze twee parameters:

```
nfsroot=ip.ad.dr.ess:/pad/naar/liveslak
nic=<driver>:<interface>:<dhcp|static>[:ipaddr:netmask[:gateway]]
```

Een aantal voorbeelden die gebruikmaken van de twee netwerk boot parameters:

```
nfsroot=192.168.1.1:/tftproot/slackware-Live
nic=auto:eth0:dhcp
nic=auto:eth0:static:10.0.0.21:24:
nic=:eth1:static:192.168.1.6:255.255.255.248:192.168.1.1
```

Nadat de PXE server is geconfigureerd aan de hand van bovenstaande richtlijnen (DHCP, TFTP en NFS servers), kun je één van je PXE-compatibele computers starten, de BIOS boot onderbreken en “netwerk boot” selecteren. Bij de PXE prompt typ of selecteer je het juiste label (in het bovenstaande voorbeeld zou dat `liveslak` zijn). Je ziet dat de kernel en initrd worden gedownload en opgestart, en dan zal het Live OS net zo starten als wanneer het wordt geladen van een lokaal medium.

Als de DHCP server te veel tijd nodig heeft voor het beantwoorden van het verzoek van de PXE client, zal het DHCP client programma tegen een time-out aanlopen en het opstarten van je Live OS zal mislukken omdat het Live bestandssysteem dat via NFS gemount moet worden niet beschikbaar zal zijn. In dat geval kun je proberen de wachttijd voor de DHCP-client te verhogen voordat dat programma besluit dat er geen IP adres van de server gaat komen. Voeg de boot parameter `dhcpwait=30` (voorbeeld) waar `30` het aantal seconden is dat de DHCP client moet wachten op een reactie van de server. Je moet natuurlijk een waarde kiezen die voldoende groot is voor je lokale netwerk.
De standaard DHCP wachttijd voor het Live OS is 20 seconden.

Persistentie wordt niet ondersteund voor het opstarten via het netwerk (PXE); op dit moment ondersteunt het overlay bestandssysteem (overlayfs) geen NFS als een beschrijfbare laag in het gecombineerde Live bestandssysteem.

PXE server

Slackware Live Edition is niet alleen in staat om op te starten als een PXE-client; het is tevens in staat om zelf als een PXE server te fungeren.

Wat betekent dat?

Een praktisch voorbeeld zou zijn dat je een USB stick met Slackware Live Edition naar een LAN-party meebrengt. Gebruik de USB stick om een van de computers op te starten. Vervolgens zijn alle andere computers in het (bekabelde) LAN in staat om vanaf het netwerk op te starten om een paar minuten later dezelfde Slackware Live Edition te hebben draaien. De computer met de USB stick fungeert als de PXE-server en alle andere computers zijn diens PXE clients, die de Slackware data lezen vanaf die USB stick. De client computers nemen een aantal instellingen over van de server, zoals tijdzone, taal en toetsenbord instellingen. Die via PXE meegegeven standaard configuratie kan alsnog worden veranderd op de boot commandoregel. De PXE clients zullen niet 'persistent' zijn. Als de server toegang tot het internet heeft, zullen de PXE clients eveneens toegang hebben.

Hoe de PXE server te starten?

Na het opstarten van het Live OS kun je een script “pxeserver” starten vanaf de console in runlevel 3 of van een X-terminal in runlevel 4. Het script zal alle benodigde informatie zelfstandig proberen te verzamelen. De ontbrekende gegevens worden via een dialoog interface aan de gebruiker opgevraagd. Als het script niet in staat is te achterhalen welke bekabelde netwerkinterface het moet gebruiken, kun je de naam van de interface (bijvoorbeeld eth1) meegeven aan de commandoregel voor het script als een enkele parameter:

```
# pxeserver eth1
```

De PXE server gebruikt het dnsmasq programma om DNS diensten aan te bieden aan de PXE clients. Het dnsmasq programma zal zijn interne DHCP server inschakelen als het lokale netwerk geen DHCP server heeft. Dnsmasq start ook een TFTP server waarmee de PXE-clients verbinding maken om de boot-bestanden (kernel en initrd) te downloaden voor het opstarten. Het `pxeserver` script start ook een NFS server die door de initial ramdisk (initrd) van het Live OS zal worden gebruikt om de squashfs modules te downloaden zodat het Live OS gestart kan worden. De PXE server kan meerdere netwerk interfaces hebben, bijvoorbeeld een draadloze interface die is verbonden met de buitenwereld en een bedrade interface die verbonden is met een andere computer (de PXE-client) of aangesloten op een switch met een heleboel potentiële PXE clients daarachter. In zo'n geval zal de PXE server “packet forwarding” aanzetten, zodat de PXE clients toegang tot de buitenwereld krijgen via de bedrade interface.

Indien de server meerdere netwerk interfaces heeft, is het belangrijk om te weten dat dnsmasq alleen zal binden aan de interface waar PXE clients contact mee zoeken. In een multi-NIC situatie waarin een tweede NIC is verbonden met de buitenwereld (je lokale netwerk), betekent dit dat de DHCP/DNS servers die gestart zijn door dnsmasq niet zullen interfereren met een bestaande DHCP server in het lokale netwerk.

Zodra de PXE server actief is, zal het `pxeserver` script de activiteiten van dnsmasq in een dialoogvenster tonen, zodat de goede werking van de PXE clients kan worden gecontroleerd.

Als de PXE-server computer over voldoende RAM geheugen beschikt, is het sterk aan te raden om het Live OS van de server op te starten vanaf de USB-stick met de boot parameter `toram`. Wanneer meer dan een paar PXE clients beginnen met het lezen van OS bestanden vanaf de PXE-server, zal de de leessnelheid van de USB-stick een knelpunt worden. Het draaien van het server OS in RAM zal dit potentiele knelpunt vermijden.

Boot parameters uitgelegd

Druk op <F2> in het syslinux opstart scherm voor een overzicht van de (meeste) boot parameters. Bij het opstarten van Grub kan het menu “Help op opstartparameters” worden gekozen. Het Grub helpscherm is lelijk, ik weet het, maar Grub kan het niet beter.

De volgende parameters worden herkend door Slackware Live Edition. Om op te starten met de standaardwaarden druk je op ENTER.

Desktop Environment

0 | 1 | 2 | 3 | 4 | 5 | 6 | S | s | single ⇒

```
Selecteer een runlevel.
De standaard is 4 voor grafische login.
```

kbd=fr xkb=ch,fr ⇒

```
Voorbeeld van een aangepaste X toetsenbordindeling.
De parameter xkb kan waarden "XkbLayout,XkbVariant,XkbOptions" krijgen.
De boot menu's zullen een aantal van deze automatisch configureren
maar uiteraard kunnen de waarden handmatig nog worden aangepast.
Let op dat de optionele XkbOptions kan bestaan uit meerdere door komma's
gescheiden waarden.
De XkbLayout en XkbVariant waarden mogen geen komma bevatten.
Je kunt ook enkel de XkbVariant instellen via bijvoorbeeld "kbd=ch xkb=,
fr"
```

livepw=“eenstring” ⇒

```
Wijzig het wachtwoord voor de gebruiker "live".
Het wachtwoord wordt doorgegeven als een leesbare tekst string.
```

locale=nl_NL kbd=nl tz=Europe/Amsterdam ⇒

```
Voorbeeld configuratie van taal, toetsenbord en/of tijdzone.
```

rootpw=“eenstring” ⇒

```
Wijzig het wachtwoord voor de gebruiker "root".
Het wachtwoord wordt doorgegeven als een leesbare tekst string.
```

Custom software

load=nvidia ⇒

```
Laden en configureren van Nvidia drivers indien beschikbaar
in de ISO (niet voor SLACKWARE en XFCE varianten).
```

load=mod1[,mod2[,...]] ⇒

```
Laad een of meer squashfs modules uit de directory "/liveslak/optional".
Standaard wordt geen van deze "optionele" modules geladen bij het opstarten.
```

noload=mod1[,mod2[,...]] ⇒

```
Voorkom laden van één of meer squashfs modules uit de directory
"/liveslak/addons".
Standaard worden alle "addons" modules geladen bij het opstarten.
```

Network boot

dhcpwait=<numseconds> ⇒

```
Maximale wachttijd voor de DHCP-client om een netwerkinterface
te configureren (standaard: 20 seconden).
```

nfsroot=ip.ad.dr.ess:/pad/naar/liveslak ⇒

```
Definieert het IP-adres van de NFS-server, en het pad naar
de uitgepakte inhoud van Slackware Live Edition.
```

nic=<driver>:<interface>:<dhcp|static>[:ipaddr:netmask[:gateway]] ⇒

```
Configureer de netwerk interface voor PXE boot, meestal is deze
parameter niet nodig wanneer je netwerk een DHCP server bevat.
Specificeer een driver als udev de interface niet detecteert. Benoem
de interface als Slackware Live die niet kan achterhalen. Bij 'static'
moet ook ipaddr en netmask worden opgeven. De gatewway is optioneel,
maar die kan nodig zijn om toegang te krijgen tot Internet bijvoorbeeld.
```

Hardware gerelateerd

localhd ⇒

```
Initialiseer RAID/LVM op de lokale harde schijven.
```

tweaks=tweak1[,tweak2,[,...]] ⇒

```
Geïmplementeerde tweaks:
nga - geen glamor 2D versnelling, dit vermijdt de foutmelding
  "EGL_MESA_drm_image vereist".
tpb - scroll met de TrackPoint terwijl middelste muisknop ingedrukt wordt.
syn - start syndaemon voor een betere ondersteuning van Synaptics touchpads.
ssh - start de SSH server (standaard uitgeschakeld).
```

nomodeset ⇒

```
Opstarten zonder kernel mode instelling, nodig met sommige machines.
```

rootdelay=10 ⇒

```
Voeg 10 seconden vertraging aan de kernel toe om USB initializatie
meer tijd te geven.  Probeer deze optie als het opstarten mislukt.
Standaard is 5.
```

swap ⇒

```
Laat het Live OS alle swappartities activeren op de lokale hardware.
Standaard wordt geen swap partitie aangeraakt.
```

Media tweaks

hostname=jouw_eigen_hostnaam[,kenmerk] ⇒

```
Geef een aangepaste hostnaam. Het kenmerk 'fixed' kan worden toegevoegd
om wijziging van hostnaam te verhinderen in geval van een netwerk boot.
```

livemedia=/dev/sdX ⇒

```
Vertel het init script welke partitie het Slackware Live besturingssysteem
bevat dat je wilt opstarten. Dit kan nodig zijn als je een ander exemplaar
van Slackware Live hebt geïnstalleerd op een andere partitie.
Ook worden geaccepteerd: UUID of LABEL.
```

livemedia=/dev/sdX:/path/to/live.iso ⇒

```
Gebruik dit als je het Live OS van een ISO-bestand wilt starten dat zich
op een lokale harde schijf partitie bevindt.
```

livemain=directoryname ⇒

```
Gebruik dit als je de inhoud hebt gekopieerd
van de ISO naar een andere directory dan "liveslak".
```

luksvol=bestand1[:/mountpoint1][,bestand2[:/mountpoint2],...] ⇒

```
Mount LUKS container "bestand1" op mount point "/mountpoint1"
in het Live bestandssysteem. Meerdere bestanden moeten worden gescheiden
door een komma.
Geef "luksvol=" om te *voorkomen* dat enige LUKS container gemount wordt,
met inbegrip van een versleutelde /home.
```

nop ⇒

```
Geen persistentie, dat wil zeggen start met het maagdelijke OS
```

```
en sla geen veranderingen op.
Nuttig indien je "persistence" directory beschadigd raakte.
Als je alle persistente data wilt negeren tijdens het opstarten,
waaronder LUKS data, gebruik dan de boot parameters "nop luksvol=".
```

nop=wipe ⇒

```
Wis alle gegevens in de persistentie directory of -container.
Nuttig ingeval de persistente gegevens corrupt geraakt zijn.
```

persistence=naam ⇒

```
Gebruik dit als je gebruik maakt van een ander
directory/bestand dan "persistence" voor het opslaan van persistente data.
```

toram ⇒

```
kopieert het besturingssysteem vanuit het opstartmedium naar RAM voordat
het start. Het opstartmedium kan dan na het opstarten verwijderd worden.
```

Problemen oplossen

blacklist=mod1[,mod2[,...]] ⇒

```
Zet een of meer kernel modules op de zwarte lijst van de kernel
om te voorkomen dat ze geladen worden, in geval ze problemen
veroorzaken tijdens het gebruik. Scheid modulenamen met komma's.
```

debug ⇒

```
Bij uitvoeren van 'init' wordt op strategische locaties gepauzeerd.
Informatie wordt getoond over de status van het overlay filesystem
en mounts.
```

rescue ⇒

```
Na de initialisatie kom je in een "rescue shell" terecht om onderhoud
uit te voeren op laag niveau.
```

Lay-out van de ISO

De Live ISO bevat drie mappen in de root van het bestandssysteem:

- EFI/
- boot/
- liveslak/

De USB-variant met persistentie kan een extra map in de root hebben:

- persistence/
- De EFI/ map bevat de Grub configuratie om het Live OS op te starten vanaf een computer met UEFI.
- De boot/ map bevat de syslinux configuratie om het Live OS op te starten vanaf een computer met een BIOS. Deze map bevat ook de kernel en initrd bestanden die worden gebruikt om het OS daadwerkelijk te starten.
- De liveslak/ directory bevat alle squashfs modules die worden gebruikt om het bestandssysteem van het Live OS samen te stellen, evenals bestanden die direct in de root van het live bestandssysteem worden gekopieerd. Het bevat vier submappen:
  - addons/ - modules die in deze map zijn opgeslagen zullen altijd aan het Live bestandssysteem worden toegevoegd, tenzij je dat verhindert met een “noload=” boot parameter;
  - optional/ - modules in deze directory worden niet toegevoegd aan het bestandssysteem van het Live OS, tenzij je dit forceert met een “load=” boot parameter;
  - system/ - deze directory bevat alle modules die zijn gemaakt door het “make_slackware_live.sh” script. Al deze modules zijn genummerd en het Live init script zal die nummering gebruiken om het Live bestandssysteem weer op te bouwen in exact dezelfde volgorde als ze in eerste instantie werden gecreëerd.
  - rootcopy/ - normaliter is deze directory leeg. Alles wat je (de gebruiker van de ISO) toevoegt aan deze map zal door het init script een-op-een naar de root van het bestandssysteem van het Live OS worden gekopieerd. Je kunt dit bijvoorbeeld gebruiken als je geen aparte squashfs module wilt maken om configuratiebestanden voor een van je toepassingen te verpakken.

Ontwikkelaar Documentatie

Scripts en gereedschappen

make_slackware_live.sh

Het eerste script:

Het script “make_slackware_live.sh” creëert een ISO-bestand dat het Live OS bevat. Dankzij Linux kernel 4.x en het squashfs-tools pakket in Slackware, is (her)compilatie van Slackware inhoud of het installeren van 3rd party pakketten niet nodig om een Slackware Live ISO te creeren.

De werking van het script kan worden onderverdeeld in meerdere afzonderlijke fases. Het proces verloopt als volgt:

Installeer de Slackware pakketten

Eerste fase:

- Het script leest een pakket sequentie voor de gekozen Live variant en installeert alle pakketten in deze volgorde naar submappen van een tijdelijke directory structuur.
- Elke Slackware pakket-serie (a, ap, d, …, y) of pakket-lijst (min, xbase, xapbase, …) wordt

geïnstalleerd in een aparte 'hoofd'-map.
- Elk van deze hoofdmappen wordt “gecomprimeerd” (met behulp van squashfs) in een aparte squashfs module. Zo'n module is een enkel archiefbestand die de gecomprimeerde directory structuur van de geïnstalleerde pakketten bevat.
- Deze module bestanden worden een voor een ge”loop“mount en vervolgens gecombineerd tot één read-only directorystructuur door middel van een “overlay mount”. Het overlay bestandssysteem is relatief nieuw; eerdere Live distributies gebruikten aufs en unionfs, die vergelijkbare functionaliteit bieden, maar die bestandssystemen zijn geen onderdeel van de kernel source en dus moest er altijd een gemodificeerde kernel worden gecompileerd voor een Live distro met aufs/unionfs.
- Deze 'overlay assemblage' vormt het bestandssysteem waarmee het Live OS zal opstarten.
- Boven op dit samengestelde read-only bestandssysteem wordt een beschrijfbaar bestandssysteem toegevoegd door het “make_slackware_live.sh” script. Dit beschrijfbare bestandssysteem wordt gebruikt om de aanpassingen op te slaan die door “mae_slackware_live.sh” worden aangebracht om van een gewone Slackware distro een Live distro te maken. Zie het volgende gedeelte voor meer informatie.
- Merk op dat wanneer je het Live OS later opstart, er nog een andere beschrijfbare overlay worden gebruikt om alle schrijfbewerkingen op te slaan die worden uitgevoerd door het besturingssysteem. Zo kan het Live OS zich voordoen als een echt besturingssysteem dat bestanden kan schrijven en verwijderen zelfs als het van een niet-beschrijfbaar DVD medium wordt opgestart. Dat beschrijfbaar bestandssysteem zal zijn:
    - Een RAM-gebaseerd bestandssysteem wanneer het Live OS draait zonder persistentie.
    - Een directory of een loop-mounted containerbestand op je USB-stick, als je een USB-stick gebruikt met persistentie.

Configureer het Live bestandssysteem met zinvolle defaults

Fase twee:

- Een zekere mate van bestandssysteem initialisatie wordt gedaan nadat de overlay is samengesteld:
    - 'root' en 'live' user accounts worden aangemaakt,
    - De login omgeving van deze user accounts wordt aangekleed,
    - De desktop-omgeving wordt geconfigureerd voor het eerste gebruik,
    - De liveslak scripts “makemod” en “iso2usb.sh” worden gekopieerd naar ”/usr/local/sbin/“ in de ISO voor je gemak,
    - Indien het Live systeem een “huge” kernel bevat (alle ISO varianten behalve XFCE) dan worden ook het “setup2hd” script en de Slackware installatie bestanden naar respectievelijk ”/usr/local/sbin“ en ”/usr/share/liveslak“ gekopieerd,
    - Slackpkg wordt geconfigureerd,
    - Een “locate” zoekdatabase wordt gemaakt,
    - enz…
- Al deze wijzigingen worden weggeschreven naar het beschrijfbare bestandssysteem dat werd aangemaakt in de vorige paragraaf. Dit bestandssysteem zal ook worden opgeslagen in de ISO als squashfs module en wanneer het Live OS opstart, zal het read-only worden gemount, net als alle andere modules. Deze module heeft de naam “0099-slackware_zzzconf-current-x86_64.sxz” of meer in het algemeen “0099-slackware_zzzconf-${SLACKVERSION}-${ARCH}.sxz”

Configureer de boot fase van het Live OS

Fase drie:

- Een initrd wordt gegenereerd, met een gemodificeerd “init” script (andere Live distro's noemen het een “linuxrc script”), dat het overlay filesystem opnieuw opbouwt uit diverse squashfs modules tijdens het opstarten en het Live OS configureert op basis van de boot parameters (taal, toetsenbord, tijdzone, runlevel, ...)
- Een Slackware “generic” kernel plus de genoemde initrd wordt toegevoegd aan een syslinux (voor BIOS computers) en een grub2 (UEFI computers) configuratie.

Maak de ISO image

Fase vier:

- Een bootable ISO-bestand wordt gemaakt met behulp van mkisofs.
- Het “isohybrid” commando wordt uitgevoerd op de ISO zodat je met “dd” of “cp” de ISO naar een USB-stick kunt kopieren en zo een opstartbaar USB medium maakt.

Klaar! Je kunt het ISO-bestand en de MD5 checksum in de /tmp directory vinden.

iso2usb.sh

Het tweede script:

Het gebruik van het “iso2usb.sh” script wordt in detail uitgelegd in een eerdere paragraaf “Overdracht van ISO inhoud naar USB-stick”.

Deze paragraaf beschrijft hoe het script de ISO aanpast voor een verbeterde USB functionaliteit.

Mount een bestandssysteem in een versleutelde container

Het script zal een bestand van de gewenste grootte aanmaken in de root van de Live partitie met het programma 'dd'. Het commando 'cryptsetup luksCreate' begint de versleuteling, waarbij de vraag wordt gesteld “weet je dit zeker, type de hoofdletters YES”, waarna een codeerwachtwoord drie keer moet worden ingevoerd (twee voor het initializeren, en één voor het openen van de container). Wanneer dit container bestand wordt gebruikt voor een versleutelde /home, is de bestandsnaam “slhome.img”. Het script kopieert de bestaande inhoud van de /home map uit de ISO naar de container, die later in de plaats van /home in het Live bestandssysteem gemount wordt (waardoor de bestaande /home gemaskeerd wordt). Het Live OS wordt geconfigureerd om de container te ontsleutelen en het bestandssysteem te mounten. Die configuratiegegevens worden geschreven in het bestand ”/luksdev“ in de initrd door het toevoegen van een regel die enkel bevat: ”/slhome.img“. Het iso2usb.sh script ondersteunt alleen het creëren en configureren van een versleutelde /home, maar je kunt additionele versleutelde containers maken en die mounten op andere locaties in het bestandssysteem van de ISO. Om dit te laten werken, moet je zelf het ”/luksdev“ bestand bewerken en een regel ”/jouw/container.img:/jouw/mountpoint“ toevoegen, dwz de padnaam van de container en de directory waar deze container gemount moet worden staan op één regel, gescheiden door een

dubbele punt. Het Live 'init' script zal de mount directory aanmaken als deze ontbreekt.

Gebruik van een container voor het opslaan van persistente data

Een tweede type versleutelde container kan worden gebruikt voor het opslaan van persistente data. Het Live 'init' script controleert of het persistentie moet inschakelen in deze volgorde:

1. Is het USB bestandssysteem beschrijfbaar? Zo ja,
    1. Bestaat een directory /persistence ? Als dat zo is, wordt die gebruikt; zo niet,
    2. Bestaat een bestand met de naam /persistence.img ? Zo ja, dan wordt het bestand gemount en als het een versleutelde container blijkt te zijn, wordt om het LUKS wachtwoord gevraagd tijdens het opstarten.

Het toevoegen van USB wachttijd

Voor langzame USB media kan de standaard 5 seconden wachttijd tijdens het opstarten soms onvoldoende zijn voor de kernel om de partities op je USB apparaat te detecteren. Het script kan optioneel meer wachttijd toevoegen. Het doet dit door het bestand ”/wait-for-root“ te bewerken in de initrd en de waarde op te hogen die daar is opgeslagen (standaard wordt de waarde “5” geschreven door het “make_slackware_live.sh” script).

makemod

Het derde script:

Met het “makemod” script kun je eenvoudig een Slackware Live module maken, met een Slackware pakket of een directory boom als de invoerparameter.

Gebruik:

```
# makemod <pakketnaam|directory> modulenaam.sxz
```

- De eerste parameter is het volledige pad naar een Slackware pakket, of anders een directory.
    - Als een pakketnaam wordt geleverd als eerste parameter, zal het worden geïnstalleerd in een tijdelijke map met behulp van Slackware's “installpkg”. De inhoud van de tijdelijke map zal in een module worden geperst door het programma “squashfs”.
    - Als een directorynaam wordt meegegeven wordt de inhoud daarvan opgenomen in een module door het programma “squashfs” .
- De tweede parameter is de volledige padnaam van de squashfs module die wordt gecreëerd.

Je kunt de module die je zojuist hebt gemaakt (let op de bestandsnaam conventies voor een Slackware Live module, zie paragraaf “Slackware Live module format”) naar de optional/ of naar de addons/ directory van het Live OS kopiëren . Als je het module bestand kopiëert naar de optional/ of addons/ directory van de liveslak brondirectory dan zal “make_slackware_live.sh” de module toevoegen bij het genereren van de ISO image.

setup2hd

Het vierde script:

Het “setup2hd” script stelt je in staat om het draaiende Live besturingssysteem te installeren op de lokale harde schijf van je computer. Het “setup2hd” script is een aangepaste Slackware 'installer', zodat het installatie proces bekend zal voorkomen. In tegenstelling tot een standaard Slackware installatie is hier geen 'SOURCE' selectie, omdat het script weet waar het de squashfs modules kan vinden. Nadat je de doelpartitie(s) hebt geselecteerd, worden alle actieve modules van de Live OS variant (SLACKWARE, PLASMA5, MATE, …) uitgepakt op de harde schijf. Nadat de extractie is voltooid, geeft het script een overzicht van de modules die zijn verwerkt. Het zal ook een voorbeeld commando laten zien om resterende inactieve of uitgezette modules handmatig uit te pakken. De laatste stap in de installatie is als vanouds de reguliere Slackware installer die de Slackware configuratiescripts start.

pxeserver

Het vijfde script:

Het `pxeserver` script werkt als volgt:

- Het vereist een bekabeld netwerk; wireless PXE boot is onmogelijk.
- Het pxeserver script probeert een bekabelde interface te vinden; je kunt een expliciete interfacenaam doorgeven als parameter aan het script (optioneel).
- Als meerdere bekabelde interfaces worden gedetecteerd, vraagt een dialoog aan de gebruiker om de juiste te selecteren.
- Een controle wordt gedaan op een eventuele DHCP-configuratie van deze bekabelde interface;
  - Als een DHCP-configuratie wordt gevonden dan zal `pxeserver` geen eigen DHCP server starten en in plaats daarvan gebruik maken van de DHCP server in het LAN.
  - Als er geen DHCP config wordt gevonden, zal het script toestemming vragen om haar eigen interne DHCP server te starten. Daarnaast wordt de gebruiker gevraagd om een IP adres te configureren voor de netwerkinterface en een IP reeks te definieren voor de interne DHCP server.
- Het script zal dan de eigenlijke PXE server starten, bestaande uit:
  - dnsmasq, dat DNS, DHCP en BOOTP diensten biedt;
  - NFS en RPC daemons;
- Het script zal een externe netwerk verbinding op een andere interface detecteren en zal IP forwarding aan zetten als dat nodig is, zodat de PXE clients ook toegang tot het externe netwerk krijgen.
- Het Live OS dat wordt opgestart via pxelinux is voorgeconfigureerd met een aantal extra boot parameters:

```
nfsroot=<server_ip_address>:/mnt/livemedia
luksvol=
nop
hostname=<distroname>
tz=<server_timezone>
locale=<server_locale>
kbd=<server_kbd_layout>
```

waaruit blijkt dat de configuratie van het Live OS waar de PXE server gestart werd grotendeels de configuratie van de PXE clients bepaalt.

- Merk op dat bij opstarten vanaf het netwerk, de hostnaam van het Live OS wordt uitgebreid met het MAC adres van de client computer om de hostnaam van elke computer die vanaf het netwerk is gestart uniek te maken.

Het creëren van een Live ISO vanuit het niets

Een ISO-image van Slackware live Edition creëren vereist dat je computer Slackware 14.2 (64-bit) draait. Oudere versies van Slackware hebben een kernel die te oud is om gebruik te maken van de “overlayfs” kernel functionaliteit - noodzakelijk om liveslak te ondersteunen. Ook ontbreken de squashfs programma's. Daarom ook kan een Slackware Live Edition alleen worden gemaakt voor Slackware 14.2 of nieuwer.

Een lokale kopie van een Slackware distributie (14.2) is noodzakelijk omdat de pakketten daarin worden gebruikt bij het opbouwen van de ISO.

Je moet ook de “liveslak” script collectie downloaden via een van de URLs aan het eind van deze pagina.

Liveslak is een directorystructuur met scripts, bitmaps en configuratiebestanden. Slechts 5 scripts zijn bedoeld om te worden uitgevoerd door jou, de gebruiker. Deze scripts (“make_slackware_live.sh”, “iso2usb.sh”, “makemod”, “setup2hd” en “pxeserver”) worden nader toegelicht in het hoofdstuk “Scripts en gereedschappen” hogerop. Bij het maken van een Live ISO vanuit het niets, hoef je alleen maar het “make_slackware_live.sh” script uit te voeren.

Layout van liveslak

De toplevel 'liveslak' map bevat de volgende submappen:

- EFI/ - bevat het skelet voor de boot-ondersteuning op UEFI computers. Sommige van de UEFI configuratiebestanden worden dynamisch gegenereerd door het “make_slackware_live.sh” script.
- addons/ - squashfs modules geplaatst in deze map zullen worden geladen in het Live bestandssysteem wanneer het besturingssysteem opstart.
- graphics/ - squashfs modules voor “gesloten” GPU-ondersteuning (zoals Nvidia) kunnen hier worden geplaatst. De module(s) worden gekopieerd naar addons/ door het “make_slackware_live.sh” script voor alle Live Desktops (behalve de pure Slackware omgeving) die profiteren van proprietary driver ondersteuning.
- local64/ , local/ - deze mappen kunnen Slackware pakketten bevatten die als 'lokaal' worden beschouwd, dat wil zeggen niet tot een repository behorend. Zulke pakketten worden ter plekke omgezet naar squashfs module(s) door het “make_slackware_live.sh” script, en gekopieerd naar de “addons/” subdirectory in de ISO zodat ze geladen worden in het Live bestandssysteem wanneer het besturingssysteem opstart.
- optional/ - squashfs modules geplaatst in deze map worden niet automatisch geladen in het Live bestandssysteem wanneer het besturingssysteem opstart. Je moet de “load=[mod]” boot parameter gebruiken om zo een module te laden.
- pkglists/ - definitie bestanden van 3rd party repositories (*.conf) en de pakketten die uit die

repositories gebruikt worden (*.lst) worden in deze map geplaatst.
- skel/ - bevat gecomprimeerde tarballs (waarvan de bestandsnamen moeten voldoen aan de wildcard “skel*.txz”). Deze bestanden worden uitgepakt naar de ”/etc/skel“ directory in het Live bestandssysteem.
- syslinux/ - bevat het skelet voor de boot-ondersteuning op BIOS computers. Sommige van zijn bestanden worden dynamisch gegenereerd door het “make_slackware_live.sh” script.
- xdm/ - Slackware thema voor de XDM grafische sessie manager die wordt gebruikt bij de ISO-varianten die niet worden geleverd met GDM, KDM of SDDM.

De toplevel 'liveslak' directory bevat de volgende bestanden:

- README.txt - deze documentatie.
- blueSW-128px.png, blueSW-64px.png - dat zijn bitmaps van het “Blue S” Slackware logo, gebruikt voor het “live” user icon en in het XDM thema.
- grub.tpl - het template bestand dat wordt gebruikt om het grub menu voor UEFI boot te genereren.
- iso2usb.sh - dit script maakt een bootable USB-versie met persistente data uit een Slackware Live ISO.
- languages - dit bestand bevat de configuratie van de taalondersteuning. Iedere regel bevat een taaldefinitie met de volgende velden: “code:naam:kbd:tz:locale:xkb”. Voorbeeld: “nl:Nederlands:nl:Europe/Amsterdam:nl_NL.utf8:”
  - code = 2 letters taalcode
  - naam = beschrijvende naam van de taal
  - kbd = naam van de console toetsenbord mapping voor deze taal
  - tz = tijdzone voor land van oorsprong van de taal
  - locale = de localizatie zoals gebruikt in het land
  - xkb = optionele aangepaste X toetsenbord variant van de taal
- liveinit - dit is het “init” script dat in het initrd image van het Live OS wordt gekopieerd. Samen met Slackware's “generic” kernel zorgt de initrd voor het opstarten van de computer. Dit “init” script assembleert het Live bestandssysteem vanuit zijn componenten, de squashfs modules.
- make_slackware_live.conf - het configuratiebestand voor het “make_slackware_live.sh” script. Je kunt standaardwaarden voor vele scriptparameters hier definiëren, zodat je het script zelf niet hoeft te bewerken.
- make_slackware_live.sh - het script dat de Live ISO genereert.
- makemod - dit script maakt een squashfs module uit een Slackware pakket (of uit een directory boom).
- menu.tpl - template bestand dat wordt gebruikt om het syslinux opstartmenu voor BIOS computers te genereren.
- pxeserver - het script dat een PXE server start die andere computers in het lokale netwerk in staat stelt om ook Slackware Live te draaien.
- setup2hd - het script waarmee Slackware Live op een harddisk kan worden ge-installeerd.
- setup2hd.local - hierin kan een ontwikkelaar van een aangepast Live OS de “post-installatie” routine zelf definiëren door de functie “live_post_install()” in het `setup2hd` script te her-implementeren.

Genereer de ISO

Het “make_slackware_live.sh' script van liveslak accepteert optionele parameters om het proces van Live OS generatie te beinvloeden:

```
Parameters van het script zijn:
 -h                  Deze helptekst.
 -a arch             Machine architectuur (default: x86_64).
                     Gebruik i586 voor een 32bit ISO, x86_64 voor 64-bit.
 -d desktoptype      SLACKWARE (full Slack), KDE4 (basis KDE4),
                     XFCE (basis XFCE), PLASMA5 (vervangt KDE4),
                     MATE (gnome2 fork vervangt KDE4), CINNAMON (fork van
                     gnome3 shell vervangt KDE4).
 -e                  Gebruik ISO boot-load-grootte van 32 voor computers
                     waar de ISO anders niet zal opstarten (standaard: 4).
 -f                  Gedwongen re-generatie van alle squashfs modules,
                     aangepaste configuraties en nieuwe initrd.img.
 -m pkglst[,pkglst] Modules toevoegen gedefinieerd door
pkglists/<pkglst>,...
 -r serie[,serie]   Vernieuw slechts één of een paar pakket series.
 -s slackrepo_dir   Directory die de Slackware repository bevat.
 -t <doc | mandoc>  Verklein de ISO (verwijder man pages en/of
documentatie).
 -v                  Toon debug/error output.
 -z versie           Bepaal de Slackware versie (standaard: current).
 -G                    Genereer ISO bestand vanuit bestaande directory
structuur
 -H <hostnaam>      Hostnaam voor het Live OS (standaard: darkstar).
 -M                    Voeg multilib toe (enkel voor x86_64).
 -O <outfile>       Custom bestandsnaam van de ISO.
 -R <runlevel>      Runlevel om mee op te starten (standaard: 4).
 -X                  Gebruik xorriso plaats van mkisofs/isohybrid.
```

Het script maakt gebruik van pakket repositories om een Live ISO te creëren. De pakketten zullen worden geïnstalleerd in een tijdelijke map.

Om een Live ISO te creëren voor een van deze varianten, moet een lokale kopie aanwezig zijn van alle gebruikte pakket repositories (dit kan een over het netwerk gemounte directory zijn). Een lokale mirror van de officiele Slackware repository is in ieder geval verplicht en moet handmatig worden klaargezet. Alle pakketten die worden gebruikt uit een 3rd party repository kunnen door “make_slackware_kive.sh” gedownload worden vanaf een externe server indien een rsync URL voor de repository in ./pkglists/*.conf is geconfigureerd.

Wanneer aan alle voorwaarden is voldaan, is een enkel commando voldoende om de ISO te genereren. Het volgende voorbeeld zal een ISO bestand voor een pure Slackware Live Edition (64bit - current) maken:

```
# ./make_slackware_live.sh
```

Hier is een ander voorbeeld waarbij een MATE variant wordt gemaakt, runlevel '3' als standaard wordt ingesteld en er een aangepast pad voor de Slackware pakket repository wordt gespecificeerd (houd er rekening mee dat het script zoekt naar een subdirectory “slackware64-current” onder deze directory als deze ISO wordt gegenereerd op basis van slackware64-current):

```
# ./make_slackware_live.sh -d MATE -R 3 -s ~ftp/pub/Slackware
```

Indien je wilt weten welke pakket sets worden gebruikt in de ondersteunde Desktop Environments, brengt de volgende commandoregel uitkomst:

```
# grep ^SEQ_ make_slackware_live.sh
```

Voor MATE vind je:

```
SEQ_MSB="tagfile:a,ap,d,e,f,k,l,n,t,tcl,x,xap,xfce,y pkglist:slackextra,mate
local:slackpkg+"
```

Dat betekent dat de meeste Slackware pakket series (met uitzondering van KDE en KDEI) worden geïnstalleerd met behulp van hun tagfiles, en daar bovenop worden twee pakket lijsten geïnstalleerd die in de pkglists/ subdirectory gedefinieerd zijn: 'slackextra' en 'mate'. Tot slot zal “slackpkg+” worden geïnstalleerd vanuit de subdirectory local64/.

Mogelijkheden tot aanpassen van het Live OS

Je kunt je eigen aangepaste Live OS creëren door het veranderen van bepaalde eigenschappen in het configuratiebestand '`make_slackware_live.conf`'. Sommige van de dingen die je kunt veranderen zijn:

- De naam van de Desktop-variant (het script zelf kent ”*SLACKWARE*”, “*PLASMA5*”, “*XFCE*”, “*MATE*” en “*CINNAMON*”),
- De naam van het gebruikersaccount (standaard is dat “*live*“),
- De lijst(en) van de pakketten die je gebruikt voor je aangepaste distributie,
- De naam van de distributie (standaard is dat ”*slackware*”),
- En tenslotte kun je een functie definiëren '`custom_config()`' waar je al je custom post-installatie stappen kunt toevoegen die niet zijn opgenomen in het '`make_slackware_live.sh`' script.

Dit is het gedeelte in “make_slackware_live.conf' dat zich bezighoudt met deze aanpassingen. Twee variabelen zijn vereist als je je eigen aangepaste Live OS wilt maken: '**`LIVEDE `**' en '**`SEQ_CUSTOM`**'. De rest is optioneel (maar meestal toch nuttig):

```
# VERPLICHT:
# Definieer een nieuwe naam voor je eigen variant van Slackware Live
Edition:
#LIVEDE="CINELERRA"

# VERPLICHT:
# Je eigen aangepaste pakket sequentie voor een aangepaste Live ISO.
# In dit voorbeeld zou je twee bestanden moeten creëren,
# "pkglists/cinelerra.conf" en "pkglists/cinelerra.lst" die respectievelijk
# de  pakket locatie en pakket lijst definiëren.
#SEQ_CUSTOM="min,xbase,xapbase,xfcebase,cinelerra"

# OPTIE:
# Gebruik een andere naam dan "min" voor de pakkettenlijst die de generieke
kernel bevat:
#MINLIST="min"
```

```
# OPTIE:
# Je aangepaste distro naam (zichtbaar in het opstartscherm en
bestandsnamen):
#DISTRO="cinelerra"

# OPTIE:
# Naam van het 'live' gebruikersaccount in de Live omgeving:
#LIVEUID="live"

# OPTIE:
# Marker gebruikt voor het vinden van de Slackware Live bestanden:
#MARKER="CINELERRA"

# OPTIE:
# Het bestandssysteem label van de ISO:
#MEDIALABEL="CINELERRA"

# OPTIE:
# De ISO hoofdmap:
#LIVEMAIN="cinelerra"

# OPTIE:
# Voeg je eigen Live OS aanpassingen toe aan de functie custom_config():
#custom_config() {
#  # Voeg je eigen code hier toe, als je functionaliteit nodig hebt
#  # die niet wordt gedekt door het hoofdscript:
#}
```

Interne werking van Slackware Live Edition

Overlayfs, squashfs

Overlayfs en squashfs zorgen voor de echte magie hier. Zoals eerder uitgelegd, maakt het squashfs programma uit een directory structuur een enkel gecomprimeerd archiefbestand. Het doet dit op een speciale manier, eerder zoals mkisofs dit doet dan zoals tar een archief creëert. De resulterende module kan worden ge”loop“mount om het bestandssysteem er binnen in te benaderen.

Wanneer je een aantal van deze loop-mounted squashfs modules hebt, en elk bevat een deel van het bestandssysteem van het Live OS, ga je vervolgens deze fractionele bestandssystemen “op elkaar stapelen” om zodoende het volledige bestandssysteem op te bouwen (net als Kuifje deed in Het Geheim van de Eenhoorn, toen hij verschillende doorschijnende stukken perkament over elkaar heen legde en de vellen tegen het licht hield om het volledige beeld te zien). Overlayfs is de kernelmodule die deze 'superpositie' van de fractionele bestandssystemen uitvoert. Overlayfs stapelt een of meer read-only fractionele bestandssystemen en precies één beschrijfbaar bestandssysteem op elkaar. Dit beschrijfbare bestandssysteem is wat het Live OS de schijn geeft dat het naar een harde schijf klan schrijven - de schrijfacties naar het overlay filesystem worden echter gedaan naar RAM (in dat geval heb je een echt Live OS) of naar een 'persistent' bestandssysteem dat op een beschrijfbaar medium ligt (een Live USB stick met “persistence” is een voorbeeld van deze situatie).

De initrd en het init script

De initiele ram disk (initrd) die wordt gebruikt voor de Slackware Live Edition is een standaard Slackware initrd image dat gemaakt wordt met Slackware's “mkinitrd” programma, met slechts één wijziging: het ”/init“ script. De juiste naam voor een 'initrd' is tegenwoordig overigens 'initramfs', een afkorting van “initiëel ram filesysteem”, omdat dit het eerste bestandssysteem bevat dat in het kernelgeheugen wordt geladen wanneer de computer opstart. De voornaamste functie van het init script van een initrd is om het “root” bestandssysteem klaar te maken vóór de daadwerkelijke start van het OS begint. Wanneer het besturingssysteem dan wordt gestart, krijgt het een “root” bestandssysteem aangeboden dat klaar voor gebruik is. Een voorbeeldgeval is natuurlijk een LUKS-versleuteld “root” bestandssysteem dat eerst middels een wachtwoord moet worden ontsleuteld. Een ander voorbeeld is een “root” bestandssysteem dat is opgeslagen op logische volumes (LVM). Er is ook dan wat werk vooraf nodig om het “root” bestandssysteem toegankelijk te maken en te kunnen beginnen met het echte “init” programma van het OS (PID 1). Net als in de voorgaande voorbeelden heb je een initializatie-script in een initrd nodig om het “root” bestandssysteem van een Live OS beschikbaar te maken. Slackware's eigen standaard initrd ondersteunt geen Live omgeving. Daarom is het standaard “init” script uit het “mkinitrd” pakket uitgebreid met functionaliteit die noodzakelijk is voor “liveslak”.

Wat doet het 'liveslak' init script?

- Het analyseert de boot parameters die je hebt ingevoerd (of werden doorgegeven door syslinux / grub) en haalt daar de relevante informatie uit.
- Het voert een aantal standaard initialisatie stappen uit net als het standaard Slackware init script (het start udev, pauzeert even om het USB subsysteem de kans te geven om apparaten te vinden, laadt kernel modules, laadt toetsenbord mapping bestand, initialiseert RAID en LVM volumes etc) voordat het toekomt aan de sectie die de Slackware Live functionaliteit bevat.
- Een RAM-gebaseerd bestandssysteem wordt gecreëerd en dat vormt de basis van al wat daarna komt.
- Middels programma's zoals 'blkid' en 'mount' probeert het init script het Live medium (CDROM, DVD, USB stick etc) te vinden. Het maakt gebruik van 'blkid' om te zoeken naar het label van het bestandssysteem van het Live medium . Als dat niet lukt (omdat je het label had gewijzigd) zal het blkid gebruiken om alle bestandssystemen op de beschikbare partities te vinden en ze één voor één te mounten en te inspecteren totdat de Live partitie is gevonden.
- Nadat het Live medium gelokaliseerd is, worden de squashfs modules één voor één ge”loop“mount en toegevoegd aan het overlay filesystem, in de juiste volgorde. Als je de 'toram' boot parameter had opgegeven, dan wordt iedere module eerst naar RAM gekopieerd worden voordat deze wordt ge”loop“mount. Hierdoor kun je het boot medium verwijderen na het opstarten, omdat het volledige OS in RAM wordt uitgevoerd.
- Modules zullen in volgorde worden geladen:
  - Eerst de systeem modules (essentiele modules in de system/ subdirectory)
  - Dan de niet-essentiele modules (in de addons/ directory). Als je niet wilt dat een addon wordt geladen, kun je “noload=modulename” meegeven op de syslinux/grub boot commandoregel.
  - Tot slot worden de optionele modules (in de optional/ subdirectory) geladen. Standaard wordt een optionele module niet geladen tenzij je het laden forceert door het toevoegen van “load=modulename” op de boot commandoregel.
- Vervolgens wordt persistentie geconfigureerd als het Live OS werd opgestart vanaf een beschrijfbaar medium zoals een USB stick. Het init script zal daartoe eerst zoeken naar een directory in de root van de Live partitie van de USB stick genaamd “persistence” en gebruikt dat om blijvende veranderingen in het Live bestandssysteem op te slaan. Als die map niet

bestaat, maar een bestand ”/persistence.img“ wordt wel gevonden, dan zal dat bestand worden ge”loop“mount en persistente wijzigingen van het Live OS worden naar het bestandssysteem in dit “container” bestand weggeschreven. De ”/persistence.img“ container kan LUKS versleuteld zijn, in welk geval het init script dat detecteert en jou vraagt om een wachtwoord voor de ontsleuteling.

- Het overlay bestandssysteem wordt dan gecompleteerd door het toevoegen van een beschrijfbare toplevel directory structuur (persistent of RAM-gebaseerd).
- Het complete RAM bestandssysteem dat aan de overlay ten grondslag ligt wordt ter beschikking gesteld aan de gebruiker van het Live OS als ”/mnt/live“
- Het bestandssysteem van het Live medium wordt eveneens ter beschikking gesteld aan de gebruiker van het Live OS; als ”/mnt/livemedia“. Als het medium een USB stick is, dan zul je schrijftoegang tot ”/mnt/livemedia“ hebben.
- Nu dat het “root” bestandssysteem compleet is, wordt het Live OS geconfigureerd voordat het daadwerkelijk opgestart kan worden:
    - Als de boot commandoregel het woord “swap” bevat, worden alle beschikbare swap partities aan ”/etc/fstab“ toegevoegd in het Live OS.
    - Als je een aangepaste toetsenbordindeling voor de console hebt opgegeven (en eventueel nog een andere indeling voor X) met behulp van de “kbd” en “xkb” boot parameters dan worden deze ingesteld in ”/etc/rc.d/rc.keymap“ en ” /etc/X11/xorg.conf.d/30-keyboard.conf “in het live OS.
    - Hetzelfde geldt voor eventuele aangepaste localizatie die werd opgegeven met de “locale” parameter, dit zal worden toegevoegd aan ”/etc/profile.d/lang.sh“.
    - Als timezone en hardware klok in de parameter “tz” zijn opgegeven, zullen deze worden geconfigureerd in ”/etc/localtime“ en ”/etc/HARDWARECLOCK“.
    - Met hulp van de boot parameters “livepw” en “rootpw” kun je aangepaste wachtwoorden opgeven voor de 'live' en 'root' gebruikers accounts; de standaardwaarden voor deze twee wachtwoorden zijn gewoon 'live' en 'root'. Deze aanpassing wordt bereikt door het uitvoeren van het “chpasswd” commando in het ge”chroot“e overlay zodat een leesbare tekst als wachtwoord kan worden meegegeven.
    - De “hostname” boot parameter kan worden gebruikt om de hostnaam van het Live OS te veranderen van de standaard naam “darkstar”. Configuratie wordt weggeschreven naar ”/etc/HOSTNAME“ en ”/etc/NetworkManager/NetworkManager.conf“.
    - Als de “blacklist” boot parameter is opgegeven, dan worden de kernel modules die daarin worden genoemd toegevoegd aan een modprobe blacklist bestand ”/etc/modprobe.d/BLACKLIST-live.conf“.
    - Het ”/var/lib/alsa/asound.state“ bestand in het Live OS wordt verwijderd zodat een goede audioconfiguratie kan worden gegenereerd voor elke computer waarop het Live medium wordt opgestart.
    - De volledige inhoud van de ”/liveslak/rootcopy“ directory op de Live partitie (kan leeg zijn) wordt gekopieerd naar de root van het bestandssysteem van het Live OS, daarmee eventueel bestanden in het Live OS 'overschrijvend'. Gebruik de /liveslak/rootcopy directory om je eigen configuratie toe te voegen aan je Live OS als je het start vanaf een USB stick.
    - En tot slot, maar wel zo belangrijk, worden alle LUKS-versleutelde container bestanden ontgrendeld (het init script zal je vragen naar het wachtwoord) en de bestandssystemen die zich daarin bevinden worden in het Live OS gemount. Op dit moment wordt een LUKS-versleutelde /home ondersteund. De bestanden ”/etc/fstab“ en ”/etc/crypttab“ zullen worden bijgewerkt, zodat het Live OS het mounten en unmounten voor rekening kan nemen.
    - Het init script eindigt met het overdragen van ons nieuwe root-bestandssysteem (de overlay) aan de kernel om daarna het Slackware init programma (PID 1, /sbin/init) te

starten.
- Vanaf dit moment start een 'normaal' Slackware systeem en het feit dat alle bewerkingen worden uitgevoerd in RAM en niet op je lokale harde schijf is niet merkbaar.

Slackware Live module format

Een Slackware Live module bevat een directory structuur, die is 'samengeperst' in een gecomprimeerd archiefbestand door het programma “squashfs”. Het compressie-algoritme dat wordt gebruikt is “XZ”, waarmee de keuze voor de bestandsextensie van de module ”.sxz“ betekenis krijgt: “squashed with xz”.

Slackware Live Edition vereist van de squashfs modules dat ze zich houden aan een conventie voor de bestandsnaam:

- Het bestandsnaam formaat is “NNNN-modnaam-*.sxz”, waarbij 'N' een cijfer is en 'modnaam' mag geen streepje '-' bevatten.
- Het “modnaam” deel is wat je als waarde moet opgeven in de boot parameters “load” en “noload”.
- Het segment '*' mag een willekeurige karakter reeks zijn, maar het meest gebruikt wordt “${VERSION}-${ARCH}”. De systeem-modules van Slackware Live gebruiken het Slackware versienummer in ${VERSION} en de Slackware architectuur in ${ARCH}. Voor de modules in addons/ en optional/ subdirectories zal ${VERSION} gewoonlijk gelijk zijn aan de versie van het programma dat in de module verpakt zit.
- De vier cijfers van een modulenaam hebben een betekenis. Sommige cijferreeksen worden geclaimd door het Slackware Live OS, gebruik die dus niet voor je eigen modules. Deze reeksen zijn gebaseerd op de pakketbron:

```
    0000 = bevat de Slackware "/boot" directory
    0010-0019 = pakketten geïnstalleerd vanuit een Slackware tagfile
(a,ap,d,...,y serie)
    0020-0029 = pakketten geïnstalleerd vanaf een pakket lijst zoals
gevonden in de submap ./pkglists van de liveslak broncode (min, xbase,
xapbase, xfcebase etc)
    0030-0039 = een 'lokaal' pakket, dat wil zeggen een pakket gevonden
in subdirectory ./local of ./local64 (afhankelijk van de architectuur)
    0099 = liveslak configuratie module (bevat alle aanpassingen aan de
geïnstalleerde pakketten die deze veranderen in een bruikbaar Live OS)
```

- Andere bereiken kunnen vrijelijk worden gebruikt. Merk op dat de volgorde waarin het bestandssysteem van het Live OS wordt samengesteld door het stapelen van directorystructuren in de squashfs modules afhangt van de module nummering. Dus als modules in bepaalde volgorde geladen moeten worden, zorg er dan gewoon voor dat hun bestandsnamen oplopende nummers hebben.

1. Voorbeeld van een systeem module: 0099-slackware_zzzconf-14.2-x86_64.sxz
2. Voorbeeld van een optionele module: 0060-nvidia-352.79_4.4.12-14.2-x86_64.sxz

Andere Slackware gebaseerde Live distributies

Natuurlijk gingen veel mensen mij voor, en omdat ik als een n00b in Linux Live land begon, heb ik veel geleerd over hoe een Live distro werkt door simpelweg te “spelen” met deze andere Slackware gebaseerde Live distributies. Sta mij toe om ze te noemen en respect te tonen:

SLAX

Website: https://www.slax.org/

SLAX was de oorspronkelijke Live spinoff van Slackware. De linux-live scripts die worden gebruikt om een SLAX ISO maken werden gegeneraliseerd, zodat ze een Live versie kunnen creëren van een willekeuring Linux besturingssysteem dat al op een harde schijf is voorgeïnstalleerd. De ontwikkeling van SLAX leek een paar jaar geleden te zijn gestopt, maar de ontwikkelaar lijkt recentelijk de draad te hebben opgepakt.

De Live functionaliteit van SLAX is gebaseerd op aufs en unionfs. Deze bestandssystemen vereisen een op maat gemaakte kernel waar aufs ondersteuning in gecompileerd is. Deze distro heeft een kleine omvang en zijn opstart scripts zijn herschreven met een focus op betere opstart snelheid.

Porteus

Website: http://ww.porteus.org/

Porteus werd opgericht als een afsplitsing van SLAX door de SLAX gemeenschap toen de ontwikkeling van SLAX leek te zijn beëindigd. Porteus heeft een actieve community van gebruikers waar het allemaal “draait om de modules”. Het gebruik van aufs plaats van overlayfs stelt Porteus in staat (net zoals SLAX) om squashfs modules nar believen in het lopende Live systeem toe te voegen en te verwijderen. Deze “killer feature” heeft de ontwikkeling van een groot aantal community modules aangewakkerd. Het lijkt erop dat de volgende generatie van Porteus zal worden gebaseerd op Arch Linux in plaats van Slackware; dit heeft te maken met het feit dat de oorspronkelijke Porteus ontwikkelaar het team verlaten heeft.

Salix Live

Website: http://www.salixos.org/download.html

Salix is een distributie op basis van Slackware met zijn eigen filosofie van “één programma per taak”. Hierdoor kon het aantal pakketten flink gereduceerd worden in vergelijking met de moeder distro, Slackware. Salix heeft het controleren van pakket-afhankelijkheden geïmplementeerd in zijn package management programma. Diverse Live edities van Salix zijn verkrijgbaar, die elk gebouwd zijn rondom en gericht op een andere Desktop Environment of Window Manager. Live-edities zijn beschikbaar voor XFCE en MATE.

Slackel

Website: http://www.slackel.gr/

Slackel is een Griekse distro op basis van zowel Slackware als Salix. Het wordt aangeboden in drie edities, die elk een Live variant hebben: KDE4, Openbox en Fluxbox. De Live scripts zijn een aanpassing van Salix-Live.

SlackEX

Website: http://slackex.exton.net/

Een website met live versies op basis van vele reguliere Linux-distributies. De SlackEX versie is losjes gebaseerd op Slackware met een aangepaste kernel en een aantal tools die geen deel uitmaken van Slackware zelf. Ik was niet in staat om de bronnen voor deze live distro te vinden.

Liveslak Bronnen

Slackware Live Edition wordt gecreëerd door de 'liveslak' scripts ontwikkeld en onderhouden door Eric Hameleers aka Alien BOB alien@slackware.com.

- Git repository: git://bear.alienbase.nl/liveslak.git
- Git repository (doorzoekbaar): http://bear.alienbase.nl/cgit/liveslak/
- Download mirror: http://www.slackware.com/~alien/liveslak/

Bronnen

- Oorspronkelijke bron: http://bear.alienbase.nl/cgit/liveslak/tree/README.txt
- Oorspronkelijk geschreven door Eric Hameleers

slackware, live, author alienbob

From:
https://docs.slackware.com/ - **SlackDocs**

Permanent link:
https://docs.slackware.com/nl:slackware:liveslak

Last update: **2016/11/29 21:31 (UTC)**