

Debian Live Manual

Debian Live Project [\[debian-live@lists.debian.org\]](mailto:debian-live@lists.debian.org)

2015-08-23

Debian Live Manual

Debian Live Project debian-live@lists.debian.org

Copyright © 2006-2015 Live Systems Project, Copyright © 2016-2025 The Debian Live team

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

The complete text of the GNU General Public License can be found in `/usr/share/common-licenses/GPL-3` file.

Contents	Usuari	10
Debian Live Manual	i	11
Sobre aquest manual	3	
Sobre aquest manual	4	
1. Sobre aquest manual	4	
1.1 Per als impacients	4	
1.2 Termes	4	
1.3 Autors	5	
1.4 Contribuir a aquest document	6	
1.4.1 Aplicar canvis	6	
1.4.2 Traducci	6	
Sobre el Debian Live Project	8	
2. Sobre el Debian Live Project	8	
2.1 Motivaci	8	
2.1.1 Qu passa amb els sistemes vius actuals	8	
2.1.2 Per qu crear el nostre prpi sistema viu?	8	
2.2 Filosofia	8	
2.2.1 Only unchanged packages from Debian main and non-free-firmware	8	
2.2.2 Paquets del sistema viu sense cap configuraci	9	
2.3 Contacte	9	
3. Instal·laci		11
3.1 Requisits		11
3.2 Instal·laci de live-build		11
3.2.1 Des del repositori de Debian		11
3.2.2 A partir del codi font		11
3.3 Instal·laci de live-boot i live-config		12
3.3.1 Des del repositori de Debian		12
3.3.2 A partir del codi font		12
Conceptes bsics		14
4. Conceptes bsics		14
4.1 Qu s un sistema viu?		14
4.2 Descarrega d'imatges prefabricades		15
4.3 Primers passos: construcci d'una imatge ISO hbrida		15
4.4 Usar una imatge ISO hbrida en viu		15
4.4.1 Gravar una imatge ISO en un medi fsic		16
4.4.2 Copiar una imatge ISO hbrida en un dispositiu USB		16
4.4.3 Utilitzar l'espai lliure en una memria USB		16
4.4.4 Arrencar el medi en viu		16
4.5 Utilitzar una mquina virtual per a fer proves		17
4.5.1 Provar una imatge ISO amb QEMU		17
4.5.2 Provar una imatge ISO amb VirtualBox		18
4.6 Construir i utilitzar una imatge HDD		18
4.7 Construir una imatge netboot		19
4.7.1 Servidor DHCP		20

4.7.2 Servidor TFTP	20	Personalitzaci dels continguts	27
4.7.3 Servidor NFS	20	7. Visi general de la personalitzaci	27
4.7.4 Com provar l'arrencada en xarxa	21	7.1 Configuraci durant la construcci vs. durant l'arrencada	27
4.7.5 Qemu	21	7.2 Etapes de la construcci	27
4.8 Webbooting	21	7.3 Suplementar lb config amb fitxers	27
4.8.1 Obtenir els fitxers webboot	21	7.4 Tasques de personalitzaci	28
4.8.2 Arrencar imatges webboot	21		
		Personalitzaci de la installaci de paquets	29
Descripci general de les eines	23	8. Personalitzaci de la installaci de paquets	29
5. Descripci general de les eines	23	8.1 Fonts dels paquets	29
5.1 El paquet live-build	23	8.1.1 Distribuci, rees d'arxiu i mode	29
5.1.1 L'ordre lb config	23	8.1.2 Miralls de distribuci	30
5.1.2 L'ordre lb build	24	8.1.3 Miralls de distribuci utilitzats en temps de construcci	30
5.1.3 L'ordre lb clean	24	8.1.4 Miralls de distribuci utilitzats en temps d'execuci	30
5.2 El paquet live-boot	24	8.1.5 Repositoris addicionals	30
5.3 El paquet live-config	24	8.2 Selecci dels paquets a installar	31
		8.2.1 Llistes de paquets	31
Gesti d'una configuraci	25	8.2.2 s dels metapaquets	31
6. Gesti d'una configuraci	25	8.2.3 Llistes locals de paquets	32
6.1 Gestionar canvis en la configuraci	25	8.2.4 Llistes locals de paquets per a l'etapa binary	32
6.1.1 Per qu utilitzar scripts auto? Qu fan?	25	8.2.5 Generar llistes de paquets	32
6.1.2 Utilitzar scripts auto d'exemple	25	8.2.6 s de condicionals dins de les llistes de paquets	33
6.2 Clonar una configuraci publicada via Git	26	8.2.7 Eliminar paquets durant la installaci	33
		8.2.8 Summary	33
		8.2.9 Tasques d'escriptori i llenguatge	34
		8.2.10 Tipus i versi del nucli	34
		8.2.11 Nuclis personalitzats	35

8.3	Installaci de paquets modificats o de tercers	35	10.3	Persistncia	45
8.3.1	Fer servir packages.chroot per a instaar paquets personalitzats	36	10.3.1	El fitxer persistence.conf	46
8.3.2	Fer servir un repositori APT per a instal·lar paquets personalitzats	36	10.3.2	Utilitzar diversos medis persistents	46
8.3.3	Paquets personalitzats i APT	36	10.3.3	Persistncia amb xifratge	47
8.4	Configurar APT en temps de construcci	37		Personalitzaci de la imatge binria	49
8.4.1	Elegir apt o aptitude	37	11.	Personalitzaci de la imatge binria	49
8.4.2	L's d'un proxy amb APT	37	11.1	Carregadors d'arrencada	49
8.4.3	Afinar APT per a estalviar espai	37	11.2	metadades ISO	49
8.4.4	Passar opcions per a apt o aptitude	38		Personalitzaci de l'installador de debian	50
8.4.5	APT pinning	38	12.	Personalitzaci de l'installador de debian	50
	Personalitzaci dels continguts	40	12.1	Tipus d'installador de Debian	50
9.	Personalitzaci dels continguts	40	12.2	Personalitzaci de l'installador de Debian amb preconfiguraci	51
9.1	Includes	40	12.3	Personalitzar el contingut de l'installador de Debian	51
9.1.1	Live/chroot local includes	40		Projecte	52
9.1.2	Binary local includes	41		Contribuir al projecte	53
9.2	Scripts ganxo (Hooks)	41	13.	Contribuir al projecte	53
9.2.1	Chroot local hooks	41	13.1	Traducci de pàgines dels manuals	53
9.2.2	Binary local hooks	41			
9.2.3	Scripts ganxo en temps d'arrencada	41			
9.3	Preconfiguraci de les preguntes de Debconf	42			
	Personalitzaci dels comportaments en temps d'execuci	43			
10.	Personalitzaci dels comportaments en temps d'execuci	43			
10.1	Personalitzar l'usuari en viu	43			
10.2	Personalitzaci de l'entorn local i el llenguatge	43			

Informar dels errors	54	Exemples	61
14. Informar dels errors	54	16. Exemples	61
14.1 Problemes coneguts	54	16.1 s dels exemples	61
14.2 Fer la recerca	54	16.2 Tutorial 1: Una imatge per defecte	61
14.3 Reconstruir des de zero	54	16.3 Tutorial 2: Una utilitat de navegador web	61
14.4 Fer servir paquets actualitzats	55	16.4 Tutorial 3: Una imatge personalitzada	62
14.5 Recopilar informaci	55	16.4.1 Primera revisi	62
14.6 Allar el cas que falla, si s possible	55	16.4.2 Segona revisi	63
14.7 Utilitzar el paquet correcte per a informar de l'error	55	16.5 Un client per a un quiosc VNC	64
14.7.1 A l'hora de construir mentre bootstrapping	56	16.6 A minimal image for a 512MB USB key	64
14.7.2 A l'hora de construir, durant la installaci de paquets	56	16.7 Un escriptori GNOME localitzat i amb installador	65
14.7.3 En el moment d'arrencar	56	Apndix	67
14.7.4 En temps d'execuci	56	Style guide	68
14.8 On informar dels errors	56	17. Guia d'estil	68
Estil de Codi	57	17.1 Instruccions per als autors	68
15. Estil de Codi	57	17.1.1 Caracterstiques lingstiques	68
15.1 Compatibilitat	57	17.1.2 Procediments	69
15.2 Indentaci	57	17.2 Directrius per als traductors	71
15.3 Ajust de lnia	57	17.2.1 Consells de traducci	71
15.4 Variables	58	SiSU Metadata, document information	73
15.5 Miscellnia	59		
Exemples	60		

₁ Debian Live Manual

₂ Sobre aquest manual

Sobre aquest manual

1. Sobre aquest manual

This manual serves as a single access point to all documentation related to the Debian Live Project and in particular applies to the software produced by the project for the Debian bookworm release. An up-to-date version can always be found at <https://live-team.pages.debian.net/live-manual/>

Si b live-manual es centra principalment en ajudar a construir un sistema viu i no en temes dels usuaris finals, un usuari final pot trobar informació en aquestes seccions: **Conceptes bsics** abasta la descarga d'imatges prefabricades i la preparació de les imatges per a ser arrencades des dels dispositius o la xarxa, ja sigui utilitzant el servei de construcció d'imatges web o executant live-build directament en el sistema. **Personalització dels comportaments en temps d'execució** descriu algunes de les opcions que es poden especificar durant l'arrencada del sistema, com ara la selecció de la disposició del teclat, la configuració regional i l's de la persistència.

Algunes de les ordres esmentades en el text s'han d'executar amb privilegis de superusuari que es poden obtenir esdevenint l'usuari root amb su o mitjanant l's de sudo. Per a distingir entre les ordres que poden ser executades per un usuari sense privilegis i aquelles que requereixen privilegis de superusuari, s'anteposa \$ o # respectivament. Aquest símbol no s'apart de l'ordre.

1.1 Per als impacients

Si b creiem que tot el que hi ha en aquest manual s'important, si ms no, per a alguns dels nostres usuaris, ens adonem que hi ha una

gran quantitat de material per a cobrir i que s'possible que es vulgui experimentar l'xit amb el programari aviat, abans d'aprofundir en els detalls. Per tant, us recomanem llegir en el segent ordre.

En primer lloc, llegir aquest capítol, **Sobre aquest manual**, des del principi i acabant amb els **Termes**. A continuació, saltar als tres tutorials abans dels **Exemples**, secció pensada per a ensenyar com fer la construcció d'una imatge i alguns aspectes bsics de la personalització. Llegir en primer lloc **s dels exemples** seguit pel **Tutorial 1: Una imatge per defecte**, **Tutorial 2: Una utilitat de navegador web** i finalment, **Tutorial 3: Una imatge personalitzada**. Al final d'aquests tutorials, es tindrà una idea del que es pot fer amb els sistemes en viu.

Us animem a tornar i a fer un estudi del manual en profunditat, la pròpia lectura pot ser **Conceptes bsics**, fregant o saltant **Construir una imatge netboot**, i acabant amb la lectura de la **Visió general de la personalització** i els capítols que segueixen. En aquest punt, esperem que estigueu ben emocionats pel que es pot fer amb els sistemes en viu i motivats per llegir la resta del manual, de principi a fi.

1.2 Termes

Sistema viu : Un sistema operatiu que pot arrencar sense necessitat d'instal·lació en un disc dur. Els sistemes vius no alteren el sistema operatiu local(s) o els fitxer(s) ja instal·lats al disc dur de l'ordinador a menys que així se'ls ho indiqui. Els sistemes vius normalment s'inicien des de dispositius, com ara CDs, DVDs o memòries USB. Alguns també poden arrencar des de la xarxa (amb les imatges netboot, veure **Construir una imatge netboot**), o mitjanant internet (amb el paràmetre fetch=URL, veure **Webbooting**).

Medi en viu : A diferència de sistema en viu, el medi en viu

es refereix al CD, DVD o memria USB on es copia el fitxer binari produït per live-build i utilitzat per a arrancar el sistema en viu. Més a més, el terme també es refereix a qualsevol lloc on resideix el fitxer binari als efectes d'iniciar el sistema en viu, com ara la ubicació dels fitxers per a arrancar en xarxa.

Debian Live Project : El projecte que manté, entre altres, els paquets live-boot, live-build, live-config, live-tools i live-manual.

Sistema amfitrió : L'entorn utilitzat per a crear el sistema en viu.

Sistema objectiu : L'entorn que s'utilitza per a executar el sistema en viu.

live-boot : Una col·lecció de scripts per a arrancar els sistemes vius.

live-build : Una col·lecció de scripts utilitzats per a construir sistemes en viu personalitzats.

live-config : Una col·lecció de scripts utilitzats per a configurar un sistema en viu durant el procés d'arrencada.

live-tools : Una col·lecció de scripts addicionals que s'utilitzen per a realitzar tasques típics en un sistema viu en execució.

live-manual : Aquest document s'ha mantingut en un paquet anomenat live-manual

Instal·lador de Debian (d-i) : El sistema oficial d'instal·lació de la distribució Debian.

Paràmetres d'arrencada : Els paràmetres que es poden introduir a l'indicador del carregador d'arrencada per a influir en el nucli o live-config

chroot : El programa chroot, chroot(8), ens permet executar diferents instàncies d'un entorn GNU/Linux a la vegada en un sol sistema sense reiniciar.

Binary image : A file containing the live system, such as live-image-amd64.hybrid.iso or live-image-amd64.img.

Distribució objectiu : La distribució en què es basa el sistema en viu. Que pot diferir de la distribució del sistema amfitrió.

stable/testing/unstable : La distribució estable, actualment anomenada bookworm, conté l'última distribució de Debian l'última oficialment. La distribució testing, temporalment anomenada trixie, s'ha creat d'assaig per a la propera versió estable. Un avantatge important de les d'aquesta distribució és que totes les versions més recents del programari en relació amb l'edició estable. La distribució unstable, permanentment anomenada sid, és on es produeix el desenvolupament actiu de Debian. En general, aquesta distribució s'utilitza pels desenvolupadors i els que els agrada viure en risc. Al llarg del manual, es tendeix a utilitzar els seus noms en clau, com ara trixie o sid, ja que són els que fan servir les pròpies eines.

1.3 Autors

Llistat d'autors (en ordre alfabètic)

Ben Armstrong

Brendan Sleight

Carlos Zuberri

Chris Lamb

Daniel Baumann

Franklin Piat

Jonas Stein

Kai Hendry

Marco Amadori

Mathieu Geli

Matthias Kirschner

Richard Nelson

Roland Clobus

Trent W. Buck

amb `PROOF=2` es construeix live-manual en format pdf, per noms el retrat A4 i carta. s per aix que amb l's de qualsevol de les possibilitats `PROOF=` es pot estalviar una quantitat considerable de temps, per exemple:

1.4 Contribuir a aquest document

Aquest manual est pensat com un projecte comunitari i totes les propostes per a millorar-lo i les contribucions sn molt benvingudes. Veure la secci **Contribuir al projecte** per a obtenir informaci detallada sobre com obtenir la clau i fer bons commits.

```
$ make build PROOF=1
```

Quan es revisa una de les traduccions, s possible construir un sol idioma mitjanant l'execuci de, per exemple:

```
$ make build LANGUAGES=de
```

1.4.1 Aplicar canvis

Per tal de realitzar canvis en el manual angls s'ha d'editar els fitxers adequats a manual/en/ per abans de presentar una contribuci, s'ha de previsualitzar el treball. Per a previsualitzar el live-manual, assegurar-se que s'han instal·lat els paquets necessaris per a la seva construcci mitjanant l'execuci de:

Tamb es possible crear per tipus de document, per exemple:

```
$ make build FORMATS=pdf
```

O combinar tot dos, per exemple:

```
# apt-get install make po4a ruby ruby-nokogiri sisu-complete
```

```
$ make build LANGUAGES=de FORMATS=html
```

Es pot crear el live-manual des del directori de nivell superior del arbre Git mitjanant l'execuci de:

```
$ make build
```

Desprs de revisar el treball i assegurar-se que tot est b, no fer un `make commit` a menys que s'actualitzin les traduccions al mateix temps, i en aquest cas, no barrejar els canvis al manual angls i les traduccions en el mateix commit, fer-ne un altre separat per a cada canvi. Veure la secci **Traducci** per a ms detalls.

Tenint en compte que es necessita un cert temps per construir el manual en tots els idiomes suportats, els autors poden utilitzar un dels mtodes abreujats per fer revisions rpides de la nova documentaci que han afegit al manual en angls. Amb `PROOF=1` es construeix live-manual en format html, per sense els fitxers html segmentats, i

1.4.2 Traducci

Nota: Per traduir les pgines dels manuals, veure **Traducci de pgines dels manuals**

Per a traduir live-manual, seguir aquests passos, depenent de si s'est començant una traducció des de zero o si es segueix treballant en una ja existent:

Començar una nova traducció des de zero

Traduir els fitxers `about-manual.ssi.pot`, `about-project.ssi.pot` i `index.html.in.pot` de `manual/pot/` a la vostra llengua amb el vostre editor favorit (per exemple poedit). Enviar els fitxers `.po` traduïts a la llista de correu per comprovar la seva integritat. La comprovació d'integritat de live-manual garanteix que els fitxers `.po` són traduïts al 100% per també detecta possibles errors.

Once checked, to enable a new language in the autobuild it is enough to add the initial translated files to `manual/po/$-LANGUAGE"/` and edit `manual/sisu/home/index.html` adding the name of the language and its name in English between brackets. And then, add the folder `manual/$-LANGUAGE"/` to the file `.gitignore`. Finally, run `make commit`.

Continuar amb una traducció ja començada

Una vegada que s'ha afegit la nova llengua, es pot continuar traduint la resta de fitxers `.po` dins de `manual/po/$-LANGUAGE"/` a l'atzar amb l'editor favorit (com per exemple poedit).

No oblidar que es necessari fer un `make commit` per a garantir que els manuals traduïts s'actualitzin a partir dels fitxers `.po` i llavors es poden revisar els canvis executant `make build` abans de `git add .`, `git commit -m Translating...` i `git push`. Recordar que `make build` pot trigar una quantitat considerable de temps, per es poden revisar els idiomes de forma individual com s'explica a la secció [Aplicar canvis](#)

Després d'executar `make commit` es podrà veure bastant text a la pantalla. Bàsicament són missatges informatius sobre l'estat del procés i també algunes pistes sobre el que es pot fer per a millorar live-manual. Si no es veu cap error fatal, generalment es pot procedir i enviar la contribució.

live-manual ve amb dues utilitats que poden ser de gran ajuda pels traductors a l'hora de trobar missatges sense traduir i difusos. La primera és `make translate`. Aquesta activa un script que diu en detall quants missatges sense traduir hi ha a cada fitxer `.po`. La segona, `make fixfuzzy`, només actua sobre els missatges difusos per ajudar a trobar-los i corregir-los un per un.

Tenir en compte que tot i que aquestes utilitats poden ser molt útils per a fer traduccions en la llista d'ordres, l'ús d'una eina especialitzada com poedit és la manera recomanada de fer la tasca. També és una bona idea llegir la documentació de localització de Debian (l10n) i, específicament dins live-manual, les [Directrius per a traductors](#).

Nota: Es pot utilitzar `make clean` per a netejar l'arbre git abans de fer un `push`. Aquest pas no és obligatori, gràcies al fitxer `.gitignore`, però és una bona pràctica per a evitar enviar fitxers de forma involuntària.

Sobre el Debian Live Project 87

2. Sobre el Debian Live Project 88

2.1 Motivaci

2.1.1 Qu passa amb els sistemes vius actuals

When Debian Live Project was initiated (around 2006), there were already several Debian based live systems available and they are doing a great job. From the Debian perspective most of them have one or more of the following disadvantages:

No sn projectes de Debian i per tant no tenen suport des de Debian.

Es barregen diferents distribucions, per exemple testing i unstable .

Noms donen suport a i386.

Es modifique el comportament i/o l'aparena dels paquets, per a estalviar espai.

S'inclouen paquets de fora de l'arxiu de Debian.

Inclouen nuclis personalitzats amb pedaos addicionals que no sn part de Debian.

Sn grans i lents a causa de la seva mida i per tant no aptes com a sistemes de rescat.

No estan disponibles en diferents formats, per exemple, CD, DVD, memries USB i imatges netboot.

2.1.2 Per qu crear el nostre prpi sistema viu?

Debian s el sistema operatiu universal: Debian t un sistema viu per a mostrar arreu i per a representar acuradament el sistema Debian amb els segents avantatges:

Es tracta d'un subprojecte de Debian. 89

Reflecteix l'estat (actual) d'una distribuci. 90

Funciona en tantes arquitectures com s possible. 91

Es tracta noms de paquets Debian sense modificacions. 92

No cont paquets que no sn a l'arxiu de Debian. 93

S'utilitza un nucli Debian sense alteracions i sense pedaos addicionals. 94

2.2 Filosofia 95

2.2.1 Only unchanged packages from Debian main and non-free-firmware 96

Noms farem servir els paquets del repositori de Debian de la secci main. La secci non-free no s part de Debian i per tant no es pot utilitzar per a les imatges oficials del sistema viu. 97

Starting with Debian 12 bookworm we added the [non-free-firmware](#) section for better support of modern hardware. 98

No canviarem cap paquet. Cada vegada que hgim de canviar alguna cosa, ho farem en coordinaci amb el mantenidor del paquet a Debian. 99

Com a excepci, els nostres propis paquets, com ara live-boot, live-build o live-config poden ser utilitzats temporalment des del nostre propi repositori per raons de desenvolupament (per exemple, per a crear instantnies de desenvolupament). Aquests paquets es pujaran a Debian de forma regular. 100

2.2.2 Paquets del sistema viu sense cap configuraci

En aquesta fase no es publicar o s'installar cap configuraci alternativa o d'exemple. Tots els paquets sn utilitzats en la seva configuraci per defecte, tal com sn desprs d'una installaci normal de Debian.

Cada vegada que ens calgui una configuraci per defecte diferent, ho farem en coordinaci amb el mantenidor del paquet Debian.

S'hi inclou un sistema per a configurar paquets mitjanant debconf que permet installar paquets configurats de forma personalitzada dins d'una imatge en viu, per per a les **imatges en viu prefabricades** noms utilitzarem una configuraci per defecte, a menys que sigui absolutament necessari per a poder treballar en l'entorn en viu. Sempre que sigui possible, preferim adaptar els paquets a l'arxiu de Debian perqu funcionin en un sistema en viu abans que realitzar canvis en la cadena d'eines en viu o en **les configuracions de les imatges prefabricades**. per a obtenir ms informaci, veure **Visi general de la personalitzaci**.

2.3 Contacte

Mailing list : The primary contact for the project is the mailing list at <https://lists.debian.org/debian-live/>. You can email the list directly by addressing your mail to debian-live@lists.debian.org. The list archives are available at <https://lists.debian.org/debian-live/>.

IRC : Un nombre d'usuaris i desenvolupadors estan presents al canal #debian-live a irc.debian.org (OFTC). Quan es pregunta al IRC, s'ha de tenir pacincia esperant una resposta, si no hi ha cap resposta, es pot enviar un correu a la llista.

BTS : **El****Informar dels errors**.

Installaci

3. Installaci

3.1 Requisites

Building live system images has very few system requirements for the host system:

Accs de superusuari (root)

Una versi actualitzada de live-build

Una shell compatible amb POSIX, com ara bash o dash

debootstrap

Linux 2.6 or newer

A mount point with dev and exec rights.

```
# mount -i your`mount`point -o dev,exec,remount
```

Tenir en compte que no cal usar Debian o una distribuci derivada de Debian ja que live-build funcionar en gaireb qualsevol distribuci amb els requisits anteriors.

3.2 Installaci de live-build

Es pot installar live-build en un nombre de maneres diferents:

Des del repositori Debian

A partir del codi font

A partir d'instants

Si s'utilitza Debian, la manera recomanada s'installar live-build des del repositori de Debian.

3.2.1 Des del repositori de Debian

Noms cal installar live-build com qualsevol altre paquet:

```
# apt-get install live-build
```

3.2.2 A partir del codi font

live-build es desenvolupa utilitzant el sistema de control de versions Git. En els sistemes basats en Debian, aix s'proporciona pel paquet git. Per a comprovar l'ltim codi, executar:

```
$ git clone https://salsa.debian.org/live-team/live-build.git
```

Es pot construir i installar un paquet Debian prpi mitjanant l'execuci de:

```
$ cd live-build
$ dpkg-buildpackage -b -uc -us
$ cd ..
```

Ara installar qualsevol dels fitxers .deb recent construïts que ens interessin, per exemple,

```
# dpkg -i live-build_4.0-1_all.deb
```

Es pot installar live-build directament al sistema mitjanant l'execuci de:

```
# make install
```


i desinstallar amb:

```
# make uninstall
```

3.3 Instal·lació de live-boot i live-config

Nota: No cal instal·lar live-boot o live-config en el sistema per a crear sistemes personalitzats en viu. No obstant, això no farà cap mal i servirà per a fins de referència. Si només es vol la documentació, ara es poden instal·lar els paquets live-boot-doc i live-config-doc per separat.

3.3.1 Des del repositori de Debian

Tots dos, live-boot i live-config, estan disponibles al arxiu de Debian, d'una manera similar a [Instal·lació de live-build](#).

3.3.2 A partir del codi font

Per a utilitzar les darreres fonts del git, es pot seguir el procés següent. Assegurar-se d'estar familiaritzat amb els termes esmentats a [Terminologia](#).

Clonar les fonts de live-boot i live-config

```
$ git clone https://salsa.debian.org/live-team/live-boot.git
$ git clone https://salsa.debian.org/live-team/live-config.git
```

Consultar les pàgines del manual de live-boot i live-config per a més detalls sobre la seva personalització si aquesta és la raó per a construir aquests paquets a partir de les fonts.

140 Crear els fitxers .deb de live-boot i live-config

S'ha de crear ja sigui en la distribució de destinació o en un chroot que contingui la plataforma de destinació: és a dir, si el objectiu és trixie llavors s'ha de construir contra trixie.

Es pot utilitzar un constructor personal, com ara pbuilder o sbuilder si es necessita crear live-boot per a una distribució de destinació diferent del sistema de construcció. Per exemple, per a les imatges en viu de trixie, crear live-boot en un chroot trixie. Si la distribució de destinació per casualitat coincideix amb la distribució del sistema de construcció, es pot construir directament en el sistema de construcció amb dpkg-buildpackage (proporcionat pel paquet dpkg-dev) :

```
$ cd live-boot
$ dpkg-buildpackage -b -uc -us
$ cd ../live-config
$ dpkg-buildpackage -b -uc -us
```

Utilitzar tots els fitxers .deb generats que calguin

Com live-boot i live-config són instal·lats per el sistema de construcció live-build, instal·lar els paquets en el sistema amfitrió no és suficient: s'ha de tractar els .deb generats com si fossin uns paquets personalitzats. Ja que el propòsit per a construir aquests paquets a partir del codi font és per a provar coses noves a curt termini abans del llanament oficial, seguir els passos de [Instal·lació de paquets modificats o de tercers](#) per a incloure temporalment els paquets rellevants en la configuració. En particular, cal observar que ambdós paquets es divideixen en una part genèrica, una part de documentació i un o més back-ends. Incloure la part genèrica, només un dels back-ends que coincideixi amb la configuració, i, opcionalment, la documentació. Suposant que s'està construint una imatge en viu en el directori actual i que s'han generat tots els .deb per a una única versió dels dos paquets

al directori superior, aquestes ordres de bash sn per a copiar tots els paquets importants, incloent-hi els back-ends per defecte:

157

```
$ cp ../live-boot-*, -initramfs-tools, -doc "*.deb  config/↵  
    packages.chroot/  
$ cp ../live-config-*, -sysvinit, -doc "*.deb  config/packages.↵  
    chroot/
```

Conceptes bsics

4. Conceptes bsics

Aquest capítol cont una breu descripció del procés de construcció i les instruccions per a l'utilització dels tres tipus d'imatge més comunes. El tipus d'imatge més versàtil iso-híbrid es pot utilitzar en una màquina virtual, en un mitjà físic o qualsevol altre dispositiu d'emmagatzematge USB. En certs casos especials, com s'explica més endavant, el tipus hdd pot ser el més adequat. El capítol cont instruccions detallades per a la construcció d'una imatge tipus netboot, que és una mica més complicat a causa de la configuració necessària en el servidor. Aquest és un tema una mica avançat per a algú que no està familiaritzat ja amb l'arrencada en xarxa, per s'inclou aquí perquè un cop que la configuració es porta a terme, es tracta d'una manera molt convenient per a provar i desplegar imatges per a l'arrencada en xarxa local sense la molèstia de tractar amb els dispositius de les imatges.

La secció acaba amb una breu introducció al **webbooting** que és, potser, la forma més fàcil d'utilitzar imatges diferents per a diferents propòsits, canviant d'una a l'altra segons sigui necessari, utilitzant internet com un mitjà.

Al llarg del capítol, sovint es fa referència als noms dels fitxers produïts per defecte per live-build. Si es **descarrega una imatge prefabricada**, els noms dels fitxers poden ser diferents.

4.1 Què és un sistema viu?

Un sistema viu és un sistema operatiu que arrenca en un equip des d'un dispositiu extraïble, com un CD-ROM o una memòria USB o des

d'una xarxa, a punt per a fer servir sense cap tipus d'instal·lació en la unitat(s) habitual(s), amb una configuració automàtica feta en temps d'execució (veure **Termes**).

With live systems, it's an operating system, built for one of the supported architectures (currently amd64 and arm64). It is made from the following parts:

Imatge del nucli Linux, generalment s'anomena **vmlinuz***

Imatge del disc RAM inicial (**initrd**): un disc RAM configurat per a l'arrencada de Linux, que cont els mòduls que possiblement es necessitaran per a muntar la imatge del sistema i algunes seqüències d'ordres per a fer-ho.

Imatge del sistema: Imatge del sistema de fitxers del sistema operatiu. Normalment, s'utilitza un sistema de fitxers comprimit SquashFS per a minimitzar la mida de la imatge. Tenir en compte que és de només lectura. Així, durant l'arrencada, el sistema Debian Live utilitza el disc RAM i un mecanisme de muntatge per a permetre l'escriptura de fitxers en el sistema en funcionament. No obstant això, totes les modificacions es perdran al apagar l'equip si no és que s'utilitza la persistència opcional (vegeu **Persistència**).

Carregador d'arrencada: Una petita peça de codi dissenyada per a arrencar des del mitjà triat, possiblement presentant un indicador d'arrencada o un menú per a permetre la selecció d'opcions/configuració. Carrega el nucli de Linux i el seu **initrd** per a funcionar amb un sistema de fitxers del sistema associat. Es poden utilitzar diverses solucions, en funció del mitjà de destinació i el format del sistema de fitxers que cont els components esmentats anteriorment: **isolinux** per a arrencar des de CD o DVD en format ISO9660, **syslinux** per a una unitat USB o HDD que s'iniciï des de particions VFAT, **extlinux** per a particions ext2/3/4 i **btrfs**, **pxelinux** per a PXE netboot, **GRUB** per a particions ext2/3/4, etc.

You can use live-build to build the system image from your specifications, set up a Linux kernel, its initrd, and a bootloader to run them, all in one medium-dependent format (ISO9660 image, disk image, etc.).

```
$ mkdir live - default && cd live - default
```

Aleshores, executar l'ordre `lb config`. Aix crear una jerarquia `config/` en el directori actual per a ser utilitzada per altres ordres:

4.2 Descarrega d'imatges prefabricades

You can download one of the prebuilt images from <https://www.debian.org/CD/live/>. For many of the popular desktop environments (GNOME, Xfce, KDE, etc.) a specific live image is prepared.

If you are unsure which file to download, use the 'Live GNOME' image from the 'stable' release. You can then skip reading the next sections and run the image in a **virtual machine**.

```
$ lb config
```

Aquí no es passa cap paràmetre a aquestes ordres, per tant s'utilitzaran les opcions per defecte. Veure **L'ordre `lb config`** per a més detalls.

Ara que la jerarquia `config/` ja existeix, crear la imatge amb l'ordre `lb build`:

4.3 Primers passos: construcció d'una imatge ISO híbrida

Independentment del tipus d'imatge, s'haur de fer els mateixos passos bàsics per a construir una imatge cada vegada. Com a primer exemple, crear un directori de treball, canviar a aquest directori i executar la següent seqüència d'ordres live-build per a crear una imatge ISO híbrida de base que conté només el sistema per defecte de Debian sense X.org. és adequat per a gravar en un CD o DVD, i també per a copiar en una memòria USB.

El nom del directori de treball és absolutament indiferent, però si es fa un cop d'ull als exemples utilitzats a live-manual, és una bona idea utilitzar un nom que ajudi a identificar la imatge amb que s'està treballant en cada directori, especialment si s'està treballant o experimentant amb diferents tipus d'imatges. En aquest cas, anem a construir un sistema per defecte així que l'anomenarem, per exemple, `live-default`.

```
# lb build
```

This process can take a while, depending on the speed of your computer and your network connection. When it is complete, there should be a `live-image-amd64.hybrid.iso` image file, ready to use, in the current directory.

Nota: Si s'està construint en un sistema amd64 el nom de la imatge resultant serà `live-image-amd64.hybrid.iso`. Tenir en compte aquesta convenció al llarg del manual.

4.4 Usar una imatge ISO híbrida en viu

After either building or downloading an ISO hybrid image the usual next step is to prepare your medium for booting, either CD-R(W) or DVD-R(W) optical media or a USB stick.

4.4.1 Gravar una imatge ISO en un medi físic

Gravar una imatge ISO és fàcil. Simplement cal instal·lar xorriso i utilitzar-lo des de la línia d'ordres per a gravar la imatge. Per exemple:

```
# apt-get install xorriso
$ xorriso -as cdrecord -v dev=/dev/sr0 blank=as-needed live -<
  image-amd64.hybrid.iso
```

4.4.2 Copiar una imatge ISO híbrida en un dispositiu USB

Les imatges ISO preparades amb xorriso, es poden copiar directament a una memòria USB utilitzant el programa cp o un altre d'equivalent. Connectar una memòria USB amb una mida prou gran per al fitxer de la imatge i determinar quin dispositiu és, que d'ara endavant anomenarem \$-USBSTICK". Aquest és el dispositiu de la memòria, com per exemple /dev/sdb, i no una partició, com ara /dev/sdb1! Es pot trobar el nom del dispositiu correcte mirant la sortida de dmesg després de connectar la memòria usb, o encara millor, ls -l /dev/disk/by-id.

Quan s'estigui segur de tenir el nom del dispositiu correcte, utilitzar l'ordre cp per a copiar la imatge a la memòria. Fent així es perdran definitivament tots els continguts anteriors de la memòria usb!

```
$ cp live-image-amd64.hybrid.iso $-USBSTICK"
$ sync
```

Nota: L'ordre sync s'utilitza per a assegurar-se que totes les dades, que el nucli emmagatzema en la memòria durant la còpia de la imatge, s'escriuen en el dispositiu USB.

4.4.3 Utilitzar l'espai lliure en una memòria USB

After copying the live-image-amd64.hybrid.iso to a USB stick, the first partition on the device will be filled up by the live system. To use the remaining free space, use a partitioning tool such as gparted or parted to create a new partition on the stick.

```
# gparted $-USBSTICK"
```

Després de crear la partició, on \$-PARTITION" és el nom de la partició, com ara /dev/sdb2, s'ha de crear un sistema de fitxers. Una opció possible seria ext4.

```
# mkfs.ext4 $-PARTITION"
```

Nota: Si es vol utilitzar l'espai addicional amb Windows, pel que sembla, aquest sistema operatiu normalment no pot accedir a altres particions més que a la primera. Algunes solucions a aquest problema han estat discutides a la nostra [llista de correu](#), per sembla que no hi ha respostes fàcils.

Remember: Every time you install a new live-image-amd64.hybrid.iso on the stick, all data on the stick will be lost because the partition table is overwritten by the contents of the image, so back up your extra partition first to restore again after updating the live image.

4.4.4 Arrencar el medi en viu

La primera vegada que s'arrenqui el medi en viu, ja sigui des de CD, DVD, memòria USB, o PXE, pot ser necessària alguna petita configuració al BIOS del ordinador en primer lloc. Atès que les BIOS varien molt en les seves funcions i dreceres de teclat, no podem

entrar en el tema en profunditat aqu. Algunes BIOS proporcionen una tecla per a obrir un men de dispositius d'arrencada, que s la manera ms fcil si es troba disponible al sistema. En cas contrari, cal entrar al men de configuraci del BIOS i canviar l'ordre d'arrencada per a situar el dispositiu del sistema en viu abans que el dispositiu d'arrencada normal.

Desprs d'arrencar des del dispositiu, es veur un menu d'inici. Si es prem entrar el sistema s'iniciar amb l'entrada Live i les seves opcions per defecte. Per a obtenir ms informaci sobre les opcions d'arrencada, llegir l'ajuda (help) al men i tamb les pgines del manual de live-boot i live-config que es troben dins del sistema en viu.

Assuming you've selected Live and booted a default desktop live image, after the boot messages scroll by, you should be automatically logged into the user account and see a desktop, ready to use. If you have booted a console-only image, you should be automatically logged in on the console to the user account and see a shell prompt, ready to use.

4.5 Utilitzar una mquina virtual per a fer proves

Pot ser un gran estalvi de temps per al desenvolupament d'imatges en viu executar-les en una mquina virtual (VM). Aix no est exempt d'avertiments:

L'execuci d'una mquina virtual requereix de suficient memria RAM, tant per al sistema operatiu convidat i l'amfitri i es recomana una CPU amb suport de maquinari per a la virtualitzaci.

Hi ha algunes limitacions inherents a l'execuci en una mquina virtual, per exemple, rendiment de vdeo pobre, opcions limitades de maquinari emulat.

En el desenvolupament per a un maquinari especfic, no hi ha cap substitut millor que el propi maquinari.

De tant en tant hi ha errors que noms sorgeixen durant l'execuci en una mquina virtual. En cas de dubte, comprovar la imatge directament al maquinari.

Sempre que es pugui treballar dins d'aquestes limitacions, examinar el programari de mquina virtual disponible i triar un que sigui adequat per a les necessitats prpies.

4.5.1 Provar una imatge ISO amb QEMU

La mquina virtual ms verstil dins Debian s QEMU. Si el processador t suport de maquinari per a la virtualitzaci, utilitzar el paquet qemu-kvm; la descripci del paquet qemu-kvm enumera breument els requisits.

Primer, instal·lar qemu-kvm si el processador ho suporta. Si no, instal·lar qemu, en aquest cas el nom del programa s qemu en lloc de kvm en els exemples segents. El paquet qemu-utils tamb s valus per a la creaci d'imatges de disc virtuals amb qemu-img.

```
# apt-get install qemu-kvm qemu-utils
```

Arrencar una imatge ISO s senzill:

```
$ kvm -cdrom live-image-amd64.hybrid.iso -m 4G
```

Veure les pgines del manual per a ms detalls

Note: For live systems containing a desktop environment that you want to test with qemu, you may wish to include the spice-vdagent package in your live-build configuration. This will automatically adjust the resolution and enable the clipboard between the virtual machine and the host.

```
$ echo "spice-vdagent" && config/package-lists/spice.list <<
chroot
```

4.5.2 Provar una imatge ISO amb VirtualBox

Per a provar la ISO amb virtualbox:

```
# apt-get install virtualbox virtualbox-qt virtualbox-dkms
$ virtualbox
```

Create a new virtual machine, change the storage settings to use live-image-amd64.hybrid.iso as the CD/DVD device, and start the machine.

Nota: Per a provar sistemes vius que contenen X.org amb virtualbox, segurament es assenyat incloure el paquet del driver VirtualBox X.org, virtualbox-guest-dkms i virtualbox-guest-x11, en la configuraci de live-build. En cas contrari, la resoluci es limita a 800x600.

```
$ echo "virtualbox-guest-dkms virtualbox-guest-x11" && config <<
/package-lists/my.list.chroot
```

Per tal de fer que el paquet dkms funcioni, s'han d'instalar tamb les capaleres del nucli per a la variant del nucli de la imatge. En lloc d'enumerar manualment el paquet linux-headers correcte en la llista de paquets creat anteriorment, la selecci del paquet adequat es pot fer automticament amb live-build.

```
$ lb config --linux-packages "linux-image linux-headers"
```

4.6 Construir i utilitzar una imatge HDD

Building an HDD image is similar to an ISO hybrid one in all respects except you specify `-b hdd` and the resulting filename is `live-image-amd64.img` which cannot be burnt to optical media. It is suitable for booting from USB sticks, USB hard drives, and various other portable storage devices. Normally, an ISO hybrid image can be used for this purpose instead, but if you have a BIOS which does not handle hybrid images properly, you need an HDD image.

Nota: si s'ha creat una imatge ISO hbrida amb l'exemple anterior, s'haur de netejar el directori de treball amb l'ordre `lb clean` (veure [L'ordre lb clean](#)):

```
# lb clean --binary
```

Executar l'ordre `lb config` com abans, excepte que aquesta vegada especificant el tipus d'imatge HDD:

```
$ lb config -b hdd
```

Ara construir la imatge amb l'ordre `lb build`:

```
# lb build
```

When the build finishes, a `live-image-amd64.img` file should be present in the current directory.

The generated binary image contains a VFAT partition and the syslinux bootloader, ready to be directly written on a USB device. Once again, using an HDD image is just like using an ISO hybrid one on USB. Follow the instructions in [Using an ISO hybrid live image](#),

except use the filename live-image-amd64.img instead of live-image-amd64.hybrid.iso.

Likewise, to test an HDD image with Qemu, install qemu as described above in [Testing an ISO image with QEMU](#). Then run kvm or qemu, depending on which version your host system needs, specifying live-image-amd64.img as the first hard drive.

```
$ kvm -hda live-image-amd64.img
```

4.7 Construir una imatge netboot

La segent seqncia d'ordres crear una imatge netboot bsica que cont el sistema per defecte de Debian sense X.org. s adequada per a l'arrencada en xarxa.

Nota: si s'ha realitzat algun dels exemples anteriors, s'haur de netejar el directori de treball amb l'ordre lb clean:

```
# lb clean
```

En aquest cas concret, un lb clean -binary no seria suficient per a netejar les etapes necessries. La causa d'aix s que en les configuracions d'arrencada en xarxa, es necessita una configuraci initramfs diferent que live-build realitza automticament quan es construeixen imatges netboot. Ja que la creaci del initramfs pertany a l'etapa chroot, fer el canvi a netboot en un directori de construcci existent significa reconstruir l'etapa chroot tamb. Per tant, s'ha de fer un lb clean (que eliminar l'etapa chroot, tamb)

Executar l'ordre segent per a configurar la imatge per a arrencar en xarxa:

```
$ lb config -b netboot --net-root-path "/srv/debian-live" --net-root-server "192.168.0.2"
```

A diferència de les imatges ISO i HDD, l'arrencada en xarxa no serveix el sistema de fitxers al client, per tant els fitxers han de ser servits a través de NFS. Amb lb config es poden elegir diferents sistemes de fitxers de xarxa. Les opcions -net-root-path i -net-root-server especifiquen la ubicaci i el servidor, respectivament, del servidor NFS on es troba la imatge del sistema de fitxers a l'hora d'arrencar. Assegurar-se que aquests s'ajusten als valors adequats per a la xarxa i el servidor.

Ara construir la imatge amb l'ordre lb build:

```
# lb build
```

En l'arrencada en xarxa, el client executa una petita pea de programari que normalment es troba a la EPROM de la targeta Ethernet. Aquest programa envia una petici DHCP per a obtenir una adrea IP i la informaci sobre qu fer a continuaci. Per regla general, el segent pas s aconseguir un carregador d'arrencada de ms alt nivell a través del protocol TFTP. Podria ser GRUB, pxelinux o fins i tot arrencar directament a un sistema operatiu com Linux.

For example, if you unpack the generated live-image-amd64.netboot.tar archive in the /srv/debian-live directory, you'll find the filesystem image in live/filesystem.squashfs and the kernel, initrd and pxelinux bootloader in tftpbboot/.

Ara hem de configurar els tres serveis al servidor per a l'arrencada en xarxa: el servidor DHCP, servidor TFTP i el servidor NFS.

4.7.1 Servidor DHCP

S'ha de configurar el servidor DHCP de la xarxa per a assegurar-se que dona una adreça IP per al client del sistema d'arrencada en xarxa, i per a anunciar la ubicació del carregador d'arrencada PXE.

Heus aquí un exemple per a servir d'inspiració, escrit per al servidor ISC DHCP `isc-dhcp-server` al fitxer de configuració `/etc/dhcp/dhcpd.conf`:

```
# /etc/dhcp/dhcpd.conf - configuration file for isc-dhcp-server
ddns-update-style none;

option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;

default-lease-time 600;
max-lease-time 7200;

log-facility local7;

subnet 192.168.0.0 netmask 255.255.255.0 {
    range 192.168.0.1 192.168.0.254;
    filename "pxelinux.0";
    next-server 192.168.0.2;
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.168.0.255;
    option routers 192.168.0.1;
}
```

4.7.2 Servidor TFTP

Aquest serveix el nucli i el disc ram inicial per al sistema en temps d'execució.

S'ha d'instal·lar el paquet `tftpd-hpa`. Aquest pot servir tots els fitxers continguts dins d'un directori arrel, per regla general `/srv/tftp`. Per

tal que es serveixin els fitxers dins de `/srv/debian-live/tftpboot`, s'ha d'executar com a superusuari la següent ordre:

```
# dpkg-reconfigure -plow tftpd-hpa
```

i omplir el nou directori del servidor tftp quan ho hagem de fer.

4.7.3 Servidor NFS

Un cop l'ordinador ha descarregat, ha arrencat el nucli de Linux i ha carregat el `initrd`, intentar muntar la imatge del sistema de fitxers en viu a través d'un servidor NFS.

S'ha d'instal·lar el paquet `nfs-kernel-server`

Llavors, fer que la imatge del sistema de fitxers estigui disponible a través de NFS afegint una línia com la següent a `/etc/exports`:

```
/srv/debian-live *(ro,async,no'root'squash,no'subtree'check)
```

i informar al servidor NFS sobre aquesta nova exportació amb la següent ordre:

```
# exportfs -rv
```

Setting up these three services can be a little tricky. You might need some patience to get all of them working together. For more information, see the `syslinux` wiki at <https://wiki.syslinux.org/wiki/index.php?title=PXELINUX> or the Debian Installer Manual's TFTP Net Booting section at <https://www.debian.org/releases/stable/amd64/ch04s05.en.html>. They might help, as their processes are very similar.

4.7.4 Com provar l'arrencada en xarxa

La creaci d'imatges d'arrencada en xarxa es senzilla amb live-build, per provar les imatges en mquines fsiques pot costar molt de temps.

Per a fer la nostra vida ms fcil, podem utilitzar la virtualitzaci.

4.7.5 Qemu

Installar qemu, bridge-utils, sudo.

Editar /etc/qemu-ifup:

```
#!/bin/sh
sudo -p "Password for $0:" /sbin/ifconfig $1 172.20.0.1
echo "Executing /etc/qemu-ifup"
echo "Bringing up $1 for bridged mode..."
sudo /sbin/ifconfig $1 0.0.0.0 promisc up
echo "Adding $1 to br0..."
sudo /usr/sbin/brctl addif br0 $1
sleep 2
```

Descarregar o crear un grub-floppy-netboot.

Llanar qemu amb -net nic,vlan=0 -net tap,vlan=0,ifname=tun0

4.8 Webbooting

Webbooting s una manera convenient d'aconseguir i arrencar sistemes vius utilitzant internet com un mitj. Els requisits per fer webbooting sn molt pocs. D'una banda, es necessita un dispositiu amb un carregador d'arrencada, un disc ram inicial i un nucli. D'altra banda, un servidor web per emmagatzemar els fitxers squashfs que contenen el sistema de fitxers.

4.8.1 Obtenir els fitxers webboot

As usual, you can build the images yourself or use the **prebuilt files**. Using prebuilt images would be handy for doing initial testing until one can fine tune their own needs. If you have built a live image you will find the files needed for webbooting in the build directory under binary/live/. The files are called vmlinuz, initrd.img and filesystem.squashfs.

Tamb s possible extreure els fitxers d'una imatge iso ja existent. Per tal d'aconseguir aix, muntar la imatge de la segent manera:

```
# mount -o loop image.iso /mnt
```

The files are to be found under the live/ directory. In this specific case, it would be /mnt/live/. This method has the disadvantage that you need to be root to be able to mount the image. However, it has the advantage that it is easily scriptable and thus, easily automated.

Per, sens dubte, la forma ms fcil d'extreure els fitxers d'una imatge iso i pujar-los al servidor web a la vegada, s utilitzant el midnight commander o mc. Si es t el paquet genisoimage instal·lat, aquest gestor de fitxers de dos panells permet examinar el contingut d'un arxiu iso en un panell i pujar els fitxers via ftp en l'altre panell. Tot i que aquest mtode requereix fer un treball manual, no requereix privilegis de root.

4.8.2 Arrencar imatges webboot

Mentre que alguns usuaris prefereixen utilitzar la virtualitzaci per prova el webbooting, en aquest cas utilitzem maquinari real perqu coincideixi amb el segent cas d's, que noms ha de ser considerat com un exemple.

291 Per a arrencar una imatge webboot s'és suficient tenir els components
esmentats anteriorment, s'és a dir, vmlinuz i initrd.img en una mem-
ria usb dins d'un directori anomenat live/ i instal·lar syslinux com
a gestor d'arrencada. Després, arrencar des de la memria usb i es-
criure `fetch=URL/RUTA/AL/FITXER` a les opcions d'arrencada.
live-boot descarregar l'arxiu squashfs i l'emmagatzemar en la mem-
ria ram. D'aquesta manera, s'és possible utilitzar el sistema de fitxers
comprimit descarregat com si fos un sistema viu normal. Per exem-
ple:

292

```
append boot=live components fetch=http://192.168.2.50/images/↵  
webboot/filesystem.squashfs
```

293

Cas d's: Tenir un servidor web en el qual s'ha emmagatzemat dos
arxius squashfs, un que cont un escriptori complet, com ara gnome,
i un standard. Si es necessita un entorn gràfic per a una mquina,
es pot connectar la memria usb i arrencar la imatge gnome. Si es
necessita una de les eines que s'inclouen en el segon tipus d'imatge,
potser per a una altra mquina, arrencar des de internet la imatge
standard.

Descripci general de les eines

5. Descripci general de les eines

Aquest capitol cont un resum de les tres eines principals utilitzades en la construcci dels sistemes en viu: live-build, live-boot i live-config.

5.1 El paquet live-build

live-build s un conjunt de scripts per a crear sistemes en viu. Aquests scripts tamb s'anomenen ordres.

La idea darrere de live-build s ser un marc que utilitza un directori de configuraci per automatitzar completament i personalitzar tots els aspectes de la construcci d'una imatge en viu.

Molts conceptes sn similars als utilitzats per a crear paquets Debian amb debhelper:

Els scripts tenen una ubicaci central per a la configuraci del seu funcionament. Amb debhelper aquest s el subdirectori debian/ d'un arbre de paquets. Per exemple, dh-install buscar, entre altres, un fitxer anomenat debian/install per a determinar quins fitxers han d'existir en un paquet binari en particular. De la mateixa manera, live-build emmagatzema la seva configuraci per complet sota un subdirectori config/.

Els scripts sn independents - s a dir, sempre s segur executar cada ordre.

A diferencia de debhelper, live-build proporciona les eines per a generar un directori de configuraci en esquelet. Aix podria ser considerat similar a eines com ara dh-make. Per a ms informaci sobre

aquestes eines, seguiu llegint, ja que la resta d'aquesta secci discuteix les quatre ordres ms importants. Tenir en compte que van precedides de lb que s una funci genrica per a les ordres de live-build.

lb config : Responsable d'inicialitzar un directori de configuraci per al sistema en viu. Consultar **L'ordre lb config** per a ms informaci.

lb build : Responsable d'iniciar la creaci d'un sistema en viu. Consultar **L'ordre lb build** per a ms informaci.

lb clean : Responsable d'eliminar parts de la construcci d'un sistema viu. Consultar **L'ordre lb clean** per a ms informaci.

5.1.1 L'ordre lb config

Com s'ha dit a **live-build**, les seqncies d'ordres que formen part de live-build llegeixen la seva configuraci amb l'ordre source d'un nic directori anomenat config/. Com la construcci d'aquest directori a m, seria molt costs i propens a errors, es pot utilitzar l'ordre lb config per a crear l'arbre inicial de configuraci en esquelet.

Executar lb config sense arguments crea el subdirectori config/ que s'omple amb alguns parmetres per defecte en fitxers de configuraci, i dos arbres de subdirectoris en esquelet que s'anomenen auto/ i local/.

```
$ lb config
[2025-02-15 12:34:56] lb config
P: Using http proxy: http://127.0.0.1:3142
P: Creating config tree for a debian/testing/amd64 system
P: Symlinking hooks...
```

Utilitzar lb config sense cap tipus d'arguments seria convenient per als usuaris que necessiten una imatge molt bsica, o que tinguin la intenci de proporcionar una configuraci ms completa ms tard

mitjanant `auto/config` (Veure **Gesti d'una configuraci** per a ms detalls).

Normalment, s'haur d'especificar algunes opcions. Per exemple, per a especificar quin gestor de paquets utilitzar durant la construcci de la imatge:

```
$ lb config --apt aptitude
```

s possible especificar diverses opcions, com ara:

```
$ lb config --binary-images netboot --bootappend-live "boot=live components hostname=live-host username=live-user" ...
```

Una llista completa d'opcions est disponible a la pgina del manual `lb.config`.

5.1.2 L'ordre `lb build`

L'ordre `lb build` llegeix la configuraci del directori `config/`. A continuaci, executa les ordres de nivell inferior necessries per a construir el sistema en viu.

5.1.3 L'ordre `lb clean`

L'ordre `lb clean` s'encarrega d'eliminar diverses parts d'una construcci per a que altres construccions posteriors puguin comenar des d'un estat net. Per defecte, es netegen les etapes `chroot`, `binary` i `source`, per la cach es mant intacta. A ms, es poden netejar etapes individuals. Per exemple, si s'han fet canvis que noms afecten a la fase `binary`, utilitzar `lb clean --binary` abans de construir un nou `binary`.

Si els canvis modifiquen el `bootstrap` i/o la cach de paquets, per exemple, canvis en les opcions `--mode`, `--architecture` o `--bootstrap`, s'ha d'utilitzar `lb clean --purge`. Veure la pgina del manual de `lb clean` per a una llista completa d'opcions.

5.2 El paquet `live-boot`

`live-boot` s un conjunt de scripts per a proporcionar hooks a `initramfs-tools`, que s'utilitzen per a generar un `initramfs` capa d'arrencar sistemes vius, com ara els creats per `live-build`. Aix inclou les ISOs dels sistemes en viu, `netboot` tarballs i imatges per a memries USB.

At boot time it will look for read-only media containing a `/live/` directory where a root filesystem (often a compressed filesystem image like `squashfs`) is stored. If found, it will create a writable environment, using `OverlayFS`, for Debian like systems to boot from.

More information on initial ramfs in Debian can be found in the Debian Linux Kernel Handbook at <https://kernel-team.pages.debian.net/kernel-handbook/> in the chapter on `initramfs`.

5.3 El paquet `live-config`

`live-config` consta dels scripts que s'executen durant l'arrencada desprs de `live-boot` per a configurar el sistema en viu de forma automtica. S'ocupa de tasques com ara l'establiment de les locales, el nom d'amfitri, la zona horria, crear l'usuari en viu, l'inhibici de tasques de cron i l'inici automtic de sessi per a l'usuari en viu.

Gesti d'una configuraci

6. Gesti d'una configuraci

En aquest capítol s'explica com gestionar una configuraci d'un sistema en viu des de la seva creaci inicial, a travs de revisions i versions successives de tant el programari live-build com de la imatge en viu en si mateixa.

6.1 Gestionar canvis en la configuraci

Les configuracions de sistemes en viu poques vegades sn perfectes al primer intent. Passar opcions a lb config des de la lnea d'ordres pot estar be per a construir una imatge una vegada, per s ms tpic revisar aquestes opcions i construir de nou fins que se'n estigui satisfet. Per a donar suport a aquests canvis, es poden utilitzar scripts auto que assegurin que la configuraci es mant en un estat consistent.

6.1.1 Per qu utilitzar scripts auto? Qu fan?

L'ordre lb config emmagatzema les opcions que se li passen als fitxers de config/*, juntament amb moltes altres opcions que estan establertes als valors per defecte. Si s'executa un cop ms, lb config no es restablir cap de les opcions dependents basades en les opcions inicials. Aix, per exemple, si s'executa de nou lb config amb un nou valor per a --binary-images, totes les opcions que en depenen que es van omplir per al tipus de imatge per defecte ja no poden funcionar amb la nova. Aquests fitxers no estan destinats a ser llegits o editats. S'emmagatzemen els valors de ms de cent opcions, i ning pot veure les opcions que s'han especificat realment. I finalment, si s'executa lb config i a continuaci s'actualitza live-build i el nom d'una opci

canvia, config/* encara contindr les variables de l'opci vella que ja no sn vlides.

Per totes aquestes raons, els scripts auto/* ens fan la vida ms fcil. Sn simples embolcalls per les ordres lb config, lb build i lb clean dissenyats per ajudar a gestionar una configuraci. Noms cal crear un script auto/config que contingui totes les opcions que es desitgin per a lb config, i un auto/clean que elimini els fitxers que continguin diversos valors de variables de configuraci, i el script auto/build guarda un build.log de cada construcci. Cada vegada que s'executi l'ordre lb corresponent, aquests fitxers seran executats automticament. L's d'aquests scripts assegurar que la configuraci sigui ms senzilla de llegir i que guardi una coherncia interna d'una reversi a una altra. A ms a ms ser ms fcil identificar i solucionar les opcions que s'han de canviar al actualitzar d'una versi de live-build a la segent desprs de llegir la documentaci.

6.1.2 Utilitzar scripts auto d'exemple

Per a ms comoditat, live-build ve amb uns scripts d'exemple per a copiar i editar. Iniciar una nova configuraci per defecte, i a continuaci, copiar els exemples:

```
$ mkdir mylive && cd mylive && lb config
$ mkdir auto
$ cp /usr/share/doc/live-build/examples/auto/* auto/
```

Editar auto/config, afegint les opcions ms adients. Per exemple:

```
#!/bin/sh
lb config noauto "
    --distribution stable "
    --binary-images hdd "
```

```
--mirror-bootstrap http://ftp.ch.debian.org/debian/ "
--mirror-binary http://ftp.ch.debian.org/debian/ "
"$-@"
```

```
$ cd images/standard
```

Editar `auto/config` i qualsevol altra cosa necessria dins l'arbre `config` per a satisfer les necessitats prpies. Per exemple, per a fer les imatges prefabricades no oficials que contenen paquets de la secci `non-free` simplement s'afegeix `--archive-areas=main contrib non-free`.

Si es desitja, es pot definir una drecera en la configuraci de `Git`, afegint el segent a `~/.gitconfig`:

```
[url "https://salsa.debian.org/live-team/"]
  insteadOf = lso:
```

This enables you to use `lso:` anywhere you need to specify the address of a `salsa.debian.org` git repository. If you also drop the optional `.git` suffix, starting a new image using this configuration is as easy as:

```
$ lb config --config lso:live-images::debian
```

Clonar tot el repositori `live-images` copia les configuracions utilitzades per diverses imatges. Si es vol construir una imatge diferent després d'haver acabat amb la primera, canviar a un altre directori i un altre cop i, opcionalment, fer els canvis per a adaptar-les a les necessitats prpies.

En qualsevol cas, recordar que cada vegada que s'ha de construir una imatge, s'ha de fer com a superusuari: `lb build`

Ara, cada vegada que s'utilitzi `lb config`, `auto/config` restablir la configuraci basada en aquestes opcions. Quan es vulgui fer canvis, editar les opcions d'aquest fitxer en lloc de passar-les a `lb config`. Quan s'utilitza `lb clean`, `auto/clean` netejar els fitxers de `config/*` juntament amb els altres productes de construcci. I, finalment, quan s'utilitza `lb build`, es crea un log de la construcci mitjanant `auto/build` anomenat `build.log`.

Nota: Aquí s'utilitza un paràmetre especial `noauto` per a suprimir un altra crida a `auto/config`, la qual cosa impedeix la recursivitat infinita. Assegurar-se de no eliminarlo accidentalment fent canvis. També, tenir cura de que quan es divideix l'ordre `lb config` a través de diverses línies per a facilitar la lectura, com es mostra en l'exemple anterior, no s'oblidi la barra invertida (al final de cada línia que segueix a la segent.

6.2 Clonar una configuraci publicada via Git

Use the `lb config --config` option to clone a Git repository that contains a live system configuration. If you would like to base your configuration on one maintained by the Debian Live Project, look at <https://salsa.debian.org/live-team/> for the repository named `live-images` in the category `Subgroups and projects`. This repository contains the configurations for the live systems **prebuilt images**.

Per exemple, per a construir una imatge `standard`, utilitzar el repositori `live-images` de la manera segent:

```
$ mkdir live-images && cd live-images
$ lb config --config https://salsa.debian.org/live-team/live-images.git::debian
```

Personalitzaci dels continguts

s'executen a menys que el sistema s'hagi configurat com a sistema viu.

7. Visi general de la personalitzaci

En aquest capítol s'ofereix una visi general de les diverses formes en qu es pot personalitzar un sistema en viu.

7.1 Configuraci durant la construcci vs. durant l'arrencada

La configuraci de un sistema en viu es divideix en opcions en temps de construcci que sn les opcions que s'apliquen durant la seva creaci i les opcions d'arrencada del sistema que s'apliquen durant l'arrencada. Les opcions d'arrencada es divideixen en les qu ocorren al principi de l'arrencada, aplicades pel paquet live-boot, i les que ocorren ms tard en l'arrencada, aplicades per live-config. Qualsevol opci durant l'arrencada pot ser modificada per l'usuari, especificant-la a l'indicador d'arrencada. La imatge tamb pot ser construïda amb els parmetres d'arrencada per defecte perqu els usuaris puguin simplement arrencar el sistema en viu sense especificar cap altra opci, ja que tots els valors per defecte sn adequats. En particular, l'argument `lb --bootappend-live` consta de les opcions de lnia d'ordres per defecte del nucli per al sistema en viu, com ara la persistncia, la distribuci del teclat o la zona horria. Veure [Personalitzaci de l'entorn local i el llenguatge](#), per exemple.

Les opcions de configuraci durant la construcci es descriuen a la pgina del manual de `lb config`. Les opcions durant l'arrencada es descriuen a les pginas del manual de `live-boot` i `live-config`. Malgrat que els paquets `live-boot` i `live-config` s'installen en el sistema en viu que s'est construint, s recomana instal·lar-los en el sistema de construcci per a tenir una referncia fcil quan s'est treballant en la configuraci. s segur fer-ho, ja que cap dels scripts continguts en ells

7.2 Etapes de la construcci

El procés de construcci es divideix en etapes, amb personalitzacions diferents aplicades successivament en cada una. La primera etapa que s'executa es la fase `bootstrap`. Aquesta s la fase inicial de poblar el directori `chroot` amb paquets per a fer un sistema Debian bsic. Aix s seguit per l'etapa `chroot`, que completa la construcci del directori `chroot`, omplint-lo amb tots els paquets que s'indiquen en la configuraci, juntament amb qualsevol altre material. La majoria de personalitzacions dels continguts es produeixen en aquesta etapa. L'etapa final de preparaci de la imatge en viu s l'etapa `binary`, quan es construeix una imatge capa d'arrencar, amb el contingut del directori `chroot` per a construir el sistema de fitxers arrel per al sistema en viu, i que inclou el programa de instal·laci i qualsevol altre material addicional en el medi de destinaci fora del sistema de fitxers del sistema en viu. Desprs de construir la imatge en viu, si est habilitat, s'inclou el codi font original a l'etapa `source`.

Dins de cadascuna d'aquestes etapes, hi ha una seqncia particular en la qual s'apliquen les ordres. Aix es fa de manera que es garanteixi que les personalitzacions es poden superposar de manera raonable. Per exemple, dins l'etapa `chroot`, les preconfiguracions (`preseeds`) s'apliquen abans que s'installin els paquets, els paquets s'installen abans que es copin els fitxers locals, i els ganxos s'executen ms tard, desprs que tots els materials estiguin al seu lloc.

7.3 Suplementar lb config amb fitxers

Encara que `lb config` crea una configuraci en esquelet al directori `config/`, per a aconseguir els objectius, pot ser necessari proporcionar

fitxers addicionals en els subdirectoris de config/. Depenent d'on s'emmagatzemen els fitxers en la configuraci, poden ser copiats en el sistema de fitxers del sistema en viu o en el sistema de fitxers de la imatge binria, o es pot proporcionar configuracions en temps de construcci del sistema que serien engorroses de passar com opcions de lnia d'ordres. Es pot incloure coses com ara llistes personalitzades de paquets, art personalitzat o scripts ganxo per a ser executats ja sigui en temps de construcci o en temps d'arrencada, augmentant la flexibilitat ja considerable de debian-live amb codi propi.

7.4 Tasques de personalitzaci

Els segents captols s'organitzen pel tipus de tasques de personalitzaci que els usuaris solen realitzar: **Personalitzaci de la installaci de paquets**, **Personalitzaci dels continguts** i **Personalitzaci de l'entorn local i el llenguatge** cobreixen noms algunes de les coses que es poden fer.

Personalitzaci de la installaci de paquets

8. Personalitzaci de la installaci de paquets

La personalitzaci ms bsica d'un sistema en viu pot ser la selecci dels paquets que seran inclosos en la imatge. Aquest capitol explica les diverses opcions de live-build per a personalitzar la installaci de paquets durant la construcci. Les opcions ms importants que influeixen en els paquets que estan disponibles per a ser installats en la imatge sn les rees de distribuci i el arxiu. Per a garantir velocitats de descarga decents, s'ha de triar un mirall de distribuci proper. Tamb es pot incloure repositoris de backports, paquets experimentals o personalitzats, o incloure paquets directament com si fossin fitxers. Es poden definir llistes de paquets, incloent-hi els metapaquets que installaran diversos paquets relacionats alhora, com ara paquets per a un ordinador d'escriptori o un llenguatge en particular. Finalment, una srie d'opcions donen un cert control sobre apt o si es prefereix aptitude, quan s'installen els paquets durant la construcci. Aix pot ser til si s'utilitza un proxy, es vol desactivar la installaci de paquets recomanats per a estalviar espai, o hi ha la necessitat de controlar quines versions dels paquets s'installen mitjanant la tcnica pinning d'APT, per nomenar algunes possibilitats.

8.1 Fonts dels paquets

8.1.1 Distribuci, rees d'arxiu i mode

The distribution you choose has the broadest impact on which packages are available to include in your live image. Specify the code name, which defaults to testing . Any current distribution carried

in the archive may be specified by its codename here. (See [Terms](#) for more details.) The `--distribution` option not only influences the source of packages within the archive, but also instructs live-build to enable other sources.

For example, to build against the stable release, with security, updates (enabled per default) and additionally proposed-updates and backports, specify:

```
$ lb config --distribution stable --proposed-updates true --backports true
```

Similarly, for the unstable release, sid , which has neither security nor updates, specify:

```
$ lb config --distribution sid
```

A l'arxiu de la distribuci, les rees sn les divisions principals de l'arxiu. A Debian, es tracta de main, contrib i non-free. Noms main cont el programari que s part de la distribuci Debian, per tant s el valor per defecte. Es poden especificar un o ms valors, per exemple:

```
$ lb config --archive-areas "main contrib non-free"
```

s dona suport experimental a alguns derivats de Debian a travs de l'opci `--mode`. Per defecte, aquesta opci s debian per noms si s'est construint en un sistema Debian o en un sistema desconegut. Si s'especifica amb `lb config` que es vol construir un dels derivats suportats alehores es modificaran les opcions per a crear aquest derivat. Si per exemple s'utilitza `lb config` amb el mode `ubuntu`, s'utilitzar el nom de la distribuci i les rees dels arxius del derivat especificat

en lloc dels de Debian. El mode també modifica el comportament de live-build per a adaptar-lo als derivats.

Nota: Els projectes per als quals s'han afegit aquests modes són els principals responsables de donar suport als usuaris d'aquestes opcions. El Debian Live Project, al seu torn, dona suport de desenvolupament només sobre una base de millor esforç, basada en les informacions proporcionades pels projectes derivats ja que nosaltres no desenvolupem ni donem suport a aquests derivats.

8.1.2 Miralls de distribució

L'arxiu de Debian es replica a través d'una mpla xarxa de miralls a tot el món perquè la gent de cada regió pugui triar un mirall proper amb la millor velocitat de descàrrega. Cadascuna de les opcions `--mirror-*` governa quin mirall de distribució s'utilitzarà en les diverses etapes de la construcció. Recordar de **Etaques de la construcció** que l'etapa bootstrap és quan el chroot s'omple inicialment per debootstrap amb un sistema mínim i l'etapa chroot és quan s'utilitza el chroot per a la construcció del sistema de fitxers del sistema en viu. D'aquesta manera, s'utilitzen els miralls corresponents per a aquestes etapes, i més tard, durant l'etapa binary s'utilitzen els valors `--mirror-binary` i `--mirror-binary-security` substituint qualsevol mirall utilitzat en una etapa anterior.

8.1.3 Miralls de distribució utilitzats en temps de construcció

Per a establir els miralls de la distribució utilitzats en temps de construcció perquè apuntin a una còpia local, és suficient establir `--mirror-bootstrap` i `--mirror-chroot-security` de la manera següent.

```
$ lb config --mirror-bootstrap http://localhost/debian/ "
```

```
--mirror-chroot-security http://localhost/debian-↔
security/
```

El mirall per al chroot, especificat per l'opció `--mirror-chroot`, per defecte pren el mateix valor que `--mirror-bootstrap`

8.1.4 Miralls de distribució utilitzats en temps d'execució

The `--mirror-binary*` options govern the distribution mirrors placed in the binary image. These may be used to install additional packages while running the live system. The defaults employ `deb.debian.org`, a service that chooses a geographically close mirror based, among other things, on the user's IP family and the availability of the mirrors. This is a suitable choice when you cannot predict which mirror will be best for all of your users. Or you may specify your own values as shown in the example below. An image built from this configuration would only be suitable for users on a network where mirror is reachable.

```
$ lb config --mirror-binary http://mirror/debian/ "
--mirror-binary-security http://mirror/debian-↔
security/ "
--mirror-binary-backports http://mirror/debian-↔
backports/
```

8.1.5 Repositoris addicionals

És possible afegir més repositoris, ampliant les opcions de paquets més enllà dels disponibles en la pròpia distribució de destinació. Aquests poden ser, per exemple, per a backports, experimentals o paquets personalitzats. Per a configurar repositoris addicionals, crear els fitxers `config/archives/your-repository.list.chroot`, i/o `config/archives/your-repository.list.binary`. Igual que amb les opcions `--mirror-*`

aquest regeix els repositoris utilitzats en l'tapa chroot durant la construcci de la imatge, i a l'tapa binary, s a dir, per a ser utilitzades quan s'executa el sistema en viu.

Per exemple, `config/archives/live.list.chroot` permet instal·lar paquets des del repositori d'instants de `debian-live` en el moment de construcci del sistema viu.

```
deb http://debian-live.alioth.debian.org/ sid-snapshots main
contrib non-free
```

Si s'afegeix la mateixa lnia a `config/archives/live.list.binary`, el repositori serà afegit al directori `/etc/apt/sources.list.d/` del sistema viu.

Si aquests fitxers existeixen, s'utilitzats de forma automàtica.

You should also put the ASCII-armored GPG key used to sign the repository into `config/archives/your-repository.key-binary,chroot` files.

En cas de necessitar un APT pinning personalitzat, es poden col·locar les preferències APT en fitxers `config/archives/your-repository.-pref-binary,chroot`, que seran afegits automàticament al sistema en viu al directori `/etc/apt/preferences.d/`.

Similarly, if you need custom APT AUTH.CONF(5) authentication configuration, this can be placed in `config/archives/your-repository.-auth-binary,chroot` files and will be automatically added to your live system's `/etc/apt/auth.conf.d/` directory

8.2 Selecció dels paquets a instal·lar

Hi ha una sèrie de formes de triar els paquets que `live-build` instal·la en la imatge, que abasta una varietat de necessitats diferents. Es pot simplement anomenar paquets individualment per a instal·lar en

una llista de paquets. També es pot optar per utilitzar metapaquets a les llistes, o seleccionar-los utilitzant camps de control de fitxers de paquets. I, finalment, es poden copiar paquets com si fossin fitxers dins del arbre `config/`, que s'un mètode que s'adapta perfectament a fer proves amb paquets nous o experimentals abans de afegirlos a un repositori.

8.2.1 Llistes de paquets

Les llistes de paquets són una forma eficaç d'expressar quins paquets han de ser instal·lats. La sintaxi de la llista suporta seccions condicionals que fa que sigui fàcil construir llistes i adaptar-les per al propi s en múltiples configuracions. Els noms dels paquets també poden ser injectats a la llista amb ajudants de l'interpret d'ordres en temps de construcci.

Nota: El comportament de `live-build` a l'hora d'especificar un paquet que no existeix està determinat per la elecció que es fa de l'eina APT. Veure [Elegir apt or aptitude](#) per a més detalls.

8.2.2 Els dels metapaquets

La forma més senzilla per a omplir la llista de paquets s'utilitza una tasca metapaquet mantinguda per una distribució. Per exemple:

```
$ lb config
$ echo task-gnome-desktop > config/package-lists/desktop.list
.chroot
```

This supersedes the older predefined list method supported in `live-build 2.x`. Unlike predefined lists, task metapackages are not specific to the Live System project. Instead, they are maintained by specialist working groups within the distribution and therefore reflect the

consensus of each group about which packages best serve the needs of the intended users. They also cover a much broader range of use cases than the predefined lists they replace.

Tots els metapaquets tenen el prefix task-, de manera que una forma rpida de determinar quins estan disponibles (encara que pot contenir un grapat d'entrades falses que coincideixin amb el nom, per que no sn metapaquets) s fer coincidir el nom del paquet amb:

```
$ apt-cache search --names-only ^task-
```

A ms d'aquests, es troben altres metapaquets amb diverses finalitats. Alguns sn subconjunts de paquets de tasques ms mplies, com gnome-core, mentre que altres sn parts individuals especialitzades de un Debian Pure Blend, com els metapaquets education-*. Per a obtenir una llista de tots els metapaquets que hi ha a l'arxiu, instal·lar el paquet debtags i llistar tots els paquets amb l'etiqueta role::metapackage de la segent manera:

```
$ debtags search role::metapackage
```

8.2.3 Llistes locals de paquets

Ja sigui afegint metapaquets a una llista, paquets individuals, o una combinaci d'ambds, totes les llistes de paquets locals s'emmagatzemen a config/package-lists/. Es pot utilitzar ms d'una llista i aix es presta molt b als dissenys modulars. Per exemple, es pot decidir dedicar una llista a una elecci particular d'escriptori, l'altra a una collecci de paquets relacionats que puguin ser fcilment utilitzats al damunt d'un escriptori diferent. Aix permet experimentar amb diferents combinacions de conjunts de paquets amb un

mnim d'esfor, intercanviant llistes comunes entre els diferents projectes d'imatges en viu.

Les llistes de paquets que es troben en aquest directori han de tenir el sufix .list per a ser processades, i a ms a ms un sufix d'etapa adicional .chroot o .binary per a indicar per a quina etapa s la llista.

The packages in the .list.chroot install list are present both in the live system and in the installed system.

Nota: Si no s'especifica el sufix d'etapa, la llista s'utilitzar per a ambdues etapes. Normalment, s'especifica .list.chroot de manera que els paquets noms s'installaran al sistema de fitxers en viu i no hi haurà una cpia extra del .deb en el medi.

8.2.4 Llistes locals de paquets per a l'etapa binary

Per a crear una llista per a l'etapa binary, crear un fitxer amb el sufix .list.binary a config/package-lists/. Aquests paquets no s'installen al sistema de fitxers en viu per s'inclouen en el medi en viu al directori pool/. Un s tpic d'aquesta llista seria amb una de les variants del instal·lador non-live. Com s'ha esmentat anteriorment, si es vol que aquesta llista sigui la mateixa que la llista de l'etapa chroot, simplement utilitzar el sufix .list.

8.2.5 Generar llistes de paquets

De vegades passa que la millor manera de crear una llista s generar-la amb un script. Qualsevol lnia que comenci amb un signe d'exclamaci indica una ordre que s'executar dins del chroot quan la imatge es construeix. Per exemple, es podria incloure la lnia ! grep-aptavail -n -sPackage -FPriority standard —sort en una llista de paquets per a produir una llista ordenada de paquets disponibles amb Priority: standard.

De fet, la selecció de paquets amb l'ordre `grep-aptavail` (del paquet `dctrl-tools`) s'ha de fer tan bonic que `live-build` proporciona un script `Packages` d'ajuda per motius de comoditat. Aquest script accepta dos arguments: `field` i `pattern`. Per tant, es pot crear una llista amb els següents continguts:

```
$ lb config
$ echo '! Packages Priority standard' >> config/package-lists/↵
  standard.list.chroot
```

```
#if ARCHITECTURES i386 amd64
memtest86+
#endif
```

També es pot provar amb variables que poden contenir més d'un valor, per exemple, per a instal·lar `vrms` si s'especifica `contrib` o `non-free` a través de l'opció `-archive-areas`:

```
#if ARCHIVEAREAS contrib non-free
vrms
#endif
```

8.2.6 Els condicionals dins de les llistes de paquets

Qualsevol de les variables de configuració de `live-build` emmagatzemades a `config/*` (menys el prefix `LB`) poden ser utilitzades en sentències condicionals en les llistes de paquets. En general, això significa qualsevol opció `lb config` en lletres majúscules i amb guions canviats a guions baixos. Per a la pràctica, només tenen sentit les que influeixen en la selecció de paquets, com ara `DISTRIBUTION`, `ARCHITECTURES` o `ARCHIVEAREAS`.

Per exemple, per a instal·lar `ia32-libs` si s'especifica `-architectures amd64`:

```
#if ARCHITECTURES amd64
ia32-libs
#endif
```

És possible fer un test d'un nombre de valors, per exemple per a instal·lar `memtest86+` si s'especifica `-architectures i386` o `-architectures amd64`:

No és possible el anidament dels condicionals.

8.2.7 Eliminar paquets durant la instal·lació

Es pot crear llistes de paquets en fitxers amb els sufixos `.list.chroot-live` i `.list.chroot-install` dins del directori `config/package-lists`. Si hi ha una llista `live` i una llista `install` els paquets de la llista `.list.chroot-live` s'eliminaran amb un script ganxo després de la instal·lació (si l'usuari utilitza l'instal·lador). Els paquets de la llista `.list.chroot-install` seran presents tant en el sistema en viu com en el sistema instal·lat. Aquest és un cas especial per al programa d'instal·lació i pot ser útil si es té `-debian-installer live` establert en la configuració i es desitja eliminar paquets específics del sistema en viu durant la instal·lació.

8.2.8 Summary

The table below shows which configuration files are required to achieve the desired availability of the package.

	X.chroot	X.chroot'- X live	X	X.binary
Package is installed in the live sys- tem	Yes	Yes	Yes	No
Package is removed in- stalling the live system	No	Yes	No	N/A
Package can be installed from the live system without network	N/A	N/A	Yes *1	Yes

*1: Because the installer needs this package
X = config/package-lists/custom.name.list

8.2.9 Tasques d’escriptori i llenguatge

Les tasques d’escriptori i el llenguatge sn casos especials que ne-
cessiten una mica de planificaci i configuraci extra. Les imatges en
viu sn diferentes de les imatges de l’installador de Debian en aquest
sentit. A l’installador de Debian, si el medi es va preparar per a
obtenir un tipus d’entorn d’escriptori en particular, la tasca corres-
ponent s’installar automticament. Per tant hi ha tasques internes
gnome-desktop, kde-desktop, lxde-desktop i xfce-desktop, cap de les
quals s’ofereixen al men de tasksel. De la mateixa manera, no hi ha
cap entrada de men per a tasques de llenges, per l’elecci del idioma
de l’usuari durant la installaci influeix en la selecci de les tasques de
les llenges corresponents.

En el desenvolupament d’una imatge en viu d’escriptori, la imatge
sol arrencar directament a un escriptori de treball, les opcions
d’escriptori i de llengua han estat fetes en temps de construcci, no en
temps d’execuci com en el cas del installador de Debian. Aix no vol

dir que una imatge en viu no es pugui construir per a donar suport a
diversos equips d’escriptori o diversos idiomes i oferir a l’usuari una
opci, per aix no s el comportament de live-build per defecte.

Com que no hi ha cap ajust automtic per a les tasques de llengua
que incloguin coses com ara tipus de lletres especfics per a una
llengua o paquets de mtdode d’entrada, si es vol, cal especificar-ho en
la configuraci. Per exemple, una imatge d’escriptori GNOME que
contingui suport per al alemany podrie incloure les segents tasques
metapaquets:

```
$ lb config  
$ echo "task-gnome-desktop task-laptop" && config/package-<←  
lists/my.list.chroot  
$ echo "task-german task-german-desktop task-german-gnome-<←  
desktop" && config/package-lists/my.list.chroot
```

8.2.10 Tipus i versi del nucli

Depenent de l’arquitectura, s’inclouran per defecte en la imatge un
o ms tipus de nuclis. Es pot triar diferents tipus a travs de l’opci
–linux-flavours. Cada tipus t un sufix per a l’arrel per defecte linux-
image per a formar el nom de cada metapaquet que al seu torn depn
d’un paquet del nucli exacte que s’ha d’incloure en la imatge.

Aix, per defecte, una imatge per a l’arquitectura amd64 inclour el
metapaquet linux-image-amd64 i una imatge per a l’arquitectura
i386 inclour el metapaquet linux-image-586.

Quan hi ha ms d’una versi del paquet del nucli disponible en els arx-
ius configurats, es pot especificar el nom d’un paquet del nucli amb
l’opci –linux-packages. Per exemple, suposem que s’est construïnt
una imatge d’arquitectura amd64 i es vol afegir l’arxiu experimen-
tal amb propsits de fer proves. Perqu es pugui installar el nucli

linux-image-3.18.0-trunk-amd64 es podria configurar la imatge de la segent manera:

```
$ lb config --linux-packages linux-image-3.18.0-trunk
$ echo "deb http://deb.debian.org/debian/ experimental main" <-
    & config/archives/experimental.list.chroot
```

8.2.11 Nuclis personalitzats

Es pot construir i incloure nuclis propis personalitzats, sempre que s'integrin en el sistema de gesti de paquets de Debian. El sistema de live-build no s compatible amb nuclis no construts com paquets .deb.

La manera apropiada i recomanable d'implementar els propis paquets del nucli s seguir les instruccions del kernel-handbook. Recordar que s'ha de modificar l'ABI i els sufixos del tipus apropiadament, i a continuaci, incloure un conjunt complet dels packets que corresponen amb linux i linux-latest al repositori.

Si s'opta per construir els paquets del nucli sense els metapaquets a joc, cal especificar una arrel --linux-packages apropiada com s'indica a **Tipus i versi del nucli**. Com expliquem a **Installaci de paquets modificats o de tercers**, s millor si s'inclouen els paquets del nucli personalitzat en un repositori propi, tot i que les alternatives discutides en aquella secci tamb funcionen.

Est ms enll de l'abast d'aquest document donar consells sobre com personalitzar un nucli. No obstant aix, cal, almenys, assegurar-se que la configuraci compleix els segents requisits mnims:

- Utilitzar una ramdisk inicial.

- Include the union filesystem module (i.e. usually OverlayFS).

Incloure tots els mduls del sistema d'arxius requerits per la configuraci (normalment squashfs).

8.3 Installaci de paquets modificats o de tercers

Si b est en contra de la filosofia d'un sistema en viu, de vegades pot ser necessaria la construcci d'un sistema amb versions modificades dels paquets que es troben al arxiu de Debian. Pot ser per a modificar o donar suport a funcions addicionals, les llenges o les marques, o fins i tot per a eliminar elements dels paquets existents que sn indesitjables. De la mateixa manera, es poden utilitzar paquets de tercers per a afegir alguna funcionalitat personalitzada i/o propietria.

This section does not cover advice regarding building or maintaining modified packages. Joachim Breitner's 'How to fork privately' method from <http://www.joachim-breitner.de/blog/archives/282-How-to-fork-privately.html> may be of interest, however. The creation of bespoke packages is covered in the Debian New Maintainers' Guide at <https://www.debian.org/doc/manuals/maint-guide/> and elsewhere.

Hi ha dues formes d'instalar paquets personalitzats modificats:

- packages.chroot

- L's d'un repositori APT personalitzat

Utilitzar packages.chroot s ms fcil d'aconseguir i til per a personalitzacions rpides, per t una srie d'inconvenients, mentre que l's d'un repositori APT personalitzat s ms costs en la quantitat de temps necessari per a posar-lo en marxa.

8.3.1 Fer servir packages.chroot per a instaar paquets personalitzats

Per a instaar un paquet personalitzat, noms s'ha de copiar al directori config/packages.chroot/. Els paquets que es troben dins d'aquest directori s'installaran automticament en el sistema en viu durant la construcci - no cal especificar res ms en cap altre lloc.

Els paquets han de ser nomenats en la forma prescrita. Una manera simple de fer aix s utilitzar dpkg-name.

Utilitzar packages.chroot per a la installaci de paquets personalitzats t els seus desavantatges:

No s possible utilitzar APT segur.

Cal posar tots els paquets apropiats al directori config/packages.chroot/.

No es adient per a l'emmagatzematge de configuracions de sistemes en viu en el control de revisi.

8.3.2 Fer servir un repositori APT per a instal·lar paquets personalitzats

A diferència de packages.chroot, quan s'utilitza un repositori APT personalitzat s'ha d'assegurar que s'especifiquen els paquets en un altre lloc. Veure **Selecció dels paquets a instal·lar** per a ms detalls.

Si b crear un repositori APT per a instal·lar paquets personalitzats pot semblar un esfor innecessari, la infraestructura pot ser feilment reutilitzada en una data posterior per a oferir actualitzacions dels paquets modificats.

The APT repository does not necessarily need to be online, you can use a local repository instead. However, in both cases the repository needs to be signed.

Example:

```
$ gpg --armor --output config/archives/custom`repo`.gpg.key$-↵
    EXTENSION" --export-options export-minimal --export $-↵
    SIGNING`KEY"
$ cat ; EOF ; config/archives/custom`repo`.list$-EXTENSION"
deb [signed-by=/etc/apt/trusted.gpg.d/custom`repo`.gpg.key$-↵
    EXTENSION".asc] $-URI" $-SUITE" $-COMPONENTS"
EOF
$ echo "$-PACKAGESFROM`REPOSITORY`" ; config/package-lists/$-↵
    custom`repo`.list$-EXTENSION"
```

Where:

\$-EXTENSION": the optional stage suffix, see the **summary**

\$-SIGNING`KEY": the keyID of the signature of the repository

\$-URI": the URI to the repository, e.g.
http://deb.debian.org/debian/ or file://\$(pwd)/my`local`-repository

\$-SUITE": the suite within the repository, e.g. my-debian-based-distro

\$-COMPONENTS": the components within the repository, e.g. main

\$-PACKAGESFROM`REPOSITORY": the names of the packages to install (dependencies will automatically be installed as well)

8.3.3 Paquets personalitzats i APT

live-build utilitza APT per a instal·lar tots els paquets al sistema en viu, per tant, heretar els comportaments d'aquest programa. Un exemple rellevant s que (assumint una configuraci per defecte) si es dna el cas que un paquet est disponible en dos repositoris diferents,

amb diferents nmeros de versi, APT triar per a instal·lar el paquet amb la versi ms alta.

A causa d'aix, s'aconsella augmentar el nombre de la versi dels paquets personalitzats als fixers debian/changelog per a assegurar-se que la versi modificada s la que s'installa en lloc d'una dels repositoris oficials de Debian. Aix tamb es pot aconseguir mitjanant l'alteraci de les preferncies d'APT del sistema en viu - veure **APT pinning** per a ms informaci.

8.4 Configurar APT en temps de construcci

Es pot configurar APT a travs d'una srie d'opcions que noms s'apliquen en temps de construcci. (La configuraci d'APT al sistema en funcionament en viu es pot fer de forma normal per als continguts del sistema en viu, s a dir, mitjanant la inclusi de les configuracions adequades a travs de config/includes.chroot/.) Per a obtenir una llista completa, buscar les opcions que comencen amb apt a la pgina del manual de lb.config.

8.4.1 Elegir apt o aptitude

Es pot optar per utilitzar apt o aptitude a l'hora d'instal·lar paquets en temps de construcci. Quina utilitat s'usa es configura grcies al argument `-apt` de lb.config. Escollir el mtode d'implementaci per al comportament preferit durant la instal·laci de paquets, la diferencia notable s la forma en que es manegen els paquets que falten.

apt: Amb aquest mtode, si un paquet que s'especifica falta, l'instal·laci de paquets fallar. Aquesta s la configuraci per defecte.

aptitude: Amb aquest mtode, si s'especifica un paquet que falta, l'instal·laci de paquets tindr xit.

8.4.2 L's d'un proxy amb APT

One commonly required APT configuration is to deal with building an image behind a proxy. You may specify your APT proxy with the `-apt-http-proxy` option as needed, e.g.

```
$ lb config --apt-http-proxy http://proxy/
```

8.4.3 Afinar APT per a estalviar espai

Pot ser necessari estalviar espai en el medi destinat a la imatge, en aquest cas una o altra o ambds de les segentes opcions poden ser d'inters.

Si no es vol incloure els ndexs d'APT en la imatge, es poden omitir amb:

```
$ lb config --apt-indices false
```

Aix no influir en les entrades de `/etc/apt/sources.list`, sin simplement si `/var/lib/apt` cont els fitxers dels ndexs o no. El desavantatge s que APT necessita aquests ndexs per tal d'operar en el sistema en viu, aix que abans d'executar per exemple `apt-cache search` o `apt-get install`, l'usuari primer ha fer un `apt-get update` per a crear aquests ndexs.

Si es considera que la instal·laci de tots els paquets recomanats infla massa la imatge, sempre que s'estigui preparat per a fer front a les conseqncies que s'analitzen a continuaci, es pot desactivar aquesta opci per defecte d'APT amb:

```
$ lb config --apt-recommends false
```

The most important consequence of turning off recommends is that live-boot and live-config themselves recommend some packages that provide important functionality used by most Live configurations.

Two packages which you most probably will want to add again are:

user-setup which live-config recommends is used to create the live user.

sudo which live-config recommends is used to obtain root access in the live-image, which is needed to shutdown the computer.

```
$ lb config --apt-recommends false
$ echo "user-setup sudo" & config/package-lists/recommends.<
list.chroot
```

In all but the most exceptional circumstances you need to add back at least some of these recommends to your package lists or else your image will not work as expected, if at all. Look at the recommended packages for each of the live-* packages included in your build and if you are not certain you can omit them, add them back into your package lists.

The more general consequence is that if you don't install recommended packages for any given package, that is, packages that would be found together with this one in all but unusual installations (**APT pinning**).

8.4.4 Passar opcions per a apt o aptitude

Si no hi ha cap opció lb config per a modificar el comportament d'APT de la manera que es necessita, es pot utilitzar `--apt-options` o `--aptitude-options` per a passar opcions a l'eina APT configurada.

Consultar les pàgines dels manuals apt i aptitude per a més detalls. Tenir en compte que ambdues opcions tenen valors per defecte que s'hauran de mantenir, a més de les opcions que es proporcionen. Així, per exemple, suposant que s'ha inclòs algun paquet de snapshot.debian.org per a fer proves i es vol especificar `Acquire::Check-Valid-Until=false` perquè APT no es queixi de que el fitxer Release ja ha caducat es podria fer com en l'exemple següent, afegint la nova opció després del valor per defecte `--yes`:

```
$ lb config --apt-options "--yes -oAcquire::Check-Valid-Until=<
=false"
```

Consultar les pàgines del manual per a entendre completament aquestes opcions i quan utilitzar-les. Així s'ha donat un exemple i no s'ha d'interpretar com un consell per a configurar la imatge. Aquesta opció no seria adequada, per exemple, per a una versió final d'una imatge en viu.

Per a configuracions més complicades que impliquen opcions apt.conf pot ser adequat crear un fitxer `config/apt/apt.conf`. Consultar també les altres opcions apt-* per a tenir algunes dreceres convenientes per a les opcions que es necessiten amb freqüència.

8.4.5 APT pinning

Com a referència, llegir primer la pàgina del manual `apt-preferences(5)`. Es pot configurar APT pinning durant la construcció, o bé durant l'execució. En el primer cas, crear `config/archives/*.pref`, `config/archives/*.pref.chroot`, i `config/apt/preferences`. Per al segon cas, crear `config/includes.chroot/etc/apt/preferences`.

Suposem que s'està construint un sistema en viu trixie per a necessitar que tots els paquets live- que acaben dins de la imatge binària s'installin des de sid en temps de construcció. Cal afegir sid a les

fonts d'APT i fer un pin dels paquets live amb una prioritat ms alta, per tots els altres paquets amb una prioritat ms baixa que la prioritat per defecte de manera que noms els paquets que es vol s'installin desde sid en el moment de la construcci i tots els altres es prenguin de la distribuci de destinaci, trixie . Aix es pot aconseguir de la manera segent:

517

```
$ echo "deb http://mirror/debian/ sid main" > config/archives↵
/sid.list.chroot
$ cat >> config/archives/sid.pref.chroot << EOF
Package: live-*
Pin: release n=sid
Pin-Priority: 600

Package: *
Pin: release n=sid
Pin-Priority: 1
EOF
```

518

Una prioritat pin negativa evitar que un paquet s'installi, com en el cas que no es vulgui un paquet que s recomanat per un altre paquet. Suposem que s'est construint una imatge LXDE afegint task-lxde-desktop a config/package-lists/desktop.list.chroot per no es desitja que al usuari se li demani que guardi les contrasenyes wifi al keyring. Aquesta llista depn de lxde-core, que recomana gksu, que al seu torn recomana gnome-keyring. Si es vol omitir el paquet recomanat gnome-keyring, es pot fer mitjanant l'addici de les segents lnies a config/apt/preferences:

519

```
Package: gnome-keyring
Pin: version *
Pin-Priority: -1
```

Personalitzaci dels continguts

9. Personalitzaci dels continguts

Aquest capítol tracta d'afinar la personalitzaci dels continguts del sistema en viu ms enll de simplement triar els paquets que es desitja incloure. Els includes permeten afegir o reemplaçar fitxers arbitraris en la imatge en viu, els scripts ganxo (hooks) permeten executar ordres arbitrries en diferents etapes de la construcci i en el moment d'arrencar, i la preconfiguraci (preseeding) permet configurar els paquets quan s'installen proporcionant respostes a les preguntes de debconf.

9.1 Includes

Tot i que l'ideal seria un sistema en viu que inclogués noms fitxers proporcionats per paquets Debian sense modificaci, de vegades s'convenient proporcionar o modificar part del contingut a través de fitxers. Amb els includes, s'possible afegir (o substituir) fitxers arbitraris en la imatge en viu. live-build ofereix dos mecanismes per al seu s:

Chroot local includes: Aquests permeten afegir o substituir fitxers dintre de chroot/Live en el sistema de fitxers. Consultar [Live/chroot local includes](#) per a ms informaci.

Binary local includes: Aquests permeten afegir o substituir fitxers dins la imatge binria. Consultar [Binary local includes](#) per a ms informaci.

Consultar [Termes](#) per a ms informaci sobre la distinció entre les imatges Live and binary.

9.1.1 Live/chroot local includes

Es poden utilitzar els chroot local includes per a afegir o reemplaçar fitxers en el sistema de fitxers chroot/Live perquè puguin ser utilitzats en el sistema en viu. Un s' típic s per a omplir l'esquelet del directori de l'usuari (/etc/skel) utilitzat pel sistema en viu per a crear el directori home de l'usuari en viu. Un altre s' el de subministrar fitxers de configuraci que poden ser simplement afegits o reemplaats en la imatge sense processar; veure [Chroot local hooks](#) si es necessita processar-los.

Per a incloure fitxers, noms s'han d'afegir al directori config/includes.chroot. Aquest directori es correspon amb el directori arrel / del sistema en viu. Per exemple, per a afegir un fitxer /var/www/index.html en el sistema en viu, fer:

```
$ mkdir -p config/includes.chroot/var/www
$ cp /path/to/my/index.html config/includes.chroot/var/www
```

La configuraci tindrà llavors l'estructura segent:

```
-- config
[... ]
-- includes.chroot
--   -- var
--     -- www
--       -- index.html
[... ]
```

Els chroot local includes s'installen després de la instal·lació del paquets de tal manera que es sobreescrueixen els fitxers instal·lats pels paquets.

9.1.2 Binary local includes

Per a incloure material com documentaci o vdeos en el sistema de fitxers del medi en viu de manera que sigui accessible immediatament desprs de la inserci del medi sense haver de arrencar el sistema en viu, es pot utilitzar els binary local includes. Aix funciona de manera similar als chroot local includes. Per exemple, si els fitxers `~/video`demo.*` sn vdeos de demostraci del sistema en viu descrits i lligats per una pgina d'ndex HTML. Noms cal copiar el material a `config/includes.binary/` de la segent manera:

```
$ cp ~/video`demo.* config/includes.binary/
```

Aquests fitxers apareixeran ara en el directori arrel del medi en viu.

9.2 Scripts ganxo (Hooks)

Els scripts ganxo permeten executar ordres en les etapes de la construcci chroot i binary per tal de personalitzar la imatge. Depenent de si es vol construir una imatge en viu o una d'un sistema normal s'han de collocar els ganxos a `config/hooks/live` o `config/hooks/-normal` respectivament. Aquests ganxos s'anomenen amb freqncia ganxos locals, ja que s'executen dins de l'entorn de construcci.

Tamb hi ha ganxos en temps d'arrencada que permeten executar ordres una vegada que la imatge ja s'ha construt, durant el procs d'arrencada.

9.2.1 Chroot local hooks

Per a executar ordres durant l'etapa chroot, crear un script ganxo que contingui les ordres amb el sufix `.hook.chroot` i afegir-lo b

als directori `config/hooks/live` o a `config/hooks/normal`. El ganxo s'executar en el chroot desprs que la resta de la configuraci del chroot s'hagi aplicat, assegurar-se que la configuraci inclou tots els paquets i els fitxers que el ganxo necessita per a funcionar. Veure els scripts chroot d'exemple per a diverses tasques comunes de personalitzaci que es poden trobar a `/usr/share/doc/live-build/examples/hooks` que es poden copiar o fer un enlla simblic per a utilitzar-los en la prpia configuraci.

9.2.2 Binary local hooks

Per a executar ordres durant l'etapa binary, crear un script ganxo que contingui les ordres amb un sufix `.hook.binary` i afegir-lo b al directori `config/hooks/live` o a `config/hooks/normal`. El ganxo s'executar desprs que s'executin totes les ordres de l'etapa binary per abans dels `binary`checksums`, la darrera ordre de l'etapa binary. Les ordres del ganxo no s'executen al chroot, per tant tenir cura de no modificar cap fitxer de fora del arbre de construcci, o es pot fer malb el sistema de construcci! Veure els scripts ganxo binary d'exemple per a diverses tasques comunes de personalitzaci a `/usr/share/doc/live-build/examples/hooks` que es poden copiar o fer un enlla simblic per a utilitzar-los en la prpia configuraci.

9.2.3 Scripts ganxo en temps d'arrencada

Per a executar ordres durant l'arrencada, es pot proporcionar scripts ganxo per a `live-config` com s'explica a la secci Personalitzaci de la seva pgina del manual. Es poden afegir els ganxos de `live-config` a `/lib/live/config/`, tenint en compte la seqncia dels nmeros. A continuaci, afegir el script ganxo propi amb un nmero de seqncia apropiat com a prefix, ja sigui com a un chroot local include a `config/includes.-chroot/lib/live/config/`, o com un paquet personalitzat com es va discutir a [Installaci de paquets modificats o de tercers](#).

9.3 Preconfiguraci de les preguntes de Debconf

Els fitxers del directory `config/preseed/` amb el sufix `.cfg` seguits del sufix de l'etapa (`.chroot` o `.binary`) son considerats fitxers de preconfiguraci de debconf i sn installats per live-build utilitzant debconf-set-selections durant l'etapa corresponent.

Per a ms informaci sobre debconf, veure `debconf(7)` del paquet debconf.

Personalitzaci dels comportaments en temps d'execuci

10. Personalitzaci dels comportaments en temps d'execuci

Tota la configuraci que es fa durant l'execuci es feta per live-config. Aquestes sn algunes de les opcions ms comunes de live-config en que els usuaris estan interessats. Una llista completa de totes les possibilitats es poden trobar a la pgina del manual de live-config.

10.1 Personalitzar l'usuari en viu

Una consideraci important s que l'usuari en viu es creat per live-boot durant l'arrencada i no per live-build en temps de construcci. Aix influeix no noms en on s'han de introduir els materials relacionats amb l'usuari durant la construcci, tal i com es va explicar a **Live/ch-root local includes**, sin tamb en els grups i els permisos associats amb l'usuari.

You can specify additional groups that the live user will belong to by using any of the possibilities to configure live-config. For example, to add the live user to the fuse group, you can either add the following file in config/includes.chroot/etc/live/config.conf.d/10-user-setup.conf:

```
LIVE'USER'DEFAULT'GROUPS="audio cdrom dip floppy video ←
plugdev netdev powerdev scanner bluetooth fuse"
```

o utilitzar live-config.user-default-groups=audio,cdrom,dip,floppy,video,plugdev,netdev,powerdev,scanner,bluetoo com parmetre d'arrencada.

Tamb s possible canviar el nom de l'usuari per defecte user i la contrasenya per defecte live. Si es vol fer aix per alguna ra, es pot aconseguir fcilment de la segent manera:

Per a canviar el nom de l'usuari per defecte noms s'ha d'especificar en la configuraci:

```
$ lb config --bootappend-live "boot=live components username=live-user"
```

Una forma possible de canviar la contrasenya per defecte s per mitj d'un ganxo com s'explica a **Scripts ganxo durant l'arrencada**. Per a fer aix, es pot utilitzar el script ganxo passwd de /usr/share/doc/live-config/examples/hooks, posar-li un prefix adequat (per exemple 2000-passwd) i afegir-lo a config/includes.chroot/lib/live/config/

10.2 Personalitzaci de l'entorn local i el llenguatge

Quan el sistema en viu arrenca, el llenguatge est implicat en dos passos:

la generaci de locales

establir la configuraci del teclat

La configuraci local per defecte en la construcci d'un sistema viu s locales=en'US.UTF-8. Per a definir la locale que s'ha de generar, utilitzar el parmetre locales de la opci --bootappend-live de lb config, per exemple.


```
$ lb config --bootappend-live "boot=live components locales=↵
de'CH.UTF-8"
```

Es poden especificar diverses locales en una llista separada per comes.

Aquest parmetre, així com els parmetres de configuració del teclat que s'indiquen a continuació, també es pot utilitzar en la línia d'ordres del nucli. Es pot especificar una configuració regional mitjançant `language`country`` (en aquest cas s'utilitza la codificació per defecte) o la forma completa `language`country`.encoding``. Una llista de locales suportades i la codificació per a cadascuna es poden trobar a `/usr/share/i18n/SUPPORTED`.

`live-config` s'encarrega de la configuració del teclat per a X i per a la consola utilitzant el paquet `console-setup`. Per a la seva configuració es pot fer servir els parmetres d'arrencada `keyboard-layouts`, `keyboard-variants`, `keyboard-options` i `keyboard-model` mitjançant l'opció `--bootappend-live`. Es poden trobar opcions vàlides per a aquests a `/usr/share/X11/xkb/rules/base.lst`. Per a trobar distribucions de teclat i variants per a un idioma determinat, s'ha d'intentar cercar el nom en anglès de la llengua i/o el país on es parla l'idioma, per exemple:

```
$ egrep -i '(!—german.*switzerland)' /usr/share/X11/xkb/↵
rules/base.lst
! model
! layout
! ch          German (Switzerland)
! variant
!   legacy    ch: German (Switzerland, legacy)
!   de`nodeadkeys` ch: German (Switzerland, eliminate dead ↵
!     keys)
!   de`sundeadkeys` ch: German (Switzerland, Sun dead keys)
!   de`mac      ch: German (Switzerland, Macintosh)
! option
```

Tinir en compte que cada variant mostra la distribució que s'aplica en la descripció.

Sovint, només la distribució necessita ser configurada. Per exemple, per a obtenir els fitxers de configuració regional per a la distribució del teclat alemany i sus-alemany per a l'entorn gràfic X:

```
$ lb config --bootappend-live "boot=live components locales=↵
de'CH.UTF-8 keyboard-layouts=ch"
```

No obstant això, per als casos d'ús molt específics, potser es vol incloure altres parmetres. Per exemple, per a establir un sistema francès, amb una distribució de teclat French-Dvorak (anomenat Bepo) en un teclat USB TypeMatrix EZ-Reach 2030, utilitzar:

```
$ lb config --bootappend-live "
"boot=live components locales=fr'FR.UTF-8 keyboard-↵
layouts=fr keyboard-variants=bepo keyboard-model=↵
tm2030usb"
```

Es poden especificar diversos valors per a cada una de les opcions `keyboard-*` en una llista separada per comes amb l'excepció de `keyboard-model`, que només accepta un valor. Veure la pàgina del manual `keyboard(5)` per a més detalls i exemples de les variables `XKBMODEL`, `XKBLAYOUT`, `XKBVARIANT` i `XKBOPTIONS`. Si s'especifiquen diversos valors de `keyboard-variants` es correspondran un a un amb els valors `keyboard-layouts` (veure `setxkbmap(1)` opció `-variant`). Es poden utilitzar valors buits, per exemple, per a definir dos dissenys, el valor predeterminat `US QWERTY` i l'altre `US Dvorak`:

```
$ lb config --bootappend-live "
"boot=live components keyboard-layouts=us,us keyboard-↵
variants=,dvorak"
```

10.3 Persistència

Un paradigma d'un live cd és ser un sistema pre-installat, que arrenca desde medis de noms lectura, com un cdrom, on les modificacions no sobreviuen als reinicis del maquinari que l'executa.

Un sistema en viu és una generalització d'aquest paradigma i per tant, compatible amb altres medis, a més dels CDs, per tot i així, en el seu comportament per defecte, s'ha de considerar de noms lectura i totes les evolucions en temps d'execució del sistema es perden al apagar l'equip.

La Persistència és un nom com per a nomenar els diferents tipus de solucions per a guardar després de reiniciar algunes, o totes, les dades d'aquesta evolució en temps d'execució del sistema. Per a entendre com funciona, seria útil saber que, encara que el sistema s'inicia i s'executa des de medis de noms lectura, les modificacions als fitxers i directoris s'escriuen en medis d'escriptura, en general un ramdisk (tmpfs) i les dades dels discos ram no sobreviuen als reinicis.

Les dades emmagatzemades en aquest disc ram han de ser guardades en un suport d'escriptura persistent com medis d'emmagatzematge locals, un recurs compartit de xarxa o fins i tot una sessió d'una multisessió de un CD/DVD (re)grabable. Tots aquests medis són compatibles amb el sistema en viu de diferents maneres, i tots menys l'últim, requereixen un paràmetre especial que s'especifica en l'arrencada: persistence.

Si s'utilitza el paràmetre d'arrencada persistence (i no s'utilitza no-persistence) es proven els medis locals d'emmagatzematge (per exemple, discs durs, unitats USB) buscant volums amb persistència durant l'arrencada. És possible restringir els tipus de volums amb persistència que s'utilitzin mitjançant l'especificació de certs paràmetres d'arrencada que es descriuen a la pàgina del manual de live-boot(7). Un volum amb persistència és qualsevol dels següents:

una partició, identificada pel seu nom GPT.

un sistema de fitxers, identificat per la seva etiqueta de sistema de fitxers.

un fitxer imatge situat en l'arrel de qualsevol sistema de fitxers llegibles (fins i tot una partició NTFS d'un altre SO), identificat pel seu nom de fitxer.

L'etiqueta de volum per als overlays ha de ser persistence per ser passat per alt a menys que contingui un fitxer anomenat persistence.conf que s'utilitza per a personalitzar completament la persistència del volum, és a dir, especificar els directoris que es volen conservar en el volum de persistència després de reiniciar. Veure [El fitxer persistence.conf](#) per a més detalls.

Aquests són alguns exemples de com preparar un volum que s'utilitza per a la persistència. Pot ser, per exemple, una partició ext4 en un disc dur o en una clau USB creat amb, per exemple:

```
# mkfs.ext4 -L persistence /dev/sdb1
```

Veure també [Utilitzar l'espai lliure en una memòria USB](#).

Si ja hi ha una partició al dispositiu, es pot canviar l'etiqueta amb un dels següents:

```
# tune2fs -L persistence /dev/sdb1 # for ext2,3,4 filesystems
```

Heus aquí un exemple de com crear un fitxer imatge basat en ext4 per a ser utilitzat per a la persistència:

```
$ dd if=/dev/null of=persistence bs=1 count=0 seek=1G # for a 1GB sized image file
$ /sbin/mkfs.ext4 -F persistence
```

Un cop s'ha creat el fitxer imatge, per exemple, per a fer /usr persistent per noms guardant els canvis que es fan en aquest directori i no tots els continguts de /usr, es pot utilitzar l'opci union. Si el fitxer imatge es troba en el directori home, copiar-lo a l'arrel del sistema de fitxers del disc dur i muntar-lo a /mnt de la segent manera:

```
# cp persistence /
# mount -t ext4 /persistence /mnt
```

A continuaci, crear el fitxer persistence.conf afegint contingut i desmuntar el fitxer imatge.

```
# echo "/usr union" >> /mnt/persistence.conf
# umount /mnt
```

Ara, reiniciar el sistema i arrencar el medi en viu amb el parmetre d'arrencada persistence.

10.3.1 El fitxer persistence.conf

Un volum amb l'etiqueta persistence ha de ser configurat mitjanant un fitxer persistence.conf per a fer directoris arbitraris persistents. Aquest fitxer, ubicat a l'arrel del sistema de fitxers del volum, controla els directoris que fa persistents, i de quina manera.

A la pgina del manual de persistence.conf(5) s'explica en detall com es configuren els muntatges de les overlays, per un simple exemple hauria de ser suficient per a la majoria d'usos. Si es vol fer el directori home i el directori del cache d'APT persistents en un sistema de fitxers ext4 a la partici /dev/sdb1:

```
# mkfs.ext4 -L persistence /dev/sdb1
# mount -t ext4 /dev/sdb1 /mnt
# echo "/home" >> /mnt/persistence.conf
# echo "/var/cache/apt" >> /mnt/persistence.conf
# umount /mnt
```

Desprs es reinicia el sistema. Durant la primera arrencada els continguts de /home i /var/cache/apt es copiaran en el volum de la persistncia, i d'aqu en endavant tots els canvis en aquests directoris es guardaran en aquest volum. Tenir en compte que les rutes que apareixen en el fitxer persistence.conf no poden contenir espais en blanc o els components especials . i ... A ms, ni /lib, /lib/live (o qualsevol dels seus subdirectoris) ni / es poden fer persistents utilitzant muntatges personalitzats. Com a soluci per a aquesta limitaci es pot afegir / union al fitxer persistence.conf per a aconseguir una persistncia completa.

10.3.2 Utilitzar diversos medis persistents

Hi ha diferents mtdes per utilitzar mltiples volums de persistncia per a diferents casos d's. Per exemple, utilitzar diversos volums al mateix temps o seleccionar-ne noms un, entre varis, per a fins molt especfics.

Es poden utilitzar diversos volums diferents de muntatges personalitzats (amb els seus propis fitxers persistence.conf per si diversos volums fan que el mateix directori sigui persistent, noms s'utilitzar un d'ells. Si qualsevol dels dos muntatges sn imbricats (s a dir, un s un sub-directori de l'altre) el directori pare es muntar abans que el directori fill per a evitar que amb el muntatge un directori no sigui ocultat per l'altre. Els muntatges personalitzats imbricats sn problemtics si estan enumerats en el mateix fitxer persistence.conf. Veure la pgina del manual persistence.conf(5) per a saber com manejar aquest cas, si realment es necessita (una pista: en general no cal

fer-ho).

Un possible cas d's: Per a guardar les dades de l'usuari, s a dir /home i les dades del superusuari, s a dir /root en diferents particions, crear dues particions amb l'etiqueta persistence i afegir un fitxer persistence.conf en cadascuna d'aquesta manera # echo /home ; persistence.conf per a la primera partici que guardar els fitxers de l'usuari i # echo /root ; persistence.conf per a la segona partici que emmagatzemar els fitxers del superusuari. Finalment, utilitzar el parmetre d'arrencada persistence.

Si un usuari necessita mltiples volums de persistncia del mateix tipus per a diferents ubicacions o proves, com private i work, el parmetre d'arrencada persistence-label utilitzat juntament amb el parmetre d'arrencada persistence permetr tenir diversos dispositius amb la mateixa, per nica, persistncia. Un exemple seria si un usuari vol utilitzar una partici amb persistncia amb l'etiqueta private per a dades personals com els marcadors d'un navegador, utilitzaria els parmetres d'arrencada: persistence persistence-label=private. I per emmagatzemar dades relacionades amb el treball, com a documents, projectes de recerca o d'un altre tipus, utilitzaria els parmetres d'arrencada: persistence persistence-label=work.

s important recordar que aquests volums, private i work, tamb necessiten tenir un fitxer persistence.conf en la seva arrel. La pgina de manual de live-boot cont ms informaci sobre com utilitzar aquestes etiquetes amb els noms ms antics.

10.3.3 Persistncia amb xifratge

Utilitzar la persistncia vol dir que algunes dades sensibles poden quedar exposades a risc. Especialment si les dades persistents s'emmagatzemen en un dispositiu porttil com una memria USB o un disc dur extern. s llavors quan el xifrat entra en joc. Fins i tot si el procediment pot semblar complicat a causa de la quantitat de

passos que s'han de fer, s molt fcil manejar particions xifrades amb live-boot. Per a utilitzar luks , que s el tipus de xifrat compatible, es necessita instal·lar cryptsetup tant en la mquina que crear la partici xifrada com en el sistema en viu amb que es va a utilitzar la partici persistent xifrada.

Per a instal·lar cryptsetup a la nostra mquina:

```
# apt-get install cryptsetup
```

Per a instal·lar cryptsetup en el sistema viu, afegir-lo a una package-lists:

```
$ lb config
$ echo "cryptsetup cryptsetup-initramfs" ; config/package-↵
lists/encryption.list.chroot
```

Una cop tinguem el nostre sistema en viu amb cryptsetup, bsicament, noms hem de crear una nova partici, xifrar-la i arrencar amb els parmetres persistence i persistence-encryption=luks. Podrem haver anticipat aquest pas i afegit els parmetres d'arrencada seguint el procediment habitual:

```
$ lb config --bootappend-live "boot=live components ↵
persistence persistence-encryption=luks"
```

Anem a entrar en els detalls per a tothom que no est familiaritzat amb el xifrat. En el segent exemple utilitzarem una partici en un dispositiu USB que correspon a /dev/sdc2. Tenir en compte que cal determinar quina partici s la que es va a utilitzar en cada cas especfic.

El primer pas s connectar la memria usb i determinar de quin dis-

positiu es tracta. La manera ms recomanable per a llistar dispositius s utilitzar `ls -l /dev/disk/by-id`. Desprs d'aix, crear una nova partici i, a continuaci, xifrar-la amb una frase de contrasenya de la segent manera:

```
# cryptsetup --verify -passphrase luksFormat /dev/sdc2
```

A continuaci, obrir la partici luks al mapeador de dispositius virtuals. Es pot utilitzar qualsevol nom que es desitgi. Aqu utilitzem `live` com a exemple:

```
# cryptsetup luksOpen /dev/sdc2 live
```

El segent pas s omplir el dispositiu amb zeros abans de crear el sistema de fitxers:

```
# dd if=/dev/zero of=/dev/mapper/live
```

Ara, estem preparats per a crear el sistema de fitxers. Noteu que estem afegint l'etiqueta `persistence` perqu el dispositiu es munti com a magatzem de persistncia durant l'arrencada.

```
# mkfs.ext4 -L persistence /dev/mapper/live
```

Per continuar amb la nostra configuraci, necessitem muntar el dispositiu, per exemple, a `/mnt`.

```
# mount /dev/mapper/live /mnt
```

I crear el fitxer `persistence.conf` a l'arrel de la partici. Aix s, com s'ha explicat abans, estrictament necessari. Veure [El fitxer persistence.conf](#).

```
# echo "/ union" > /mnt/persistence.conf
```

Desmuntar el punt de muntatge:

```
# umount /mnt
```

I opcionalment, encara que podria ser una bona manera de protegir les dades que acabem d'agregar a la partici, podem tancar el dispositiu:

```
# cryptsetup luksClose live
```

Anem a resumir el procs. Fins ara, hem creat un sistema viu capa de manejar xifratge, que es pot copiar a una memria USB com s'explica a [Copiar una imatge ISO hbrida en un dispositiu USB](#). Tamb hem creat una partici xifrada, que es pot situar en la mateixa memria USB per portar a tot arreu i hem configurat la partici xifrada per ser utilitzada com a magatzem de persistncia. Aix que ara, noms hem de arrencar el sistema en viu. En el moment d'arrencar, `live-boot` ens preguntar la frase de contrasenya i muntar la partici xifrada per a ser utilitzada per a la persistncia.

633

634

635

636

637

638

Personalitzaci de la imatge binria

```
include menu.cfg
default vesamenu.c32
prompt 0
timeout 50
```

11. Personalitzaci de la imatge binria

11.1 Carregadors d'arrencada

live-build utilitza syslinux i alguns dels seus derivats (depenent del tipus d'imatge) com carregadors d'arrencada per defecte. Es poden personalitzar fcilment per satisfer totes les necessitats.

Per a utilitzar un tema complet, copiar /usr/share/live/build/-bootloaders a config/bootloaders i editar els fitxers all. Si no es vol modificar totes les configuracions dels carregadors d'arrencada disponibles, noms cal utilitzar una cpia local personalitzada d'un dels carregadors, per exemple, copiar la configuraci d'isolinux a config/bootloaders/isolinux ja s suficient, depenent del cas d's.

Quan es modifica un dels temes per defecte, si es vol utilitzar una imatge de fons personalitzada que es mostrar juntament amb el men d'arrencada, es pot afegir una imatge de 640x480 pxels. Aleshores, esborrar el fitxer splash.svg.

Hi ha moltes possibilitats a l'hora de fer canvis. Per exemple, els derivats de syslinux estan configurats per defecte amb un temps d'espera de 0 (zero) el que significa que faran una pausa indefinida en la seva pantalla inicial fins que es premi una tecla.

Per a modificar el temps d'espera d'arrencada d'una imatge iso-hybrid es pot editar el fitxer isolinux.cfg especificant el temps d'espera en unitats de segons 1/10. Un fitxer isolinux.cfg modificat per a arrencar desprs de cinc segons seria semblant a aquest:

11.2 metadades ISO

Quan es crea una imatge binria ISO9660, es poden utilitzar les segents opcions per a afegir diverses metadades textuais. Aix pot ajudar a identificar fcilment la versi o la configuraci d'una imatge sense arrencar-la.

LB'ISO'APPLICATION/-iso-application NAME: Ha de descriure l'aplicaci que ser a la imatge. La longitud mxima per a aquest camp s de 128 carcters.

LB'ISO'PREPARER/-iso-preparer NAME: Ha de descriure al preparador de la imatge, en general amb algunes dades de contacte. El valor per defecte per a aquesta opci s la versi de live-build utilitzada, la qual cosa pot ajudar amb la depuraci d'errors posterior. La longitud mxima per a aquest camp s de 128 carcters.

LB'ISO'PUBLISHER/-iso-publisher NAME: Ha de descriure l'editor de la imatge, en general amb algunes dades de contacte. La longitud mxima per a aquest camp s de 128 carcters.

LB'ISO'VOLUME/-iso-volume NAME: Aix ha d'especificar l'ID de volum de la imatge. Aix s'utilitza com una etiqueta visible per a l'usuari en algunes plataformes com Windows i Apple Mac OS. La longitud mxima per a aquest camp s de 32 carcters.

Personalitzaci de l'installador de debian

12. Personalitzaci de l'installador de debian

Les imatges del sistema en viu es poden integrar amb l'installador de Debian. Hi ha un nombre de diferents tipus d'installaci, que varien en el que s'inclou i en com opera l'installador.

Tenir en compte l's acurat de les lletres majscules quan es refereix a l'installador de Debian en aquesta secci - quan s'utilitza aix ens referim explicitament a l'installador normal del sistema Debian, i no a una altra cosa. Es veu sovint abreuajat com d-i.

12.1 Tipus d'installador de Debian

Els tres principals tipus d'installador sn els segents:

Installador de Debian Normal : Aquesta s una imatge normal de sistema en viu amb un nucli i initrd independents que (quan es seleccionen des del carregador d'arrencada adequat) realitzen una installaci estndard de Debian, igual que si s'hagus descarregat i arrencat una imatge de Debian des d'un CD. Les imatges que contenen un sistema viu i aquest tipus d'installador independent s'anomenen sovint imatges combinades.

Amb aquest tipus d'imatges, Debian s'installa descarregant i instal·lant paquets .deb mitjanant debootstrap, des dels medis locals o alguna xarxa, aix resulta en un sistema Debian per defecte instal·lat al disc dur.

Tot aquest procs pot ser preconfigurat i personalitzadat de moltes formes, veure les pgines corresponents al manual de l'installador de

Debian per a ms informaci. Quan es t un fitxer de preconfiguraci que funcioni, live-build pot posar-lo automticament a la imatge i activar-lo.

Installador de Debian Live : Aquesta s una imatge en viu amb un nucli i initrd independents que (quan es seleccionen des del carregador d'arrencada adequat) llancen un installador de Debian.

La installaci continuar de forma idntica a l'installaci que s'ha descrit anteriorment, per en la fase d'installaci dels paquets, en lloc d'utilitzar debootstrap per a buscar-los i instal·lar-los, es copia el sistema de fitxers viu a la destinaci. Aix s'aconsegueix amb un udeb especial anomenat live-installer.

Desprs d'aquesta etapa, l'installador de Debian continua de forma normal, installant i configurant elements com ara els gestors d'arrencada i els usuaris locals, etc

Nota: per a donar suport a les entrades de l'installador normal i live en el gestor d'arrencada del mateix medi s'ha de desactivar el live-installer mitjanant la preconfiguraci live-installer/enable=false.

Installador de Debian d'escriptori : Independentment del tipus d'installador de Debian incls, es pot iniciar el d-i des de l'escriptori fent clic damunt una icona. Aix s senzill per a l'usuari per perqu funcioni s'ha d'afegir el paquet debian-installer-launcher.

Tenir en compte que, per defecte, live-build no inclou imatges de l'installador de Debian en les imatges, ha de ser especficament activat amb lb config. A ms, tenir en compte que per a que funcioni l'installador d'escriptori el nucli del sistema viu ha de coincidir amb el nucli que el d-i utilitza per a l'arquitectura especificada. Per exemple:

```
$ lb config --debian-installer live
```

```
$ echo debian-installer-launcher && config/package-lists/my.↵  
list.chroot
```

12.2 Personalitzaci de l'installador de Debian amb preconfiguraci

As described in the Debian Installer Manual, Appendix B at <https://www.debian.org/releases/stable/amd64/apb.en.html>, Preseeding provides a way to set answers to questions asked during the installation process, without having to manually enter the answers while the installation is running. This makes it possible to fully automate most types of installation and even offers some features not available during normal installations. This kind of customization is best accomplished with live-build by placing the configuration in a preseed.cfg file included in config/includes.installer/. For example, to preseed setting the locale to en`US:

```
$ echo "d-i debian-installer/locale string en`US" "  
&& config/includes.installer/preseed.cfg
```

12.3 Personalitzar el contingut de l'installador de Debian

Per motius experimentals o de depuraci d'errors, s possible que es vulgui incloure paquets udeb creats localment per al d-i. Per a afegir-los a la imatge posar-los a config/packages.binary/. Es poden incloure fitxers addicionals o de substituci i alguns directoris a l'initrd de l'installador d'una manera similar a **Live/chroot local includes**, posant el material a config/includes.installer/.

Projecte

Contribuir al projecte

13. Contribuir al projecte

Quan s'envia una contribuci, s'ha d'identificar clarament el titular dels drets d'autor i incloure la declaraci de concessi de llicncies aplicables. Recordar que per a ser acceptada, la contribuci ha de tenir una llicencia igual que la resta del document, a saber, la versi de la GPL 3 o superior.

Contributions to the project, such as translations and patches, are greatly welcome. Anyone can send merge requests. The projects are hosted on Salsa: <https://salsa.debian.org/live-team> follow Salsa's documentation for instructions on how to contribute.

Tot i que tots els lliuraments poden ser revisats, demanem que s'utilitzi el sentit com i es facin bons lliuraments amb bons missatges.

Escriure missatges de lliurament que consisteixen en oracions completes i significatives en angls, comenament amb una lletra majúscula i acabant amb un punt. En general, aquests comenaran amb la forma 'Fixing/Adding/Removing/Correcting/Translating/...'.

Escriure bons missatges de lliurament. La primera línia ha de ser un resum exacte dels continguts del lliurament, que s'inclour en la llista de canvis. Si es necessita fer algunes explicacions ms, escriure a sota deixant una línia en blanc després de la primera línia i després una altra línia en blanc després de cada paràgraf. Les línies dels paràgrafs no han de superar els 80 caràcters de longitud.

Fer lliuraments de manera atmica, s a dir, no barrejar coses no relacionades en el mateix lliurament. Fer un lliurament diferent per a cada canvi que es faci.

13.1 Traducci de pàgines dels manuals

Tamb es pot contribuir al projecte treballant en la traducci de les pàgines dels manuals dels diferents paquets live-* que el projecte mant. El procediment pot ser diferent, depenent de si s'est comenament una traducci des de zero o si es segueix treballant en una ja existent:

Continuar amb una traducci ja comenada

Si es vol treballar en la traducci d'una llengua ja existent s'ha de fer els canvis als fitxers que hi ha dins `manpages/po/$-LANGUAGE"/*-*.po` i després executar `make rebuild` des de dins del directori `manpages/`. Així actualitzar les pàgines a `manpages/$-LANGUAGE"/*-*`

Comenar una nova traducci des de zero

Per afegir una nova traducci de qualsevol de les pàgines dels manuals del projecte s'ha de seguir un procediment similar. Es podria resumir de la següent manera:

Obrir el fitxer o fitxers que hi ha a `manpages/pot/` en el teu editor preferit, com per exemple `poedit`, i guardar-lo com un fitxer `.po` a `manpages/po/$-LANGUAGE"/`. (S'haur de crear el directori del `$-LANGUAGE"/` corresponent).

Executar `make rebuild` dins el directori `manpages/` per a crear els fitxers a `manpages/$-LANGUAGE"/` que contindran les pàgines dels manuals.

Recordar que cal afegir tots els directoris i fitxers, i després fer el `commit` i finalment fer un `git push` al servidor `git`.

Informar dels errors

14. Informar dels errors

Live systems est lluny de ser perfecte, per volem que sigui el ms perfecte possible - amb la vostra ajuda. No dubtar d'informar sobre un error. s millor omplir un informe dues vegades que mai. No obstant aix, aquest capitol inclou recomanacions sobre com presentar bons informes d'errors.

Per als impacients

First check whether the bugs has been reported already. You can see the full list of bugs that are assigned to the live-team at <https://bugs.debian.org/cgi-bin/pkgreport.cgi?maint=debian-live%40lists.debian.org>.

Before submitting a bug report always try to reproduce the bug with the most recent versions of the packages of live-build, live-boot, live-config and live-tools that you're using.

Intentar donar la informaci ms especifica possible sobre l'error. Aix inclou (almenys) la versi de live-build, live-boot, live-config i live-tools i la distribuci del sistema en viu que s'est construint.

14.1 Problemes coneguts

Currently known issues are listed in the BTS at <https://bugs.debian.org/cgi-bin/pkgreport.cgi?maint=debian-live%40lists.debian.org>.

Note: Since Debian testing and Debian unstable distributions are moving targets, when you specify either of them as the target system distribution, a successful build may not always be possible.

Si aix s massa difcil, no construir un sistema basat en testing o

unstable , sin ms aviat, utilitzar stable . live-build sempre construeix la versi stable per defecte.

It is out of the scope of this manual to train you to correctly identify and fix problems in packages of the development distributions, however, you can always try the following: If a build fails when the target distribution is testing , try unstable . If unstable does work, revert to testing and pin the newer version of the failing package from unstable (see [APT pinning](#) for details).

14.2 Fer la recerca

Abans de presentar l'informe d'errors, cercar a la web el missatge d'error o smptoma que s'est rebent. Ja que s molt poc probable que sigui l'nica persona que t un problema en particular. Sempre hi ha una possibilitat que hagi estat discutit en un altre lloc i hi hagi una possible soluci, pegat o s'hagi proposat una soluci alternativa.

S'ha de prestar especial atenci a la llista de correu dels sistemes en viu, aix com a la pgina web, ja que s probable que continguin la informaci ms actualitzada. Si aquesta informaci existeix, incloure una refernca a aquesta en l'informe d'errors.

A ms, s'hauria de comprovar les llistes d'errors actuals de live-build, live-boot, live-config i live-tools per a veure si ja s'ha informat sobre alguna cosa semblant .

14.3 Reconstruir des de zero

Per a assegurar-se que un error en particular no s causat per un sistema mal construt, reconstruir sempre tot el sistema en viu a partir de zero per veure si l'error s reproducible.

14.4 Fer servir paquets actualitzats

Using outdated packages can cause significant problems when trying to reproduce (and ultimately fix) your problem. Make sure your build system is up-to-date and any packages included in your image are up-to-date as well. If possible, try to reproduce the bug with the newest code from source, see [Installation](#) for details.

14.5 Recopilar informaci

Proporcionar informaci suficient amb l'informe. Incloure, com a mnim, la versi exacta de live-build i els passos per a reproduir-lo. Utilitzar el sentit com i proporcionar tota altra informaci pertinent si es pensa que aix pot ajudar a resoldre el problema.

Per a treure el mxim profit del informe d'errors, es requereix com a mnim la informaci segent:

Arquitectura del sistema amfitri

Distribuci del sistema amfitri

Versi de live-build al sistema amfitri

Versi de debootstrap al sistema amfitri

Arquitectura del sistema en viu

Distribuci del sistema en viu

Versi de live-boot al sistema amfitri

Versi de live-config al sistema amfitri

Versi de live-tools al sistema amfitri

Es pot generar un log del procs de construcci mitjanant l'ordre tee. Recomanem fer-ho automticament amb un script auto/build (veure [Gesti d'una configuraci](#) per a ms detalls).

```
# lb build 2i&1 -- tee build.log
```

Durant l'arrencada, live-boot i live-config emmagatzemen els seus logs a /var/log/live/. Comprovar aquest fitxers per a detectar misatges d'error.

A ms, per a descartar altres errors, sempre s una bona idea comprimir el directori config/ i pujar-lo a algun lloc (no enviar-lo com arxiu adjunt a la llista de correu), perqu puguem tractar de reproduir els errors que s'han trobat. Si aix s difcil (per exemple, a causa de la mida del arxiu) es pot utilitzar la sortida de lb config --dump que produeix un resum del arbre de configuraci (s a dir, fa un llistat dels fitxers dins els subdirectoris de config/, per no els inclou).

Recordar que s'ha d'enviar qualsevol log que es produeixi amb la configuraci regional en angls, per exemple, executar les ordres de live-build comenant per LC'ALL=C o LC'ALL=en'US.

14.6 Allar el cas que falla, si s possible

Si pot ser, allar el cas que falla al canvi ms petit possible que fa que no funcioni. No sempre s fcil fer aix, per tant, si no es possible fer-ho pel informe, no preocupar-se. No obstant aix, si es planeja b el cicle de desenvolupament, i s'utilitzen petits conjunts de canvis suficients per iteraci, es pot ser capa d'allar el problema mitjanant la construcci d'una configuraci 'base' ms senzilla que s'ajusti a la configuraci desitjada ms el conjunt de canvis que fa que no funcioni. Si es difcil classificar quins canvis fan que falli, pot ser que s'inclougui massa en cada conjunt de canvis i s'ha de desenvolupar en petits increments.

14.7 Utilitzar el paquet correcte per a informar de l'error

En general, s'ha d'informar dels errors en temps de construcci contra

el paquet live-build, dels errors durant l'arrencada contra live-boot i dels errors de temps d'execuci contra live-config. Si no s'est segur de quin paquet s l'adequat o es necessita ms ajuda abans d'enviar un informe d'errors, informar contra el pseudopaquet debian-live. Nosaltres ens farem crrec d'ell i el reassignarem on sigui procedent.

No obstant aix, estarem molt agrats si s'intenta limitar la recerca segons el lloc on apareix l'error.

14.7.1 A l'hora de construir mentre bootstrapping

live-build crea primer un sistema Debian bsic amb debootstrap. Si un error apareix aqu, comprovar si l'error est relacionat amb un paquet especfic de Debian (el ms probable), o si est relacionat amb l'eina debootstrap en si mateixa.

En ambds casos, aix no s un error del sistema en viu, sin de Debian en si mateix i, probablement, no ho podem arreglar directament. Informar del error sobre l'eina de debootstrapping o el paquet que falla.

14.7.2 A l'hora de construir, durant la installaci de paquets

live-build installa paquets addicionals del arxiu de Debian i en funci de la distribuci Debian utilitzada i de l'estat diari de l'arxiu, pot fallar. Si un error apareix aqu, comprovar si l'error s tamb reproducible en un sistema normal.

Si aquest s el cas, no es tracta d'un error del sistema en viu, sin de Debian - Informar d'aix sobre el paquet que falla. Executar debootstrap per separat de la construcci del sistema Live o executar lb bootstrap -debug per a tenir ms informaci.

A ms, si es fa servir un mirall local i/o qualsevol tipus de proxy i s'est

experimentant algun problema, sempre s'ha de mirar de reproduir-lo fent un bootstrapping a partir d'un mirall oficial.

14.7.3 En el moment d'arrencar

Si la imatge no arrenca, informar a la llista de correu, juntament amb la informaci sollicitada a **Recopilar informaci**. No oblidar-se d'esmentar, com/quan la imatge falla, ja sigui amb virtualitzaci o maquinari real. Si s'utilitza una tecnologia de virtualitzaci d'algun tipus, sempre fer la prova amb maquinari real abans d'informar d'un error. Proporcionar una captura de pantalla de l'error s tamb molt til.

14.7.4 En temps d'execuci

If a package was successfully installed, but fails while actually running the Live system, this is probably a bug in live-config.

14.8 On informar dels errors

El Debian Live Project mant un registre de tots els errors en el sistema de seguiment d'errors de Debian (BTS). per a obtenir informaci sobre la utilitzaci del sistema, es pot consultar <https://bugs.debian.org/>. Tamb es poden enviar els informes dels errors mitjanant l'ordre reportbug del paquet amb el mateix nom.

Tenir en compte que els errors trobats en les distribucions derivades de Debian (com Ubuntu i altres) no han de ser enviats al BTS de Debian tret que puguin ser reproduits tamb en sistemes Debian utilitzant paquets oficials de Debian.

Estil de Codi

esac

15. Estil de Codi

En aquest capítol es documenta l'estil de codi utilitzat a live systems.

15.1 Compatibilitat

Avoid bashisms, the codebase must be POSIX compliant and thus universally compatible.

Furthermore it must comply with the version of the POSIX specification chosen by the current Debian Policy.

Es pot comprovar els scripts amb 'sh -n' i 'checkbashisms'.

Assegurar-se que tot el codi funciona amb 'set -e'.

15.2 Indentació

Utilitzar sempre tabuladors en lloc d'espais.

Keep case branch terminators (;;) aligned with the content of the branch, rather than the branch entry.

B:

```
case "$1" in
    foo)
        foobar
        ;;
    bar)
        foobar
        ;;
esac
```

15.3 Ajust de l'ínia

Generally, lines should be 80 chars at maximum.

Placement of keywords like then and do should be chosen with good judgement with respect to clutter and readability. For small bits of code in particular it should be preferred to have them on the same line as the prior keyword they relate to (if; for; etc). Only place on the next line where it makes good sense to do so; typically this might only be to comply with maximum line length restrictions. One situation where they should always be placed on the next line is where what they follow is broken up onto multiple lines, and thus it being on a new line creates clear separation between that and the body of code following it. I.e. :

Preferred:

```
if foo; then
    bar
fi

for FOO in $ITEMS; do
    bar
done

if [ "$MYLOCATIONVARIABLE" = "something" ] && [ -e "$MYOUTPUTFILE" ]
then
    MYOTHERVARIABLE="$(some`bin $-FOOBAR" — awk -F' ' —
    print $1 " ")"
fi

if [ "$MYFOO" = "something" ] && [ -e "path/$-FILE1" ]
[ "$MYBAR" = "something else" ] && [ $-ALLOW = "true" ]
```

```
then
    foobar
fi
```

```
Foo () -
    bar
"
```

Less ideal:

Awful:

```
if [ "$MYLOCATIONVARIABLE" = "something" ] && [ -e "$MYOUTPUTFILE" ]; then
    MYOTHERVARIABLE="$(some`bin $-FOOBAR" — awk -F' ' —
    print $1 " ")"
fi
```

```
Foo ()
-
    bar
"
```

Horrible:

```
if [ "$MYLOCATIONVARIABLE" = "something" ] && [ -e "$MYOUTPUTFILE" ] — [ "$MYLOCATIONVARIABLE" = "something-else" ] && [ -e "$MYOUTPUTFILE2" ]; then
    MYOTHERVARIABLE="$(some`bin $-FOOBAR" — awk -F' ' —
    print $1 " ")"
fi
```

Prefer placing the opening brace of a function on a new line (for consistency with established style), and keep the braces aligned with the function name:

```
Foo ()
-
    bar
"
```

Bad (inconsistent with existing style):

15.4 Variables

Les variables van sempre en majúscules.

Config variables used in live-build should start with an LB` prefix.

Local function variables should be restricted to local scope.

Les variables en relaci a un parmetre d'arrencada de live-config comencen amb LIVE`.

Totes les altres variables de live-config comencen amb el prefix `

Utilitzar claus al voltant de les variables, per exemple, escriure \$-FOO` en lloc de \$FOO`.

Always protect variables with quotes to respect potential whitespaces (except where necessary to achieve correct word splitting): write \$-FOO` not \$-FOO`.

Per raons de coherencia, utilitzar sempre cometes al assignar valors a les variables:

Mal:

```
FOO=bar
```

789 B:

```
790 FOO="bar"
```

791 If multiple variables are used, prefer quoting the full expression:

792 Typically bad:

```
793 if [ -f "$-FOO"/foo/"$-BAR"/bar ]; then  
    foobar  
fi
```

794 B:

```
795 if [ -f "$-FOO"/foo/"$-BAR"/bar ]; then  
    foobar  
fi
```

796 15.5 Miscellnia

797 Prefer `—` (without the surround quotes) as a separator in calls
to `sed`, e.g. `sed -e 's—'` (without `"`).

798 Don't use the test command for comparisons or tests, use `[` and `]`
(without `"`); e.g. `if [-x /bin/foo]; ...` and not `if test -x /bin/foo;`
....

799 Use `case` wherever it makes code more readable than conditional
checks (`if foo; ...` and tests without the actual `if` keyword, e.g. `[`
`-e $-FILE"`] `—exit 0`).

800 Use `Foo'bar` style names for functions, i.e. a capital first let-
ter, then all lowercase, with sensible use of underscores for better
readability.

Exemples

Exemples

16. Exemples

En aquest capítol s'inclouen exemples de construccions per a casos d'ús específics amb sistemes en viu. Si s'és nou en la construcció d'imatges en viu pròpies, us suggerim mirar els tres tutorials en seqüència, ja que cada un ensenya noves tècniques que ajuden a utilitzar i entendre els exemples restants.

16.1 Els exemples

per a utilitzar aquests exemples es necessita un sistema de construcció que compleixi les exigències enumerades a [Requisits](#) y que tingui live-build instal·lat com es descriu a [Instal·lació de live-build](#).

Cal notar que, per a abreviar, en aquests exemples no s'especifica un mirall local per a utilitzar en la construcció. Es poden accelerar les descàrregues considerablement si s'utilitza un mirall local. Es pot especificar les opcions quan s'utilitza lb config, com es descriu a [Miralls de distribució utilitzats en temps de construcció](#), o per a major comoditat, establir el valor predeterminat per al sistema de construcció en el fitxer /etc/live/build.conf. Noms cal crear aquest fitxer i establir a les variables al mirall preferit. Tots els altres miralls que s'utilitzin en la construcció adoptaran valors per defecte segons aquests valors. Per exemple:

```
LB_MIRROR_BOOTSTRAP="http://mirror/debian/"
LB_MIRROR_CHROOT_SECURITY="http://mirror/debian-security/"
LB_MIRROR_CHROOT_BACKPORTS="http://mirror/debian-updates/"
```

16.2 Tutorial 1: Una imatge per defecte

Cas d'ús: Crear una primera imatge senzilla, aprenent els conceptes bàsics de live-build.

En aquest tutorial, anem a construir una imatge ISO híbrida per defecte que contingui noms els paquets de base (no t'Xorg) i altres paquets de suport de sistema en viu, com un primer exercici en l'ús de live-build.

No pot ser més senzill que això:

```
$ mkdir tutorial1 ; cd tutorial1 ; lb config
```

Examinar el contingut del directori config/ si es vol. Es veurà emmagatzemada aquí una configuració en esquelet, a punt per a ser personalitzada o, en aquest cas, per a ser utilitzada immediatament per a construir una imatge per defecte.

Ara, com a superusuari, construir la imatge, guardant un log del que es construeix amb tee.

```
# lb build 2;&1 — tee build.log
```

Assuming all goes well, after a while, the current directory will contain live-image-amd64.hybrid.iso. This ISO hybrid image can be booted directly in a virtual machine as described in [Testing an ISO image with Qemu](#) and [Testing an ISO image with VirtualBox](#), or else imaged onto optical media or a USB flash device as described in [Burning an ISO image to a physical medium](#) and [Copying an ISO hybrid image to a USB stick](#), respectively.

16.3 Tutorial 2: Una utilitat de navegador web

Cas d'ús: Crear una imatge d'una utilitat de navegador web, apre-

nent a aplicar personalitzacions.

En aquest tutorial, anem a crear una imatge adequada per al seu s com a una utilitat de navegador web, que serveix com introducci a la personalitzaci de les imatges de sistemes en viu.

```
$ mkdir tutorial2
$ cd tutorial2
$ lb config
$ echo "task-lxde-desktop firefox-esr" && config/package-<
lists/my.list.chroot
```

La nostra elecci de LXDE per a aquest exemple reflecteix el nostre desig d'oferir un entorn d'escriptori mnim, ja que l'objectiu de la imatge s l'nic s que tenim al cap, el navegador web. Podrem anar encara ms lluny i oferir una configuraci per defecte per al navegador web a config/includes.chroot/etc/iceweasel/profile/, o paquets addicionals de suport per a la visualitzaci de diversos tipus de contingut web, per deixem aix com a un exercici per al lector.

Construir la imatge, de nou com a superusuari, guardant un log com al **Tutorial 1**:

```
# lb build 2&1 — tee build.log
```

Un cop ms, verificar que la imatge est b i provar-la, com al **Tutorial 1**.

16.4 Tutorial 3: Una imatge personalitzada

Cas d's: Crear un projecte per a construir una imatge personalitzada, que contingui el programari favorit per a portar en una memria USB all on es vagi i que evolucionar en revisions successives tal i com les necessitats i les preferncies canvien.

Com la nostra imatge personalitzada canviar durant un nombre de revisions i volem fer un seguiment d'aquests canvis, provar coses experimentals i possiblement revertir-les si les coses no surten b, anem a mantenir la nostra configuraci en el popular sistema de control de versions git. Tamb utilitzarem les millors prctiques de configuraci automtica mitjanant scripts auto com s'explica a **Gesti d'una configuraci**.

16.4.1 Primera revisi

```
$ mkdir -p tutorial3/auto
$ cp /usr/share/doc/live-build/examples/auto/* tutorial3/auto<
/
$ cd tutorial3
```

Editar auto/config de la manera segent:

```
#!/bin/sh

lb config noauto "
--distribution stable "
"$@"
```

Executar lb config per a crear l'arbre de configuraci, utilitzant el script auto/config que s'acaba de crear:

```
$ lb config
```

Ara, omplir la llista local de paquets:

```
$ echo "task-lxde-desktop spice-vdagent hexchat" && config/<
package-lists/my.list.chroot
```

First, `--distribution stable` ensures that `stable` is used instead of the default `--testing`. Second, we have added `spice-vdagent` for easier testing the image in `qemu`. And finally, we have added an initial favourite package: `hexchat`.

Ara, construir la imatge:

```
# lb build
```

Tenir en compte que a diferència dels dos primers tutorials, ja no s'ha d'escriure `2>&1 | tee build.log` ja que ara s'inclou al script `auto/build`.

Quan s'ha provat la imatge (com al [Tutorial 1](#)) i s'est satisfet del seu funcionament, s'el moment d'iniciar el repositori `git`, afegint noms els scripts `auto` que s'han creat, i llavors fer el primer lliurament:

```
$ git init
$ cp /usr/share/doc/live-build/examples/gitignore .gitignore
$ git add .gitignore auto config
$ git commit -m "Initial import."
```

16.4.2 Segona revisi

In this revision, we're going to clean up from the first build, replace the `smplayer` package with `vlc` package, rebuild, test and commit.

L'ordre `lb clean` netejar tots els fitxers generats en la construcció anterior a excepció del `cache`, el que estalvia haver de tornar a descarregar els paquets. Així assegura que el `lb build` següent tornar a executar totes les etapes per a regenerar els fitxers de la nostra nova configuració.

```
# lb clean
```

Now install the `vlc` package before the `lxde` package chooses between `smplayer`, `vlc` and `mplayer-gui` in our local package list in `config/package-lists/my.list.chroot`:

```
$ echo "vlc task-lxde-desktop spice-vdagent hexchat" >> \
  config/package-lists/my.list.chroot
```

Construir de nou:

```
# lb build
```

Provar, i quan s'estigui satisfet, fer el lliurament de la propera revisió:

```
$ git commit -a -m "Replacing smplayer with vlc."
```

Per descomptat, es possible fer canvis més complicats en la configuració, potser afegint fitxers en els subdirectoris de `config/`. Quan es fa un lliurament de les noves revisions, s'ha de tenir cura de no editar a més o incloure en el lliurament els fitxers de nivell superior de `config` que contenen variables `LB*`, ja que són productes de construcció també, i sempre són netejats per `lb clean` i tornats a crear per `lb config` a través dels seus respectius scripts `auto`.

Hem arribat al final de la nostra sèrie de tutorials. Molts més tipus de personalització són possibles, amb les poques característiques explorades en aquests senzills exemples, es poden crear una varietat gairebé infinita d'imatges diferents. Els exemples que queden d'aquesta secció tracten diferents casos d'extrets de les experiències recollides dels usuaris de sistemes en viu.

16.5 Un client per a un quiosc VNC

Cas d's: Crear una imatge amb live-build per a connectar-se directament a un servidor VNC al arrencar.

Fer un directori de treball i crear una configuraci d'esquelet en el seu interior, desactivant els recommends per a fer un sistema mnim. I a continuaci, crear dues llistes inicials de paquets: La primera generada per un script proporcionat per live-build anomenat Packages (veure [Generar llistes de paquets](#)), i la segona incloent-hi xorg, gdm3, metacity i xvnc4viewer.

```
$ mkdir vnc-kiosk-client
$ cd vnc-kiosk-client
$ lb config --apt-recommends false
$ echo '! Packages Priority standard' & config/package-lists/←
  standard.list.chroot
$ echo "xorg gdm3 metacity xtightvncviewer" & config/package-←
  lists/my.list.chroot
```

Com s'explica a [Afinar APT per a estalviar espai](#) pot ser que s'hagi de tornar a afegir alguns paquets recomanats per a fer que la imatge funcioni correctament.

Una manera fcil d'enumerar els recommends s utilitzar apt-cache. Per exemple:

```
$ apt-cache depends live-config live-boot
```

En aquest exemple, ens vam assabentar que havem de tornar a incloure diversos paquets recomanats per live-config i live-boot: user-setup perqu funcioni l'autologin i sudo com a programa essencial per a apagar el sistema. A ms, podria ser til afegir live-tools per a poder copiar la imatge en la memria RAM i eject per a expulsar, finalment, el medi en viu. Per tant:

```
$ echo "live-tools user-setup sudo eject" & config/package-←
  lists/recommends.list.chroot
```

Desprs, crear el directori /etc/skel a config/includes.chroot i posar un fitxer .xsession personalitzat per a l'usuari per defecte que posar en marxa metacity i iniciar xvncviewer, connectant al port 5901 d'un servidor ubicat a 192.168.1.2:

```
$ mkdir -p config/includes.chroot/etc/skel
$ cat & config/includes.chroot/etc/skel/.xsession ; EOF
#!/bin/sh

/usr/bin/metacity &
/usr/bin/xvncviewer 192.168.1.2:1

exit
EOF
```

Construir la imatge:

```
# lb build
```

Gaudir-ne.

16.6 A minimal image for a 512MB USB key

Use case: Create a default image with some components removed in order to fit on a 512MB USB key with a little space left over to use as you see fit.

When optimizing an image to fit a certain media size, you need to understand the tradeoffs you are making between size and functionality. In this example, we trim only so much as to make room for additional material within a 512MB media size, but without doing

anything to destroy the integrity of the packages contained within, such as the purging of locale data via the `localepurge` package, or other such intrusive optimizations. Of particular note, we use `--debootstrap-options` to create a minimal system from scratch and `--binary-image hdd` to create an image that can be copied to a USB key.

```
$ lb config --binary-image hdd --apt-indices false --apt-recommends false --debootstrap-options "--variant=minbase" --firmware-chroot false --memtest none
```

Per a fer que la imatge funcioni correctament, hem de tornar a afegir, com a `mnim`, dos paquets recomanats, que es queden fora per l'opció `--apt-recommends false`. Veure [Afinar APT per a estalviar espai](#)

```
$ echo "user-setup sudo" && config/package-lists/recommends.list.chroot
```

Additionally, you'll want to have network access, so another two recommended packages need to be re-added:

```
$ echo "ifupdown isc-dhcp-client" && config/package-lists/recommends.list.chroot
```

Ara, crear la imatge de la forma habitual:

```
# lb build 2i&1 --tee build.log
```

On the author's system at the time of writing this, the above configuration produced a 298MiB image. This compares favourably with

the 380MiB image produced by the default configuration in [Tutorial 1](#), when `--binary-image hdd` is added.

Deixar els ndexs d'APT amb `--apt-indices false` permet estalviar una bona quantitat d'espai, el desavantatge és que es necessita fer `apt-get update` abans d'utilitzar `apt` en el sistema en viu. Deixar els paquets recomanats amb `--apt-recommends false` estalvia una mica d'espai addicional, a costa d'ometre alguns paquets que d'una altra manera es podria esperar que hi fossin. `--debootstrap-options --variant=minbase` preinstalla un sistema `mnim` des del principi. Al no incloure automàticament paquets de firmware amb `--firmware-chroot false` també es guanya una mica d'espai. I finalment, `--memtest none` impedeix la instal·lació d'un comprovador de memòria.

Note: A minimal system can also be achieved using hooks, like for example the `stripped.hook.chroot` hook found in `/usr/share/doc/live-build/examples/hooks`. It may shave off additional small amounts of space and produce an image of 277MiB. However, it does so by removal of documentation and other files from packages installed on the system. This violates the integrity of those packages and that, as the comment header warns, may have unforeseen consequences. That is why using a minimal `debootstrap` is the recommended way of achieving this goal.

16.7 Un escriptori GNOME localitzat i amb installador

Cas d'ús: Crear una imatge amb l'escriptori GNOME, localitzat per Sussa i que inclogui un installador.

We want to make an iso-hybrid image using our preferred desktop, in this case GNOME, containing all of the same packages that would be installed by the standard Debian installer for GNOME.

El nostre primer problema és descobrir els noms de les tasques del llenguatge apropiades. En l'actualitat, `live-build` no ens pot ajudar

amb aix. Tot i que es podria tenir sort i trobar-ho per assaig i error, hi ha una eina, `grep-dctrl`, per extreure les descripcions de les tasques de `tasksel-data`. Per a preparar-ho tot, assegurar-se de tenir totes dues coses:

```
# apt-get install dctrl-tools tasksel-data
```

Ara podem buscar les tasques apropiades, primer amb:

```
$ grep -dctrl -FTest-lang de /usr/share/tasksel/descs/debian-↵
tasks.desc -sTask
Task: german
```

Mitjanant aquesta ordre, es descobreix que la tasca s'anomena, amb suficient claredat, `german`. Ara, per a trobar les tasques relacionades:

```
$ grep -dctrl -FEnhances german /usr/share/tasksel/descs/↵
debian-tasks.desc -sTask
Task: german-desktop
Task: german-kde-desktop
```

En el moment d'arrencar es generar la variant regional de `CH.UTF-8` i es selecciona la disposici del teclat `ch`. Ara posarem les peces juntes. Recordant de **s dels metapaquets** que els metapaquets tenen el prefix `task-`, simplement especificar aquests parmetres del llenguatge a l'arrencada, i després afegir els paquets de prioritat estndard i tots els metapaquets descoberts a la nostra llista de paquets de la segent manera:

```
$ mkdir live-gnome-ch
$ cd live-gnome-ch
```

```
$ lb config "
--bootappend-live "boot=live components locales=de`CH.↵
UTF-8 keyboard-layouts=ch" "
--debian-installer live
$ echo '! Packages Priority standard' & config/package-lists/↵
standard.list.chroot
$ echo task-gnome-desktop task-german task-german-desktop && ↵
config/package-lists/desktop.list.chroot
$ echo debian-installer-launcher && config/package-lists/↵
installer.list.chroot
```

Note that we have included the `debian-installer-launcher` package to launch the installer from the live desktop.

Apndix

Style guide

17. Guia d'estil

17.1 Instruccions per als autors

Aquesta secci s'ocupa d'algunes consideracions generals a tenir en compte al escriure documentaci tcnica per a live-manual. Es divideixen en aspectes lingstics i procediments recomanats.

Nota: Els autors han de llegir primer **Contribuir a aquest document**

17.1.1 Caracterstiques lingstiques

Utilitzar un angls planer

Tenir en compte que un alt percentatge dels lectors no sn parlants nadius d'angls. Aix que, com a regla general, intentar utilitzar frases curtes i significatives, seguides d'un punt i apart.

Aix no vol dir que s'hagi d'utilitzar un estil simplista i ingenu. s un suggeriment per intentar evitar, en la mesura del possible, les oracions subordinades complexes que fan que el text sigui difcil d'entendre per als parlants no nadius d'angls.

Varietat d'angls

Les varietats d'angls ms esteses sn el britnic i l'americ, aix que s molt probable que la majoria dels autors acabin utilitzant l'una o l'altra. En un entorn de collaboraci, la varietat ideal seria l'angls internacional, per s molt difcil, per no dir impossible, decidir quina varietat d'entre totes les existents, s la millor.

Esperem que les diferents varietats es puguin barrejar sense crear malentesos, per en termes generals s'ha d'intentar ser coherent i

abans de decidir sobre l's de l'angls britnic, l'angls americ o qualsevol altra varietat, fer una ullada a com escriuen altres persones i tractar d'imitar l'estil.

Ser equilibrat

S'ha de ser imparcial. Evitar incloure referncies a ideologies totalment alienes a live-manual. L'escriptura tcnica ha de ser el ms neutral possible. Est en la naturalesa mateixa de l'escriptura cientfica.

Ser polticament correcte

Evitar el llenguatge sexista tant com sigui possible. Si es necessita fer referncia a la tercera persona del singular utilitzar preferiblement they en lloc de he or she o invents estranys com per exemple s/he o s(he).

Ser concs

Anar directament al gra i no fent voltes. Donar tota la informaci necessria, per no afegir ms informaci de la necessria, s a dir, no explicar detalls innecessaris. Els lectors sn intelligents. Es presumeix algun coneixement previ per la seva part.

Minimitzar la feina de traducci

Tenir en compte que qualsevol cosa que s'escrigui haur de ser traduïda a diverses llenges. Aix implica que un nombre de persones hauran de fer un treball extra si s'agrega informaci innecessria o redundant.

Ser coherent

Com s'ha suggerit abans, s gaireb impossible estandarditzar un document escrit en collaboraci en un tot perfectament unificat. No obstant aix, s'aprecia tot esfor per escriure d'una manera coherent amb la resta dels autors.

Cohesi

918	Utilitzar connectors del discurs perquè el text sigui coherent i sense ambigüitats. (Normalment s'anomenen connectors).	Compte amb els falsos amics. No importa com de competent un s en un idioma estranger, de tant en tant es pot caure en el parany dels anomenats falsos amics, paraules que s'assemblen en dos idiomes per els significats o usos poden ser completament diferents.	928
919	Ser descriptiu		
920	s preferible descriure l'assumpte en un o diversos paràgrafs en lloc d'utilitzar una srie de oracions en un tpic estil de changelog. Cal descriure-ho! Els Lectors ho agrairan.	Intentar evitar els modismes tant com sigui possible. Els modismes sn expressions que tenen un significat completament diferent del que les seves paraules per separat semblen voler dir. De vegades, els modismes poden ser difícils d'entendre fins i tot per als parlants nadius d'angls!	929
921	Diccionari		
922	Buscar el significat de les paraules en un diccionari o una enciclopèdia si no es sap com expressar certs conceptes en anglès. Per cal tenir en compte que un diccionari pot ser el millor amic o pot convertir-se en el pitjor enemic si no es sap com utilitzar-lo correctament.	Evitar l'argot, les abreviatures, les contraccions...	930
923	L'anglès t el vocabulari més gran que existeix (amb més d'un mili de paraules). Moltes d'aquestes paraules sn préstecs d'altres llengües. Al buscar el significat de les paraules en un diccionari bilingüe la tendència d'un parlant no nadiu d'anglès s triar la que sona més semblant en la seva llengua materna. Sovint, així es converteix en un discurs excessivament formal que no sona ben natural en anglès.	Tot i que s'anima a utilitzar un anglès senzill i planer, l'escriptura tcnica pertany al registre formal de la llengua.	931
924	Com a regla general, si un concepte es pot expressar utilitzant diferents sinònims, s un bon consell triar la primera paraula proposada pel diccionari. En cas de dubte, s sovint correcte elegir les paraules d'origen germànic (Normalment paraules monosíl·labes). Tenir en compte que aquestes dues tècniques poden produir un discurs més aviat informal, per almenys la elecció de paraules ser d'ampli s i acceptació general.	Intentar evitar l'argot, les abreviatures inusuals que sn difícils d'entendre i per sobre de tot, les contraccions que tracten d'imitar el llenguatge parlat. Per no parlar d'expressions familiars o típiques de l'írc.	932
925	L's d'un diccionari de frases fetes es recomanable. Sn molt tils quan es tracta de saber quines paraules solen aparixer juntes.	17.1.2 Procediments	933
926	Com s'ha dit abans, s una bona pràctica aprendre del treball dels altres. L's d'un motor de recerca per comprovar com altres autors utilitzen certes expressions pot ajudar molt.	Provar abans d'escriure	934
927	Falsos amics, modismes i altres expressions idiomàtiques	s important que els autors provin els seus exemples abans d'afegir-los a live-manual per assegurar-se que tot funciona com es descriu. Fer les proves en un entorn chroot net o en una màquina virtual pot ser un bon punt de partida. A més, seria ideal si les proves fossin dutes a terme en diferents ordinadors amb un maquinari diferent per detectar els possibles problemes que puguin sorgir.	935
		Exemples	936
		Quan es posa un exemple mirar de ser el més específic possible. Un exemple s, després de tot, noms un exemple.	937
		Sovint s millor utilitzar una llnia que noms s'aplica a un cas concret	938

que l's d'abstraccions que poden confondre als lectors. En aquest cas es pot donar una breu explicaci dels efectes de l'exemple proposat.

Hi pot haver algunes excepcions quan l'exemple suggereixi l's d'algunes ordres potencialment perilloses que, si s'utilitzen incorrectament, poden provocar la prdua de dades o altres efectes indesitjables similars. En aquest cas, s'haur de proporcionar una explicaci detallada sobre els possibles efectes secundaris.

Enllaos externs

Els enllaos a llocs externs noms s'han d'utilitzar quan la informaci en aquests llocs s crucial per a comprendre un punt especial. Tot i aix, intentar utilitzar els enllaos a llocs externs el menys possible. Els enllaos d'internet poden canviar de tant en tant, donant lloc a enllaos trencats i deixant els arguments en un estat incomplet.

A ms, la gent que llegeix el manual sense connexi no podr seguir els enllaos.

Evitar les marques i les coses que violen la llicencia sota la qual es publica el manual

Intentar evitar les marques tant com sigui possible. Tenir en compte que altres projectes derivats poden fer s de la documentaci que escrivim. Aix que estem complicant les coses per a ells si s'afegix determinat material especfic.

live-manual es publica sota la llicencia GNU GPL. Aix t una srie d'implicacions que s'apliquen a la distribuci dels materials (de qualsevol tipus, incloent-hi grfics o logotips amb drets d'autor) que es publica amb ell.

Escriure un primer esborrany, revisar, editar, millorar, fer de nou si s necessari

- Pluja d'idees!. Es necessari organitzar les idees primer en una

seqncia lgica d'esdeveniments.

- Quan d'alguna manera ja s'han organitzat aquestes idees en la ment, escriure un primer esborrany.

- Revisar la gramtica, la sintaxi i l'ortografia. Tenir en compte que els noms propis de les versions, com ara trixie o sid , no s'han d'escriure en majúscula quan es refereixen als noms en clau. Per tal de comprovar l'ortografia es pot executar el target spell. s a dir, make spell

- Millorar les frases i refer qualsevol part si s necessari.

Captols

Utilitzar el sistema de numeraci convencional dels captols i subttols. Per exemple 1, 1.1, 1.1.1, 1.1.2 ... 1.2, 1.2.1, 1.2.2 ... 2, 2.1 ... i aix successivament. Veure marcat a continuaci.

Si s'ha d'enumerar una srie de passos o etapes en la descripci, tamb es poden utilitzar els nombres ordinals: primer, segon, tercer ... o en primer lloc, llavors, desprs, per fi ... Alternativament, es poden utilitzar punts.

Marcat

And last but not least, live-manual uses **SiSU** to process the text files and produce a multiple format output. It is recommended to take a look at **SiSU's manual** to get familiar with its markup, or else type:

```
$ sisu --help markup
```

Aquests sn alguns exemples de marcat que poden ser tils:

- Per a l'mfasi/negreta:

```
*-foo"* o !-foo"!
```

960 produeixen: foo o foo . S'usen per emfatitzar certes paraules clau.

961 - Per a la cursiva:

```
/-foo"/
```

963 produeix: foo. S'usa, per exemple, per als noms dels paquets Debian.

964 - Per a monospace:

```
#-foo"#
```

966 produeix: foo. S'usa per exemple, per als noms de les ordres. I també per ressaltar algunes paraules clau o coses com les rutes.

967 - Per a blocs de codi:

```
code-
    $ foo
    # bar
"code
```

969 produeix:

```
$ foo
# bar
```

971 S'utilitza code- per a obrir i "code per a tancar els blocs. És important recordar deixar un espai al principi de cada línia de codi.

17.2 Directrius per als traductors

Aquesta secció s'ocupa d'algunes consideracions generals a tenir en compte a l'hora de traduir el contingut de live-manual.

Com a recomanació general, els traductors haurien d'haver llegit i entès les regles de traducció que s'apliquen als seus llenguatges específics. En general, els grups de traductors i les llistes de correu proporcionen informació sobre com produir traduccions que compleixin amb els estàndards de qualitat de Debian.

Nota: Els traductors també han de llegir **Contribuir a aquest document**. En particular, la secció **Traducció**.

17.2.1 Consells de traducció

Comentaris

El paper del traductor és transmetre el més fidelment possible el significat de les paraules, oracions, paràgrafs i textos de com van ser escrits pels autors originals al seu idioma.

Per tant, aquests, s'han d'abstenir d'afegir comentaris personals o informacions addicionals pel seu compte. Si es vol afegir un comentari per als traductors que treballen en els mateixos documents, es poden deixar a l'espai reservat per a això. És a dir, la capçalera de les cadenes dels fitxers pot precedir-se pel signe # . La majoria dels programes gràfics de traducció poden manejar aquest tipus de comentaris automàticament.

NT, Nota del Traductor

És perfectament acceptable per, incloure una paraula o una expressió entre parèntesi en el text traduït si, i només si, això fa que el significat d'una paraula o expressió difícil sigui més clara per al lector. Dins dels parèntesis, el traductor ha de posar de manifest que l'addició s'ha utilitzant l'abreviatura NT o Nota del traductor.

982	Frases impersonals	992	No confondre's, aix no vol dir que el text tradut hagi de tenir la mateixa longitud que la versi en angl. Aix s gaireb impossible.	
983	Els documents escrits en angls fan un gran s de la forma impersonal you. En alguns altres idiomes que no comparteixen aquesta caracterstica, aix pot donar la falsa impressi que els textos originals s'adresen directament el lector quan en realitat n'ho fan. Els traductors han de ser conscients d'aquest fet i ho han de reflectir en el seu idioma amb la major precisi possible.		Cadenes intradubles	993
			Els traductors no han de traduir mai:	994
			- Els noms en clau de les versions (que han de ser escrits en minscules)	995
984	Falsos amics		- Els noms dels programes	996
985	El perill dels falsos amics explicat anteriorment s'aplica especialment als traductors. Tornar a comprovar el significat dels falsos amics sospitosos en cas de dubte.		- Les ordres que es posen com a exemples	997
			- Les metadades (apareixen sovint entre dos punts :metadata:)	998
986	Marcat		- Els enllaos	999
987	Els traductors que treballin inicialment amb els fitxers pot i posteriorment amb els fitxers po trobar moltes caracterstiques de marcat en les cadenes. Es pot traduir el text, sempre que sigui traduble, per s extremadament important que s'utilitzi exactament el mateix marcat que a la versi original en angls.		- Les rutes	1000
988	Blocs de codi			
989	Tot i que els blocs de codi sn generalment intradubles, incloure'ls en la traducci s l'nica manera de conseguir una traducci completa al 100%. I encara que aix signifiqui ms feina al principi, ja que pot requerir la intervenci dels traductors s'hi ha canvis en el codi, s la millor manera, a la llarga, per identificar el que s'ha tradut i el que no al comprovar la integritat dels fitxers .po.			
990	Salts de lnia			
991	Els texts traduts han de tenir exactament els mateixos salts de lnia que els texts originals. Aneu amb compte de pressionar la tecla Enter o escriure si apareix als fitxers originals. Aquests salts de lnies apareixen sovint, per exemple, en els blocs de codi.			

SiSU Metadata, document information

Title: Debian Live Manual

Author: Debian Live Project <debian-live@lists.debian.org>

Rights: Copyright: Copyright (C) 2006-2015 Live Systems Project, Copyright (C) 2016-2025 The Debian Live team

License: This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

The complete text of the GNU General Public License can be found in /usr/share/common-licenses/GPL-3 file.

Publisher: Debian Live Project <debian-live@lists.debian.org>

Date: 2025-02-26

Version Information

Sourcefile: live-manual.ssm.sst

Filetype: SiSU text 2.0, Unicode text, UTF-8 text, with very long lines (745)

Source Digest: SHA2-256(live-manual.ssm.sst)=729d94465549cf4aa742d865-deae03a44d4fd226521a8155bd948981783a7f22

Generated

Document (ao) last generated: 2025-02-26 23:55:25 +0000

Generated by: SiSU 7.3.0 of 2023w44/1 (2023-10-30)

Ruby version: ruby 3.3.7 (2025-01-15 revision be31f993d7) [x86_64-linux-gnu]