

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	Usuario	11
Debian Live Manual	i	Instalacin 12
Acerca de este manual	3	3. Instalacin 12
Acerca de este manual	4	3.1 Requisitos 12
1. Acerca de este manual	4	3.2 Instalacin de live-build 12
1.1 Para el impaciente.	4	3.2.1 Desde el repositorio Debian. 12
1.2 Trminos	4	3.2.2 A partir del cdigo fuente 12
1.3 Autores	5	3.3 Instalacin de live-boot y live-config 13
1.4 Cmo contribuir a este documento	6	3.3.1 Desde el repositorio Debian. 13
1.4.1 Aplicar cambios	6	3.3.2 A partir del cdigo fuente 13
1.4.2 Traduccin	7	Conceptos bsicos 15
Contribuir al Debian Live Project	9	4. Conceptos bsicos 15
2. Acerca del Debian Live Project	9	4.1 Qu es un sistema en vivo? 15
2.1 Motivacin	9	4.2 Descarga de imgenes prefabricadas 16
2.1.1 Desventajas de los sistemas en vivo actuales	9	4.3 Primeros pasos: creacin de una imagen ISO hbrida 16
2.1.2 El porqu de crear un sistema en vivo propio.	9	4.4 Usar una imagen ISO hbrida 16
2.2 Filosofa	9	4.4.1 Grabar una imagen ISO en un medio fsico. 17
2.2.1 Only unchanged packages from Debian main and non-free-firmware	9	4.4.2 Copiar una imagen ISO hbrida a un dispositivo USB 17
2.2.2 Sin configuracin especial para el sistema en vivo	10	4.4.3 Usar el espacio libre en el dispositivo USB 17
2.3 Contacto	10	4.4.4 Arrancar el medio en vivo 17
		4.5 Usar una mquina virtual para pruebas 18
		4.5.1 Probar una imagen ISO con QEMU 18
		4.5.2 Probar una imagen ISO con VirtualBox 19
		4.6 Construir y utilizar una imagen HDD 19
		4.7 Creacin de una imagen de arranque en red 20
		4.7.1 Servidor DHCP 21

4.7.2 Servidor TFTP	21	Personalizacin de contenidos	28
4.7.3 Servidor NFS	21	7. Descripcin general de la personalizacin	28
4.7.4 Cmo probar el arranque en red	22	7.1 Configuracin en el momento de la creacin vs en el momento del arranque	28
4.7.5 Qemu	22	7.2 Etapas de la creacin	28
4.8 Arrancar desde internet	22	7.3 Opciones para lb config en ficheros	29
4.8.1 Conseguir los ficheros para arrancar desde internet	22	7.4 Tareas de personalizacin	29
4.8.2 Arrancar imgenes webboot	23		
		Personalizacin de la instalacin de paquetes	30
Descripcin general de las herramientas	24	8. Personalizacin de la instalacin de paquetes	30
5. Descripcin general de las herramientas	24	8.1 Origen de los paquetes	30
5.1 El paquete live-build	24	8.1.1 Distribucin, reas de archivo y modo	30
5.1.1 El comando lb config	24	8.1.2 Rplcas de Distribucin Debian	31
5.1.2 El comando lb build	25	8.1.3 Rplcas de Distribution utilizadas durante la creacin	31
5.1.3 El comando lb clean	25	8.1.4 Rplcas de distribucin Debian utilizadas en la ejecucin.	31
5.2 El paquete live-boot	25	8.1.5 Repositorios adicionales	32
5.3 El paquete live-config	25	8.2 Seleccin de los paquetes a instalar	32
		8.2.1 Listas de paquetes	32
Gestionar una configuracin	26	8.2.2 Utilizar metapaquetes	32
6. Gestionar una configuracin	26	8.2.3 Listas de paquetes locales	33
6.1 Gestionar cambios en la configuracin	26	8.2.4 Listas de paquetes locales para la etapa binary	33
6.1.1 Por qu utilizar scripts auto? Qu hacen?	26	8.2.5 Generar listas de paquetes	34
6.1.2 Usar scripts auto de ejemplo	26	8.2.6 Utilizacin de condiciones dentro de las listas de paquetes	34
6.2 Clonar una configuracin publicada a travs de Git	27	8.2.7 Eliminacin paquetes durante la instalacin	34
		8.2.8 Summary	35
		8.2.9 Tareas de Escritorio e Idioma	35
		8.2.10 Versin y tipo de kernel	35

8.2.11 Kernels personalizados	36	10.2 Personalizacin de las variantes locales e idioma	44
8.3 Instalar paquetes modificados o de terceros	36	10.3 Persistencia	46
8.3.1 Mtodo packages.chroot para instalar paquetes per- sonalizados	37	10.3.1 El fichero persistence.conf	47
8.3.2 Mtodo de repositorio APT para instalar paquetes personalizados	37	10.3.2 Utilizar varios medios persistentes	47
8.3.3 Paquetes personalizados y APT	37	10.3.3 Utilizar persistencia con cifrado	48
8.4 Configurar APT en la creacin	38	Personalizacin de la imagen binaria	51
8.4.1 Utilizar apt o aptitude	38	11. Personalizacin de la imagen binaria	51
8.4.2 Utilizacin de un proxy con APT	38	11.1 Gestores de arranque	51
8.4.3 Ajuste de APT para ahorrar espacio	38	11.2 Metadatos ISO	51
8.4.4 Pasar opciones a apt o a aptitude	39	Personalizacin del Instalador de Debian	52
8.4.5 APT pinning	39	12. Personalizacin del Instalador de Debian	52
Personalizacin de contenidos	41	12.1 Tipos de imgenes segn el instalador	52
9. Personalizacin de contenidos	41	12.2 Personalizando el Instalador de Debian mediante precon- figuracin	53
9.1 Includes	41	12.3 Personalizar el contenido del Instalador de Debian	53
9.1.1 Includes locales en Live/chroot	41	Proyecto	54
9.1.2 Includes locales en Binary	42	Contribuir al proyecto	55
9.2 Scripts gancho (Hooks)	42	13. Contribuir al proyecto	55
9.2.1 Scripts gancho locales en el chroot	42	13.1 Traduccin de las pginas de manual	55
9.2.2 Scripts gancho locales en Binary	42		
9.2.3 Scripts gancho en tiempo de arranque	43		
9.3 Preconfiguracin de las preguntas de Debconf	43		
Personalizacin del comportamiento en tiempo de ejecucin.	44		
10. Personalizacin del comportamiento en tiempo de ejecucin.	44		
10.1 Personalizacin del usuario por defecto del sistema en vivo	44		

Cmo informar acerca de errores.	56	Ejemplos	64
14. Informes de errores.	56	16. Ejemplos	64
14.1 Problemas conocidos	56	16.1 Uso de los ejemplos	64
14.2 Hacer la investigacin	56	16.2 Tutorial 1: Una imagen predeterminada	64
14.3 Reconstruir desde cero	56	16.3 Tutorial 2: Una utilidad de navegador web	65
14.4 Utilizar paquetes actualizados	57	16.4 Tutorial 3: Una imagen personalizada	65
14.5 Recopilar informacin	57	16.4.1 Primera revisin	65
14.6 Aislar el fallo si es posible	57	16.4.2 Segunda revisin	66
14.7 Utilizar el paquete correcto sobre el que informar del error	58	16.5 Un cliente VNC kiosk	67
14.7.1 En la preinstalacin (bootstrap) en tiempo de	58	16.6 A minimal image for a 512MB USB key	68
creacin.	58	16.7 Un escritorio GNOME con variante local e instalador . .	69
14.7.2 Mientras se instalan paquetes en tiempo de creacin.	58		
14.7.3 En tiempo de arranque	58	Apndice	70
14.7.4 En tiempo de ejecucin	58		
14.8 Dnde informar de los fallos	58	Style guide	71
Estilo de cdigo	60	17. Gua de estilo	71
15. Estilo de cdigo	60	17.1 Instrucciones para los autores	71
15.1 Compatibilidad	60	17.1.1 Aspectos lingsticos	71
15.2 Sangrado	60	17.1.2 Procedimientos	72
15.3 Ajuste de lneas	60	17.2 Directrices para los traductores	74
15.4 Variables	61	17.2.1 Consejos de traduccin	74
15.5 Miscelnea	62		
		SiSU Metadata, document information	76
Ejemplos	63		

₁ Debian Live Manual

₂ Acerca de este manual

Acerca de este manual

1. Acerca de este 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/>

live-manual est principalmente enfocado a ayudar en la creacin de un sistema en vivo y no est dirigido al usuario final de estos sistemas. Un usuario final puede encontrar informacin til en las siguientes secciones: **Conceptos bsicos** que cubre la descarga de imgenes prefabricadas y la preparacin de imgenes para arrancar un sistema desde un medio de almacenamiento o desde una red local, ya sea utilizando el constructor web o ejecutando live-build directamente en el sistema. **Personalizacin del comportamiento en tiempo de ejecucin** que describe algunas de las opciones que pueden especificarse en el indicador de arranque, como pueden ser la seleccin de la distribucin del teclado, las variantes locales o la persistencia.

Algunos de los comandos mencionados en el texto deben ser ejecutados con privilegios de superusuario, que pueden ser obtenidos accediendo a la cuenta de root mediante la orden `su` o mediante la orden `sudo`. Para distinguir las ordenes que deben ser ejecutadas como usuario no privilegiado de las que si requieren privilegios de superusuario se ha prefijado con `$` las primeras y con `#` las segundas. Estos smbolos no son parte de la orden.

1.1 Para el impaciente.

Aunque se cree que todo lo descrito en este manual es importante para la mayora de los usuarios, es cierto que hay mucho material a interiorizar y que los usuarios desean experimentar con las herramientas de forma rpida y satisfactoria antes de entrar en detalles. Por lo tanto, se sugiere leer siguiendo el siguiente orden.

Primero, leer el captulo **Acerca de este manual**, desde el principio y terminando en la seccin **Trminos**. Despus, saltar hasta los tres tutoriales que estn al principio de la seccin **Ejemplos** pensados para aprender a configurar y construir imgenes de forma bsica. Se deber leer primero **Uso de los ejemplos**, seguido de **Tutorial 1: Una imagen predeterminada** y **Tutorial 2: Una utilidad de navegador web**, para finalizar con **Tutorial 3: Una imagen personalizada**. Al final de estos tutoriales, el lector tendr una visin de lo que se puede hacer con los sistemas en vivo.

Se anima a profundizar en el estudio del manual con la lectura detenida del siguiente captulo: **Conceptos bsicos**, y de una manera ms somera el captulo **La creacin de una imagen netboot**, para acabar con la lectura de **Descripcin general de la personalizacin** y los captulos que le siguen. Se espera que, en este punto, el lector est lo suficientemente motivado con lo que se puede hacer con los sistemas en vivo para leer el resto del manual, de principio a fin.

1.2 Trminos

Sistema en vivo : Se entiende por sistema en vivo aquel sistema operativo que se puede arrancar sin instalacin previa en el disco duro. Un sistema en vivo no altera ningn sistema operativo previamente instalado ni ningn fichero existente en el disco duro de la mquina a menos que se le instruya para hacerlo. Los sistemas en vivo son arrancados tpicamente desde medios extrables como CDs, DVDs o llaves USB. Al-

gunos pueden tambien arrancar desde la red local (utilizando imagenes tipo netboot, ver [Creacin de una imagen de arranque en red](#)), o incluso desde internet utilizando el parmetro de arranque fetch=URL, ver [Arrancar desde internet](#)).

Medio en vivo : A diferencia de sistema en vivo, el medio en vivo se refiere al CD, DVD o memoria USB donde se copia el fichero binario producido por live-build y utilizado para arrancar el sistema en vivo. Ms ampliamente, el trmino tambien se refiere a cualquier lugar en el que reside el fichero binario a los efectos de iniciar el sistema en vivo, tales como la ubicacin de los ficheros de arranque de red.

Debian Live Project : Es un proyecto que mantiene, entre otros, los paquetes live-boot, live-build, live-config, live-tools y live-manual.

Sistema huped : Es el conjunto de herramientas y equipo utilizado para crear el sistema en vivo.

Sistema objetivo : Es el conjunto de herramientas y equipo donde se ejecutar el sistema en vivo.

live-boot : Es una coleccin de scripts que sern responsables de arrancar el sistema en vivo.

live-build : Una coleccin de scripts utilizados para construir sistemas en vivo personalizados.

live-config : Es una coleccin de scripts utilizados para configurar un sistema en vivo durante el proceso de arranque.

live-tools : Una coleccin de scripts adicionales que se utilizan para realizar tareas tiles en un sistema en vivo en ejecucin.

live-manual : Este documento forma parte de un paquete llamado live-manual.

Instalador de Debian (Debian Installer o d-i) : Es el mecanismo oficial de instalacin para la distribucin Debian.

Parmetros de arranque : Parmetros que son entregados al gestor de arranque (bootloader) para modificar el comportamiento del kernel o del conjunto de scripts live-config. Son llamados tambien Parmetros de kernel u Opciones de arranque.

chroot : El programa chroot, chroot(8), permite ejecutar diferentes instancias de un entorno GNU/Linux en un solo sistema de manera simultnea sin necesidad de reiniciar el sistema.

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

Distribucin objetivo : Es la versin de la distribucin Debian en la cual estar basado el sistema en vivo que puede diferir de la versin de la distribucin en el sistema huped.

stable/testing/unstable : La distribucin stable , actualmente llamada bookworm , contiene la ltima distribucin Debian publicada oficialmente. La distribucin testing , temporalmente llamada trixie , est en la rampa de salida para ser la prxima distribucin stable . La principal ventaja de utilizar esta distribucin es que tiene versiones de programas ms recientes si se compara con la versin stable . La distribucin unstable , permanentemente llamada sid , es dnde se realiza el desarrollo de Debian. Generalmente esta distribucin es usada por los desarrolladores y aquellos que les gusta vivir al filo de lo imposible. A lo largo del manual, se usan sus nombres en clave, como por ejemplo trixie o sid , ya que es lo que las mismas herramientas reconocen.

1.3 Autores 29

Lista de autores (en orden alfabtico): 30

Ben Armstrong 31

Brendan Sleight 32

Carlos Zuferri

Chris Lamb

Daniel Baumann

Franklin Piat

Jonas Stein

Kai Hendry

Marco Amadori

Mathieu Geli

Matthias Kirschner

Richard Nelson

Roland Clobus

Trent W. Buck

1.4 Cmo contribuir a este documento

Este manual se ha creado como un proyecto comunitario y cualquier propuesta para su mejora u otras contribuciones son siempre bienvenidas. Ver la seccin **Contribuir al proyecto** para obtener informacin detallada sobre cmo obtener la clave pblica y hacer buenos commits.

1.4.1 Aplicar cambios

Para realizar cambios en el manual en Ingls se debe editar los ficheros adecuados en manual/en/ pero antes de enviar una contribucin se debera realizar una visualizacin del trabajo realizado. Para ello asegurarse de tener instalados los paquetes necesarios para la construccin de live-manual ejecutando la siguiente orden:

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

Se puede realizar la construccin del manual posicionndose en el directorio de nivel superior, o sea, en el directorio clonado mediante Git y ejecutando la siguiente orden:

```
$ make build
```

Ya que construir el manual completo en todos los idiomas disponibles cuesta bastante rato, los autores seguramente estarn interesados en utilizar alguno de los siguientes atajos a la hora de revisar la documentacin que hayan aadido al manual en ingls. Utilizando PROOF=1 se crea live-manual en formato html, pero sin los documentos html segmentados, y utilizando PROOF=2 se crea live-manual en formato pdf pero slo en retrato A4 y carta. Por este motivo, utilizar cualquiera de las opciones PROOF= puede llegar a ahorrar una cantidad de tiempo considerable, por ejemplo.

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

Cuando se revisa alguna de las traducciones, es posible construir slo un idioma ejecutando, por ejemplo:

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

Es posible generar el documento por formato:

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

O combinar ambos, por ejemplo:

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

Despus de revisar el trabajo y asegurarse de que todo est bien, no ejecutar make commit a menos de que se actualicen las traducciones al mismo tiempo, y en ese caso, no mezclar los cambios al manual en ingls con las traducciones en el mismo commit, sino en commits separados. Ver la seccin [Traduccin](#) para ms detalles.

1.4.2 Traduccin

Nota: Para la traduccin de las pginas de manual, ver [Traduccin de las pginas de manual](#)

Para traducir live-manual, seguir estos pasos, dependiendo de si se est comenzando una traduccin desde cero o si se continua trabajando en una traduccin ya comenzada:

Empezar una nueva traduccin desde cero

Traducir los ficheros about`manual.ssi.pot , about`project.ssi.pot y index.html.in.pot de manual/pot/ al idioma deseado con cualquier editor (como puede ser poedit) . Enviar los ficheros .po traducidos a la lista de correo para revisar su integridad. La comprobacin de integridad de live-manual no slo se asegura de que los ficheros .po estn 100% traducidos sino que tambin detecta posibles errores.

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 con una traduccin ya comenzada

Si el nuevo idioma ya ha sido aadido, se puede continuar la traduccin de los ficheros .po restantes en manual/po/`\$-LANGUAGE`/ de manera aleatoria utilizando el editor preferido (como por ejemplo poedit) .

No se debe olvidar la ejecucin del comando make commit para actualizar los manuales traducidos a partir de los ficheros .po. Entonces se puede revisar los cambios ejecutando make build antes de git add ., git commit -m Translating... y git push. Recordar que como make build puede tardar una cantidad considerable de tiempo, se pueden revisar las diferentes lenguas de forma individual como se explica en [Aplicar cambios](#)

Despus de ejecutar make commit se podr observar bastante texto en la pantalla. Bsicamente son mensajes informativos sobre el estado del proceso y tambin algunas pistas sobre lo que se puede hacer para mejorar live-manual. A menos que se vea un error fatal, generalmente se puede proceder y enviar la contribucin.

live-manual incluye dos utilidades que pueden ser de gran ayuda para los traductores a la hora de encontrar mensajes sin traducir y mensajes difusos. La primera es make translate. Esta activa un script que muestra en detalle cuantos mensajes sin traducir hay en cada fichero .po. La segunda, make fixfuzzy, slo acta sobre los mensajes difusos pero ayuda a encontrarlos y corregirlos uno por uno.

Hay que tener en cuenta que aunque estas utilidades pueden ser de gran ayuda para traducir en la linea de comandos, se recomienda el uso de una herramienta especializada como por ejemplo poedit. Adems, es una buena idea leer la documentacin de debian sobre localizacion (l10n) y, especficamente para live-manual, las [Directrices para los traductores](#).

Nota: Se puede utilizar make clean para limpiar el rbol git antes de

enviar los cambios. Este paso no es obligatorio, gracias al fichero `.gitignore`, pero es una buena práctica para evitar enviar ficheros involuntariamente.

74	Contribuir al Debian Live Project ⁸⁷	
75	2. Acerca del Debian Live Project	
76	2.1 Motivacin	
77	2.1.1 Desventajas de los sistemas en vivo actuales	
78	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:	
79	No eran proyectos de Debian y por lo tanto no contaban con soporte desde dentro de Debian.	
80	Mezclaban paquetes de diferentes versiones, por ejemplo testing y unstable .	
81	Solamente soportaban la arquitectura i386.	
82	Modificaban el comportamiento y/o la apariencia de los paquetes, eliminando contenido para reducirlos de tamao.	
83	Incluan paquetes de fuera del archivo de Debian.	
84	Utilizaban kernels personalizados con parches que no eran parte de Debian.	
85	Eran demasiado lentos, debido a su gran tamao, para ser utilizados como sistemas de rescate.	
86	No disponan de diferentes medios de instalacin, como CDs, DVDs, llaves USB o imgenes de arranque por red netboot.	
	2.1.2 El porqu de crear un sistema en vivo propio.	
	Debian es el Sistema Operativo Universal: Debian tiene un sistema en vivo para mostrar y representar el autntico y verdadero Debian con las siguientes ventajas fundamentales:	88
	Es un subproyecto de Debian.	89
	Refleja el estado (actual) de una versin Debian.	90
	Se ejecuta en tantas arquitecturas como es posible.	91
	Consiste solamente de paquetes Debian sin modificar.	92
	No contiene ningn paquete que no forma parte del archivo de Debian.	93
	Utiliza kernels que provienen de Debian inalterados sin parches adicionales.	94
	2.2 Filosofia	95
	2.2.1 Only unchanged packages from Debian main and non-free-firmware	96
	Solamente se utilizarn paquetes del repositorio de Debian de la seccin main. La seccin non-free no es parte de Debian y por lo tanto no puede ser utilizada de ninguna de las maneras para generar imgenes de sistema oficiales.	97
	Starting with Debian 12 bookworm we added the non-free-firmware section for better support of modern hardware.	98
	No se modificar ningn paquete. Siempre que se necesite modificar algo, se har en coordinacin con el correspondiente mantenedor del paquete en Debian.	99
	Como excepcin, los paquetes del proyecto como son live-boot, live-build o live-config, pueden ser utilizados temporalmente desde el repositorio	100

del proyecto, por razones de desarrollo (por ejemplo para crear instantaneas de pruebas). Estos paquetes sern actualizados en Debian de manera regular.

presente en el canal #debian-live de irc.debian.org (OFTC). Por favor, se debe ser paciente cuando se espera una respuesta en el IRC. Si la respuesta no llega, se puede enviar la pregunta a la lista de correos.

BTS : El [Informes de errores](#).

108

101 2.2.2 Sin configuracin especial para el sistema en vivo

102 En esta fase, no se crear o instalarn configuraciones alternativas o de ejemplo. Se utilizarn todos los paquetes con su configuracin por defecto, tal y como quedan despues de una instalacin normal de Debian.

103 Siempre que se necesite una configuracin diferente a la de por defecto, se har en coordinacin con el mantenedor del paquete Debian correspondiente.

104 Se puede emplear un sistema para configurar paquetes que utiliza debconf, permitiendo la personalizacin de la configuracin de los paquetes que van a ser instalados en la imagen en vivo que se genere, pero las [imgenes en vivo prefabricadas](#) solamente utilizarn la configuracin por defecto, a menos que sea absolutamente necesario hacer cambios para que funcionen en los sistemas en vivo. Siempre que sea posible, preferimos adaptar los paquetes en el archivo de Debian para que funcionen mejor en un sistema en vivo en lugar de realizar cambios en nuestra cadena de herramientas o en [las configuraciones de las imgenes prefabricadas](#). Para ms informacin, ver [Descripcin general de la personalizacin](#).

105 2.3 Contacto

106 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/>.

107 IRC : Un nmero importante de usuarios y desarrolladores suele estar

110	Instalacin	126	Usando instantneas	
			Si se usa Debian, el mtodo recomendado es instalar live-build a travs del repositorio de Debian.	127
111	3. Instalacin			
112	3.1 Requisitos		3.2.1 Desde el repositorio Debian.	128
113	Building live system images has very few system requirements for the host system:		Simplemente instalar live-build como cualquier otro paquete:	129
114	Acceso de superusuario (root)			130
115	Una versin actualizada de live-build			
116	Un intrprete de comandos compatible con POSIX, como por ejemplo bash o dash		3.2.2 A partir del cdigo fuente	131
117	debootstrap		live-build se desarrolla utilizando el sistema de control de versiones Git. En los sistemas basados en Debian se encuentra el paquete git. Para ver el ltimo cdigo, ejecutar:	132
118	Linux 2.6 or newer			
119	A mount point with dev and exec rights.			133
120				
	<pre># mount -i your`mount`point -o dev,exec,remount</pre>		<pre>\$ git clone https://salsa.debian.org/live-team/live-build.git</pre>	
121	Tener en cuenta que no es necesario el uso de Debian o una distribucin derivada de Debian - live-build funcionar en casi cualquier distribucin que cumpla con los requisitos anteriores.		Se puede crear e instalar el paquete Debian ejecutando:	134
				135
122	3.2 Instalacin de live-build			
123	Se puede instalar live-build de varias maneras diferentes:			
124	Desde el repositorio Debian		Si se desea, se podr instalar cualquiera de los paquetes .deb recién creados con el procedimiento anterior, p.ej.	136
125	A partir del cdigo fuente			137
			<pre># dpkg -i live-build`4.0-1`all.deb</pre>	

Tambin se puede instalar live-build directamente en el sistema ejecutando:

```
# make install
```

y desinstalarlo con:

```
# make uninstall
```

3.3 Instalacin de live-boot y live-config

Nota: No es necesario instalar live-boot o live-config en el sistema para crear sistemas personalizados en vivo. Sin embargo, eso no causar ningn dao y es til por motivos de referencia. Si nicamente se desea tener la documentacin, es posible instalar los paquetes live-boot-doc y live-config-doc de forma independiente.

3.3.1 Desde el repositorio Debian.

Tanto live-boot como live-config estn disponibles en el repositorio Debian siguiendo un procedimiento similar al explicado en la [Instalacin de live-build](#).

3.3.2 A partir del cdigo fuente

Para utilizar el ltimo cdigo fuente a partir de git, se puede seguir el proceso siguiente. Asegurarse de estar familiarizado con los trminos mencionados en [Trminos](#).

Comprobar el cdigo fuente de live-boot y live-config

138

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

Si se desea generar estos paquetes a partir del cdigo fuente, se puede consultar las pginas del manual para ms detalles sobre la personalizacin de live-boot y live-config.

Creacin de los paquetes .deb de live-boot y live-config

Se debe crear ya sea en la distribucin de destino o en un entorno chroot que contenga la plataforma de destino: es decir, si el objetivo es trixie entonces se debe crear usando trixie .

Utilizar un programa creador personal como pbuilder o sbuilder si se necesita crear live-boot para una distribucin de destino diferente del sistema de creacin. Por ejemplo, para las imgenes en vivo de trixie , crear live-boot en un entorno chroot trixie . Si la distribucin de destino coincide con la distribucin actual, se puede crear directamente sobre el sistema de creacin con dpkg-buildpackage (proporcionada por el paquete dpkg-dev):

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

Utilizar los ficheros .deb generados que proceda

Como live-boot y live-config son instalados por el sistema de construccin live-build, la instalacin de esos paquetes en el sistema anfitrión no es suficiente: se debe tratar los .deb generados como si fueran paquetes personalizados. Puesto que el propsito de la construccin de estos paquetes a partir del cdigo fuente es probar cosas nuevas a corto plazo antes de su lanzamiento oficial, seguir las instrucciones de [Instalar paquetes modificados](#)

o de terceros para incluir temporalmente los ficheros necesarios en la configuracin. En particular, observar que ambos paquetes se dividen en una parte genrica, una parte de documentacin y uno o ms back-ends. Incluir la parte genrica, slo uno de los back-ends que coincida con la configuracin y opcionalmente, la documentacin. Suponiendo que se est construyendo una imagen en vivo en el directorio actual y se han generado todos los .deb para una nica versin de los dos paquetes en el directorio superior, estos comandos bash copiaran todos los paquetes necesarios, incluyendo sus back-ends por defecto:

157

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

Conceptos bsicos

4. Conceptos bsicos

Este captulo contiene una breve descripcin del proceso de creacin de las imgenes en vivo y las instrucciones para el uso de los tres tipos de imgenes ms utilizadas. El tipo de imagen ms verstil, iso-hybrid, se puede utilizar en una mquina virtual, en medios pticos u otros dispositivos de almacenamiento USB. En ciertos casos especiales, como se explica ms adelante, las imgenes hdd, pueden ser las ms adecuadas. El captulo incluye instrucciones detalladas para crear y utilizar una imagen de tipo netboot, que es un poco ms complicado debido a la configuracin necesaria en el servidor. Es un tema ligeramente avanzado para cualquier persona que no est familiarizada con el arranque en red, pero se incluye aqu porque una vez que se realiza toda la configuracin, es una forma muy conveniente para probar y desplegar imgenes de arranque en red local sin la molestia de tratar con los dispositivos de almacenamiento de la imagen.

La seccin termina con una rpida introduccin al **arranque desde internet**, que es, quizs, la manera ms rpida de utilizar diferentes imgenes para diferentes propsitos, cambiando de una a otra segn las necesidades, utilizando internet como medio.

A lo largo de todo el captulo se hace a menudo referencia al nombre de las imgenes producidas por defecto por live-build. Si se **descarga una imagen ya creada** el nombre puede variar.

4.1 Qu es un sistema en vivo?

Por lo general, un sistema en vivo se refiere a un sistema operativo que arranca en un equipo desde un medio extrable, como un CD-ROM, dis-

positivo USB, o desde una red, listo para usar sin ningn tipo de instalacin en la unidad de costumbre, con configuracin automtica en tiempo de ejecucin (Ver **Trminos**).

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:

Imagen del kernel de Linux , normalmente llamada vmlinuz*

Imagen del Disco RAM inicial (initrd) : Un Disco RAM configurado para el arranque de Linux, que incluya los mdulos posiblemente necesarios para montar la imagen del sistema y algunos scripts para ponerlo en marcha.

Imagen del sistema : La imagen del sistema de ficheros raz. Por lo general, se utiliza un sistema de ficheros comprimido SquashFS para reducir al mnimo el tamao de la imagen en vivo. Hay que tener en cuenta que es de slo lectura. Por lo tanto, durante el arranque del sistema en vivo se utiliza un disco RAM y un mecanismo de unin que permite escribir ficheros en el sistema en funcionamiento. Sin embargo, todas las modificaciones se pierden al apagar el equipo a menos que se use de modo opcional la persistencia (ver **Persistencia**).

Gestor de arranque : Una pequea pieza de cdigo diseada para arrancar desde el medio de almacenamiento escogido, posiblemente mostrando un men o un indicador de arranque para permitir la seleccin de opciones/configuracin. Carga el kernel de Linux y su initrd para funcionar con un sistema de ficheros asociado. Se pueden usar soluciones diferentes, dependiendo del medio de almacenamiento de destino y el formato del sistema de ficheros que contenga los componentes mencionados anteriormente: isolinux para arrancar desde un CD o DVD en formato ISO9660, syslinux para arrancar desde el disco duro o unidad USB desde una particin VFAT, extlinux para formatos ext2/3/4 y par-

ticiones btrfs, pxelinux para arranque de red PXE, GRUB para particiones ext2/3/4 , etc.

utilizando los valores por defecto, as que lo vamos a llamar, por ejemplo, live-default.

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
```

Entonces, ejecutar el comando lb config. Esto crear una jerarquia config/ en el directorio actual que ser usada por otros comandos:

4.2 Descarga de imagenes prefabricadas

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.

```
$ lb config
```

Al no pasar ningn parmetro a estos comandos, se utilizarn todas las opciones por defecto. Ver **El comando lb config** para ms detalles.

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**.

Ahora que existe un jerarquia config/, se puede crear la imagen con el comando lb build:

```
# lb build
```

4.3 Primeros pasos: creacin de una imagen ISO hbrida

Independientemente del tipo de imagen, cada vez se tendr que realizar los mismos pasos bsicos para construir una imagen. Como primer ejemplo, crear un directorio de trabajo, cambiar a ese directorio y ejecutar la siguiente secuencia de comandos live-build para crear una imagen ISO hbrida bsica que contiene slo el sistema por defecto de Debian sin X.org. Es adecuada para grabarla en un CD o DVD y tambin para copiarla en un dispositivo USB.

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 se est construyendo en un sistema amd64 el nombre de la imagen resultante ser live-image-amd64.hybrid.iso. Tener en cuenta esta convencion a lo largo del manual.

El nombre del directorio de trabajo es indiferente, pero si se da un vistazo a los ejemplos utilizados en live-manual, es una buena idea utilizar un nombre que ayude a identificar la imagen con la que est trabajando en cada directorio, especialmente si se est trabajando o experimentando con distintos tipos de imagenes. En este caso, vamos a construir un sistema

4.4 Usar una imagen ISO hbrida

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 Grabar una imagen ISO en un medio físico.

Grabar una imagen ISO es fácil. Simplemente instalar xorriso y usarlo desde el intérprete de comandos para grabar la imagen. Por ejemplo:

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

4.4.2 Copiar una imagen ISO híbrida a un dispositivo USB

Las imágenes ISO preparadas con xorriso, pueden sencillamente copiarse a una llave USB con la orden `cp` o con un programa equivalente. Insertar una llave USB con un tamaño suficiente para la imagen y determinar qué dispositivo es, al cual nos referiremos de ahora en adelante como `$-USBSTICK`. Este nombre de dispositivo se refiere a la llave entera como por ejemplo `/dev/sdb` y no a una partición como `/dev/sdb1`. Se puede encontrar el nombre del dispositivo correcto mirando la salida de `dmesg` después de conectar la llave, o mejor aún, ejecutando `ls -l /dev/disk/by-id`.

Cuando se está seguro de tener el nombre del dispositivo correcto, usar la orden `cp` para copiar la imagen a la llave. Esto borra de forma definitiva cualquier contenido previo en la llave!

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

Nota: El comando `sync` se utiliza para asegurarse de que todos los datos, que el kernel almacena en la memoria mientras se copia la imagen, se escriben en la llave USB.

4.4.3 Usar el espacio libre en el dispositivo 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
```

Después de crear la partición, donde `$-PARTITION` es el nombre de la partición, por ejemplo `/dev/sdb2` se tiene que crear un sistema de archivos en ella. Una opción posible será `ext4`.

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

Nota: Si se desea usar el espacio extra con Windows, según parece, ese sistema operativo no puede acceder normalmente a otra partición más que a la primera. Se han comentado algunas soluciones a este problema en nuestra [lista de correo](#) pero según parece no hay una solución fácil.

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 Arrancar el medio en vivo

La primera vez que se arranque desde el medio de almacenamiento en vivo, ya sea CD, DVD, llave USB, o de arranque en red PXE, primero puede ser necesario algún tipo de configuración en la BIOS de la máquina. Dado que las BIOS varían mucho en sus características y combinaciones de

teclas, no se puede entrar en el tema en profundidad aqu. Algunas BIOS proporcionan una tecla para abrir un men de dispositivos de arranque que es la manera ms fcil de hacerlo si se encuentra disponible en el sistema. De lo contrario, se tiene que entrar en el men de configuracin de la BIOS y cambiar el orden de arranque y colocar el dispositivo de arranque del sistema en vivo antes que el dispositivo de arranque habitual.

Una vez que se haya arrancado desde el medio de almacenamiento, se accede a un men de arranque. Si se pulsa la tecla enter, el sistema arrancar usando el modo por defecto Live y las opciones predeterminadas. Para obtener ms informacin acerca de las opciones de arranque, ver la opcin help del men y tambin las pginas del manual de live-boot y live-config que se encuentran en el sistema en vivo.

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 Usar una mquina virtual para pruebas

Ejecutar las imgenes en vivo en una mquina virtual (VM) puede ser un gran ahorro de tiempo para su desarrollo. Esto no est exento de advertencias:

Para ejecutar una mquina virtual se requiere tener suficiente memoria RAM para el sistema operativo huésped y el anfitrión y se recomienda una CPU con soporte de hardware para la virtualización.

Existen algunas limitaciones inherentes a la ejecución en una mquina virtual, por ejemplo, rendimiento de video pobre o limitada gama de hardware emulado.

Cuando se desarrolla para un hardware específico, no hay sustituto mejor que el propio hardware.

A veces hay errores causados nicamente por la ejecución en una mquina virtual. En caso de duda, probar la imagen directamente en el hardware.

Siempre que se pueda trabajar dentro de estas limitaciones, mirar que software VM hay disponible y elegir uno que sea adecuado según las necesidades.

4.5.1 Probar una imagen ISO con QEMU

La mquina virtual ms verstil en Debian es QEMU. Si el procesador tiene soporte de hardware para virtualización, utilizar el paquete qemu-kvm. En la descripción del paquete qemu-kvm se enumera brevemente la lista de requisitos.

En primer lugar, instalar qemu-kvm si el procesador lo soporta. Si no es as, instalar qemu, en cuyo caso el nombre del programa ser qemu en vez de kvm en los siguientes ejemplos. El paquete qemu-utils también es útil para la creación de imágenes virtuales de disco con qemu-img.

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

Arrancar una imagen ISO es sencillo:

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

Consultar las páginas del manual para más detalles.

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 Probar una imagen ISO con VirtualBox

Para probar una imagen ISO con 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: Para probar los sistemas en vivo con soporte X.org en virtualbox, se puede incluir el paquete del driver de VirtualBox X.org, virtualbox-guest-dkms y virtualbox-guest-x11, en la configuracin de live-build. De lo contrario, la resolucin se limita a 800x600.

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

Para que el paquete dkms funcione, hace falta tener instalados tambin los kernel-headers para la variante del kernel utilizado. En lugar de enumerar manualmente el paquete linux-headers correspondiente en la lista de paquetes creados anteriormente, live-build puede seleccionarlo automticamente.

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

4.6 Construir y utilizar una imagen 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 se ha creado una imagen ISO hbrida con el ejemplo anterior, se tendr que limpiar el directorio de trabajo con el comando `lb clean` (ver [El comando lb clean](#)):

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

Ejecutar el comando `lb config` como antes pero esta vez especificando el tipo de imagen HDD:

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

Crear ahora la imagen con el comando `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 Creacin de una imagen de arranque en red

La siguiente secuencia de comandos crear una imagen de arranque en red bsica que contendr el sistema por defecto de Debian sin X.org. Se puede usar para el arranque en red.

Nota: si se ha seguido alguno de los ejemplos anteriores, se tendr que limpiar el directorio de trabajo con el comando `lb clean`:

```
# lb clean
```

En este caso concreto, un `lb clean --binary` no sera suficiente para eliminar las etapas necesarias. La razn de esto es que en las configuraciones de arranque en red, se debe utilizar una configuracin `initramfs` diferente que `live-build` ejecuta automticamente al crear imgenes `netboot`. Ya que la creacin del `initramfs` pertenece a la etapa `chroot`, realizar el cambio a `netboot` en un directorio de construccin ya existente significa reconstruir

la etapa `chroot` tambin. Por lo tanto, se tiene que ejecutar un `lb clean` (que tambin eliminar la etapa `chroot`).

Ejecutar el comando `lb config` de la siguiente manera para configurar la imagen de arranque en red:

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

A diferencia de las imgenes ISO y HDD, el sistema de arranque en red en s mismo no enva la imagen del sistema de ficheros al cliente, por eso los ficheros se deben enviar mediante NFS. Con `lb config` se puede seleccionar diferentes sistemas de ficheros en red. Las opciones `--net-root-path` y `--net-root-server` especifican la ubicacin y el servidor, respectivamente, del servidor NFS en el que se encuentra la imagen del sistema de ficheros en el arranque. Se debe asegurar que estos se ajustan a los valores adecuados para la red y el servidor deseados.

Crear ahora la imagen con el comando `lb build`:

```
# lb build
```

En un arranque en red, el cliente ejecuta una pequea pieza de software que generalmente se encuentra en la EPROM de la tarjeta Ethernet. Este programa enva una solicitud de DHCP para obtener una direccin IP e informacin sobre qu hacer a continuacin. Por lo general, el siguiente paso es conseguir un gestor de arranque de alto nivel a travs del protocolo TFTP. Este gestor podra ser PXELINUX, GRUB, o incluso arrancar directamente un sistema operativo como 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/.

Ahora se debe configurar tres servicios en el servidor para el arranque en red: el servidor DHCP, el servidor TFTP y el servidor NFS.

4.7.1 Servidor DHCP

Hay que configurar el servidor DHCP de red para asegurar que proporciona una direccin IP al cliente, y para anunciar la ubicacin del gestor de arranque PXE.

He aqu un ejemplo que puede servir de inspiracin. Fue escrito para el servidor ISC DHCP isc-dhcp-server en su fichero de configuracin /etc/dhcp/dhcpd.conf:

```
# /etc/dhcp/dhcpd.conf - fichero de configuracin para 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

Se encarga de suministrar el kernel y el Disco RAM inicial para el sistema.

Se debe instalar el paquete tftpd-hpa. Este servidor podr suministrar todos los ficheros contenidos de un directorio raz, normalmente /srv/tftp. Para permitirle que pueda servir los ficheros de /srv/debian-live/tftpbboot, se debe ejecutar el siguiente comando con privilegios de superusuario:

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

y escribir el directorio del nuevo servidor tftp cuando sea requerido.

4.7.3 Servidor NFS

Una vez el equipo cliente ha descargado y arrancado el kernel de Linux junto a su initrd, intentar montar el sistema de archivos de la imagen en vivo a travs de un servidor NFS.

Se debe instalar el paquete nfs-kernel-server.

Entonces, se debe hacer que la imagen del sistema de archivos est disponible a travs de NFS aadiendo una lnea como la siguiente para /etc/exports:

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

e informar al servidor NFS sobre esta nueva exportacin con el siguiente comando:

```
# 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 Cmo probar el arranque en red

La creacin de una imagen de arranque en red es sencilla con live-build, pero probar las imgenes en mquinas fsicas puede ser un proceso mucho ms lento.

Para hacer nuestra vida ms fcil, se puede utilizar la virtualizacin.

4.7.5 Qemu

Instalar qemu, bridge-utils, sudo.

Se debe editar el fichero `/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
```

Obtener o crear un grub-floppy-netboot.

Lanzar qemu con `-net nic,vlan=0 -net tap,vlan=0,ifname=tun0`

4.8 Arrancar desde internet

Arrancar desde internet, o Webbooting, es una manera muy adecuada de descargar y arrancar sistemas en vivo utilizando internet como medio, ya que hay muy pocos requisitos para arrancar desde internet utilizando webbooting. Por un lado, se necesita un medio en vivo con un gestor de arranque, un disco ram inicial y un kernel. Por otro lado, un servidor web para almacenar los ficheros squashfs que contienen el sistema de ficheros.

4.8.1 Conseguir los ficheros para arrancar desde internet

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`.

Tambin es posible extraer los ficheros de una imagen iso ya existente. Para ello, hay que montar la imagen de la siguiente 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.

Pero, sin duda alguna, la forma ms fcil de extraer los ficheros de una imagen iso y subirlos al servidor web al mismo tiempo, es utilizando el `midnight commander` o `mc`. Si se tiene el paquete `genisoimage` instalado, este administrador de ficheros de dos paneles permite examinar el contenido de un archivo iso en un panel y subir los ficheros a travs de ftp en

el otro panel. A pesar de que este mtodo requiere un trabajo manual, no requiere privilegios de root.

4.8.2 Arrancar imgenes webboot

Aunque algunos usuarios pueden preferir la virtualizacin para probar el arranque desde internet, en este caso se utiliza hardware real para ilustrar el caso de uso que se explica a continuacin y que debe considerarse slo como un ejemplo.

Para arrancar una imagen webboot es suficiente copiar los elementos mencionados anteriormente, es decir, vmlinuz y initrd.img en una llave usb dentro de un directorio llamado live/ e instalar syslinux como gestor de arranque. Entonces, arrancar desde la llave usb y teclear `fetch=URL/-RUTA/AL/FICHERO` en las opciones de arranque. live-boot se encargar de descargar el archivo squashfs y almacenarlo en la memoria ram. De este modo, es posible utilizar el sistema de ficheros comprimido descargado como si fuera un sistema en vivo normal. Por ejemplo:

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

Caso de uso: Se tiene dos archivos squashfs almacenados en un servidor web, uno que contiene un escritorio completo, como gnome, y uno standard. Si se necesita un entorno grfico para una mquina, se puede insertar la llave usb y arrancar desde internet la imagen gnome. Si se necesita una de las herramientas incluidas en el segundo tipo de imagen, quizs para otra mquina, se puede arrancar desde internet la imagen standard.

294

295

296

297

298

299

300

301

302

303

Descripci3n general de las herramientas

5. Descripci3n general de las herramientas

Este captulo contiene una descripci3n general de las tres herramientas principales utilizadas en la creaci3n de sistemas en vivo: live-build, live-boot y live-config.

5.1 El paquete live-build

live-build es una colecci3n de scripts para generar los sistemas en vivo. A estos scripts tambi3n se les conoce como comandos.

La idea detr3s de live-build es ser un marco que utiliza un directorio de configuraci3n para automatizar completamente y personalizar todos los aspectos de la creaci3n de una imagen de un sistema en vivo.

Muchos conceptos son similares a los utilizados para crear paquetes Debian con debhelper:

Los scripts tienen una ubicaci3n central para la configuraci3n de su funcionamiento. En debhelper, ste es el subdirectorio `debian/` de un rbol de paquetes. Por ejemplo, `dh_install` buscar, entre otros, un fichero llamado `debian/install` para determinar qu3 ficheros deben existir en un paquete binario en particular. De la misma manera, live-build almacena toda su configuraci3n bajo un subdirectorio `config/`.

Los scripts son independientes - es decir, siempre es seguro ejecutar cada comando.

A diferencia de debhelper, live-build contiene las herramientas para crear

un directorio de configuraci3n en esqueleto. Esto podr3a ser considerado como similar a herramientas tales como `dh-make`. Para obtener m3s informaci3n acerca de estas herramientas, seguir leyendo, ya que el resto de esta secci3n trata sobre los cuatro comandos m3s importantes. En interesante notar que est3n precedidos por `lb` que es una funci3n gen3rica para todos los comandos de live-build.

`lb config` : Responsable de inicializar un directorio de configuraci3n para la creaci3n de un sistema en vivo. Ver [El comando lb config](#) para m3s informaci3n.

`lb build` : Responsable de iniciar la creaci3n de un sistema en vivo. Ver [El comando lb build](#) para m3s informaci3n.

`lb clean` : Responsable de la eliminaci3n de partes de la creaci3n de un sistema en vivo. Ver [El comando lb clean](#) para m3s informaci3n.

5.1.1 El comando lb config

Como se coment3 en [live-build](#), los scripts que componen live-build obtienen su configuraci3n gracias al comando `source` desde un nico directorio llamado `config/`. Como la creaci3n de este directorio a mano ser3a largo y propenso a errores, se puede utilizar el comando `lb config` para crear el esqueleto de directorios de configuraci3n inicial.

Ejecutar `lb config` sin argumentos crea el subdirectorio `config/` que se completa con algunas opciones por defecto en ficheros de configuraci3n y dos rboles de subdirectorios en forma de esqueleto llamados `auto/` y `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...
```

Utilizar `lb config` sin ningn argumento sera conveniente para los usuarios que necesitan una imagen muy bsica, o que tienen intencin de proporcionar, ms adelante, una configuracin ms completa a travs de `auto/config` (ver **Gestionar una configuracin** para ms detalles).

Normalmente, se tendr que especificar algunas opciones. Por ejemplo, para especificar que gestor de paquetes se desea utilizar durante la construccin de la imagen:

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

Es posible especificar muchas opciones, tales como:

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

Una lista completa de opciones est disponible en la pgina de manual `lb.config`.

5.1.2 El comando `lb build`

El comando `lb build` lee la configuracin del directorio `config/`. A continuacin, ejecuta los comandos de nivel inferior necesarios para crear el sistema en vivo.

5.1.3 El comando `lb clean`

El comando `lb clean` es el encargado de eliminar varias partes de una creacin de forma que las creaciones posteriores puedan comenzar de forma limpia. Por defecto se eliminan las etapas `chroot`, `binary` y `source` pero

se deja el `cache` intacto. Adems, se pueden limpiar etapas de forma individual. Por ejemplo, si se han realizado cambios que slo afectan a la etapa `binary`, se debe usar `lb clean --binary` antes de crear una nueva `binary`. Si los cambios modifican el `bootstrap` y/o los `caches` de paquetes, por ejemplo, cambios en las opciones `--mode`, `--architecture` o `--bootstrap`, se debe utilizar `lb clean --purge`. Ver el manual de `lb-clean` para una lista detallada de todas sus opciones.

5.2 El paquete `live-boot`

`live-boot` es una coleccin de scripts que proporcionan ganchos (hooks) para `initramfs-tools`, que sirve para generar un `initramfs` capaz de arrancar sistemas en vivo, tales como los creados por `live-build`. Esto incluye imgenes ISO, archivos comprimidos en formato `tar` para el arranque en red, e imgenes para llaves 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 paquete `live-config`

`live-config` consiste en una serie de scripts que se ejecutan en el arranque despus de `live-boot` para configurar el sistema en vivo de forma automtica. Se ocupa de tareas como la creacin del nombre del equipo (`hostname`), las variantes locales y la zona horaria, crear el usuario en vivo, la inhibicin de trabajos de `cron` y el inicio de sesin automtico del usuario en vivo.

Gestionar una configuracin

6. Gestionar una configuracin

Este captulo explica como gestionar una configuracin para crear un sistema en vivo desde el principio, pasando por sucesivas versiones tanto de la herramienta live-build como de la imagen del sistema en vivo propiamente dicha.

6.1 Gestionar cambios en la configuracin

Las configuraciones en vivo rara vez son perfectas al primer intento. Puede estar bien pasar opciones a lb config en la lnea de comandos para realizar una construccin nica, pero es ms tpico revisar esas opciones y construir de nuevo hasta quedar satisfecho. Para gestionar estos cambios, se pueden utilizar scripts auto que garanticen que la configuracin se mantiene en un estado coherente.

6.1.1 Por qu utilizar scripts auto? Qu hacen?

El comando lb config almacena las opciones que se le pasan en ficheros en el directorio config/*, junto con muchas otras opciones que figuran en sus valores predeterminados. Si se ejecuta lb config una vez ms, no restablecer ninguna opcin que se estableci como por defecto en funcin de las opciones iniciales. As, por ejemplo, si se ejecuta lb config otra vez con un nuevo valor para `--binary-images`, todas las opciones que se establecieron como predeterminadas segn la opcin anterior ya no pueden funcionar con la nueva. Estos ficheros tampoco estan destinados a ser ledos o editados. Almacenan valores para ms de cien opciones, y nadie es capaz de ver las opciones que se especific realmente. Y por ltimo, si se

ejecuta lb config y a continuacin se actualiza live-build y hay alguna opcin que cambi de nombre, config/* todava tendr variables con las opciones viejas que ya no son vlidas.

Por todas estas razones, los scripts auto/* nos hacen la vida ms fcil. Son simples envoltorios para los comandos lb config, lb build y lb clean diseados para ayudar a gestionar una configuracin. El script auto/config contiene el comando lb config con todas las opciones que se desea, el script auto/clean elimina los ficheros que contienen variables de configuracin y el fichero auto/build crea un build.log de cada creacin. Cada uno de estos scripts se ejecuta automticamente cada vez que se ejecuta la orden lb correspondiente. Mediante el uso de estos scripts, la configuracin es ms fcil de leer y se mantiene internamente coherente de una revisin a la siguiente. Adems, ser mucho ms fcil identificar y corregir las opciones que necesitan cambiarse tras actualizar live-build y leer la documentacin actualizada.

6.1.2 Usar scripts auto de ejemplo

Para mayor comodidad, live-build incluye scripts auto de ejemplo que se pueden copiar y editar. Iniciar una nueva configuracin por defecto y a continuacin, copiar los ejemplos:

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

Editar auto/config, aadiendo las opciones que se desee. Por ejemplo:

```
#!/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/ "
"$@"
```

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

Ahora, cada vez que se utilice `lb config`, `auto/config` reiniciar la configuracin basndose en estas opciones. Cuando se desee realizar cambios, se deben editar las opciones en este fichero en lugar de pasarlas a `lb config`. Cuando se utilice `lb clean`, `auto/clean` limpiar los ficheros en `config/*` junto a los otros productos de construccin. Y, por ltimo, cuando se utilice `lb build`, `auto/build` crear un log del proceso de construccin llamado `build.log`.

Nota: Aqu se utiliza `noauto`, un parmetro especial para suprimir otra llamada a `auto/config`, evitando as una repeticin infinita. Asegurarse de no eliminarlo accidentalmente al hacer cambios en el fichero. Tener cuidado al dividir el comando `lb config` en varias lneas para facilitar la lectura, como se muestra en el ejemplo anterior, ya que no debe olvidarse la barra invertida (al final de cada lnea que sigue en la siguiente.

Editar `auto/config` y cualquier otra cosa que se necesite en el `rbol config` para adaptarlo a las propias necesidades. Por ejemplo, las imgenes prefabricadas con paquetes de la seccin `non-free` se crean simplemente aadiendo `--archive-areas main contrib non-free`.

Si se desea, se puede definir un mtodo abreviado en la configuracin de Git, aadiendo lo siguiente al fichero `$HOME"/.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
```

6.2 Clonar una configuracin publicada a travs de 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**.

Por ejemplo, para construir una imagen `standard`, utilizar el repositorio `live-images` de la siguiente manera:

Clonar el repositorio `live-images` completo copiar todas las configuraciones utilizadas para varias imgenes. Si se quiere construir una imagen diferente despus de haber terminado con la primera, cambiar a otro directorio y de nuevo, y opcionalmente, hacer los cambios necesarios para adaptarlo segn las necesidades.

En cualquier caso, recordar que cada vez que se tiene que construir una imagen hay que hacerlo como superusuario: `lb build`

Personalizacin de contenidos

7. Descripcin general de la personalizacin.

Este captulo presenta un resumen de las diversas formas en que se puede personalizar un sistema en vivo.

7.1 Configuracin en el momento de la creacin vs en el momento del arranque

Las opciones de configuracin de un sistema Debian Live se pueden dividir en opciones que se aplican en el momento de la creacin de la imagen del sistema en vivo y opciones que se tendrn en cuenta cuando el sistema en vivo arranque. Estas ltimas se pueden dividir a su vez en opciones que se ejecutan en la etapa inicial del arranque, aplicadas por el paquete live-boot, y otras que se llevarn a cabo posteriormente y que son aplicadas por el paquete live-config. Cualquier opcin en tiempo de arranque puede ser modificada por el usuario indicndola en los parmetros de arranque del kernel mediante el indicador de arranque. La imagen puede ser creada por defecto con los parmetros de arranque adecuados, de manera que los usuarios solamente tendrn que arrancar el sistema en vivo, directamente, sin necesidad de especificar ninguna opcin adicional, ya que las opciones por defecto sern las adecuadas. En particular, la opcin `lb -bootappend-live` permite introducir cualquier parmetro del kernel para el sistema en vivo, como pueden ser la persistencia, distribucin del teclado, zonas horarias, etc. Ver un ejemplo en [Personalizacin de las variantes locales e idioma](#).

Las opciones de configuracin en tiempo de creacin se describen en la pgina de manual del comando `lb config`. Las opciones en tiempo de arranque se describen en las pginas de manual de los paquetes live-boot y live-config.

Aunque los paquetes live-boot y live-config se instalan en el sistema en vivo que se est creando, tambin se recomienda que sean instalados en el sistema hupsed, que se utiliza para crear la imagen del sistema en vivo, con el fin de facilitar la referencia cuando se trabaja en una configuracin. No hay ningn problema en hacerlo, ya que ninguno de los scripts que contiene el sistema hupsed ser ejecutado, a menos que se configure el sistema hupsed como sistema en vivo.

7.2 Etapas de la creacin

El proceso de creacin de la imagen est dividido en etapas en las que se aplican diferentes personalizaciones en cada una de ellas. La primera etapa que se ejecuta es la etapa bootstrap. Esta fase inicial crea y rellena el directorio chroot con paquetes que constituyen un sistema Debian bsico. A continuacin la etapa chroot completa la creacin del directorio chroot, rellenndolo con todos los paquetes que han sido listados en la configuracin y material adicional. En esta etapa se utiliza la mayora de las personalizaciones de contenido. La etapa binary es la etapa final en la que se prepara la imagen del sistema en vivo utilizando el contenido del directorio chroot para construir el sistema de ficheros raz del futuro sistema en vivo, se incluye el instalador y cualquier otro material adicional de la imagen que no es parte del sistema de ficheros raz, como puede ser el gestor de arranque (bootloader) o ficheros de documentacin. Posteriormente, en la etapa opcional source se crea el fichero comprimido (tarball) que contiene los ficheros de cdigo fuente de los paquetes utilizados.

En cada una de estas etapas hay una secuencia particular en la que se aplican las acciones a realizar. Estas acciones son organizadas en forma de capas de tal manera que aseguran la personalizacin de una manera razonable. Por ejemplo, dentro de la etapa chroot, las preconfiguraciones (preseeds) se aplican antes que cualquier paquete sea instalado, los paquetes son instalados antes de incluir ningn fichero localmente y los scripts gancho

(hooks) sern ejecutados al final de todo, una vez que todos los materiales estn ubicados en su lugar.

7.3 Opciones para lb config en ficheros

Aunque la orden lb config crea un esqueleto de configuracin en el directorio config/, quizs sea necesario escribir ficheros de configuracin adicionales dentro de la jerarquía de subdirectorios de config/ con el fin de alcanzar los objetivos propuestos. En el proceso de creacin de la imagen estos ficheros adicionales sern copiados o en el sistema de ficheros que se utilizar en el sistema en vivo, o en el sistema de ficheros de la propia imagen binaria o quizs podrn suministrar opciones de configuracin al sistema en vivo que sera incomodo pasar en la línea de parámetros del kernel. Esto depender de en qu parte de la jerarquía de subdirectorios de config/ se copian estos ficheros. Se puede incluir cosas como listas de paquetes personalizadas, imágenes gráficas personalizadas o scripts gancho (hook scripts) para ejecutar o en el momento de creacin de la imagen o en el momento de arranque del sistema en vivo, aumentando la ya por otra parte considerable flexibilidad de Debian Live con código creado ex profeso.

7.4 Tareas de personalizacin

Los siguientes capítulos se organizan por tareas de personalizacin que el usuario realiza típicamente: Los capítulos de **Personalizacin de la instalacin de paquetes**, **Personalizacin de contenidos** y **Personalizacin de las variantes locales e idioma** cubren solamente unas pocas de las tareas que pueden realizarse.

Personalizacin de la instalacin de paquetes

8. Personalizacin de la instalacin de paquetes

Quizs la tarea ms bsica de personalizacin de un sistema en vivo es la seleccin de paquetes que sern incluidos en la imagen. Este captulo orienta a travs de las diferentes opciones de live-build que, en el momento de la creacin de la imagen, personalizan la instalacin de paquetes. Las opciones que seleccionan la distribucin base y las reas del archivo a utilizar son las que ms influyen a la hora de conocer qu paquetes estarn disponibles para su instalacin en la imagen. Para asegurar una buena velocidad de descarga de paquetes, se debera elegir el repositorio ms cercano. Se pueden aadir repositorios para backports, experimentales, paquetes personalizados o incluir ficheros de paquetes directamente. Se pueden definir listas de paquetes personalizadas, incluyendo metapaquetes que instalan muchos paquetes relacionados, como por ejemplo paquetes de un entorno de escritorio o lenguaje particular. Por ltimo existen varias opciones que dan algn control sobre cuando son instalados los paquetes por la herramienta apt o la herramienta aptitude, segn sea la elegida. Estas opciones pueden ser tiles si se utiliza un proxy, se quiere desactivar la instalacin de paquetes recomendados para ahorrar espacio o se necesita controlar las versiones de los paquetes a instalar mediante APT pinning, por nombrar algunas posibilidades.

8.1 Origen de los paquetes

8.1.1 Distribucin, reas de archivo y modo

The distribution you choose has the broadest impact on which packages are available to include in your live image. Specify the codename, 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
```

Las reas del archivo Debian son la principal divisin de paquetes dentro de una distribucin dada. En Debian las reas del archivo establecidas son main, contrib y non-free. Solamente los paquetes contenidos en main son parte de la distribucin Debian. sta es el rea definida por defecto en live-build. Se pueden indicar uno o ms valores tal y como se muestra en el siguiente ejemplo:

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

Experimentalmente se da soporte a alguna distribucin derivada de Debian mediante la opcin `--mode`. Por defecto, esta opcin toma el valor `debian` slo si se est construyendo en un sistema Debian o en un sistema desconocido. Si se utiliza `lb config` en cualquiera de las distribuciones derivadas a las que se da soporte, por defecto se construir una imagen de esa distribucin derivada. Por ejemplo, si `lb config` se ejecuta en modo `ubuntu` se utilizar el nombre de esa distribucin y las reas de archivos especficas de esa distribucin derivada en lugar de los propios de Debian y `live-build` modificar su comportamiento para adecuarlo al modo seleccionado.

Nota: La ayuda a los usuarios de las distribuciones para las cuales se aadieron estos modos son responsabilidad de los desarrolladores de dichas distribuciones. El Debian Live Project proporciona ayuda al desarrollo de la mejor manera posible, basndose en la informacin recogida de dichas distribuciones derivadas a pesar de que no desarrolla ni da soporte a las mismas.

8.1.2 Rplcas de Distribucin Debian

Los repositorios de Debian estn replicados en una gran red alrededor del mundo, de manera que se puede seleccionar la rplica ms cercana con el fin de obtener la mejor velocidad de descarga. Cada una de las opciones `--mirror-*` gobierna qu rplica de repositorio Debian se utiliza en las diferentes etapas de creacin. Si se recuerda de **Etapas de la creacin**, en la etapa `bootstrap` es cuando se crea el directorio `chroot` con un sistema mnimo mediante la herramienta `debootstrap`, y en la etapa `chroot` es cuando el directorio `chroot` es completado con los paquetes necesarios para crear el sistema de ficheros que ser utilizado en el sistema en vivo. A cada una de estas etapas le corresponde su propia opcin `--mirror-*`. Posteriormente, en la etapa `binary` se utilizarn las rplicas Debian indicadas en los valores de las opciones `--mirror-binary` y `--mirror-binary-security` en lugar de utilizar los indicados para las etapas anteriores.

8.1.3 Rplcas de Distribution utilizadas durante la creacin

Para indicar qu rplicas deben ser utilizadas en el momento de crear la imagen es suficiente con utilizar las opciones `--mirror-bootstrap` y `--mirror-chroot-security` como se muestra a continuacin.

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

El valor indicado en `--mirror-chroot` es utilizado como valor por defecto para la opcin `--mirror-bootstrap` si esta no es especificada.

8.1.4 Rplcas de distribucin Debian utilizadas en la ejecucin.

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 Repositorios adicionales

Se pueden añadir más repositorios, ampliando la lista de paquetes seleccionables más allá de aquellos disponibles para la distribución indicada, como pueden ser paquetes de backports, paquetes experimentales o personalizados. Para configurar repositorios adicionales se debe crear los ficheros `config/archives/your-repository.list.chroot` y/o `config/archives/your-repository.list.binary`. Al igual que en las opciones `-mirror-*`, estos ficheros gobiernan los repositorios utilizados en las etapas `chroot` y `binary` respectivamente, esto es, los repositorios que serán utilizados cuando se ejecute el sistema en vivo.

Por ejemplo, `config/archives/live.list.chroot` permite instalar paquetes de las instantáneas del repositorio Debian Live en el momento de crear la imagen.

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

Si se añade la misma línea a `config/archives/live.list.binary`, el repositorio será añadido al directorio `/etc/apt/sources.list.d/` del sistema en vivo.

Estos ficheros serán seleccionados automáticamente si existen.

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

En caso de necesitar un APT pinning personalizado, las preferencias de APT se pueden colocar mediante ficheros `config/archives/your-repository.pref`. `-binary, chroot`, y serán añadidos automáticamente al sistema en vivo en el directorio `/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 Selección de los paquetes a instalar

Hay varias maneras de seleccionar qué paquetes serán instalados por live-build en la imagen que cubren una variedad de necesidades diversas. Se puede nombrar paquetes individuales para instalar en una lista de paquetes. También se puede utilizar metapaquetes en esas listas, o seleccionarlos utilizando campos de ficheros de control de paquetes. Por último, también se pueden utilizar ficheros de paquetes de prueba o experimentales obtenidos antes de que aparezcan en los repositorios oficiales simplemente depositando estos ficheros directamente en el árbol de directorios `config/`.

8.2.1 Listas de paquetes

Las listas de paquetes proporcionan una potente forma de expresar qué paquetes deberán ser instalados. La sintaxis de las listas soporta las expresiones condicionales, que facilitan la creación de listas, adaptando su utilización a diversas configuraciones. También se pueden añadir nombre de paquetes en las listas utilizando shell helpers en tiempo de construcción.

Nota: El comportamiento de live-build cuando se especifica un paquete que no existe es determinado por lo que se haya configurado en la utilidad APT. Para más detalles ver [Utilizar apt o aptitude](#).

8.2.2 Utilizar metapaquetes

La manera más sencilla de rellenar una lista de paquetes es utilizar una tarea metapaquete mantenida por una distribución. Por ejemplo:

```
$ 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.

Todos los metapaquetes de tareas tienen el prefijo task-, por lo que una forma rpida de determinar cuales estn disponibles (aunque puede contener un puado de entradas falsas que coincidan con el nombre, pero que no son metapaquetes) es buscar el nombre del paquete con:

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

Adems de stos, se encuentran otros metapaquetes con diversos fines. Algunos son subconjuntos de paquetes de tareas ms amplias, como gnome-core, mientras que otros son partes especializadas individuales de un Debian Pure Blend, como los metapaquetes education-*. Para tener una lista de todos los metapaquetes en el archivo, instalar el paquete deb-tags y listar todos los paquetes con la etiqueta role::metapackage de la siguiente manera:

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

8.2.3 Listas de paquetes locales

Ya sea incluyendo metapaquetes en una lista, paquetes individuales, o una combinacin de ambos, todas las listas de paquetes locales se deben almacenar en config/package-lists/. Ya que se puede utilizar ms de una lista, esto se presta muy bien a los diseos modulares. Por ejemplo, se puede dedicar una lista a una eleccin particular de escritorio, la otra a una coleccin de paquetes relacionados que puedan ser fcilmente utilizados sobre un escritorio diferente. Esto permite experimentar con diferentes combinaciones de conjuntos de paquetes con un mnimo esfuerzo, as como compartir listas comunes entre diferentes proyectos de imgenes en vivo.

Para que sean procesadas, las listas de paquetes que se depositen en este directorio deben tener la extensin .list adems de la extensin de la etapa .chroot o .binary para indicar a qu etapa corresponde la lista.

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

Nota: Si no se especifica el sufijo, la lista ser usada en las dos etapas. En consecuencia, es conveniente especificar .list.chroot de modo que los paquetes se instalen nicamente en el sistema en vivo y no exista otra copia extra del paquete .deb.

8.2.4 Listas de paquetes locales para la etapa binary

Para crear una lista para la etapa binary crear un fichero con el sufijo .list.binary en config/package-lists/. Estos paquetes no son instalados en el sistema en vivo, pero son incluidos en pool/. El uso tpico de una de estas lista sera para una de las variantes de instalador normal (non-live N.del T.). Tal y como se mencionaba anteriormente, si se desea usar la misma lista para la etapa chroot basta con solamente aadir el sufijo .list

8.2.5 Generar listas de paquetes

A veces ocurre que la mejor manera de crear una lista es generarla con un script. Cualquier línea que comience con un signo de exclamación indica un comando que se ejecutará dentro del chroot cuando la imagen se construya. Por ejemplo, se podrá incluir la línea `! grep-aptavail -n -sPackage -FPriority standard --sort` en una lista de paquetes para producir una lista ordenada de los paquetes disponibles con `Priority: standard`.

De hecho, la selección de paquetes con la orden `grep-aptavail` (del paquete `dctrl-tools`) es tan útil que `live-build` proporciona un script de ayuda llamado `Packages`. Este script acepta dos argumentos: `field` y `pattern`. Por lo tanto, se puede crear una lista con los siguientes contenidos:

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

```
ia32-libs
#endif
```

En la expresión condicional pueden utilizarse varios valores. Por ejemplo para instalar el paquete `memtest86+` si la arquitectura es `i386` (`-architectures i386`) o es `amd64` (`-architectures amd64`) se puede especificar:

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

En la expresión condicional también pueden utilizarse variables que pueden contener más de un valor. Por ejemplo para instalar `vrms` si se utilizan las áreas del archivo `contrib` o `non-free` mediante la opción `-archive-areas` se puede indicar:

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

8.2.6 Utilización de condiciones dentro de las listas de paquetes

En las sentencias condicionales de las listas de paquetes pueden utilizarse cualquier variable disponible en `config/*` (excepto las que tienen el prefijo `LB`). En general esto significa que puede utilizarse cualquier opción válida para `lb config` cambiando las letras minúsculas por mayúsculas y los guiones por barras bajas. En la práctica solamente tiene sentido utilizar aquellas variables relacionadas con la selección de paquetes, como pueden ser `DISTRIBUTION`, `ARCHITECTURES` o `ARCHIVEAREAS`.

Por ejemplo, para instalar el paquete `ia32-libs` si se ha especificado la arquitectura `amd64` (`-architectures amd64`) se puede utilizar:

```
#if ARCHITECTURES amd64
```

No se permite el anidamiento de estructuras condicionales.

8.2.7 Eliminación de paquetes durante la instalación

Se puede crear listas de paquetes en ficheros con los sufijos `.list.chroot`live` y `.list.chroot`install` dentro del directorio `config/package-lists`. Si existe una lista `live` y una lista `install` los paquetes de la lista `.list.chroot`live` se eliminan con un script gancho después de la instalación (si el usuario utiliza el instalador). Los paquetes de la lista `.list.chroot`install` estarán presentes tanto en el sistema en vivo como en el sistema instalado. Este es un caso especial para el instalador y puede ser útil si se tiene `-debian-installer live`

establecido en la configuracin y se desea eliminar paquetes especificos del sistema en vivo durante la instalacin.

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'-live	X	X.binary
Package is installed in the live system	is	Yes	Yes	Yes	No
Package is removed after installing the live system	is	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 Tareas de Escritorio e Idioma

Las tareas de escritorio y de idioma son casos especiales que necesitan un poco de planificacin y configuracin extra. Si el medio de instalacin fue preparado para una clase particular de entorno de escritorio, el Instalador de Debian instalar automticamente la tarea de entorno de escritorio correspondiente. Para ello existen las tareas internas gnome-desktop, kde-desktop, lxde-desktop y xfce-desktop pero ninguna de ellas son presentadas en el men de tasksel. De igual forma, las tareas para idiomas

tampoco son presentadas en el men de tasksel, pero la seleccin del idioma, al inicio de la instalacin repercute en la seleccin de las correspondientes tareas del idioma.

Cuando se desarrolla una imagen de escritorio, la imagen normalmente arranca directamente a un escritorio de trabajo, las opciones de escritorio y de idioma por defecto han sido elegidas en tiempo de creacin, no en tiempo de ejecucin como en el caso del instalador de Debian. Eso no quiere decir que una imagen en vivo no pueda ser creada para admitir mltiples escritorios o varios idiomas y ofrecer al usuario una eleccin, pero ese no es un comportamiento por defecto de live-build.

Ya que no se ha previsto la instalacin automtica de tareas de idiomas, que incluyen cosas tales como tipos de letra especificos de cada lengua o paquetes de mtodos de entrada, si se quiere incluirlos, es necesario especificarlo en la configuracin. Por ejemplo, una imagen de escritorio GNOME que contenga soporte para el alemn podra incluir los siguientes metapaquetes de tareas:

```
$ 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 Versin y tipo de kernel

Dependiendo de la arquitectura, se incluyen por defecto en las imgenes uno o ms tipos de kernels. Se puede elegir entre diferentes tipos utilizando la opcin -linux-flavours. Cada tipo tiene el sufijo de la raz predeterminada linux-image para formar el nombre de cada metapaquete que a su vez depende del paquete del kernel exacto que debe incluirse en la imagen.

As, por defecto, una imagen de arquitectura amd64 incluir el metapaquete linux-image-amd64 y una imagen de arquitectura i386 incluir el metapaquete linux-image-586.

Cuando hay ms de una versin diferente del paquete del kernel disponible en los archivos configurados, se puede especificar el nombre de un paquete del kernel diferente con la opcin `--linux-packages`. Por ejemplo, suponer que se est construyendo una image de arquitectura amd64 y se quiere aadir el archivo experimental a fin de realizar pruebas. Para que se pueda instalar el kernel linux-image-3.18.0-trunk-amd64, se podra configurar la imagen de la siguiente 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 Kernels personalizados

Se pueden crear e incluir kernels personalizados, pero hay que tener en cuenta que live-build slo soporta los kernels que se integran en el sistema de gestin de paquetes de Debian y no es compatible con kernels que no esten en paquetes .deb.

La manera apropiada y recomendada de implementar los propios paquetes del kernel es seguir las instrucciones del kernel-handbook. Recordar modificar el ABI y los sufijos de los tipos del kernel e incluir los paquetes del kernel completo en un repositorio que coincidan con los paquetes linux y linux-latest.

Si se opta por construir los paquetes del kernel sin los metapaquetes adecuados, es necesario especificar una raz `--linux-packages` apropiada como se indica en **Versin y tipo de kernel**. Tal y como se explica en **Instalar paquetes modificados o de terceros**, es mejor si se incluyen los paquetes

del kernel personalizado en un repositorio propio, aunque las alternativas discutidas en esa seccin tambin funcionan.

Est ms all del alcance de este documento dar consejos sobre cmo personalizar un kernel. Sin embargo, se debe por lo menos, asegurarse de que la configuracin cumple los siguientes requisitos mnimos:

Utilizar un ramdisk inicial.

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

Incluir todos los mdulos de sistemas de ficheros requeridos por la configuracin (normalmente squashfs).

8.3 Instalar paquetes modificados o de terceros

Si bien est en contra de la filosofa de un sistema en vivo, en ocasiones es necesario crear un sistema con versiones de paquetes modificados a partir de los disponibles en el repositorio de Debian. Estos paquetes pueden modificar caractersticas existentes o dar soporte a caractersticas adicionales, idiomas y marcas, o eliminar elementos existentes en los paquetes que no son de interese. De manera similar, se pueden incluir paquetes de terceros para aadir funcionalidades a medida o propietarias.

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.

Existen dos formas de instalar paquetes personalizados:

packages.chroot

Utilizando un repositorio APT personalizado

El mtodo `packages.chroot` es el ms simple para aadir paquetes personalizados. Es muy til para personalizaciones rpidas pero tiene unos cuantos inconvenientes mientras que la utilizacin de un repositorio APT personalizado es ms lento de poner en marcha.

8.3.1 Mtodo `packages.chroot` para instalar paquetes personalizados

Para instalar paquetes personalizados solamente hay que copiar el paquete en el directorio `config/packages.chroot/`. Los paquetes contenidos en este directorio sern automticamente instalados en el sistema en vivo durante el proceso de creacin. No es necesario especificar nada ms.

Los paquetes deben nombrarse de la forma prescrita. La forma ms simple es usar `dpkg-name`.

El mtodo `packages.chroot` para la instalacin de paquetes personalizados tiene desventajas:

- No es posible utilizar secure APT.

- Se deben depositar todos los paquetes apropiados en el directorio `config/packages.chroot/`.

- No es adecuado para almacenar configuraciones en vivo en un control de versiones.

8.3.2 Mtodo de repositorio APT para instalar paquetes personalizados

A diferencia del mtodo `packages.chroot`, cuando se utiliza el mtodo de repositorio APT personalizado se debe asegurar que se especifica dnde se deben buscar los paquetes a instalar. Para ms informacin ver [Seleccin de los paquetes a instalar](#).

Aunque crear un repositorio APT para instalar paquetes personalizados puede parecer un esfuerzo innecesario, la infraestructura puede ser

fcilmente reutilizada posteriormente para ofrecer nuevas versiones de los paquetes.

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 "$-PACKAGES`FROM`REPOSITORY"" ` config/package-lists/ custom`repo`.list$-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`

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

8.3.3 Paquetes personalizados y APT

live-build utiliza APT para instalar todos los paquetes en el sistema

en vivo, as que hereda sus comportamientos. Un punto a resaltar es que (asumiendo una configuracin de APT por defecto) dado un paquete en dos repositorios diferentes con diferentes nmeros de versiones, APT seleccionar para instalar el paquete con nmero de versin superior.

Esta sera una buena razn para incrementar el nmero de version en los ficheros debian/changelog de los paquetes personalizados y as asegurar que sern estos los paquetes instalados en lugar de los contenidos en los repositorios oficiales de Debian. Esto puede tambin lograrse alterando las preferencias de pinning de APT del sistema en vivo. Para ms informacin ver [APT pinning](#).

8.4 Configurar APT en la creacin

Se puede configurar APT mediante varias opciones que se aplicarn en el momento de crear la imagen. (La configuracin que APT utilizar cuando se ejecute el sistema en vivo puede ser configurada de la manera que habitualmente se utiliza para introducir contenidos del sistema en vivo, esto es, incluyendo las configuraciones apropiadas en el directorio `config/includes.chroot/.).` Se puede encontrar una lista completa de las opciones para configurar APT en la pgina de manual de `lb.config`. Son aquellas opciones que comienzan con `apt`.

8.4.1 Utilizar apt o aptitude

Se puede seleccionar qu herramienta se utilizar para instalar paquetes, `apt` o `aptitude`, en el momento de crear la imagen mediante la opcin `--apt` de `lb config`. Esta seleccin definir el comportamiento preferido en la instalacin de paquetes, siendo la mayor diferencia la manera de tratar los paquetes no disponibles.

`apt`: Con este mtodo, si se especifica un paquete no existente, la instalacin fallar. Es el comportamiento por defecto.

`aptitude`: Con este mtodo, si se especifica un paquete no existente, la instalacin continuar sin error.

8.4.2 Utilizacin de un proxy con 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 Ajuste de APT para ahorrar espacio

En ocasiones es necesario ahorrar un poco de espacio en el medio de instalacin. Las dos opciones descritas a continuacin pueden ser de interres.

Si no se desea incluir los ndices de APT en la imagen creada se puede utilizar la siguiente opcin:

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

Esto no modificar el comportamiento de las entradas definidas en `/etc/apt/sources.list`, sino que solo afecta a si existirn o no ficheros de ndice en el directorio `/var/lib/apt`. El compromiso viene de que APT necesita estos ficheros ndices para funcionar en el sistema en vivo, as que, si no existen, el usuario deber ejecutar la orden `apt-get update` para crear estos ndices antes de poder ejecutar una orden del tipo `apt-cache search` o `apt-get install`.

Si la instalacin de los paquetes recomendados aumenta demasiado el

tamao de la imagen, siempre y cuando se est preparado para hacer frente a las consecuencias que se mencionan a continuacin, se puede desactivar el valor por defecto de esta opcin de APT con:

```
$ 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 Pasar opciones a apt o a aptitude

Si no hay una opcin lb config para modificar el comportamiento de APT

en la forma que se necesita, utilizar `--apt-options` o `--aptitude-options` para pasar opciones a la herramienta APT configurada. Consultar las pginas de manual apt y aptitude para ms detalles. Tener en cuenta que ambas opciones tienen valores por defecto que tendran que mantenerse, adem de las opciones que se pueden especificar. As, por ejemplo, supongamos que se ha incluido algo con fines de prueba de snapshot.debian.org y se desea especificar `Acquire::Check-Valid-Until=false` para que APT est feliz con el fichero Release caducado, se hara como en el ejemplo siguiente, aadiendo la opcin de nuevo despus del valor por defecto `--yes`:

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

Consultar las pginas de manual para entender completamente estas opciones y cundo utilizarlas. Esto es slo un ejemplo y no debe ser interpretado como consejo para configurar la imagen. Esta opcin no sera apropiada para, por ejemplo, una versin final de una imagen en vivo.

Para configuraciones ms complicadas que implican opciones apt.conf puede ser necesario crear un fichero `config/apt/apt.conf`. Ver tambin las otras opciones apt-* para tener algunos atajos convenientes para las opciones que se necesitan con frecuencia.

8.4.5 APT pinning

Como informacin bsica, sera recomendable leer la pgina de manual apt'-preferences(5). APT pinning puede ser configurado o en tiempo de creacin de la imagen, creando los ficheros `config/archives/*.pref`, `config/archives/*.pref.chroot`, y `config/apt/preferences`. o en tiempo de ejecucin del sistema en vivo creando el fichero `config/includes.chroot/etc/apt/preferences`.

Supongamos que se est creando un sistema en vivo basado en trixie pero

se necesita instalar todos los paquetes live que terminan instalados en la imagen binaria final desde la versin inestable sid en el momento de crear la imagen. Se deber aadir sid a los orgenes (sources) de APT y fijar (pin) los paquetes live con una prioridad ms alta pero todos los otros paquetes con una prioridad ms baja que la prioridad por defecto de manera que solamente los paquetes fijados sean instalados desde sid mientras que el resto ser obtenido desde la distribucin base, trixie . Esto se puede realizar de la siguiente forma:

```
Pin: version *
Pin-Priority: -1
```

517

```
$ echo "deb http://mirror/debian/ sid main" > config/archives/sid.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 prioridad pin negativa previene la instalacin de un paquete, como puede ser el caso de que no se desee que un paquete recomendado por otro sea instalado al instalar el primero. Supongamos que se est creando una imagen LXDE aadiendo task-lxde-desktop en config/package-lists/-desktop.list.chroot, pero no se desea preguntar al usuario si desea almacenar las claves wifi en el keyring. Este metapaquete depende de lxde-core, el cual recomienda gksu que a su vez recomienda gnome-keyring. As que el objetivo es omitir la instalacin del paquete gnome-keyring, que puede conseguirse aadiendo un fichero con el siguiente contenido a config/apt/-preferences:

519

```
Package: gnome-keyring
```

Personalizacin de contenidos

9. Personalizacin de contenidos

Este captulo trata, no solamente de una mera descripcin de cmo seleccionar los paquetes a incluir en el sistema en vivo, sino que adems presenta cmo hacer el ajuste fino de la personalizacin de los contenidos del propio sistema. Los includes permiten adjuntar o reemplazar cualquier fichero en la imagen en vivo a crear, los scripts gancho (hooks) permiten ejecutar cualquier orden en las diferentes etapas de creacin y en el momento del arranque y por ltimo, la preconfiguracin permite configurar paquetes cuando son instalados, suministrando las respuestas a las preguntas de debconf.

9.1 Includes

Idealmente, un sistema en vivo debera incluir solamente los ficheros proporcionados por los paquetes sin modificar. Sin embargo, algunas veces es conveniente incluir o modificar algn contenido mediante ficheros. La utilizacin de includes posibilita la inclusin, modificacin o cambio de cualquier fichero en la imagen en vivo a crear. live-build utiliza dos mecanismos:

Includes locales en chroot : Estos includes permiten incluir o reemplazar ficheros en el sistema de ficheros chroot. Para ms informacin ver [Includes locales en Live/chroot](#)

Includes locales en Binary: Estos includes permiten incluir o reemplazar ficheros en la propia imagen binaria generada. Para ms informacin ver [Includes locales en Binary](#)

Para ms informacin acerca de la diferencia entre las imgenes Live y binary

ver [Trminos](#)

9.1.1 Includes locales en Live/chroot

Los includes locales en chroot se utilizan para incluir o reemplazar ficheros en el sistema de ficheros Live/chroot de manera que puedan ser utilizados en el sistema en vivo. Una utilizacin tpica de estos includes puede ser rellenar el directorio (/etc/skel) usado por el sistema Live para crear el directorio home del usuario. Otra utilizacin tpica es suministrar ficheros de configuracin que pueden ser incluidos o reemplazados en la imagen sin necesidad de realizar procesamiento alguno; Si se necesita realizar algn procesamiento de estos ficheros ver la seccin [Scripts gancho locales en el chroot](#)

Para incluir ficheros solamente hace falta aadirlos al directorio de configuracin config/includes.chroot. Habr una relacin directa entre este directorio y el directorio raz / del sistema en vivo. Por ejemplo, si se desea aadir un fichero para que sea el fichero /var/www/index.html del sistema en vivo se puede hacer lo siguiente:

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

El directorio de configuracin presentar la siguiente jerarquía:

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

Los includes locales en chroot sern instalados despues de la instalacin de los paquetes de manera que los includes sobreescribirn cualquier fichero que los paquetes puedan haber instalado.

9.1.2 Includes locales en Binary

Se puede incluir material como documentacin, videos, etc en el sistema de ficheros del medio (USB, CDROM, etc) donde se grabar la imagen de manera que sea accesible nada ms insertar el medio sin necesidad de arrancar el sistema en vivo. Para esto se utilizan los includes locales en Binary. Funciona de manera similar a los includes locales en chroot comentados anteriormente. Por ejemplo, supongamos que en el medio de instalacin se desea aadir unos ficheros con videos de demostracin `~/video`demo.*` sobre el funcionamiento del sistema en vivo de manera que el usuario pueda acceder a ellos a travs de la pgina de indice HTML. Simplemente se debe copiar el material en `config/includes.binary/` de la siguiente manera:

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

Los ficheros aparecern ahora en el directorio raz del medio en vivo.

9.2 Scripts gancho (Hooks)

Los scripts gancho permiten ejecutar rdenes para personalizar la imagen en las etapas chroot y binary. Dependiendo de si se est construyendo una imagen en vivo o una imagen de un sistema normal, hay que colocar los ganchos en `config/hooks/live` o `config/hooks/normal` respectivamente. A stos se les llama con frecuencia ganchos locales, ya que se ejecutan dentro del entorno de construccin.

Tambin hay ganchos en tiempo de arranque que permiten ejecutar rdenes

una vez que la imagen ya se ha construido, durante el proceso de arranque.

9.2.1 Scripts gancho locales en el chroot

Para ejecutar rdenes en la etapa chroot se deben crear scripts gancho (hooks) con el sufijo `.hook.chroot` que contengan dichas ordenes a ejecutar y depositarlos en el directorio `config/hooks/live` o en `config/hooks/normal`. Estos scripts sern ejecutados en el entorno del chroot despues de que el resto de las tareas de preparacin del chroot han sido realizadas. Se debe asegurar que previamente se han instalado en el entorno chroot cualquier paquete, fichero u rden que necesiten los scripts gancho. El paquete `live-build` instala en el directorio `/usr/share/doc/live-build/examples/hooks` del sistema husped unos cuantos scripts gancho de ejemplo para realizar tareas habituales de personalizacin del entorno chroot que pueden ser copiados o referenciados mediante enlace simblico en la propia configuracin.

9.2.2 Scripts gancho locales en Binary

Para ejecutar comandos en la etapa Binary se deben crear scripts gancho con el sufijo `.hook.binary` que contengan las ordenes y depositarlos en el directorio `config/hooks/live` o en `config/hooks/normal`. Los scripts gancho se ejecutarn despues de finalizar el resto de procesos de la etapa pero antes de crear los checksum con `binary`checksum` que es el ltimo proceso que se ejecuta en esta etapa. Los scripts gancho no se ejecutan en el entorno del chroot, as que hay que tener cuidado de no modificar cualquier fichero fuera del rbol de creacin, o se daar el sistema de creacin. En `/usr/share/doc/live-build/examples/hooks` se pueden ver varios ejemplos de scripts gancho genericos que permiten tareas de personalizacin para la etapa binary. Estos scripts pueden ser utilizados en la propia configuracin

copindolos o creando enlaces simbólicos.

546 9.2.3 Scripts gancho en tiempo de arranque

547 Para ejecutar ordenes en el arranque del sistema en vivo, se puede suministrar scripts gancho a live-config depositndolos en el directorio config/-includes.chroot/lib/live/config/, tal y como se explica en la seccin de Personalizacin de la pgina de manual de live-config. Es interesante examinar los scripts gancho que trae de serie live-config que pueden verse en /lib/-live/config/ y fijarse en la secuencia de nmeros. Cuando se vaya a utilizar scripts propios deben ser prefijados con un nmero para indicar el orden de ejecucin. Otra posibilidad es utilizar un paquete personalizado tal y como se describe en [Instalar paquetes modificados o de terceros](#).

548 9.3 Preconfiguracin de las preguntas de Debconf

549 Los ficheros del directorio config/preseed/ con el sufijo .cfg seguido por la etapa (.chroot o .binary) son ficheros de preconfiguracin para debconf. live-build instalar estos ficheros mediante debconf-set-selections durante la etapa correspondiente.

550 Ver debconf(7) en el paquete debconf para obtener ms informacin acerca de debconf.

Personalizacin del comportamiento en tiempo de ejecucin.

10. Personalizacin del comportamiento en tiempo de ejecucin.

Toda la configuracin que se hace en tiempo de ejecucin es realizada por live-config. stas son algunas de las opciones ms comunes de live-config en las que los usuarios estn ms interesados. Se puede encontrar una lista completa de todas las posibilidades en la pgina de manual de live-config.

10.1 Personalizacin del usuario por defecto del sistema en vivo

Una consideracin importante es que el usuario por defecto del sistema en vivo es creado por live-boot en el arranque y no live-build durante la creacin de la imagen. sto no slo influye dnde se introducen los materiales relacionados con este usuario durante la creacin de la imagen tal y como se explica en [Includes locales en Live/chroot](#) sino tambin a cualquier grupo y a los permisos asociados con el usuario por defecto del sistema en vivo.

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 utilizar live-config.user-default-groups=audio,cdrom,dip,floppy,video,plugdev,netdev como parmetro de arranque.

Adems, es posible cambiar el usuario por defecto user y la contrasea por defecto live. Si se desea cambiarlos por cualquier motivo, se puede conseguir de forma sencilla tal y como se explica a continuacin:

Cambiar el nombre del usuario por defecto es tan sencillo como especificarlo en la configuracin:

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

Una posible forma de cambiar la contrasea por defecto es usando un script gancho (hook) tal y como se describe en [Scripts gancho en tiempo de arranque](#). Para conseguirlo se puede usar el script gancho passwd de /usr/share/doc/live-config/examples/hooks, ponerle un prefijo adecuado (p.ej. 2000-passwd) y aadirlo a config/includes.chroot/lib/live/config/

10.2 Personalizacin de las variantes locales e idioma

Cuando el sistema en vivo arranca, el idioma est implicado en dos pasos:

Generar las variantes locales

Establecer la distribucin del teclado

La variante local predeterminada en la creacin de un sistema en vivo es locales=en`US.UTF-8. Para definir la variante local que se debe generar,

```
de'sundeadkeys  ch: German (Switzerland, Sun dead keys)
de'mac         ch: German (Switzerland, Macintosh)
! option
```

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

Se pueden especificar diversas variantes locales separandolas con comas.

Este parametro se puede utilizar en la linea de comandos del kernel, al igual que los parametros de configuracin del teclado indicados a continuacin. Es posible configurar una variante local con idioma`pas (en cuyo caso se utiliza el tipo de codificacin por omisin) o tambien con la expresin completa idioma`pas.codificacin. La lista de todas las variantes locales est en /usr/share/i18n/SUPPORTED.

live-config se encarga de la configuracin del teclado de la consola y del entorno grfico X utilizando el paquete console-setup. Para configurarlos se puede utilizar los parmetros de arranque keyboard-layouts, keyboard-variants, keyboard-options y keyboard-model a travs de la opcin `-bootappend-live`. Se puede encontrar una lista de opciones vlidas para estos parmetros en `/usr/share/X11/xkb/rules/base.lst`. Para hallar la distribucin del teclado y la variante que corresponde a un idioma se puede buscar el nombre en ingls de la nacin donde se habla el idioma, por ejemplo:

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

Cada variante muestra una descripci3n de la disposici3n que aplica.

Normalmente, slo es necesario configurar la disposicin del teclado. Por ejemplo, para obtener los ficheros de la variante local de la disposicin del teclado alemn y suizo-alemn en X utilizar:

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

Sin embargo, para casos de uso muy específicos, se puede incluir otros parámetros. Por ejemplo, para configurar un sistema Francs con una disposición French-Dvorak (también llamado Bepo) en un teclado USB Type-Matrix EZ-Reach 2030, utilizar:

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

Para cada una de las variables de configuracin del teclado keyboard-* se puede especificar varios valores separados por comas. A excepcin de keyboard-model, que slo acepta un valor. En la pgina de manual keyboard(5) se explican los detalles y algunos ejemplos de cmo utilizar las variables XKBMODEL, XKBLAYOUT, XKBVARIANT y XKBOPTIONS. Si se especifican diferentes valores en keyboard-variants estos se corresponderan uno a uno con los valores keyboard-layouts (ver setxkbmap(1) opcin -variant). Se admiten valores vacos; por ejemplo para definir dos distribuciones de teclado, la que se usa por omisin US QWERTY y otra US Dvorak, utilizar:

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

10.3 Persistencia

Un paradigma de un cd en vivo (live cd N. del T.) es ser un sistema pre-instalado que funciona desde medios de almacenamiento de slo lectura, como un CD-ROM, donde los cambios y las modificaciones no se guardan tras reiniciar el sistema en que se ejecuta.

Un sistema en vivo es una generalizacin de este paradigma pero que es compatible con otros medios de almacenamiento, no slo en CDs. An as, en su comportamiento predeterminado, se debe considerar un sistema de slo lectura y todos los cambios en tiempo de ejecucin del sistema se pierden al apagar el equipo.

La persistencia es un nombre comn que se da a los diferentes tipos de soluciones para guardar algunos o todos los cambios realizados durante la ejecucin tras reiniciar el sistema. Para entender cmo funciona es til saber que incluso si el sistema se inicia y se ejecuta desde los medios de almacenamiento de slo lectura, las modificaciones de los ficheros y directorios se escriben en medios de escritura, por lo general en la memoria ram (tmpfs) y los datos guardados en la ram se pierden al reiniciar.

Los datos almacenados en esta memoria ram se pueden guardar en un soporte grabable, como un medio de almacenamiento local, un recurso compartido en red o incluso en una sesin de un CD/DVD regrabable en multisesis. Todos estos medios son compatibles de diferentes maneras y todos, menos el ltimo, requieren un parmetro de arranque especial que se especifica en el momento del arranque: persistence.

Si se usa el parmetro de arranque persistence (y no se usa la opcin

nopersistence), se busca en los medios de almacenamiento locales (p.ej. discos duros, llaves USB) volmenes con persistencia durante el arranque. Es posible restringir qu tipos de volmenes persistentes se pueden usar especificando ciertos parmetros de arranque descritos en la pgina del manual de live-boot(7). Un volumen persistente es cualquiera de los siguientes:

una particin, identificada por su nombre GPT.

Un sistema de ficheros, identificado por su etiqueta de sistema de ficheros.

una fichero imagen situado en la raz de cualquier sistema de ficheros que pueda ser leído (incluso una particin NTFS de otro sistema operativo), identificado por su nombre de fichero.

La etiqueta del volumen para las overlays debe ser persistence pero ser ignorado a menos que contenga en su raz un fichero llamado persistence.conf que se utiliza para personalizar la persistencia del volumen, esto es, especificar los directorios que se desea guardar en un volumen de persistencia despus de reiniciar. Ver [El fichero persistence.conf](#) para ms detalles.

He aqu algunos ejemplos de cmo preparar un volumen para ser usado para la persistencia. Puede ser, por ejemplo, una particin en un disco duro o en una llave usb creada con, p.ej.

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

Ver [Usar el espacio libre en el dispositivo USB](#).

Si ya existe una particin en el dispositivo, slo se tiene que cambiar la etiqueta con uno de los siguientes:

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

Un ejemplo de cmo crear un fichero imagen basado en ext4 para ser usado para la persistencia:

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

Despus de crear el fichero imagen, a modo de ejemplo, para hacer /usr persistente pero nicamente guardando los cambios que se realizan en ese directorio en lugar de todos los contenidos de /usr, se puede utilizar la opcin union. Si el fichero imagen se encuentra en el directorio home, copiarlo a la raz del sistema de ficheros del disco duro y montarlo en /mnt como se explica a continuacin:

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

Despus, crear el fichero persistence.conf aadiendo contenido y desmontar el fichero imagen.

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

Ahora, reiniciar y arrancar el medio en vivo con el parmetro de arranque persistence.

10.3.1 El fichero persistence.conf

Un volumen con la etiqueta persistence debe ser configurado a travs de un fichero persistence.conf para crear directorios arbitrarios persistentes.

Ese fichero, situado en el sistema de ficheros raz del volumen, controla que directorios hace persistentes y tambin de que manera.

En la pgina de manual de persistence.conf(5) se explica en detalle cmo se configura el montaje de las overlays, pero un sencillo ejemplo es suficiente para la mayora de los casos. Supongamos que queremos crear nuestro directorio home y APT cache persistentes en un sistema de ficheros ext4 en la particin /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
```

Entonces reiniciamos. Durante el primer arranque los contenidos de /home y /var/cache/apt se copiarn en el volumen persistente y a partir de ese momento todos los cambios en esos directorios se guardarn all. Tener en cuenta que las rutas listadas en el fichero persistence.conf no pueden contener espacios en blanco ni los componentes especiales . y ... Adems, ni /lib, /lib/live (o ninguno de sus sub-directorios) ni / pueden hacerse persistentes montndolos de forma personalizada. Una posible alternativa a esta limitacin es aadir / union al fichero persistence.conf para conseguir una persistencia completa.

10.3.2 Utilizar varios medios persistentes

Existen diferentes mtodos para utilizar mltiples volmenes de persistencia para diferentes casos de uso. Por ejemplo, utilizar varios volmenes al mismo tiempo o seleccionar slo uno, entre varios, para fines muy especficos.

Se puede usar diferentes volmenes de overlays al mismo tiempo (con

sus propios ficheros persistence.conf) pero si varios volmenes hacen que un mismo directorio sea persistente, slo uno de ellos ser usado. Si dos unidades montadas estn anidadas (es decir, una es un sub-directorio de la otra) el directorio superior ser montado antes que el inferior de este modo no quedar uno escondido por el otro. La personalizacin de los montajes anidadados es problemtica si estn listados en el mismo fichero persistence.conf. Consultar la pgina de manual de persistence.conf(5) para ver como manejar ese caso si realmente es necesario. (aclaracin: normalmente no lo es).

Un posible caso de uso: Si se desea guardar los datos del usuario, es decir /home y los datos del superusuario, es decir /root en particiones diferentes, crear dos particiones con la etiqueta persistence y aadir un fichero persistence.conf en cada una de este modo, # echo /home & persistence.conf para la primera particin que guardar los ficheros del usuario y # echo /root & persistence.conf para la segunda particin que almacenar los ficheros del superusuario. Finalmente, utilizar el parmetro de arranque persistence.

Si un usuario necesita un almacenamiento persistente mltiple del mismo tipo para diferentes lugares o pruebas, tales como private y work, el parmetro de arranque persistence-label usado junto con el parmetro de arranque persistence permitir medios de almacenamiento persistentes mltiples pero nicos. Un ejemplo sera, si un usuario desea utilizar una particin persistente etiquetada private para datos de uso personal como los marcadores de un navegador o similares utilizara los parmetros de arranque: persistence persistence-label=private. Y para almacenar datos relacionados con el trabajo, como documentos, proyectos de investigacin o de otro tipo, utilizara los parmetros de arranque: persistence persistence-label=work.

Es importante recordar que cada uno de estos volmenes, private y work, necesita tambin un fichero persistence.conf en su raz. La pgina de manual

de live-boot contiene ms informacin acerca de cmo utilizar estas etiquetas con los antiguos nombres que se utilizaban en anteriores versiones.

10.3.3 Utilizar persistencia con cifrado

Utilizar la persistencia significa que algunos datos sensibles pueden estar expuestos a riesgo. Especialmente si los datos persistentes se almacenan en un dispositivo portable, como una memoria USB o un disco duro externo. Es entonces cuando el cifrado cobra sentido. Incluso aunque todo el procedimiento puede parecer complicado debido a la cantidad de pasos que hay que hacer, es muy fcil manejar particiones cifradas con live-boot. Para utilizar luks, que es el tipo de cifrado soportado, se necesita instalar cryptsetup tanto en la mquina que va a crear la particin cifrada como en el sistema en vivo con que se va a utilizar la particin persistente cifrada.

Para instalar cryptsetup en nuestra mquina:

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

Para instalar cryptsetup en nuestro sistema en vivo, lo aadimos a una package-lists:

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

Una vez se tiene el sistema en vivo con cryptsetup, bsicamente, slo se necesita crear una nueva particin, cifrarla y arrancar con los parmetros persistence y persistence-encryption=luks. Podramos habernos anticipado a este paso y haber aadido esos parmetros de arranque siguiendo el procedimiento habitual:

aadiendo la etiqueta persistence para que el dispositivo se monte como almacén de persistencia durante el arranque.

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

Vamos a entrar en detalles para quien que no est familiarizado con el cifrado. En el siguiente ejemplo vamos a utilizar una particin en un dispositivo usb que corresponde a /dev/sdc2. Tener en cuenta que es necesario determinar qu particin es la que se va a utilizar en cada caso especfico.

El primer paso es conectar la memoria usb y determinar de qu dispositivo se trata. El mtodo recomendado para los dispositivos en live-manual es utilizando ls -l /dev/disk/by-id. Despus de eso, crear una nueva particin y, a continuacin, cifrarla con una frase de contrasea de la siguiente manera:

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

A continuacin, abrir la particin luks en el mapeador de dispositivos virtuales. Se puede utilizar cualquier nombre que se desee. Aqu utilizamos live como ejemplo:

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

El siguiente paso es llenar el dispositivo con ceros antes de crear el sistema de ficheros:

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

Ahora, estamos listos para crear el sistema de ficheros. Ntese que estamos

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

Para continuar con nuestra configuracin, necesitamos montar el dispositivo, por ejemplo, en /mnt.

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

Y crear el fichero persistence.conf en la raz de la particin. Esto es, como se ha explicado antes, estrictamente necesario. Ver [El fichero persistence.conf](#).

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

Entonces, desmontar el punto de montaje:

```
# umount /mnt
```

Y opcionalmente, aunque puede ser una buena manera de salvaguardar los datos que acabamos de agregar a la particin, podemos cerrar el dispositivo:

```
# cryptsetup luksClose live
```

Vamos a resumir el proceso. Hasta ahora, hemos creado un sistema vivo capaz de utilizar el cifrado, que se puede copiar en una memoria usb como se explica en [Copiar una imagen ISO hbrida en un dispositivo](#)

USB. Tambin hemos creado una particin cifrada, que se puede crear en la misma memoria usb para llevarla a todas partes y hemos configurado la particin cifrada para ser utilizada como almacn de persistencia. As que ahora, slo tenemos que arrancar el sistema en vivo. En el momento del arranque, live-boot nos preguntará la frase de contrasea y montar la particin cifrada para ser utilizada para la persistencia.

Personalizacin de la imagen binaria

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

11. Personalizacin de la imagen binaria

11.1 Gestores de arranque

live-build utiliza syslinux y algunos de sus derivados (en funcin del tipo de imagen) como gestores de arranque por defecto. Se pueden personalizar fcilmente para satisfacer todas las necesidades.

Para utilizar un tema completo, copiar /usr/share/live/build/bootloaders en config/bootloaders y editar los ficheros all. Si no se desea modificar todas las configuraciones de los gestores de arranque disponibles, es suficiente con slo proporcionar una copia local personalizada de uno, por ejemplo, copiar la configuracin de isolinux en config/bootloaders/isolinux es suficiente, dependiendo del caso de uso.

Cuando se modifica uno de los temas predeterminados, si se quiere utilizar una imagen de fondo personalizada que se mostrar junto con el men de arranque, aadir una imagen splash.png de 640x480 pxeles. Y entonces, borrar el fichero splash.svg.

Hay muchas posibilidades a la hora de hacer cambios. Por ejemplo, los derivados de syslinux estn configurados por defecto con un tiempo de espera de 0 (cero) lo que significa que harn una pausa indefinida en su pantalla de inicio hasta que se pulse una tecla.

Para modificar el tiempo de espera de arranque de una imagen iso-hybrid se puede editar el fichero isolinux.cfg especificando el tiempo en unidades de segundo 1/10. Un fichero isolinux.cfg modificado para arrancar despus de cinco segundos sera as:

11.2 Metadatos ISO

Al crear una imagen binaria ISO9660 se pueden utilizar las siguientes opciones para aadir varios metadatos textuales a la imagen. Esto puede ayudar a identificar fcilmente la versin o la configuracin de una imagen sin arrancarla.

LB`ISO`APPLICATION/--iso-application NAME: Esto debera especificar la aplicacin que estar en la imagen. La longitud mxima para este campo es de 128 caracteres.

LB`ISO`PREPARER/--iso-preparer NAME: Esto debera identificar quin prepara la imagen, por lo general con algunos detalles de contacto. El valor predeterminado para esta opcin es la versin de live-build que se est utilizando, lo que puede ayudar con la posterior depuracin de errores. La longitud mxima para este campo es de 128 caracteres.

LB`ISO`PUBLISHER/--iso-publisher NAME: Esto debera identificar quin publica la imagen, por lo general con algunos detalles de contacto. La longitud mxima para este campo es de 128 caracteres.

LB`ISO`VOLUME/--iso-volume NAME: Esto debera especificar el volumen de identificacin de la imagen. Esto se utiliza como etiqueta visible para el usuario en algunas plataformas como Windows y Apple Mac OS. La longitud mxima para este campo es de 32 caracteres.

Personalizacin del Instalador de Debian

12. Personalizacin del Instalador de Debian

Las imgenes de los sistemas en vivo pueden integrarse con el Instalador de Debian. Hay varios tipos de instalacin que se diferencian en qu se incluye en la imagen y en cmo opera el instalador.

En esta seccin se debe estar atento a la utilizacin de las mayculas. Cuando se utiliza Instalador de Debian, con mayculas, se hace referencia explcita al instalador oficial del sistema Debian, y a nada ms ni a ningn otro instalador. A menudo se abrevia con d-i.

12.1 Tipos de imgenes segn el instalador

Principalmente existen tres tipos de imgenes segn el instalador:

Imgenes con Instalador Debian normal : Esta imagen en vivo se puede considerar como la imagen habitual. Dispone de un kernel y un initrd diferenciados que, al ser seleccionados desde el gestor de arranque, ejecutan un Instalador de Debian estndar, de la misma manera que lo haran si se arrancase desde una imagen de CD descargada desde el sitio oficial de Debian. Las imgenes que contienen un sistema en vivo con otro instalador independiente se suelen llamar imgenes combinadas.

En estas imgenes, el sistema operativo Debian se instala mediante la herramienta debootstrap que descarga paquetes .deb desde medios locales o por red. El resultado final es un sistema Debian por defecto instalado en el disco duro.

El conjunto de este proceso puede ser preconfigurado (preseeded) y per-

sonalizado de muchas maneras; Para ms informacin, ver las pginas relevantes en el manual del Instalador de Debian. Una vez que se ha generado el fichero de preconfiguracin adecuado a las necesidades, live-build puede encargarse de depositarlo en la imagen y activarlo de forma automtica.

Imgenes con Instalador Debian Live : Estas imgenes en vivo tambin disponen de un kernel y un initrd diferenciados que, al ser seleccionados desde el gestor de arranque, ejecutan un Instalador de Debian.

El procedimiento de instalacin es idntico al realizado por las imgenes Regulares pero, en lugar de utilizar debootstrap para obtener e instalar paquetes .deb, lo que hace es copiar al disco duro la imagen del sistema de ficheros que se haba preparado para lanzar el sistema en vivo. Esto se logra mediante un .udeb especial llamado live-installer.

Una vez finalizada esta etapa, el Instalador de Debian continua normalmente, instalando y configurando los siguientes elementos como pueden ser gestor de arranque, creacin de usuarios locales, etc.

Nota: Para poder incluir los dos tipos de instalador, normal y live, en el mismo medio, se debe deshabilitar el live-installer. Esto se hace utilizando la variable de preconfiguracin (preseed) live-installer/enable=false.

Instalador Debian del escritorio : Una vez el sistema en vivo est ejecutandose, se puede lanzar el Instalador de Debian haciendo clic en el icono correspondiente, sin importar el tipo de Instalador Debian utilizado en el arranque. Esta manera de instalar Debian es ms sencilla para el usuario y aconsejable en algunas situaciones. Para poder realizar esta accin se debe instalar el paquete debian-installer-launcher.

Por defecto, live-build no incluye las imgenes que utilizan el Instalador de Debian. Esto debe ser habilitado de forma especfica en lb config. Tambin hay que hacer notar que, para que la instalacin desde el escritorio funcione, el kernel del sistema en vivo debe ser el mismo que el kernel que utiliza

d-i en la arquitectura especificada. Por ejemplo:

una manera similar a la que se describe en **Includes locales en Live/chroot**, depositando el material en el directorio `config/includes.installer/`.

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

12.2 Personalizando el Instalador de Debian mediante preconfiguracin

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" "ll config/includes.installer/preseed.cfg
```

12.3 Personalizar el contenido del Instalador de Debian

Es posible que, con propsitos experimentales o para depuracin de errores, se desee incluir paquetes udeb creados localmente para el d-i. Estos paquetes udeb son componentes del Instalador de Debian que definen su comportamiento. Para incluirlos en la imagen, basta con depositarlos en el directorio de configuracin `config/packages.binary/`. Tambin pueden incluirse o reemplazarse ficheros y directorios en el `initrd` del instalador de

Proyecto

Contribuir al proyecto

13. Contribuir al proyecto

Cuando se enva una contribucion se debe identificar claramente al titular de los derechos de autor e incluir la declaracion de las licencias aplicables. Se hace notar que para ser aceptada, una contribucion debe ser publicada bajo la misma licencia que el resto del documento, es decir, GPL versin 3 o posterior.

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.

A pesar de que todas las entregas pueden ser revisadas, pedimos usar el sentido comn y hacer buenos commits con mensajes de commit adecuados.

Hay que escribir mensajes de entrega que consistan en una frase en ingles con significado completo, comenzando con una letra mayuscula y acabando con un punto final. Es habitual comenzar estas frases con la forma 'Fixing/Adding/Removing/Correcting/Translating/...'.

Escribir buenos mensajes de entrega. La primera frase debe ser un resumen exacto de los contenidos del commit, que se incluir en la lista de cambios. Si se necesita hacer algunas aclaraciones, escribirlas debajo dejando una linea en blanco despues de la primera y luego otra linea en blanco despues de cada prrafo. Las lineas de los prrafos no deben superar los 80 caracteres de longitud.

Hacer entregas de forma atmica, es decir, no mezclar cosas no rela-

cionadas en el mismo commit. Hacer un commit diferente para cada cambio que se realice.

13.1 Traduccin de las pginas de manual

Tambin se puede contribuir al proyecto trabajando en la traduccin de las pginas de manual de los diferentes paquetes live-* que el proyecto mantiene. El proceso es diferente, dependiendo de si se est comenzando una traduccin desde cero o si se continua trabajando en una traduccin ya comenzada:

Continuar con una traduccin ya comenzada

Si se desea mantener la traduccin de una lengua ya existente hay que realizar los cambios en el fichero o ficheros presentes en manpages/po/\$-LANGUAGE"/*.po y despues ejecutar make rebuild desde dentro del directorio manpages/. Esto actualizar las pginas de manual de manpages/\$-LANGUAGE"/*

Empezar una nueva traduccin desde cero

Para aadir una nueva traduccin de cualquiera de las pginas de manual del proyecto hay que seguir un proceso similar. Se puede resumir del modo siguiente:

Abrir el fichero o ficheros de manpages/pot/ en el editor preferido, como por ejemplo poedit, y guardarlo como fichero .po en manpages/po/\$-LANGUAGE"/. (Se tendr que crear el directorio del \$-LANGUAGE"/ correspondiente).

Ejecutar make rebuild desde dentro del directorio manpages/ para crear los ficheros manpages/\$-LANGUAGE"/ que contendrn las pginas de manual.

Recordar que se tendr que aadir todos los directorios y ficheros antes de escribir el mensaje de presentacin y hacer la entrega al servidor git.

Cmo informar acerca de errores.

14. Informes de errores.

Los sistemas en vivo estn lejos de ser perfectos, pero queremos que sean lo ms perfectos posible - con su ayuda. No dudar en informar de un error. Es mejor llenar un informe dos veces que no hacerlo nunca. Sin embargo, este captulo incluye recomendaciones sobre cmo presentar buenos informes de errores.

Para los impacientes:

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.

Se debe intentar proporcionar una informacin tan especfica como sea posible acerca del error. Esto incluye (al menos) la versin de live-build, live-boot, live-config y live-tools utilizada y la distribucin del sistema en vivo que se est construyendo.

14.1 Problemas conocidos

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 esto causa mucha dificultad, no se debe crear un sistema basado en testing o unstable, sino que debe utilizarse stable. live-build siempre crea, por defecto, la versin stable.

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 Hacer la investigacin

Antes de presentar el informe de errores, buscar en la web el mensaje de error en particular o el sntoma que se est percibiendo. Como es muy poco probable que sea la nica persona que tiene ese problema en concreto, siempre existe la posibilidad de que se haya discutido en otras partes y exista una posible solucin, parche o se haya propuesto una alternativa.

Se debe prestar especial atencin a la lista de correo del sistema en vivo, as como a su pgina web, ya que seguramente tienen la informacin ms actualizada. Si esa informacin existe, se debe incluir una referencia a ella en el informe de errores.

Adems, se debe comprobar las listas de errores actuales de live-build, live-boot, live-config y live-tools y verificar si se ha informado ya de algo similar.

14.3 Reconstruir desde cero

Para asegurarse de que un error en particular no es causado por crear el sistema basndose en los datos de un sistema anterior, se debe reconstruir

el sistema en vivo entero, desde el principio y comprobar si el error es reproducible.

Se puede generar un log del proceso de creacin mediante el comando tee. Se recomienda hacer esto de forma automtica con un script auto/build (ver [Gestionar una configuracin](#) para ms detalles).

14.4 Utilizar paquetes actualizados

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.

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

En el momento del arranque, live-boot y live-config guardan sus logs en /var/log/live/. Comprobar si hay algn mensaje de error en estos ficheros.

14.5 Recopilar informacin

Se debe proporcionar informacin suficiente con el informe. Como mnimo, la versin exacta de live-build donde se encuentra el error y los pasos para reproducirlo. Se debe utilizar el sentido comn e incluir cualquier informacin pertinente si se cree que podra ayudar a resolver el problema.

Adems, para descartar otros errores, siempre es una buena idea comprimir en un .tar el directorio config/ y subirlo a algn lugar, para que el equipo de Debian Live pueda reproducir el error (No se debe enviar como documento adjunto a la lista de correo). Si esto es difcil (por ejemplo, debido a su tamao) se puede utilizar la salida del comando lb config-dump que produce un resumen del rbol de configuracin (es decir, listas de archivos de los subdirectorios de config / pero no los incluye).

Para sacar el mximo provecho de un informe de errores, se requerir al menos la siguiente informacin:

Hay que recordar que los informes a enviar se deben hacer en ingls, por lo que para generar los logs en este idioma se debe utilizar la variante local English, p.ej. ejecutar los comandos de live-build o cualquier otro precedidos de LC`ALL=C o LC`ALL=en`US.

Arquitectura del sistema anfitriin

Distribucin del sistema anfitriin.

La versin de live-build del sistema anfitriin.

Versin de debootstrap en el sistema anfitriin.

Arquitectura del sistema en vivo.

Distribucin del sistema en vivo.

Versin de live-boot en el sistema en vivo.

Versin de live-config en el sistema en vivo.

Versin de live-tools en el sistema en vivo.

14.6 Aislar el fallo si es posible

Si es posible, aislar el caso del fallo al menor cambio posible que lo produzca. No siempre es fcil hacer esto, as que si no se consigue para el informe, no hay que preocuparse. Sin embargo, si se planea el ciclo de desarrollo bin, con conjuntos de cambios lo bastante pequeos por iteracin, puede ser posible aislar el problema mediante la construccin de una simple base de configuracin que se ajuste a la configuracin actual deseada, ms el conjunto del cambio que falla aadido. Si resulta difcil determinar que

cambios produjeron el error, puede ser que se haya incluido demasiado en cada conjunto de cambios y se deba desarrollar en incrementos ms pequeos.

14.7 Utilizar el paquete correcto sobre el que informar del error

En general, se debe informar sobre los errores en tiempo de creacin contra el paquete live-build. De los fallos en tiempo de arranque contra el paquete live-boot, y de los errores en tiempo de ejecucin contra el paquete live-config. Si no se est seguro de qu paquete es el adecuado o se necesita ms ayuda antes de presentar un informe de errores, lo mejor es enviar un informe contra el pseudo-paquete debian-live. Nosotros nos encargaremos de reasignarlo donde sea apropiado.

Sin embargo, se agradece si se intenta limitar la bsqueda a donde aparece el error.

14.7.1 En la preinstalacin (bootstrap) en tiempo de creacin.

live-build crea primero un sistema Debian bsico con debootstrap. Si un error aparece en este momento, se debe comprobar si est relacionado con un paquete especfico de Debian (es lo ms probable), o si est relacionado con la herramienta de preinstalacin en s.

En ambos casos, esto no es un error en el sistema en vivo, sino de Debian en s mismo, por lo cual nosotros probablemente no podamos solucionarlo directamente. Informar del error sobre la herramienta de preinstalacin o el paquete que falla.

14.7.2 Mientras se instalan paquetes en tiempo de creacin.

live-build instala paquetes adicionales del archivo de Debian que pueden fallar en funcin de la distribucin Debian utilizada y del estado diario del

archivo Debian. Se debe comprobar si el error es reproducible en un sistema Debian normal, si el fallo aparece en esta etapa.

Si este es el caso, esto no es un error en el sistema en vivo, sino de Debian - se debe informar sobre el paquete que falla. Se puede obtener ms informacin ejecutando debootstrap de forma separada del sistema de creacin en vivo o ejecutando lb bootstrap -debug.

Adems, si se est utilizando una rplica local y/o cualquier tipo de proxy y se experimenta un problema, se debe intentar reproducir siempre preinstalando desde una rplica oficial.

14.7.3 En tiempo de arranque

Si la imagen no arranca, se debera informar a la lista de correo, junto con la informacin solicitada en **Recopilar informacin**. No hay que olvidar mencionar, cmo y cundo la imagen falla, si es utilizando virtualizacin o hardware real. Si se est utilizando una tecnologa de virtualizacin de cualquier tipo, se debe probar la imagen en hardware real antes de informar de un error. Proporcionar una captura de pantalla del error tambin es muy til.

14.7.4 En tiempo de ejecucin

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

14.8 Dnde informar de los fallos

El Debian Live Project realiza un seguimiento de todos los errores en el sistema de seguimiento de errores de Debian (BTS). Para obtener informacin sobre cmo utilizar el sistema, consultar <https://bugs.debian.org/>. Tambin

se puede enviar los errores mediante el comando `reportbug` del paquete con el mismo nombre.

748

Hay que tener en cuenta que los errores que se encuentran en las distribuciones derivadas de Debian (como Ubuntu y otras) no deben enviarse al BTS de Debian a menos que también se puedan reproducir en un sistema Debian usando paquetes oficiales de Debian.

Estilo de cdigo

15. Estilo de cdigo

En este captulo se documenta el estilo de cdigo utilizado en los sistemas en vivo.

15.1 Compatibilidad

- 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.
- Se puede comprobar las secuencias de comandos con ‘sh -n’ y ‘checkbashisms’.
- Asegurarse de que el cdigo funcione con ‘set -e’.

15.2 Sangrado

- Utilizar siempre los tabuladores en lugar de espacios.
- Keep case branch terminators (;;) aligned with the content of the branch, rather than the branch entry.

Bien:

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

```
        bar)
            foobar
            ;;
esac
```

15.3 Ajuste de lineas

- 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 [ "$-MY`LOCATION`VARIABLE" = "something" ] && [ -e "$-↵
MY`OUTPUT`FILE" ]; then
    MY`OTHER`VARIABLE="$(some`bin $-FOOBAR" — awk -F' ' - ↵
    print $1 "`)"
fi
```

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

Horrible:

```
if [ "$-MY`LOCATION`VARIABLE" = "something" ] && [ -e "$-↵
MY`OUTPUT`FILE" ] — [ "$-MY`LOCATION`VARIABLE" = "something↵
-else" ] && [ -e "$-MY`OUTPUT`FILE`2" ]; then
    MY`OTHER`VARIABLE="$(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:

Bien:

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

Bad (inconsistent with existing style):

15.4 Variables

Las variables deben escribirse siempre en letras mayusculas.

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

Local function variables should be restricted to local scope.

Las variables en relacin a un parmetro de arranque en live-config comienzan con LIVE`.

Todas las dems variables de live-config comienzan con el prefijo `.

Utilizar llaves para las variables, por ejemplo, escribir "\$-FOO" en lugar de \$FOO.

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

Por motivos de coherencia, se debe utilizar siempre comillas en la asignacin de valores a las variables:

Mal:

```
FOO=bar
```

Bien:

```
FOO="bar"
```

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

Typically bad:

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

Bien:

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

15.5 Miscelnea

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

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; ...`

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`).

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

Ejemplos

Ejemplos

16. Ejemplos

Este capítulo ofrece ejemplos de creación de imágenes de sistemas en vivo para casos de uso específicos. Si se es nuevo en la creación de una imagen en vivo propia, se recomienda leer primero los tres tutoriales en secuencia, ya que cada uno enseña nuevas técnicas que ayudan a utilizar y entender los ejemplos restantes.

16.1 Uso de los ejemplos

Para poder seguir estos ejemplos es necesario un sistema donde crearlos que cumpla con los requisitos enumerados en [Requisitos](#) y tener live-build instalado tal y como se describe en [Instalación de live-build](#).

Hay que tener en cuenta que, para abreviar, en estos ejemplos no se especifica una réplica local para la creación de la imagen. Es posible acelerar las descargas considerablemente si se utiliza una réplica local. Se puede especificar las opciones cuando se usa `lb config`, tal y como se describe en [Réplicas de Distribution utilizadas durante la creación](#), o para más comodidad, establecer el valor por defecto para la creación del sistema en `/etc/live/build.conf`. Basta con crear este fichero y en el mismo, establecer las variables `LB*MIRROR*` correspondientes a la réplica preferida. Todas las demás réplicas usadas en el proceso de creación usarán estos valores por defecto. Por ejemplo:

```
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 imagen predeterminada

Caso práctico: Crear una primera imagen sencilla, aprendiendo los fundamentos de live-build.

En este tutorial, vamos a construir una imagen ISO híbrida por defecto que contenga únicamente los paquetes base (sin Xorg) y algunos paquetes de soporte, como un primer ejercicio en el uso de live-build.

No puede ser más fácil que esto:

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

Si se examina el contenido del directorio `config/` se verá almacenada allí una configuración en esqueleto preparada para ser personalizada o en este caso para ser usada inmediatamente para construir una imagen por defecto.

Ahora, como superusuario, crear la imagen, guardando un log con `tee` mientras se crea.

```
# lb build 2i&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 utilidad de navegador web

Caso práctico: Crear una utilidad de navegador web, aprendiendo a aplicar personalizaciones.

En este tutorial, se creará una imagen adecuada para su uso como utilidad de navegador web, esto sirve como introducción a la personalización de las imágenes de sistemas en vivo.

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

La elección de LXDE para este ejemplo refleja el deseo de ofrecer un entorno de escritorio mínimo, ya que el enfoque de la imagen es el uso individual que se tiene en mente, el navegador web. Se podrá ir a más lejos y ofrecer una configuración por defecto para el navegador web en `config/includes.chroot/etc/iceweasel/profile/`, o paquetes adicionales de soporte para la visualización de diversos tipos de contenido web, pero se deja esto como un ejercicio para el lector.

Crear la imagen, de nuevo como superusuario, guardando un log como en el [Tutorial 1](#):

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

De nuevo, verificar que la imagen está bien y probarla igual que en el [Tutorial 1](#).

16.4 Tutorial 3: Una imagen personalizada

Caso práctico: Crear un proyecto para conseguir una imagen personal-

izada, que contenga el software favorito para llevarse en una memoria USB donde quiera que se vaya, y hacerlo evolucionar en revisiones sucesivas, tal y como vayan cambiando las necesidades y preferencias.

Como nuestra imagen personalizada irá cambiando durante un número de revisiones, si se quiere ir siguiendo esos cambios, probar nuevas cosas de forma experimental y posiblemente volver atrás si no salen bien, se guardará la configuración en el popular sistema de control de versiones git. También se utilizarán las mejores prácticas de configuración automática a través de scripts auto como se describe en [Gestionar una configuración](#).

16.4.1 Primera revisión

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

Editar `auto/config` del siguiente modo:

```
#!/bin/sh

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

Ejecutar `lb config` para generar el árbol de configuración, utilizando el script `auto/config` que justo se acaba de crear:

```
$ lb config
```

Completar la lista de paquetes local:

```
$ 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`.

Ahora, crear la imagen:

```
# lb build
```

Tener en cuenta que a diferencia de los dos primeros tutoriales, ya no se tiene que escribir `2i&1` `--tee build.log` ya que esto se incluye ahora en `auto/build`.

Una vez que se ha probado la imagen (como en el [Tutorial 1](#)) y se ha asegurado de que funciona, es el momento de iniciar el repositorio git, aadiendo slo los scripts `auto` que se acaba de crear, y luego hacer el primer commit:

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

16.4.2 Segunda revisin

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

El comando `lb clean` limpiar todos los ficheros generados en las primeras

creaciones a excepcin del cach, lo cual ahorra tener que volver a descargar de nuevo los paquetes. Esto asegura que el siguiente `lb build` vuelva a ejecutar todas las fases para regenerar los ficheros de nuestra nueva configuracin.

```
# 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
```

Crear de nuevo:

```
# lb build
```

Probar, y cuando se est satisfecho, enviar la prxima revisin al git:

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

Por supuesto, es posible hacer cambios ms complicados en la configuracin, tal vez aadiendo ficheros en los subdirectorios de `config/`. Cuando se envian nuevas revisiones, hay que tener cuidado de no editar a mano o enviar los ficheros del nivel superior en `config` que contienen variables `LB*` ya que estos son productos de creacin tambin y son siempre limpiados por `lb clean` y recreados con `lb config` a travs de sus respectivos scripts `auto`.

Hemos llegado al final de nuestra serie de tutoriales. Si bien son posibles muchos ms tipos de personalizacin, aunque slo sea con las pocas caractersticas explicadas en estos sencillos ejemplos, se puede crear una variedad casi infinita de imgenes diferentes. Los ejemplos que quedan en esta seccin abarcan varios casos de usos diferentes procedentes de las experiencias recogidas de los usuarios de sistemas en vivo.

16.5 Un cliente VNC kiosk

Caso Prctico: Crear una imagen con live-build para que se conecte directamente a un servidor VNC al arrancar.

Crear un directorio de construccin y lanzar una configuracin de esqueleto en su interior, desactivando recommends para conseguir un sistema mnimo. Y a continuacin, crear dos listas iniciales de paquetes: La primera generada con un script proporcionado por live-build llamado Packages (ver [Generar listas de paquetes](#)), y la segunda lista una que incluya xorg, gdm3, metacity y 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
```

Como se explica en [Ajuste de APT para ahorrar espacio](#) puede ser necesario volver a agregar algunos paquetes recomendados para que la imagen funcione correctamente.

Una manera fcil de conocer todos los recommends es utilizar apt-cache. Por ejemplo:

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

En este ejemplo, descubrimos que tenemos que volver a incluir varios paquetes recomendados por live-config y live-boot: user-setup para hacer funcionar el inicio automtico de sesin y sudo programa esencial para apagar el sistema. Adems, podra ser til aadir live-tools para poder copiar la imagen en la memoria RAM y eject para finalmente poder expulsar el medio en vivo. Por eso:

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

Despus, crear el directorio /etc/skel en config/includes.chroot y poner dentro un fichero .xsession personalizado para el usuario que por defecto ejecutar metacity e iniciar el xvncviewer, conectndo al puerto 5901 de un servidor en 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
```

Crear la imagen:

```
# lb build
```

Disfrutarlo.

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
```

Para hacer que la imagen funcione correctamente, tenemos que volver a añadir, al menos, dos paquetes recomendados, que son excluidos por la opción `--apt-recommends false`. Ver [Ajuste de APT para ahorrar espacio](#)

```
$ 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
```

Ahora, crear la imagen de 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.

Dejar fuera los índices de APT con `--apt-indices false` ahorra una cantidad importante de espacio, la desventaja es que es necesario hacer un `apt-get update` antes de usar `apt` en el sistema en vivo. Excluyendo los paquetes recomendados con `--apt-recommends false` se ahorra un poco de espacio adicional a costa de omitir algunos paquetes que de otro modo podría esperarse que estuvieran allí. `--debootstrap-options --variant=minbase` preinstala un sistema mínimo desde el principio. El hecho de no incluir automáticamente paquetes de firmware con `--firmware-chroot false` también ahorra un poco de espacio. Y por último, `--memtest none` evita la instalación de un comprobador de memoria.

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 escritorio GNOME con variante local e instalador

Caso práctico: Crear una imagen que contenga el escritorio gráfico GNOME, la variante local Suiza y un instalador.

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 primer problema es descubrir los nombres de las tareas adecuadas. En la actualidad, live-build no puede ayudar en esto. Aunque podríamos tener suerte y encontrarlos a base de pruebas, hay una herramienta, `grep-dctrl`, para extraerlos de las descripciones de tareas en `tasksel-data`, para proceder, asegurarse de tener ambas cosas:

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

Ahora podemos buscar las tareas apropiadas, primero con:

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

Con este comando, se descubre que la tarea se llama, sencillamente, `german`. Ahora, para encontrar las tareas relacionadas:

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

En el momento del arranque se va a generar la variante local `de`CH.UTF-8` y seleccionar la distribución del teclado `ch`. Ahora vamos a poner las

piezas juntas. Recordando de [Utilizar metapaquetes](#) que los metapaquetes tienen el prefijo `task-`, especificamos estos parámetros del lenguaje en el arranque y a continuación añadimos los paquetes de prioridad estándar y los metapaquetes que hemos descubierto a la lista de paquetes de la siguiente 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.

Apndice

895	Style guide	906	Esperamos que las diferentes variedades puedan mezclarse sin crear malentendidos, pero en trminos generales se debe tratar de ser coherente y antes de decidir sobre el uso de las variantes britnica o americana, o cualquier otro tipo de ingls a su discrecin, por favor, dar un vistazo a cmo escriben otras personas y tratar de imitarlas.	
896	17. Gua de estilo			
897	17.1 Instrucciones para los autores		Ser equilibrado	907
898	Esta seccin se ocupa de algunas consideraciones generales a tener en cuenta al escribir documentacin tcnica para live-manual. Se dividen en aspectos lingsticos y procedimientos recomendados.		Hay que ser imparcial. Evitar incluir referencias a ideologas totalmente ajenas a live-manual. La escritura tcnica debe ser lo ms neutral posible. Est en la naturaleza misma de la escritura cientfica.	908
899	Nota: Los autores deberan leer primero contribuir a este documento		Ser polticamente correcto	909
900	17.1.1 Aspectos lingsticos		Evitar el lenguaje sexista tanto como sea posible. Si se necesita hacer referencia a la tercera persona del singular, utilizar preferiblemente they en lugar de he o she o inventos extraos como s/he o s(he).	910
901	Utilizar un ingls llano		Ser concisos	911
902	Tener en cuenta que un alto porcentaje de lectores no son hablantes nativos de ingls. As que, como regla general, se debe utilizar frases cortas y significativas, seguidas de un punto y aparte.		Ir directamente al grano y no dar vueltas a las cosas. Dar toda la informacin necesaria, pero no dar ms informacin de la necesaria, es decir, no explicar detalles innecesarios. Los lectores son inteligentes. Se presume algn conocimiento previo de su parte.	912
903	Esto no significa que se tenga que utilizar un estilo simplista. Es una sugerencia para tratar de evitar, en la medida de lo posible, las oraciones subordinadas complejas que hacen que el texto sea difcil de entender para los hablantes no nativos de ingls.		Minimizar la labor de traduccin	913
904	Variedad de ingls		Tener en cuenta que cualquier cosa que se escriba tendr que ser traducido a otros idiomas. Esto implica que un nmero de personas tendrn que hacer un trabajo extra si se agrega informacin innecesaria o redundante.	914
905	Las variedades ms extendidas del ingls son la britnica y la americana, as que es muy probable que la mayora de los autores utilicen una u otra. En un entorno de colaboracin, la variedad ideal sera el ingls internacional, pero es muy difcil, por no decir imposible, decidir qu variedad entre todas las existentes, es la mejor.		Ser coherente	915
			Como se ha sugerido anteriormente, es casi imposible estandarizar un documento creado en colaboracin en un todo perfectamente unificado. Sin embargo, se aprecia todo esfuerzo por escribir de manera coherente con el resto de los autores.	916
			Cohesin	917

918	Utilizar conectores de discurso para conseguir un texto coherente y sin ambigüedades. (Normalmente se llaman connectors).		
919	Ser descriptivo		
920	Es preferible describir el asunto en uno o varios párrafos que la mera utilización de una serie de oraciones en un típico estilo de changelog. Hay que describirlo! Los lectores lo agradecerán.		
921	Diccionario		
922	Buscar el significado de las palabras en un diccionario o una enciclopedia si no se sabe cómo expresar ciertos conceptos en inglés. Pero hay que tener en cuenta que un diccionario puede ser el mejor amigo o puede convertirse en el peor enemigo si no se utiliza correctamente.		
923	El inglés tiene el mayor vocabulario que existe (con más de un millón de palabras). Muchas de estas palabras son préstamos de otras lenguas. Al buscar el significado de las palabras en un diccionario bilingüe la tendencia de un hablante no nativo de inglés es elegir las que suenan más similares en su lengua materna. A menudo, esto se convierte en un discurso excesivamente formal que no suena muy natural en inglés.		
924	Como regla general, si un concepto se puede expresar utilizando diferentes sinónimos, es un buen consejo elegir la primera palabra propuesta por el diccionario. En caso de duda, la elección de palabras de origen germánico (Normalmente palabras monoslabas) es a menudo lo correcto. Tener en cuenta que estas dos técnicas pueden producir un discurso más bien informal, pero al menos la elección de palabras será de amplio uso y aceptación general.		
925	Se recomienda el uso de un diccionario de frases hechas. Son muy útiles cuando se trata de saber qué palabras suelen aparecer juntas.		
926	Una vez más, es una buena práctica aprender del trabajo de los otros. El uso de un motor de búsqueda para comprobar cómo otros autores utilizan ciertas expresiones puede ayudar mucho.		
	Falsos amigos, modismos y otras expresiones idiomáticas		927
	Cuidado con los falsos amigos. No importa lo competente que se sea en un idioma extranjero, se puede caer de vez en cuando en la trampa de los llamados falsos amigos, palabras que se parecen en dos idiomas pero cuyos significados o usos pueden ser completamente diferentes.		928
	Intentar evitar los modismos tanto como sea posible. Los modismos son expresiones que tienen un significado completamente diferente de lo que sus palabras parecen decir por separado. A veces, los modismos pueden resultar difíciles de entender incluso para los hablantes nativos de inglés!		929
	Evitar el argot, las abreviaturas, las contracciones...		930
	A pesar de que se anime a utilizar un inglés sencillo, del da a da, la escritura técnica pertenece al registro formal de la lengua.		931
	Intentar evitar el argot, las abreviaturas inusuales que son difíciles de entender y por encima de todo, las contracciones que tratan de imitar el lenguaje hablado. Por no hablar de las expresiones familiares o típicas del IRC.		932
	17.1.2 Procedimientos		933
	Probar antes de escribir		934
	Es importante que los autores prueben sus ejemplos antes de añadirlos a live-manual para asegurarse de que todo funciona como se describe. Hacer las pruebas en un entorno chroot limpio o una máquina virtual puede ser un buen punto de partida. Además, será ideal si las pruebas fueran llevadas a cabo en diferentes máquinas con un hardware diferente para detectar los posibles problemas que puedan surgir.		935
	Ejemplos		936

	Cuando se pone un ejemplo hay que tratar de ser lo ms especifico posible. Un ejemplo es, despus de todo, tan slo un ejemplo.	937	Escribir un primer borrador, revisar, editar, mejorar, rehacer de nuevo si es necesario	946
938	A menudo es mejor utilizar un ejemplo que slo se aplica a un caso concreto que el uso de abstracciones que puedan confundir a los lectores. En este caso se puede dar una breve explicacin de los efectos del ejemplo propuesto.		- Lluvia de ideas!. Es necesario organizar las ideas primero en una secuencia lgica de eventos.	947
			- Una vez que, de alguna manera, se han organizado esas ideas en la mente, escribir un primer borrador.	948
939	Puede haber algunas excepciones cuando el ejemplo sugiera el uso de comandos potencialmente peligrosos que, si se utilizan incorrectamente, pueden provocar la prdida de datos u otros efectos indeseables similares. En este caso se debe proporcionar una explicacin detallada acerca de los posibles efectos secundarios.		- Revisar la gramtica, la sintaxis y la ortografa. Tener en cuenta que los nombres propios de las versiones, tales como trixie o sid , no se deben escribir en maycula cuando se refieren a los nombres en clave. Para comprobar la ortografa se puede ejecutar el target spell. Es decir, make spell	949
940	Enlaces externos		- Mejorar las frases y rehacer cualquier parte si es necesario.	950
941	Los enlaces a sitios externos slo se deben utilizar cuando la informacin en esos sitios es crucial para comprender un punto concreto. A pesar de eso, tratar de utilizar enlaces a sitios externos lo menos posible. Los enlaces de Internet pueden cambiar de vez en cuando, dando lugar a enlaces rotos y dejando los argumentos en un estado incompleto.		Captulos	951
			Utilizar el sistema de numeracin convencional de los captulos y subtítulos. Por ejemplo 1, 1.1, 1.1.1, 1.1.2 ... 1.2, 1.2.1, 1.2.2 ... 2, 2.1 ... y as sucesivamente. Ver marcado a continuacin.	952
942	Adems, las personas que leen el manual sin conexin no tendrán la oportunidad de seguir los enlaces.		Si es necesario enumerar una serie de pasos o etapas en la descripcin, también se puede utilizar los números ordinales: primero, segundo, tercero ... o en primer lugar, entonces, despus, por fin ... Alternativamente, se pueden utilizar puntos.	953
943	Evitar las marcas y las cosas que violan la licencia bajo la que se publica el manual		Marcado	954
944	Tratar de evitar las marcas tanto como sea posible. Tener en cuenta que otros proyectos derivados pueden hacer uso de la documentacin que escribimos. As que estamos complicando las cosas para ellos si se aade determinado material especifico.		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:	955
945	live-manual se publica bajo la GNU GPL. Esto tiene una serie de implicaciones que se aplican a la distribucin de los materiales (de cualquier tipo, incluyendo grficos o logos con derechos de autor) que se publican en l.		<pre>\$ sisu --help markup</pre>	956
			Estos son algunos ejemplos de marcado que pueden ser tiles:	957

958 - Para el nfasis/negrita:

959

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

960 producen: foo o foo . Se usan para enfatizar ciertas palabras clave.

961 - Para la cursiva:

962

```
/-foo"/
```

963 produce: foo. Se usa, por ejemplo, para los nombres de paquetes De-
bian.

964 - Para monospace:

965

```
#-foo"#
```

966 produce: foo. Se usa, por ejemplo, para los nombres de los comandos. Y
tambin para resaltar algunas palabras clave o cosas como las rutas.

967 - Para los bloques de cdigo:

968

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

969 produce:

970

```
$ foo  
# bar
```

Se utiliza code- para abrir y "code para cerrar los bloques. Es importante 971
recordar dejar un espacio al principio de cada lnea de cdigo.

17.2 Directrices para los traductores 972

Esta seccin se ocupa de algunas consideraciones generales a tener en 973
cuenta al traducir el contenido de live-manual.

Como recomendacin general, los traductores deberan haber ledo y enten- 974
dido las reglas de traduccin que se aplican a sus lenguas especificas. Por
lo general, los grupos de traduccin y las listas de correo proporcionan
informacin sobre cmo hacer traducciones que cumplan con los estndares
de calidad de Debian.

Nota: Los traductores tambn deberan leer **Cmo contribuir a este docu-** 975
mento. En particular, la seccin **Traduccin**

17.2.1 Consejos de traduccin 976

Comentarios 977

El papel del traductor es transmitir lo ms fielmente posible el significado 978
de las palabras, oraciones, prrafos y textos de como fueron escritos por
los autores originales a su idioma.

As que ellos deben abstenerse de aadir comentarios personales o infor- 979
maciones adicionales por su cuenta. Si se desea agregar un comentario
para los traductores que trabajan en los mismos documentos, se pueden
dejar en el espacio reservado para ello. Es decir, el encabezado de las
cadenas de los ficheros po precedidos por el signo # . La mayora de los
programas grficos de traduccin pueden manejar ese tipo de comentarios
automticamente.

NT, Nota del Traductor 980

Es perfectamente aceptable, sin embargo, incluir una palabra o una expresin entre parntesis en el texto traducido si, y slo si, hace que el significado de una palabra o expresin difcil sea ms clara para el lector. Dentro de los parntesis, el traductor debe poner de manifiesto que la adicin es suya utilizando la abreviatura NT o Nota del traductor.

982 Frases impersonales

983 Los documentos escritos en Ingls utilizan muchsimo la forma impersonal you. En algunos otros idiomas que no comparten esta caracterstica, esto podra dar la falsa impresin de que los textos originales se dirigen directamente el lector cuando en realidad no lo hacen. Los traductores deben ser conscientes de este hecho y reflejarlo en su idioma con la mayor precisin posible.

984 Falsos amigos

985 La trampa de los falsos amigos explicada anteriormente se aplica especialmente a los traductores. Volver a comprobar el significado de los falsos amigos sospechosos en caso de duda.

986 Marcado

987 Los traductores que trabajen inicialmente con los ficheros pot y posteriormente con los ficheros po encontrarn muchas caractersticas de marcado en las cadenas. Se puede traducir el texto, siempre que sea traducible, pero es extremadamente importante que se utilice exactamente el mismo marcado que la versin original en ingls.

988 Bloques de cdigo

989 A pesar de que los bloques de cdigo son generalmente intraducibles, incluirlos en la traduccin es la nica manera de conseguir una traduccin completa al 100%. Y aunque eso signifique ms trabajo al principio, ya que puede ser necesaria la intervencin de los traductores cuando hay cambios en el cdigo, es la mejor manera, a la larga, de identificar lo que se ha traducido y lo que no al comprobar la integridad de los ficheros .po.

981 Saltos de lnea

Los textos traducidos deben tener exactamente los mismos saltos de lnea que los textos originales. Tener cuidado de presionar la tecla Enter o escribir si aparece en los ficheros originales. Estas nuevas lneas aparecen a menudo, por ejemplo, en los bloques de cdigo.

No hay que confundirse, esto no significa que el texto traducido tenga que tener la misma longitud que la versin en ingls. Eso es prcticamente imposible.

Cadenas intraducibles

Los traductores nunca deben traducir:

- Los nombres en clave de las versiones (que debe ser escrito en minsculas)

- Los nombres de los programas

- Los comandos que se ponen como ejemplos

- Los metadatos (aparecen a menudo entre dos puntos :metadata:)

- Los enlaces

- Las rutas

SiSU Metadata, document information

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

Ttulo: Debian Live Manual

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

Derechos: 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.

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

Fecha: 2025-02-26

Version Information

Fichero fuente: 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)=d9a70b544267eeec1255721fa50c3a62-7149fd9d7000338439fd0afc838a059b

Generated

ltima generacin (metaverse) del documento: 2025-02-26 23:57:53 +0000

Generado por: SiSU 7.3.0 of 2023w44/1 (2023-10-30)