Conference Paper

Marco de trabajo para construir el modo vivo en distribuciones From Scratch

Huriviades Calderón-Gómez and Miguel Vargas-Lombardo

Universidad Tecnológica de Panamá, GISES, Panamá, Panamá

Abstract

The main objective of this research is to develop a framework to build automated scripts that generate the Live mode in the Distributions from Scratch. Special emphasis is placed on the methodology of the initramfs and the use of GNU packages to adequate the scripts to the needs of developers. Moreover, the purpose of this, is to demonstrate the stages involved in developing the scripts and the benefits of running it, based on the results of this research.

Keywords: Live-USB Customization, Linux From Scratch, GNU/Linux Custom Distributions, Initramfs, BusyBox

Corresponding Author:
Huriviades Calderón-Gómez
huriviades.calderon@utp.ac.pa

Received: 15 November 2017
Accepted: 5 January 2018
Published: 4 February 2018

Publishing services provided by Knowledge E

Selection and Peer-review under the responsibility of the ESTEC Conference Committee.

Resumen

El principal objetivo de esta investigación es desarrollar un marco de trabajo para construir scripts automáticos que generen el modo vivo en las distribuciones From Scratch. Se hace especial énfasis en la metodología de Initramfs y utilizar paquetes GNU para adecuar los scripts a las necesidades de los desarrolladores. Además, el propósito es demostrar las etapas que conlleva desarrollar los scripts y las ventajas de ejecutarlo, basado en los resultados de esta investigación.

Palabras claves: Modo vivo-USB personalizable, Linux From Scratch, Distribuciones GNU/Linux Personalizadas, Initramfs, BusyBox

1. Introducción

La documentación del proyecto Linux From Scratch (LFS) es un área de gran interés para los desarrolladores, debido a que provee las instrucciones generales de cómo construir sus propias distribuciones GNU/Linux personalizando el código fuente, según sus necesidades (Beekmans et al. 2017a).

Antes de entrar en consideración, es necesario mencionar algunos trabajos relacionados orientado a la construcción de distribuciones personalizadas como: FLUKA-LIVE (Cohen et al. 2008), Image customized through Linux from scratch on cloud platform (Xu *et al.* 2014), SpicaxOS (Calderón et al. 2015) y IXCHEL (Molina et al. 2016). Cada trabajo relacionado explica desde su punto de vista, las diversas metodologías para desarrollar una distribución GNU/Linux para diferentes propósitos (servidores para nubes, ejecución de computación paralela, simulación y entre otras). Dentro de ese marco, nos centramos en cómo generar y ejecutar el modo vivo provenientes de las distribuciones creadas por estas metodologías, para así evitar el empotramiento y limitantes de ejecución de la misma.

Actualmente, las versiones más recientes (7.x o superior) de la documentación, se han optimizado los procesos de configuración, compilación y ensamblaje en los sistemas construidos modernos en comparación a sus anteriores versiones (Beekmans et al. 2017b); sin embargo, la comunidad LFS cesó el soporte al modo vivo desde la versión 7.x, en consecuencia se pierde el esquema bajo un entorno ideal para inicializar el sistema independientemente del equipo ejecutado (Huntwork et al. 2012).

La presente investigación surge de la necesidad de construir un script para el modo vivo, acorde a los Linux Kernel y paquetes GNU más reciente, si bien es cierto que existen script desarrollados por la comunidad LFS para elaborar el modo vivo del sistema a construir, por ejemplo, el código "Beginners Guide To Creating A Live CD With LFS 6.0" (Hernandez, 2005), pero la mayoría de los scripts se encuentran obsoletos o incompatibles con las versiones más recientes de Núcleos de Sistemas Operativos Abiertos y traen como resultados problemas como: agujero de seguridad, corrupción de los enlaces simbólico, sobrecarga u otro evento desfavorable para los sistemas From Scratch (TracReports: Linux From Scratch, 2007a, 2009b, 2010c).

Por otra parte, la falta del soporte ha permanecido como un problema en los sistemas construidos no bifurcados, en comparación con las distribución GNU/Linux principales (Debian, Red Hat, SUSE y entre otras) (Universi *et al.* 2006), dado que cuentan con software prediseñado (Anaconda – Red Hat y Casper – Ubuntu) para construir el modo vivo de sus derivados (Fedora, 2017; Ubuntu, 2016). Por ello se hacen necesario, realizar un marco de trabajo preliminar basado en el sistema initramfs y BusyBox para construir el modo vivo para las distribuciones personalizadas (Gentoo Linux, 2016; Minnich et al. 2015).

2. Discusión y Resultados

Para evaluar la implementación del modo vivo personalizado, se utilizaron los métodos: el torito, debootstrap y anaconda modificados (Negus, 2006); sin embargo, se encontraron varios errores de incompatibilidad con los ficheros del sistema y los gestores de arranque (Grub, Syslinux y LILO). Algunos de los errores más significativos son:

- Bucle infinito al inicio del sistema.
- Corrupción de la raíz (root).
- Incompatibilidad del initramfs generado de la misma distribución.
- Discrepancia para montar la imagen en un dispositivo de almacenamiento externo con formato ext4 bajo la opción 64Bit.

El resultado que emerge de los anteriores métodos es desarrollar una serie de scripts personalizables escrito en Bash, C/C++ o Python propio del sistema para iniciarse a si mismo. Por ello, se eligió el método de initramfs para montar el init de la imagen al binario de BusyBox (Linux Minimal Precompilado) y realizar el intercambio del root simulado "BusyBox" hacia el root del modo vivo.

Cabe destacar que las características de este método son: portabilidad, adaptabilidad e independencia de preinstalaciones, para posteriormente ejecutarse en otras computadoras; sin embargo, se debe tener habilitados los módulos y firmwares del kernel para dichos equipos (Gentoo Linux, 2017). En este sentido resulta beneficioso para el propósito educativo e investigativo de las ingenierías en TIC desarrollar sus propias variantes.

Las pruebas realizadas al script, escrito en Bash, determino la eficiencia del modo vivo personalizado a través de la métrica del tiempo de arranque. De acuerdo con las distribuciones principales en su versión minimal, el tiempo promedio de ejecución es de 9~12 segundos aproximadamente con una computadora moderna (Diamond et al. 2015). A continuación, en la Tabla 1 se muestran los resultados obtenidos al ejecutar las imágenes.

Para la realización de este muestreo, se utilizó una computadora con procesador Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz, memoria RAM de 8192MB, 1 HDD de 1TB, 2 USB 3.0/2.0 y ambas imágenes en sus versiones Linux Minimal (sin entorno gráfico). Además, se determinó un rango de aceptación de 8.0 ~12 segundos, ambas distribuciones se encuentra dentro del rango de aceptación.

Tabla 1: Resultado de las muestras

Gestor de arranque	Memoria de almacenamiento	Tiempo promedio de ejecución en segundos (Sistema From Scratch)	Tiempo promedio de ejecución en segundos (Knoppix)
Grub	Memoria 3.0 USB 16GB	9.60	9.00
	Memoria 2.0 USB 16GB	11.15	10.80
Syslinux	Memoria 3.0 USB 16GB	8.60	8.20
	Memoria 2.0 USB 16GB	10.50	10.10

De igual manera, se realizaron varias pruebas de ejecución del modo vivo en una distribución From Scratch desarrollada, se tomó como referencia a SpicaxOS (Calderón et al. 2015). A continuación, en la Figura 1 se muestra el montaje del modo vivo de una imagen personalizada bajo el marco de trabajo desarrollado.

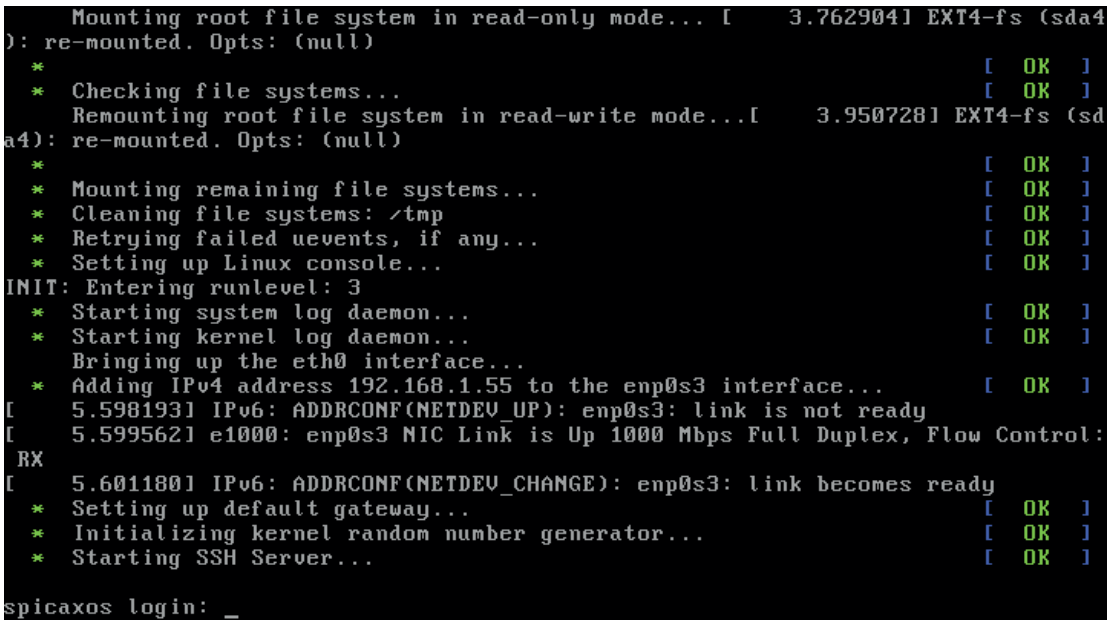

Figura 1: Arranque de un Sistema From Scratch desde la memoria USB

Un hallazgo importante obtenido fue que la primera versión del script se enfoca principalmente en la construcción del modo vivo-USB, todavía resta brindar soporte al formato ISO 9660 (CD). No obstante, este problema se puede superar a través del uso de SquashFS e IsoHybrid.

3. Materiales y métodos

3.1. Implementación

Este proyecto consiste en dos scripts escrito en Bash (A y B), ofreciendo un entorno necesario para la instalación del modo vivo desde una memoria USB.

Para la realización del primer script (**A**) se aplicó el concepto de Chroot "cambio de raíz", cuya función es intercambiar el fichero de raíz y sus procesos actuales en ejecución del sistema anfitrión a un directorio determinado de otro sistema huésped (van Tilborg et al. 2011). Este cambio, prepara el entorno ideal y separado para la ejecución del segundo script (**B**). Como se puede apreciar en la Figura 2, se muestra un fragmento del código fuente del script A.

```
echo "¿Ya hizo la copia de los archivos del ISO al disco seleccionado?"
read -p "Elija una opción [y/n]: " OPb
if [ "$OPb" = "y" ] || [ "$OPb" = "Y" ]
then
  mount -v -t "${FRM_DISC}" "${DISC}" $SPICAX
  mount -v --bind /dev $SPICAX/dev
  mount -vt devpts devpts $SPICAX/dev/pts -o gid=5,mode=620
  mount -vt proc proc $SPICAX/proc
  mount -vt sysfs sysfs $SPICAX/sys
  mount -vt tmpfs tmpfs $SPICAX/run
  if [ -h $SPICAX/dev/shm ]; then
      mkdir -pv $SPICAX/$(readlink $SPICAX/dev/shm)
  fi
  echo "Entrando a modo CHROOT..."
  chroot "$SPICAX" /usr/bin/env -i              \
  HOME=/root TERM="$TERM" PS1='\u:\w\$ ' \
  PATH=/bin:/usr/bin:/sbin:/usr/sbin     \
  /bin/bash --login
```

Figura 2: Fragmento de Chroot en el script A

Como seguimiento de esta actividad, el segundo script (B) se encargará de las siguientes tareas dentro del ambiente Chroot.

- Crear un initramfs del sistema con busybox estático.
- Incluir los directorios bin, mnt, root, proc y sys dentro del initramfs.
- Crear el script iniciador (init) para el intercambio del punto de montaje de la raíz.
- Incluir el soporte a hotplug.
- Crear un Master Boot Record (MBR) a la memoria USB.
- Habilitar el uso de etiqueta de la memoria USB
- Sobrescribir el archivo FSTAB del sistema huésped con la etiqueta dinámica.
- Configurar e instalar un gestor de arranque (Grub o Syslinux) a la memoria USB.

Como se puede apreciar en la Figura 3, se muestra un fragmento del código fuente del script B.

```
            chmod +x /sources/initrd/init
            cd /sources/initrd
            find . -print0 | cpio --null -ov --format=newc | gzip -9 > /boot/initrd-4.9.9.cpio.gz
            echo " AVISO: El siguiente comando sobrescribirá el gestor de arranque actual. No ejecute el comando si esta no es la deseada
            read -p "Elija una opción [y/n]: " OPa
            if [ "$OPa" = "y" ] || [ "$OPa" = "Y" ]
            then
                e2label "${DISC}" spicax
                mkdir "${SYSLINUX_PATH}"
                extlinux --install "${SYSLINUX_PATH}"
                cd "${SEKAI}"
                wget -c --quiet --show-progress --progress=bar --no-check-certificate -t 5 -T 10 "${SYSLINUX}"
                cp "${SYSLINUX_BUILD}" "${SYSLINUX_PATH}"
                cd "${SEKAI}"
                cat > /etc/fstab << "EOF"
# Begin /etc/fstab

# file system  mount-point  type     options             dump  fsck
#                                                              order

LABEL=spicax   /           auto     defaults            1     1
proc           /proc       proc     nosuid,noexec,nodev 0     0
sysfs          /sys        sysfs    nosuid,noexec,nodev 0     0
devpts         /dev/pts    devpts   gid=5,mode=620      0     0
tmpfs          /run        tmpfs    defaults            0     0
devtmpfs       /dev        devtmpfs mode=0755,nosuid    0     0

# End /etc/fstab
EOF
```

Figura 3: Fragmento de Chroot en el script B

3.2. Instalación

Los scripts A y B son capaces de iniciar el sistema desde una memoria USB o instalarse en un disco duro. En primer lugar, se puede iniciar el sistema sin necesidad de hacer cambios en el sistema operativo subyacente. Segundo, los scripts ha sido probados en diversos equipos modernos de 64-bit PC. Cabe señalar, que ambos scripts son de licenciamiento MIT para los sistemas operativos GNU/Linux y se encuentran disponible en el repositorio BitBucket de los autores.

En cuanto a cómo ejecutar los scripts en una distribución GNU/Linux, se encuentra descrito detalladamente en la Tabla 2.

3.3. Paquetes de software

En la tabla 3 se incluirán los paquetes de softwares fundamentales para el funcionamiento de los scripts A y B; no utilizar los scripts si el sistema From Scratch está en **fase de construcción temporal**.

Tabla 2: Comandos de los scripts A & B

Script	Funciones	Parámetros		Información	Ejemplo
A	Mount	Nombre del disco		Crea el Chroot entre los sistemas huésped y anfitrión	../live_usb_A.sh mount sda4 imagen.img
		ruta de la imagen			
B	Install	grub	nombre del disco	Instala el modo Live USB a la memoria USB con grub	../live_usb_B.sh install grub sda4
		syslinux	nombre del disco	Instala el modo Live USB a la memoria USB con syslinux	../live_usb_B.sh install syslinux sda4
	Set	nombre del disco		Desmonta el Chroot de la memoria USB	../live_usb_B.sh set sda4

Tabla 3: Paquetes de software para el modo live

Nombre	Versión	Dependencias
cpio	2.12	texlive-20160523b
BusyBox	1.21.1	–
Grub	2.02	–
Syslinux	6.03	libffi-3.2.1 NASM-2.12.02 Python-2.7.12 libusb-1.0.20 libusb-compat-0.1.5
Dos2unix	7.3.4	–

4. Conclusiones

Con el desarrollo del presente estudio se logró completar el objetivo propuesto, obtener un marco de trabajo preliminar con los scripts A y B modificables, según las necesidades del sistema a construir. Al mismo tiempo, se logró demostrar todos los pasos involucrados en la construcción del modo vivo**.** Cabe destacar, que los scripts permiten crear variantes o mejoras al código.

Como trabajos futuros se tiene contemplado el soporte de un entorno grafico utilizando QT y sería de interés implementar SquashFS e IsoHybrid para soportar en su totalidad el modo vivo-CD.

5. Agradecimiento

Este trabajo fue apoyado en parte por el grupo de investigación en salud electrónica y supercomputación (GISES) de la Universidad Tecnológico de Panamá y al programa Sistema Nacional de Investigación (SNI) de la SENACYT, el cual uno de los dos autores es miembro.

Referencias

[1] Beekmans, G., Burgess, M. and Dubbs, B. (2017) *Linux From Scratch: Version SVN-20170624*. Available at: http://www.linuxfromscratch.org/lfs/view/development/.

[2] Beekmans, G., Huntwork, J. and Burgess, M. (2017) *LFS Stable Version 8.0 Release*. Available at: http://www.linuxfromscratch.org/news.html.

[3] Calderón, H. and Dominguez, L. (2015) *Desarrollo De Una Distribución Linux Para La Implementación De Aplicaciones Paralelas Científicas De Código Abierto*. Universidad Tecnológica de Panamá.

[4] Cohen, A., Battistoni, G. and Mark, S. (2008) 'Short communication F LUKA -LIVE — an embedded framework, for enabling a computer to execute F LUKA under the control of a Linux OS', 43, pp. 121–124. doi: 10.1016/j.radmeas.2007.11.010.

[5] Fedora (2017) *Anaconda*. Available at: https://fedoraproject.org/wiki/Anaconda.

[6] Gentoo Linux (2016) *Initramfs/Guide*. Available at: `https://wiki.gentoo.org/wiki/Initramfs/Guide/es`.

[7] Gentoo Linux (2017) *Kernel/Configuration*. Available at: `https://wiki.gentoo.org/wiki/Kernel/Configuration`.

[8] Hernandez, M. (2005) *Beginners Guide To Creating A Live CD With LFS 6.0*. Available at: http://www.linuxfromscratch.org/hints/downloads/files/OLD/bootcd-2.6-udev-nptl.txt.

[9] Huntwork, J., Patrakov, A. and Pegg, T. (2012) *What is the LFS LiveCD?* Available at: `http://www.linuxfromscratch.org/livecd/`.

[10] Minnich, R. G. and Mirtchovski, A. (2015) 'U-root: A Go-based, Firmware Embeddable Root File System with On-demand Compilation', *2015 USENIX Annual Technical Conference (USENIX ATC 15)*, pp. 577–586. Available at: https://www.usenix.org/conference/atc15/technical-session/presentation/minnich.

[11] Molina, I. and Camacho, J. (2016) *DISTRIBUCIÓN LINUX 'IXCHEL'*.

[12] Negus, C. (2006) 'Using Boot Loaders in Live CDs', in *Live Linux CDs: Building and Customizing Bootables*. Available at: at:https://books.google.com.pa/books?id=2S3eUgb39C8C&pg=PA136&lpg=PA136&dq=el+torito+live+cd&source=bl&ots=mwTzywn9BJ&sig=1FiVVxj8RmkTuVK2Zb71DWSSe68&hl=es&sa=X&redir_esc=y#v=onepage&q=eltoritolivecd&f=false..

[13] van Tilborg, H. C.. and Jajodia, S. (2011) 'Chroot', in *Encyclopedia of Cryptography and Security*.

[14] TracReports: Linux From Scratch (2007) *Create initramfs*. Available at: http://wiki.linuxfromscratch.org/lfs/ticket/2033.

[15] TracReports: Linux From Scratch (2009) *getpids() bug in lfs/init.d/functions, as of lfs-bootscripts-20090523*. Available at: http://wiki.linuxfromscratch.org/lfs/ticket/2472.

[16] TracReports: Linux From Scratch (2010) 'Nouveau Part 1: Intergrating it into the Linux kernel'. Available at: http://wiki.linuxfromscratch.org/lfs/ticket/2585.

[17] Ubuntu (2016) *LiveCDCustomization*. Available at: https://help.ubuntu.com/community/LiveCDCustomization.

[18] Universi, G. M. C. *et al.* (2006) 'Linux based Live CD on optical disks', (May 2014).

[19] Xu, G. *et al.* (2014) 'An approach of VM image customized through Linux from scratch on cloud platform', 18(4), pp. 62–67.

Authorization and Disclaimer

Authors authorize ESTEC to publish the paper in the conference proceedings. Neither ESTEC nor the editors are responsible either for the content or for the implications of what is expressed in the paper.