GUIX SYSTEM AND LIBREBOOT

Raghav "RG" Gururajan

GUIX SYSTEM AND LIBREBOOT

Guix System with Full Disk Encryption on Libreboot

Raghav "RG" Gururajan

FLOSS Manuals

Guix System and Libreboot
Guix System with Full Disk Encryption on Libreboot
Raghav "RG" Gururajan

Published by FLOSS Manuals, Toronto, 1.0, 2019-08-18
ISBN: booktype:guix-system-and-libreboot

© 2019-08-18 Raghav "RG" Gururajan. Creative Commons Attribution-ShareAlike

https://www.raghavgururajan.com

This book was created with Booktype. For more information, please visit: www.booktype.pro
Discover Omnibook, a social platform for writing books, and create your own book at www.omnibook.pro

INTRODUCTION

Guix System is an exotic distribution of GNU/Linux Operating System; with Guix as package+system manager, Linux-Libre as kernel and Shepherd as init system.

Libreboot is a de-blobbed distribution of Coreboot Firmware. By default, Libreboot comes with GRUB Bootloader as a payload.

The objective of this mini-book is to provide step-by-step guide for setting up guix system (stand-alone guix) with full disk encryption (including /boot) on devices powered by libreboot.

Any users, for their generalized use cases, need not stumble away from this guide to accomplish the setup. Advancers, for deviant use cases, will have to explore outside this guide for customization; although this guide provides information that is of paramount use.

Let us begin!

PREPARATION

In your current GNU/Linux System, open terminal as root user.

Insert USB drive and get the USB device name /dev/sdX, where “X” is the variable to make a note of.

lsblk

Unmount the USB drive just in case if it’s auto-mounted.

umount /dev/sdX

Download the latest (a.b.c) Guix System ISO Installer Package (sss) and it’s GPG Signature; where “a.b.c” is the variable for version number and “sss” is the variable for system architecture.

wget https://ftp.gnu.org/gnu/guix/guix-system-install-a.b.c.sss-linux.iso.xz

wget https://ftp.gnu.org/gnu/guix/guix-system-install-a.b.c.sss-linux.iso.xz.sig

Import required public key.

gpg --keyserver pool.sks-keyservers.net –recv-keys 3CE464558A84FDC69DB40CF-B090B11993D9AEBB5

Verify the GPG Signature of the downloaded package.

gpg --verify guix-system-install-a.b.c.sss-linux.iso.xz.sig

Extract the ISO Image from the downloaded package.

xz -d guix-system-install-a.b.c.sss-linux.iso.xz

Write the extracted ISO Image to the USB drive.

dd if=guix-system-install-a.b.c.sss-linux.iso of=/dev/sdX; sync

Reboot the device.

reboot

PRE-INSTALLATION

On reboot, as soon as you see the Libreboot Graphic Art, press arrow keys to change the menu entry.

Choose “Search for GRUB2 configuration on external media [s]” and wait for the Guix System from USB drive to load.

Set your keyboard layout lo, where “lo” is the two-letter keyboard layout code (example: us or uk).

loadkeys lo

Unblock network interfaces (if any).

rfkill unblock all

Get the names of your network interfaces.

ifconfig -a

Bring your required network interface nwif (wired or wireless) up, where “nwif” is the variable for interface name. For wired connections, this should be enough.

ifconfig nwif up

For wireless connection, create a configuration file using text editor, where “fname” is the variable for any desired filename.

nano fname.conf

Choose, type and save ONE of the following snippets, where ‘nm’ is the name of the network you want to connect, ‘pw’ is the corresponding network’s password or passphrase and ‘un’ is user identity.

For most private networks:

```
network={
  ssid="nm"
  key_mgmt=WPA-PSK
  psk="pw"
}
```

(or)

For most public networks:

```
network={
  ssid="nm"
  key_mgmt=NONE
}
```

(or)

For most organizational networks:

```
network={
  ssid="nm"
  scan_ssid=1
  key_mgmt=WPA-EAP
  identity="un"
  password="pw"
  eap=PEAP
  phase1="peaplabel=0"
  phase2="auth=MSCHAPV2"
}
```

Connect to the configured network, where “fname” is the filename and “nwif” is the network interface name.

```
wpa_supplicant -c fname.conf -i nwif -B
```

Assign an IP address to your network interface, where “nwif” is the network interface name.

```
dhclient -v nwif
```

Obtain the device name /dev/sdX in which you would like to deploy and install Guix System, where “X” is the variable to make a note of.

```
lsblk
```

Wipe the respective device. Wait for the command operation to finish.

```
dd if=/dev/urandom of=/dev/sdX; sync
```

Load device-mapper module in the current kernel.

```
modprobe dm_mod
```

Partition the respective device. Just do, GPT --> New --> Write --> Quit; defaults will be set.

```
cfdisk /dev/sdX
```

Encrypt the respective partition.

```
cryptsetup -v --cipher serpent-xts-plain64 --key-size 512 --hash whirlpool --iter-time 500 --use-ran-
```

dom --verify-passphrase luksFormat /dev/sdX1

Obtain and note down the “LUKS UUID”.

cryptsetup luksUUID /dev/sdX1

Open the respective encrypted partition, where “partname” is any desired partition name.

cryptsetup luksOpen /dev/sdX1 partname

Make filesystem on the respective partition, where “fsname” is any desired filesystem name.

mkfs.btrfs -L fsname /dev/mapper/partname

Mount the respective filesystem under the current system.

mount LABEL=fsname /mnt

Create a swap file and make it readable cum writable only by root.

dd if=/dev/zero of=/mnt/swapfile bs=1MiB count=2048

chmod 600 /mnt/swapfile

mkswap /mnt/swapfile

swapon /mnt/swapfile

INSTALLATION

Make the installation packages to be written on the respective mounted filesystem.

herd start cow-store /mnt

Create the required directory.

mkdir /mnt/etc

Create, edit and save the configuration file by typing the following code snippet. WATCH-OUT for variables in the code snippet and replace them with your relevant values.

nano /mnt/etc/config.scm

Snippet:

```
(use-modules
   (gnu)
   (gnu system nss))
(use-service-modules
   xorg
   desktop)
(use-package-modules
   certs
   gnome)
(operating-system
   (host-name "hostname")
   (timezone "Zone/SubZone")
   (locale "ab_XY.1234")
   (keyboard-layout
      (keyboard-layout
         "xy"
         "altgr-intl"))
   (bootloader
      (bootloader-configuration
         (bootloader
            (bootloader
               (inherit grub-bootloader)
               (installer #~(const #t))))
         (keyboard-layout keyboard-layout)))
   (mapped-devices
      (list
         (mapped-device
            (source
```

```
            (uuid "luks-uuid"))
            (target "partname")
            (type luks-device-mapping))))
  (file-systems
    (append
      (list
        (file-system
          (device
            (file-system-label "fsname"))
          (mount-point "/")
          (type "btrfs")
          (dependencies mapped-devices)))
      %base-file-systems))
  (users
    (append
      (list
        (user-account
          (name "username")
          (comment "Full Name")
          (group "users")
          (supplementary-groups '("wheel" "netdev" "audio" "video" "lp" "cdrom" "tape" "kvm"))))
      %base-user-accounts))
  (packages
    (append
      (list
        nss-certs)
      %base-packages))
  (services
    (append
      (list
        (extra-special-file "/usr/bin/env"
          (file-append coreutils "/bin/env"))
        (set-xorg-configuration
          (xorg-configuration
            (keyboard-layout keyboard-layout)))
        (service gnome-desktop-service-type))
      %desktop-services))
  (name-service-switch %mdns-host-lookup-nss))
```

Initialize new Guix System.

```
guix system init /mnt/etc/config.scm /mnt
```

Reboot the device.

```
reboot
```

POST-INSTALLATION

On reboot, as soon as you see the Libreboot Graphic Art, choose the option 'Load Operating System [o]'

Enter LUKS Key, for libreboot's grub, as prompted.

You may have to go through warning prompts by repeatedly pressing the "enter/return" key.

You will now see guix's grub menu from which you can go with the default option.

Enter LUKS Key again, for kernel, as prompted.

Upon GNOME Login Screen, login as "root" with password field empty.

Open terminal from the GNOME Dash.

Set passkey for "root" user. Follow the prompts.

passwd root

Set passkey for "username" user. Follow the prompts.

passwd username

Update the guix distribution. Wait for the process to finish.

guix pull

Update the search paths.

export PATH="$HOME/.config/guix/current/bin:$PATH"

export INFOPATH="$HOME/.config/guix/current/share/info:$INFOPATH"

Update the guix system. Wait for the process to finish.

guix system reconfigure /etc/config.scm

Reboot the device.

reboot

COMPLETION

On reboot, as soon as you see the Libreboot Graphic Art, press arrow keys to change the menu entry. Now, press “C” to enter command-line.

Unplug any other attached storage devices.

Enter following commands and respond to first command with your LUKS Key.

cryptomount -a

set root=’crypt0’

configfile /boot/grub/grub.cfg

As soon as you see Guix Boot menu, Press “E” to edit the command.

Right after ‘linux /[…]’; type *iomem=relaxed*. Then press “F10” key.

Type your LUKS key again.

Login as “root” with leaving password empty.

Open terminal from the GNOME Dash.

Download libreboot utilities.

wget https://www.mirrorservice.org/sites/libreboot.org/release/stable/20160907/ libreboot_r20160907_util.tar.xz

Extract the downloaded file.

tar -xf libreboot_r20160907_util.tar.xz

Rename the respective file.

mv "*libreboot_r20160907_util*" "*libreboot_util*"

Install flashrom application.

guix package –i flashrom

Find your ROM Chip Model and Size. Look for the output line “Found […] flash chip […]”.

flashrom -p internal –V

Create image from your current ROM and move it for respective utility; where “chip-model” is the variable for flash chip model.

flashrom -c chip-model -p internal -r /libreboot_util/cbfstool/i686/libreboot.rom

(or)

flashrom -c chip-model -p internal -r /libreboot_util/cbfstool/x86_64/libreboot.rom

Change directory to the respective utility.

cd /libreboot_util/cbfstool/i686/

(or)

cd /libreboot_util/cbfstool/x86_64/

Extract two types of grub configuration files from the image.

./cbfstool libreboot.rom extract -n grub.cfg -f grub.cfg

./cbfstool libreboot.rom extract -n grubtest.cfg -f grubtest.cfg

Edit “grub.cfg” and insert the following code snippet above the first menu entry line “menuentry 'Load Operating System [o]' --hotkey='o' –unrestricted { [...] }”.

nano grub.cfg

Snippet:

menuentry ‘Guix System (An Advanced GNU System Distribution) [g]’ --hotkey=’g’ --unrestricted
{
cryptomount -a
set root=’crypt0’
configfile /boot/grub/grub.cfg
}

Edit “grubtest.cfg” and insert the following code snippet above the first menu entry line “menuentry 'Load Operating System [o]' --hotkey='o' –unrestricted { [...] }”.

nano grubtest.cfg

Snippet:

menuentry ‘Guix System (An Advanced GNU System Distribution) [g]’ --hotkey=’g’ --unrestricted
{
cryptomount -a
set root=’crypt0’
*configfile /gnu/store/*grub.cfg*

}

Remove old config files from ROM Image.

./cbfstool libreboot.rom remove -n grub.cfg

./cbfstool libreboot.rom remove -n grubtest.cfg

Insert new config files into the ROM Image

./cbfstool libreboot.rom add -n grub.cfg -f grub.cfg -t raw

./cbfstool libreboot.rom add -n grubtest.cfg -f grubtest.cfg -t raw

Move ROM Image.

mv libreboot.rom ~/libreboot_util

Flash the ROM with the new Image.

./flash update libreboot.rom

(or)

./flash forceupdate libreboot.rom

Reboot the device.

reboot

CONCLUSION

Everything should be stream-lined from now. You can follow your regular boot steps without requiring manual intervention.

Generally, you can use libreboot's initial/default grub.cfg, whose guix menu-entry invokes guix's grub.cfg located at "/boot/grub/". For contingency, you can also use libreboot's grubtest.cfg, whose guix menu-entry randomly invokes one of the guix's special grub.cfg files located at guix's special directory "/gnu/store/".

During the boot process, as prompted, you will have to type LUKS key twice; once for libreboot's grub and once more for linux-libre kernel.

That is it! You have now setup guix system with full-disk encryption on your device powered by libreboot. Enjoy!

REFERENCES

[1] Guix Manual (http://guix.gnu.org/manual/en/).

[2] Libreboot Documentation (https://libreboot.org/docs/).

ACKNOWLEDGEMENTS

[1] Thanks to Guix Developer, Clement Lassieur (clement@lassieur.org), for helping me with the Guile Scheme Code for the Bootloader Configuration.

[2] Thanks to Libreboot Founder and Developer, Leah Rowe (leah@libreboot.org), for helping me to understand the libreboot’s functionalities better.

LICENSE

This work by Raghav "RG" Gururajan is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.

To view a copy of this license, visit https://creativecommons.org/licenses/by-sa/4.0/.