01. Getting Started

What is Kali Linux ?

Kali Linux is a Debian-based Linux distribution aimed at advanced Penetration Testing and Security Auditing. Kali contains several hundred tools which are geared towards various information security tasks, such as Penetration Testing, Security research, Computer Forensics and Reverse Engineering. Kali Linux is developed, funded and maintained by Offensive Security, a leading information security training company.

Kali Linux was released on the 13th March, 2013 as a complete, top-to-bottom rebuild of BackTrack Linux, adhering completely to Debian development standards.

- **More than 600 penetration testing tools included:** After reviewing every tool that was included in BackTrack, we eliminated a great number of tools that either simply did not work or which duplicated other tools that provided the same or similar functionality. Details on what's included are on the Kali Tools site.
- **Free (as in beer) and always will be:** Kali Linux, like BackTrack, is completely free of charge and always will be. You will never, ever have to pay for Kali Linux.
- **Open source Git tree:** We are committed to the open source development model and our development tree is available for all to see. All of the source code which goes into Kali Linux is available for anyone who wants to tweak or rebuild packages to suit their specific needs.
- **FHS compliant:** Kali adheres to the Filesystem Hierarchy Standard, allowing Linux users to easily locate binaries, support files, libraries, etc.
- **Wide-ranging wireless device support:** A regular sticking point with Linux distributions has been supported for wireless interfaces. We have built Kali Linux to support as many wireless devices as we possibly can, allowing it to run properly on a wide variety of hardware and making it compatible with numerous USB and other wireless devices.
- **Custom kernel, patched for injection:** As penetration testers, the development team often needs to do wireless assessments, so our kernel has the latest injection patches included.
- **Developed in a secure environment:** The Kali Linux team is made up of a small group of individuals who are the only ones trusted to commit packages and interact with the repositories, all of which is done using multiple secure protocols.
- **GPG signed packages and repositories:** Every package in Kali Linux is signed by each individual developer who built and committed it, and the repositories subsequently sign the packages as well.
- **Multi-language support:** Although penetration tools tend to be written in English, we have ensured that Kali includes true multilingual support, allowing more users to operate in their native language and locate the tools they need for the job.
- **Completely customizable:** We thoroughly understand that not everyone will agree with our design decisions, so we have made it as easy as possible for our more adventurous users to customize Kali Linux to their liking, all the way down to the kernel.
- **ARMEL and ARMHF support:** Since ARM-based single-board systems like the Raspberry Pi and

BeagleBone Black, among others, are becoming more and more prevalent and inexpensive, we knew that Kali's ARM support would need to be as robust as we could manage, with fully working installations for both ARMEL and ARMHF systems. Kali Linux is available on a wide range of ARM devices and has ARM repositories integrated with the mainline distribution so tools for ARM are updated in conjunction with the rest of the distribution.

Kali Linux is specifically tailored to the needs of penetration testing professionals, and therefore all documentation on this site assumes prior knowledge of, and familiarity with, the Linux operating system in general. Please see Should I Use Kali Linux? for more details on what makes Kali unique.

Should I Use Kali Linux?

What's Different About Kali Linux?

Kali Linux is specifically geared to meet the requirements of professional penetration testing and security auditing. To achieve this, several core changes have been implemented in Kali Linux which reflect these needs:

1. **Single user, root access by design:** Due to the nature of security audits, Kali Linux is designed to be used in a "single, root user" scenario. Many of the tools used in penetration testing require escalated privileges, and while it's generally sound policy to only enable root privileges when necessary, in the use cases that Kali Linux is aimed at, this approach would be a burden.

2. **Network services disabled by default:** Kali Linux contains systemd hooks that disable network services by default. These hooks allow us to install various services on Kali Linux, while ensuring that our distribution remains secure by default, no matter what packages are installed. Additional services such as Bluetooth are also blacklisted by default.

3. **Custom Linux kernel:** Kali Linux uses an upstream kernel, patched for wireless injection.

4. **A *minimal* and *trusted* set of repositories:** given the aims and goals of Kali Linux, maintaining the integrity of the system as a whole is absolutely key. With that goal in mind, the set of upstream software sources which Kali uses is kept to an absolute minimum. Many new Kali users are tempted to add additional repositories to their **sources.list**, but doing so runs a *very serious risk* of breaking your Kali Linux installation.

Is Kali Linux Right For You?

As the distribution's developers, you might expect us to recommend that everyone should be using Kali Linux. The fact of the matter is, however, that Kali is a Linux distribution specifically geared towards professional penetration testers and security specialists, and given its unique nature, it is **NOT** a recommended distribution if you're unfamiliar with Linux or are looking for a general-purpose Linux desktop distribution for development, web design, gaming, etc.

Even for experienced Linux users, Kali can pose some challenges. Although Kali is an open source project, it's not a *wide*-open source project, for reasons of security. The development team is small and trusted, packages in the repositories are signed both by the individual committer and the team, and — importantly — the set of upstream repositories from which updates and new packages are drawn is very small. Adding repositories to your software sources which have not been tested by the Kali Linux development team is a good way to cause problems on your system.

While Kali Linux is architected to be highly customizable, don't expect to be able to add random unrelated packages and repositories that are "out of band" of the regular Kali software sources and have it Just Work. In particular, there is absolutely no support whatsoever for the apt-add-repository command, LaunchPad, or PPAs. Trying to install ***Steam*** on your Kali Linux desktop is an experiment that will not end well. Even getting a package as mainstream as NodeJS onto a Kali Linux installation can take a little extra effort and tinkering.

If you are unfamiliar with Linux generally, if you do not have at least a basic level of competence in administering a system, if you are looking for a Linux distribution to use as a learning tool to get to know your way around Linux, or if you want a distro that you can use as a general purpose desktop installation, *Kali Linux is probably not what you are looking for*.

In addition, misuse of security and penetration testing tools within a network, particularly without specific authorization, may cause irreparable damage and result in significant consequences, personal and/or legal. "Not understanding what you were doing" is not going to work as an excuse.

However, if you're a professional penetration tester or are studying penetration testing with a goal of becoming a certified professional, there's no better toolkit — at any price — than Kali Linux.

If you are looking for a Linux distribution to learn the basics of Linux and need a good starting point, Kali Linux is ***not*** the ideal distribution for you. You may want to begin with Ubuntu, Mint, or Debian instead. If you're interested in getting hands-on with the internals of Linux, take a look the "Linux From Scratch" project.

Downloading Kali Linux

IMPORTANT! ***Never*** download Kali Linux images from *anywhere other than the official sources*. ***Always*** be sure to *verify the SHA256 checksums of the file you've downloaded against our* official values. It would be easy for a malicious entity to modify a Kali installation to contain exploits or malware and host it unofficially. Downloads are rate limited to 5 concurrent connections.

Where to Get Official Kali Linux Images

ISO Files for Intel-based PCs

In order to run Kali "Live" from a USB drive on standard Windows and Apple PCs, you'll need a Kali Linux bootable ISO image, in either 32-bit or 64-bit format.

If you're not sure of the architecture of the system you want to run Kali on, on Linux or OS X, you can run the command

uname -m

at the command line. If you get the response, "x86_64", use the 64-bit ISO image (the one containing "amd64" in the file name); if you get "i386", use the 32-bit image (the one containing "i386" in the file name). If you're on a Windows system, the procedure for determining whether your architecture is detailed on Microsoft's website.

The Kali Linux images are available both as directly downloadable ".iso/.img" files or via ".torrent" files.

- Official Kali ISOs for Intel-based PCs

Building your own Kali Linux ISO, standard or customized, is a very simple process.

VMware Images

If you want to run Kali Linux as a "guest" under VMware, Kali is available as a pre-built VMware virtual machine with VMware Tools already installed. The VMware image is available in a 64-bit (amd64), 32-bit (i686), and 32-bit PAE (i486) formats.

- Official Kali Linux VMware Images

ARM Images

The hardware architectures of ARM-based devices vary considerably, so it is not possible to have a single image that will work across all of them. Pre-built Kali Linux images for the ARM architecture are available for the wide range of devices.

Scripts for building your own ARM images locally are also available on GitHub. For more details see the articles on setting up an ARM cross-compilation environment, and building a custom Kali Linux ARM chroot.

Verifying Your Downloaded Kali Image

Why do I need to do this?

Before you run Kali Linux Live, or install it to your hard disk, you want to be very sure that what you've got actually *is* Kali Linux, and not an imposter. Kali Linux is a professional penetration testing and forensics toolkit. As a professional penetration tester, having absolute confidence in the integrity of your tools is critical: if your tools aren't trustworthy, your investigations won't be trustworthy, either.

Moreover, as the leading penetration testing distribution, Kali's strengths mean that a bogus version of Kali Linux could do *a tremendous amount of damage* if it were deployed unwittingly. There are plenty of people with plenty of reason to want to stick very sketchy stuff into something that *looks* like Kali, and you absolutely don't want to find yourself running something like that.

Avoiding this is simple:

- *only* download Kali Linux via the official download pages at https://www.kali.org/downloads or https://www.offensive-security.com/kali-linux-vmware-arm-image-download/ — you won't be able to browse to these pages without SSL: encrypting the connection makes it much harder for an attacker to use a "man-in-the-middle" attack to modify your download. There are a few potential weaknesses to even these sources — see the sections on verifying the download with the SHA256SUMS file and its signature against the official Kali Development team private key for something much closer to absolute assurance.
- once you've downloaded an image, and *before you run it*, *always* validate that it really *is* what it's supposed to be by verifying its checksum using one of the procedures detailed below.

There are several methods for verifying your download. Each provides a certain level of assurance, and involves a corresponding level of effort on your part.

- You can download an ISO image from an official Kali Linux "Downloads" mirror, calculate the ISO's SHA256 hash and compare it by inspection with the value listed on the Kali Linux site. This is quick and easy, but potentially susceptible to subversion via a DNS poisoning: it assumes that the site to which, for example, the domain "kali.org" resolves is in fact the actual Kali Linux site. If it somehow weren't, an attacker could present a "loaded" image and a matching SHA256 signature on the fake web page. See the section "Manually Verify the Signature on the ISO (Direct Download)", below.
- You can download an ISO image through the torrents, and it will also pull down a file — unsigned — containing the calculated SHA256 signature. You can then use the shasum command (on Linux and OS X) or a utility (on Windows) to automatically verify that the file's computed signature matches the signature in the secondary file. This is even easier than the "manual" method, but suffers from the same weakness: if the torrent you pulled down isn't really Kali Linux, it could still have a good signature. See the section "Verify the Signature on the ISO Using the Included Signature File (Torrent Download)", below.
- To be as close to absolutely certain as possible that the Kali Linux download you've obtained is the real thing, you can download both a cleartext signature file and and version of the same file that has been signed with the official Kali Linux private key and use GNU Privacy Guard (GPG) to first, verify that the computed SHA256 signature and the signature in the cleartext file match and second, verify that the signed version of the file containing the SHA256 hash has been correctly signed with the official key. If you use this more complicated process and successfully validate your downloaded ISO, you can proceed with pretty complete assurance that what you've got is the official image and that it has not been tampered with in any way. This method, while the most complex, has the advantage of providing independent assurance of the integrity of the image. The only way this method can fail is if the official Kali Linux private key is not only subverted by an attacker, but also not subsequently revoked by the Kali Linux development team. For this method, see the section on verification using the SHA256SUMS file.

***What* do I need to do this?**

If you're running on Linux, you probably already have GPG (GNU Privacy Guard) installed. If you're on Windows or OS X, you'll need to install the appropriate version for your platform.

- If you're on a PC running Windows, download and install GPG4Win from here.
- If you're on a Macintosh running OS X, download and install GPGTools from here. Since Windows does not have the native ability to calculate SHA256 checksums, you will also need a utility such as Microsoft File Checksum Integrity Verifier or Hashtab to verify your download.

Once you've installed GPG, you'll need to download and import a copy of the Kali Linux official key. Do this with the following command:

```
$ wget -q -O - https://archive.kali.org/archive-key.asc | gpg --import
```

or the command

```
$ gpg --keyserver hkp://keys.gnupg.net --recv-key 7D8D0BF6
```

Your output should look like this:

```
gpg: key 7D8D0BF6: public key "Kali Linux Repository <devel@kali.org>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

Verify that the key is properly installed with the command:

```
gpg --fingerprint 7D8D0BF6
```

The output will look like this:

```
pub rsa4096 2012-03-05 [SC] [expires: 2021-02-03]
44C6 513A 8E4F B3D3 0875 F758 ED44 4FF0 7D8D 0BF6
uid [ full ] Kali Linux Repository <devel@kali.org>
sub rsa4096 2012-03-05 [E] [expires: 2021-02-03]
```

You’re now set up to validate your Kali Linux download.

How Do I Verify My Downloaded Image?

Manually Verify the Signature on the ISO (Direct Download)

If you downloaded the ISO directly from the downloads page, verify it using the following procedure.

On Linux, or OS X, you can generate the SHA256 checksum from the ISO image you’ve downloaded with the following command (assuming that the ISO image is named “kali-linux-2016.2-amd64.iso”, and is in your current directory):

```
shasum -a 256 kali-linux-2016.2-amd64.iso
```

The output should look like this:

```
1d90432e6d5c6f40dfe9589d9d0450a53b0add9a55f71371d601a5d454fa0431
kali-linux-2016.2-amd64.iso
```

The resulting SHA256 signature, “1d90432e6d5c6f40dfe9589d9d0450a53b0add9a55f71371d601a5d454fa0431”, can be seen to match the signature displayed in the “sha256sum” column on the official download page for the 64-bit Intel architecture Kali Linux 2016.2 ISO image:

Download Kali Linux Images

We generate fresh Kali Linux image files every few months, which we make available for download. This page provides the links to **download Kali Linux** in its latest official release. For a release history, check our Kali Linux Releases page. Please note: You can find unofficial, untested weekly releases at http://cdimage.kali.org/kali-weekly/.

Image Name	Download	Size	Version	sha256sum
Kali 64 bit	ISO | Torrent	2.9G	2016.2	1d90432e6d5c6f40dfe9589d9d0450a53b0add9a55f71371d601a5d454fa0431
Kali 32 bit	ISO | Torrent	2.9G	2016.2	c94772c4fd71f50b245c7b15f4f225ad7c751879f501fa1cf698beb1460c0bf5
Kali 64 bit Light	ISO | Torrent	1.1G	2016.2	997f5ed0f7c99c4518288c7e2c4b684b1bdcc2fbe02c152d7ecbd17f0536c29f
Kali 32 bit Light	ISO | Torrent	1.1G	2016.2	590e6df2e8e0b4d42bf3dd4e4c7d6acf24b7262fabda52a0c6c3b35006def295
Kali 64 bit e17	ISO | Torrent	2.7G	2016.2	404d0fd917a404cf6c894b5bd87171ebf8eb445bd5573a3e78f88629067d694b
Kali 64 bit Mate	ISO | Torrent	2.8G	2016.2	cd11b7085cc7d71546488106c2eedf85386fe73d731bedf38991661270dd91db
Kali 64 bit Xfce	ISO | Torrent	2.7G	2016.2	3e08e5420b368183606b105cf2cb1276dd024afe3e2b2e3187d7d37ec1320c41
Kali 64 bit LXDE	ISO | Torrent	2.7G	2016.2	7461882843e5a0fc37979850994fb5755249a176429f9e67805bd7f6baa5bb62
Kali armhf	Image | Torrent	0.7G	2016.2	f192289b6bc64bab7197a90627ced2477c7c98bd20c1d29f442a152e169dae42
Kali armel	Image | Torrent	0.7G	2016.2	efb6f487feab1c9141f28da22c73cbc5217325bf46298f899ac89c39c19aa5f5

Verify the Signature on the ISO Using the Included Signature File (Torrent Download)

If you downloaded your copy of the Kali Linux ISO image via the torrents, in addition to the ISO file (e.g. kali-linux-2016.2-amd64.iso), there will be a second file containing the computed SHA256 signature for the ISO, with the extension ".txt.sha256sum" (e.g. kali-linux-2016.2-amd64.txt.sha256sum). You can use this file to verify the authenticity of your download on Linux or OS X with the following command:

```
grep kali-linux-2016.2-amd64.iso kali-linux-2016.2-amd64.txt.sha256sum | shasum -a 256 -c
```

If the image is successfully authenticated, the response will look like this:

```
kali-linux-2016.2-amd64.iso: OK
```

IMPORTANT! If you are unable to verify the authenticity of the Kali Linux image you have downloaded as described in the preceding section, ***do NOT use it!*** **Using it could endanger not only your own system, but any network you connect to as well as the other systems on that network.** ***Stop***, and ***ensure*** that you have downloaded the images from a ***legitimate Kali Linux mirror***.

Verify the ISO Using the SHA256SUMS File

This is a more complex procedure, but offers a much higher level of validation: it does not rely on the integrity of the web site you downloaded the image from, only the official Kali Linux development team key that you install independently. To verify your image this way for an Intel architecture version of Kali, you will need to download three files from the Kali "Live CD Image" site for the current release (v2016.2, as of this writing):

- The ISO image itself (e.g. kali-linux-2016.2-amd64.iso)
- The file containing the calculated SHA256 hash for the ISO, SHA256SUMS
- The signed version of that file, SHA256SUMS.gpg

Before verifying the checksums of the image, you must ensure that the SHA256SUMS file is the one generated by Kali. That's why the file is signed by Kali's official key with a detached signature in SHA256SUMS.gpg. If you have not already done so, Kali's official key can be downloaded and imported into your keychain with this command:

```
$ wget -q -O - https://www.kali.org/archive-key.asc | gpg --import
```

or this command

```
$ gpg --keyserver hkp://keys.gnupg.net --recv-key 7D8D0BF6
```

Your output should look like this:

```
gpg: key 7D8D0BF6: public key "Kali Linux Repository <devel@kali.org>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

You should verify that the key is properly installed with the command:

```
gpg --fingerprint 7D8D0BF6
```

The output will look like this:

```
pub rsa4096 2012-03-05 [SC] [expires: 2021-02-03]
44C6 513A 8E4F B3D3 0875 F758 ED44 4FF0 7D8D 0BF6
uid [ full ] Kali Linux Repository <devel@kali.org>
sub rsa4096 2012-03-05 [E] [expires: 2021-02-03]
```

Once you have downloaded both SHA256SUMS and SHA256SUMS.gpg, you can verify the signature as follows:

```
$ gpg --verify SHA256SUMS.gpg SHA256SUMS
gpg: Signature made Thu 16 Mar 08:55:45 2017 MDT using RSA key ID 7D8D0BF6
gpg: Good signature from "Kali Linux Repository <devel@kali.org>"
```

If you don't get that "Good signature" message or if the key ID doesn't match, then you should *stop* and review whether you downloaded the images from a *legitimate Kali Linux mirror*. The failed verification strongly suggests that the image you have may have been tampered with.

If you did get the "Good signature" response, you can now be assured that the checksum in the SHA256SUMS file was actually provided by the Kali Linux development team. All that remains to be done to complete the verification is to validate that the signature you compute from the ISO you've downloaded matches the one in the SHA256SUMS file. You can do that on Linux or OS X with the following command (assuming that the ISO is named "kali-linux-2016.2-amd64.iso" and is in your working directory):

```
grep kali-linux-2016.2-amd64.iso SHA256SUMS | shasum -a 256 -c
```

If the image is successfully authenticated, the response will look like this:

```
kali-linux-2016.2-amd64.iso: OK
```

If you don't get "OK" in response, then *stop* and review what's happened: the Kali image you have has apparently been tampered with. Do **NOT** use it.

Once you've downloaded and verified your image, you can proceed to create a bootable "Kali Linux Live" USB drive.

Kali Linux Default Passwords

Kali Linux Default root Password is toor

Default root Password

During installation, Kali Linux allows users to configure a password for the *root* user. However, should you decide to boot the live image instead, the i386, amd64, VMWare and ARM images are configured with the **default root password – "toor"**, without the quotes.

02. Kali Linux Live

Making a Kali Bootable USB Drive

Our favorite way, and the fastest method, for getting up and running with Kali Linux is to run it “live” from a USB drive. This method has several advantages:

- It’s non-destructive — it makes no changes to the host system’s hard drive or installed OS, and to go back to normal operations, you simply remove the “Kali Live” USB drive and restart the system.
- It’s portable — you can carry Kali Linux in your pocket and have it running in minutes on an available system
- It’s customizable — you can roll your own custom Kali Linux ISO image and put it onto a USB drive using the same procedures
- It’s potentially persistent — with a bit of extra effort, you can configure your Kali Linux “live” USB drive to have persistent storage, so the data you collect is saved across reboots

In order to do this, we first need to create a bootable USB drive which has been set up from an ISO image of Kali Linux.

What You’ll Need

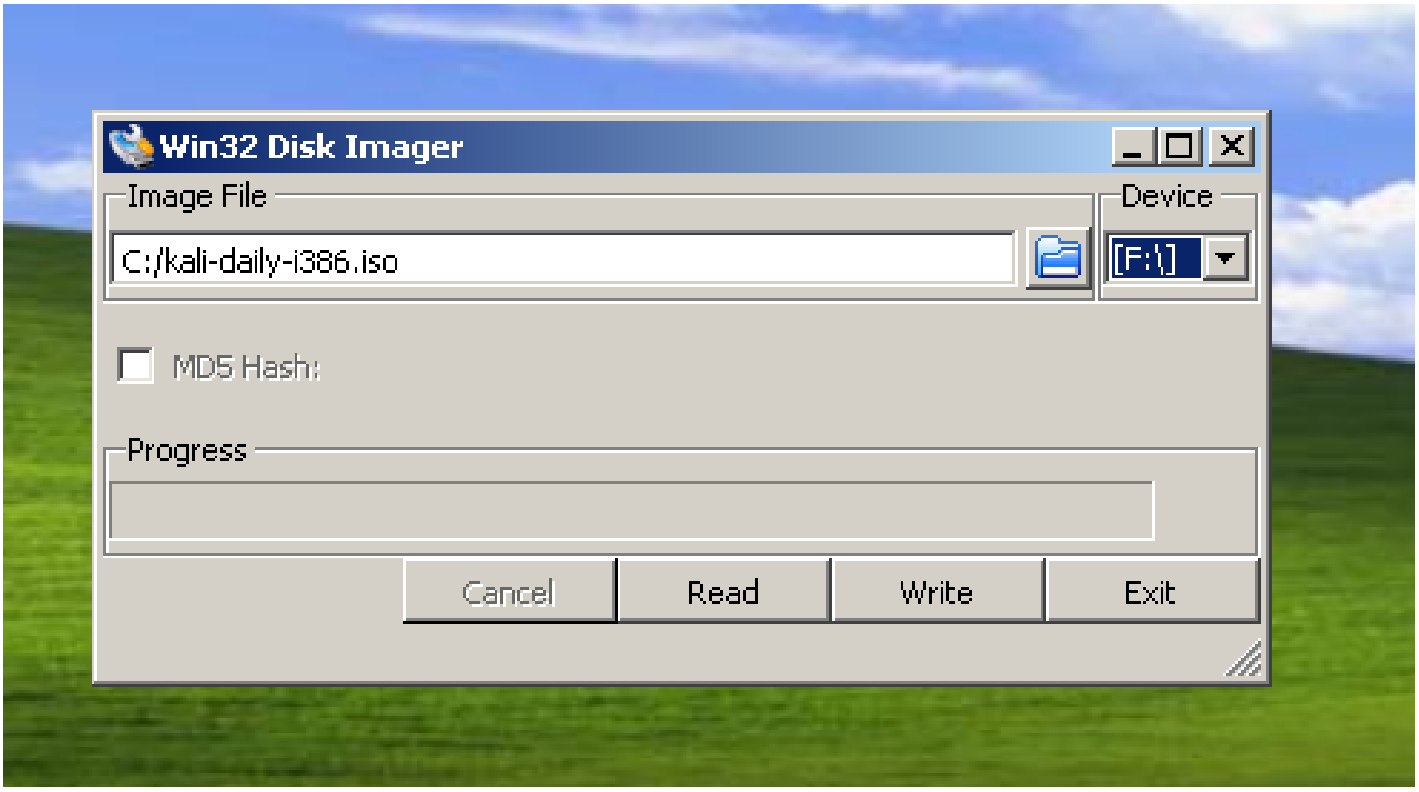
1. A *verified* copy of the appropriate ISO image of the latest Kali build image for the system you’ll be running it on: see the details on downloading official Kali Linux images.
2. If you’re running under Windows, you’ll also need to download the Win32 Disk Imager utility. On Linux and OS X, you can use the **dd** command, which is pre-installed on those platforms.
3. A USB thumb drive, 4GB or larger. (Systems with a direct SD card slot can use an SD card with similar capacity. The procedure is identical.)

Kali Linux Live USB Install Procedure

The specifics of this procedure will vary depending on whether you’re doing it on a Windows, Linux, or OS X system.

Creating a Bootable Kali USB Drive on Windows

1. Plug your USB drive into an available USB port on your Windows PC, note which drive designator (e.g. “F:\”) it uses once it mounts, and launch the Win32 Disk Imager software you downloaded.
2. Choose the Kali Linux ISO file to be imaged and verify that the USB drive to be overwritten is the correct one. Click the “Write” button.



3. Once the imaging is complete, safely eject the USB drive from the Windows machine. You can now use the USB device to boot into Kali Linux.

Creating a Bootable Kali USB Drive on Linux

Creating a bootable Kali Linux USB key in a Linux environment is easy. Once you've downloaded and verified your Kali ISO file, you can use the **dd** command to copy it over to your USB stick using the following procedure. Note that you'll need to be running as root, or to execute the **dd** command with sudo. The following example assumes a Linux Mint 17.1 desktop — depending on the distro you're using, a few specifics may vary slightly, but the general idea should be very similar.

Kali Linux Live USB Persistence

Adding Persistence to a Kali Linux “Live” USB Drive

Kali Linux “Live” has two options in the default boot menu which enable persistence — the preservation of data on the “Kali Live” USB drive — across reboots of “Kali Live”. This can be an extremely useful enhancement, and enables you to retain documents, collected testing results, configurations, etc., when running Kali Linux “Live” from the USB drive, even across different systems. The persistent data is stored in its own partition on the USB drive, which can also be optionally LUKS-encrypted.

To make use of the USB persistence options at boot time, you’ll need to do some additional setup on your “Kali Linux Live” USB drive; this article will show you how.

This guide assumes that you have already created a Kali Linux “Live” USB drive as described in the section on that subject. For the purposes of this article, we’ll assume you’re working on a Linux-based system.

You’ll need to have root privileges to do this procedure, or the ability to escalate your privileges with the command “sudo su”. In this example, we assume

- you are running as the **root user**
- your USB drive is **/dev/sdb**
- your USB drive has a capacity of **at least 8GB** — the Kali Linux image takes over 3GB, and for this guide, we’ll be creating a new partition of about 4GB to store our persistent data in.

In this example, we’ll create a new partition to store our persistent data into, starting right above the second Kali Live partition and ending at 7GB, put an ext3 file system onto it, and create a **persistence.conf** file on the

new partition.

1. First, begin by imaging the latest Kali Linux ISO (currently 2016.2) to your USB drive as described in this article. We're going to assume that the two partitions created by the imaging are **/dev/sdb1** and **/dev/sdb2**. This can be verified with the command "**fdisk -l**".

2. Create and format an additional partition on the USB drive.
   First, let's create the new partition in the empty space above our Kali Live partitions.

```
end=7gb
read start _ < <(du -bcm kali-linux-2016.2-amd64.iso | tail -1); echo $start
parted /dev/sdb mkpart primary $start $end
```

   The **parted** command may advise you that it can't use the exact start values you specified; if so, accept the suggested value instead. If advised that the partition isn't placed at an optimal location, "ignore" it. When parted completes, the new partition should have been created at **/dev/sdb3**; again, this can be verified with the command "**fdisk -l**".
3. Next, create an **ext3** file system in the partition and label it "persistence".

```
mkfs.ext3 -L persistence /dev/sdb3
e2label /dev/sdb3 persistence
```

4. Create a mount point, mount the new partition there, and then create the configuration file to enable persistence. Finally, unmount the partition.

```
mkdir -p /mnt/my_usb
mount /dev/sdb3 /mnt/my_usb
echo "/ union" > /mnt/my_usb/persistence.conf
umount /dev/sdb3
```

Adding USB Persistence with LUKS Encryption

Alternatively, you can create a LUKS-encrypted persistent storage area. This adds an extra layer of security to your sensitive files when traveling with Kali Live on USB devices. In the following example, we'll create a new partition to store our persistent data into, starting right above the second Kali Live partition and ending at 7GB, set up LUKS encryption on the new partition, put an ext3 file system onto it, and create a **persistence.conf** file on it.

1. Image the latest Kali Linux ISO (currently 2016.2) to your USB drive as described in this article.
2. Create the new partition in the empty space above our Kali Live partitions.

```
end=7gb
read start _ < <(du -bcm kali-linux-2016.2-amd64.iso | tail -1); echo $start
parted /dev/sdb mkpart primary $start $end
```

   The parted command may advise you that it can't use the exact start value you specified; if so, accept the suggested value instead. If advised that the partition isn't placed at an optimal location, "ignore" it. When parted completes, the new partition should have been created at /dev/sdb3; again, this can be verified with the command "**fdisk -l**".
3. Initialize the LUKS encryption on the newly-created partition. You'll be warned that this will overwrite any data on the partion. When prompted whether you want to proceed, type "YES" (all upper case). Enter your selected passphrase twice when asked to do so, and be sure to pick a passphrase you're going to remember: if you forget it, your data will still be persistent, just irretrievable (and unusable).

```
cryptsetup --verbose --verify-passphrase luksFormat /dev/sdb3
cryptsetup luksOpen /dev/sdb3 my_usb
```

4. Create the ext3 filesystem, and label it "persistence".

```
mkfs.ext3 -L persistence /dev/mapper/my_usb
e2label /dev/mapper/my_usb persistence
```

5. Create a mount point, mount our new encrypted partition there, set up the **persistence.conf** file, and unmount the partition.

```
mkdir -p /mnt/my_usb
```

```
mount /dev/mapper/my_usb /mnt/my_usb
echo "/ union" > /mnt/my_usb/persistence.conf
umount /dev/mapper/my_usb
```

6. Close the encrypted channel to our persistence partition.

```
cryptsetup luksClose /dev/mapper/my_usb
```

That’s really all there is to it! To use the persistent data features, simply plug your USB drive into the computer you want to boot up Kali Live on — make sure your BIOS is set to boot from your USB device — and fire it up. When the Kali Linux boot screen is displayed, choose the persistent option you set up on your USB drive, either normal or encrypted.

Live Build a Custom Kali ISO

An Introduction to Building Your Own Kali ISO

Building a customized Kali ISO is easy, fun, and rewarding. You can configure virtually any aspect of your Kali ISO build using the Debian live-build scripts. These scripts allow developers to easily build live system images by providing a framework that uses a configuration set to automate and customize all aspects of building the image. The Kali Linux development team has adopted these scripts and they're used to produce the official Kali ISO releases.

Where Should You Build Your ISO?

Ideally, you should build your custom Kali ISO from **within a pre-existing Kali environment**.

Getting Ready — Setting up the live-build system

We first need to prepare the Kali ISO build environment by installing and setting up live-build and its requirements with the following commands:

```
apt install -y curl git live-build cdebootstrap
git clone git://gitlab.com/kalilinux/build-scripts/live-build-config.git
```

Now you can simply build an updated Kali ISO by entering the "live-build-config" directory and running our **build.sh** wrapper script, as follows:

```
cd live-build-config/
./build.sh --verbose
```

The "build.sh" script will take a while to complete, as it downloads all of the required packages needed to create your ISO. Good time for a coffee.

Configuring the Kali ISO Build (Optional)

If you want to customize your Kali Linux ISO, this section will explain some of the details. Through the **kali-config** directory, the Kali Linux live build supports a wide range of customization options, which are well-documented on the Debian live build 4.x page. However, for the impatient, here are some of the highlights.

Building Kali with Different Desktop Environments

Since Kali 2.0, we now support built in configurations for various desktop environments, including KDE, Gnome, E17, I3WM, LXDE, MATE and XFCE. To build any of these, you would use syntax similar to the following:

```
# These are the different Desktop Environment build options:
#./build.sh --variant {gnome,kde,xfce,mate,e17,lxde,i3wm} --verbose

# To build a KDE ISO:
./build.sh --variant kde --verbose
# To build a MATE ISO:
./build.sh --variant mate --verbose

#...and so on.
```

Controlling the packages included in your build

The list of packages included in your build will be present in the the respective kali-$variant directory. For example, if you're building a default Gnome ISO, you would use the following package lists file – **kali-config/variant-gnome/package-lists/kali.list.chroot**. By default, this list includes the "kali-linux-full" metapackage, as well as some others. These can be commented out and replaced with a manual list of packages to include in the ISO for greater granularity.

Build hooks, binary and chroot

Live-build hooks allows us to hook scripts in various stages of the Kali ISO live build. For more detailed information about hooks and how to use them, refer to the live build manual. As an example, we recommend you check out the existing hooks in **kali-config/common/hooks/**.

Overlaying files in your build

You have the option to include additional files or scripts in your build by overlaying them on the existing

filesystem, inside the **includes.{chroot,binary,installer}** directories, respectively. For example, if we wanted to include our own custom script into the **/root/** directory of the ISO (this would correspond to the "chroot" stage), then we would drop this script file in the **kali-config/common/includes.chroot/** directory before building the ISO.

Building a Kali Linux ISO for older i386 architectures

The Kali Linux i386 ISO has PAE enabled. If you require a default kernel for older hardware with PAE disabled, you will need to rebuild a Kali Linux ISO. The rebuilding process is much the same as described above, except that the **686-pae** parameter that needs to be changed to **586** in **auto/config** as follows. First, install the prerequisites.

```
apt install -y git live-build cdebootstrap debootstrap
git clone git://gitlab.com/kalilinux/build-scripts/live-build-config.git
```

Next, make the change in auto/config for the appropriate architecture:

```
cd live-build-config/
sed -i 's/686-pae/686/g' auto/config
```

Finally, run your build.

```
./build.sh --arch i386 --verbose
```

Building Kali on Non-Kali Debian Based Systems

You can easily run live-build on Debian based systems other than Kali. The instructions below have been tested to work with both Debian and Ubuntu.

First, we prep the system by ensuring it is fully updated, then proceed to download the Kali archive keyring and live-build packages.

```
sudo apt update
sudo apt upgrade
cd /root/

wget http://http.kali.org/pool/main/k/kali-archive-keyring/kali-archive-keyring_2018.1_all.deb
wget https://archive.kali.org/kali/pool/main/l/live-build/live-build_20180618kali1_all.deb
```

With that completed, we install some additional dependencies and the previously downloaded files.

```
sudo apt install -y git live-build cdebootstrap debootstrap curl
sudo dpkg -i kali-archive-keyring_2018.1_all.deb
sudo dpkg -i live-build_20180618kali1_all.deb
```

With the environment all prepared, we start the live-build process by setting up the build script and checking out the build config.

```
cd /usr/share/debootstrap/scripts/
echo "default_mirror http://http.kali.org/kali"; sed -e
"s/debian-archive-keyring.gpg/kali-archive-keyring.gpg/g" sid > /tmp/kali
sudo mv /tmp/kali .
sudo ln -s kali kali-rolling

cd ~
git clone git://gitlab.com/kalilinux/build-scripts/live-build-config.git

cd live-build-config/
```

At this point, we have to edit the `build.sh` script to bypass a version check. We do this by commenting out the “exit 1” below.

```
# Check we have a good debootstrap
ver_debootstrap=$(dpkg-query -f '${Version}' -W debootstrap)
if dpkg --compare-versions "$ver_debootstrap" lt "1.0.97"; then
if ! echo "$ver_debootstrap" | grep -q kali; then
echo "ERROR: You need debootstrap >= 1.0.97 (or a Kali patched debootstrap). Your current version:
$ver_debootstrap" >&2
exit 1
fi
fi
```

With that change made, the script should like as follows:

```
# Check we have a good debootstrap
ver_debootstrap=$(dpkg-query -f '${Version}' -W debootstrap)
if dpkg --compare-versions "$ver_debootstrap" lt "1.0.97"; then
if ! echo "$ver_debootstrap" | grep -q kali; then
echo "ERROR: You need debootstrap >= 1.0.97 (or a Kali patched debootstrap). Your current version:
$ver_debootstrap" >&2
# exit 1
fi
fi
```

At this point, we can build our ISO as normal

```
sudo ./build.sh --variant light --verbose
```

© Offensive Security 2019

03. Installing Kali Linux

Kali Linux Hard Disk Install

Kali Linux Installation Requirements

Installing Kali Linux on your computer is an easy process. First, you'll need compatible computer hardware. Kali is supported on i386, amd64, and ARM (both armel and armhf) platforms. The hardware requirements are minimal as listed below, although better hardware will naturally provide better performance. The i386 images have a default PAE kernel, so you can run them on systems with over 4GB of RAM. Download Kali Linux and either burn the ISO to DVD, or prepare a USB stick with Kali Linux Live as the installation medium. If you do not have a DVD drive or USB port on your computer, check out the Kali Linux Network Install.

Installation Prerequisites

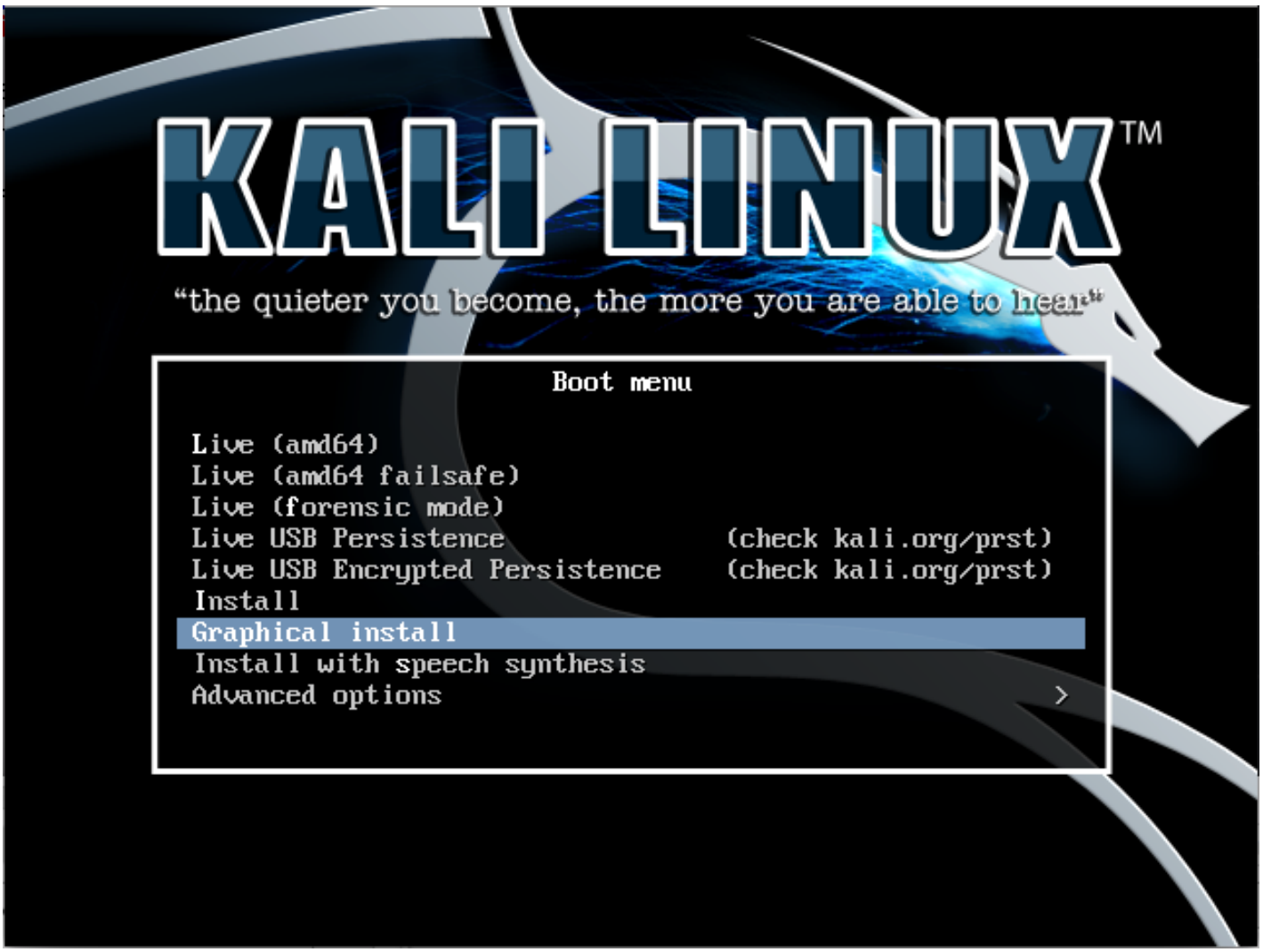
- A minimum of 20 GB disk space for the Kali Linux install.
- RAM for i386 and amd64 architectures, minimum: 1GB, recommended: 2GB or more.
- CD-DVD Drive / USB boot support

Preparing for the Installation

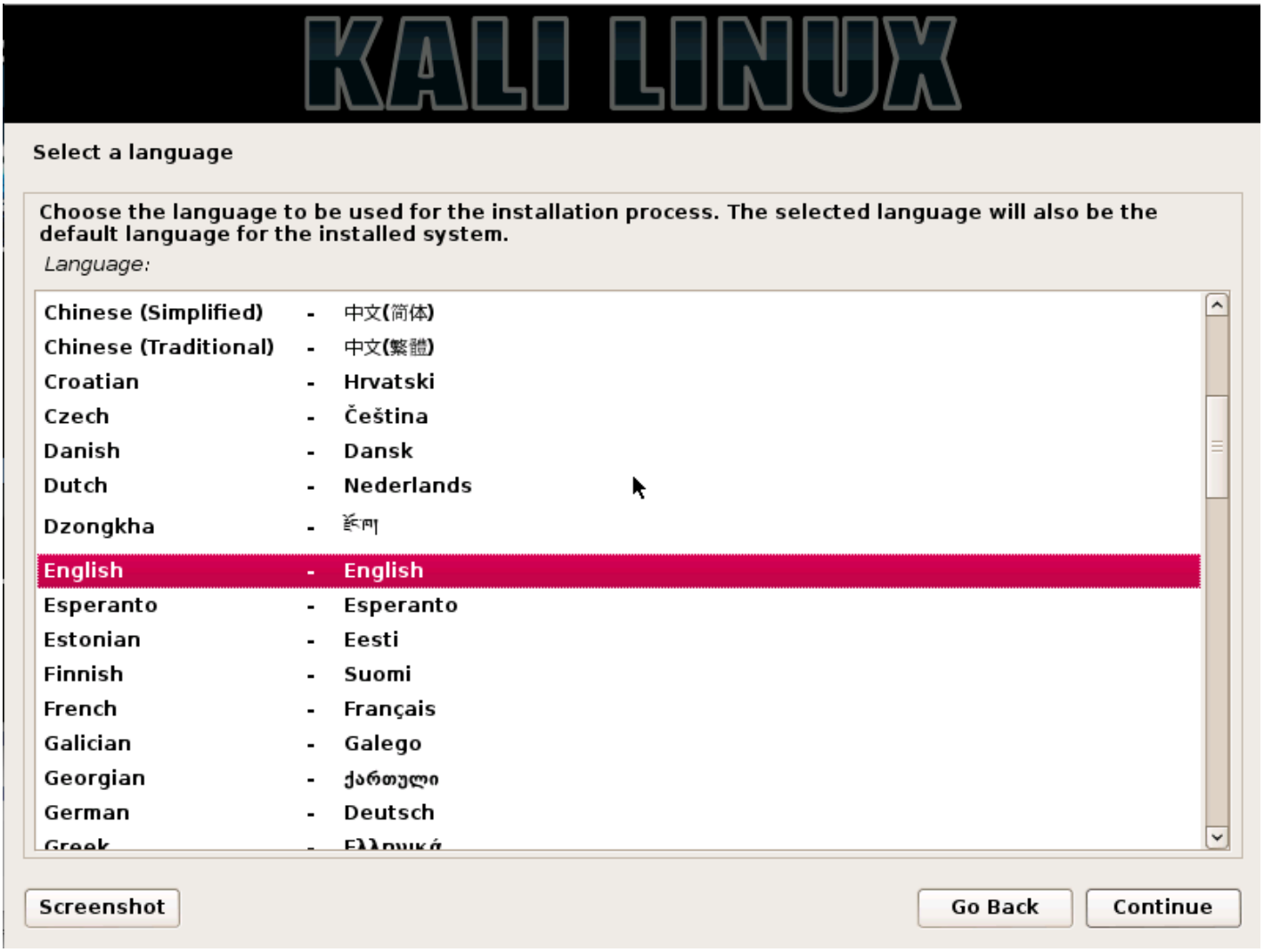
1. Download Kali linux.
2. Burn The Kali Linux ISO to DVD or Image Kali Linux Live to USB.
3. Ensure that your computer is set to boot from CD / USB in your BIOS.

Kali Linux Installation Procedure

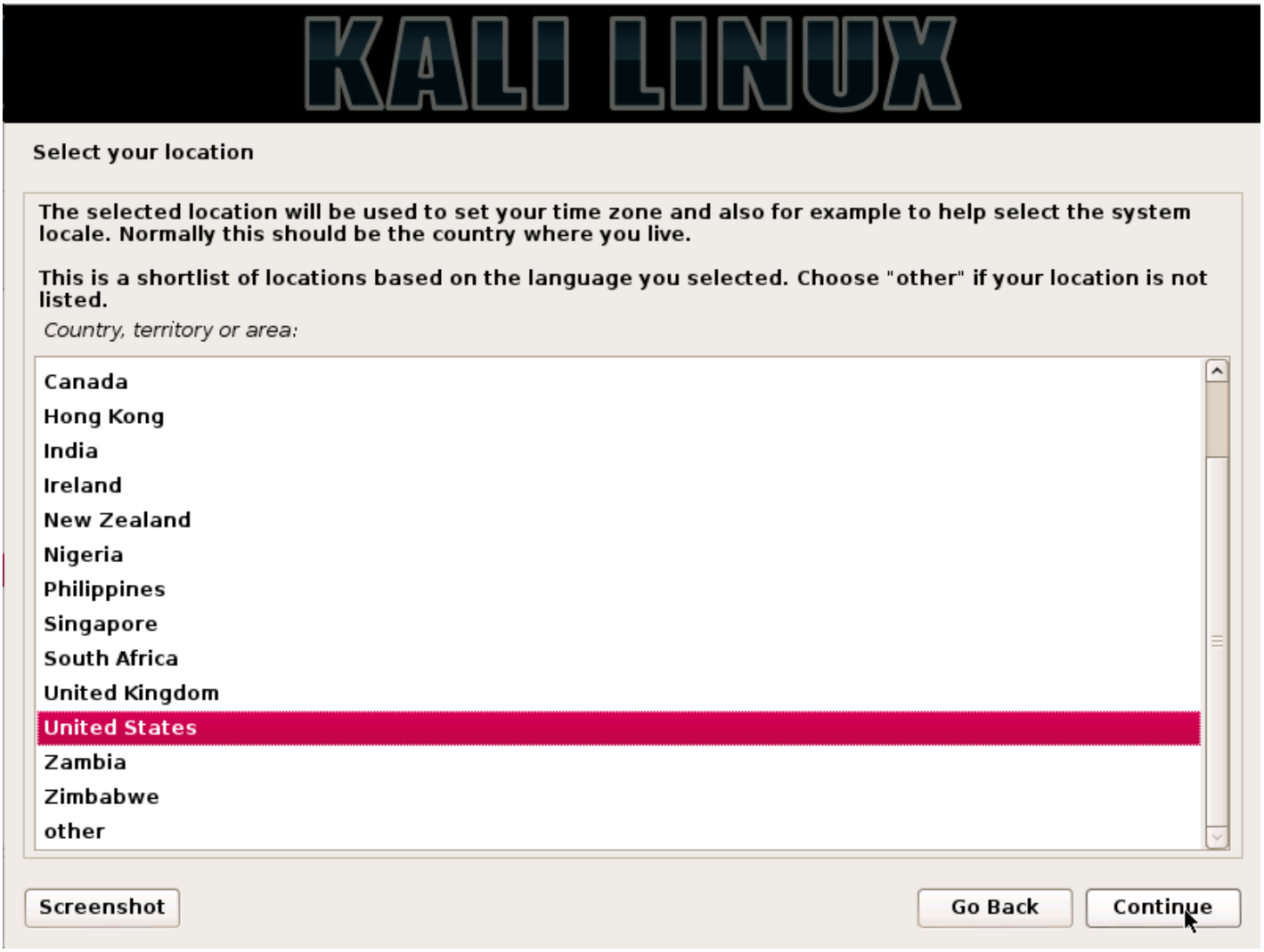
1. To start your installation, boot with your chosen installation medium. You should be greeted with the Kali Boot screen. Choose either *Graphical* or *Text-Mode* install. In this example, we chose a GUI install.



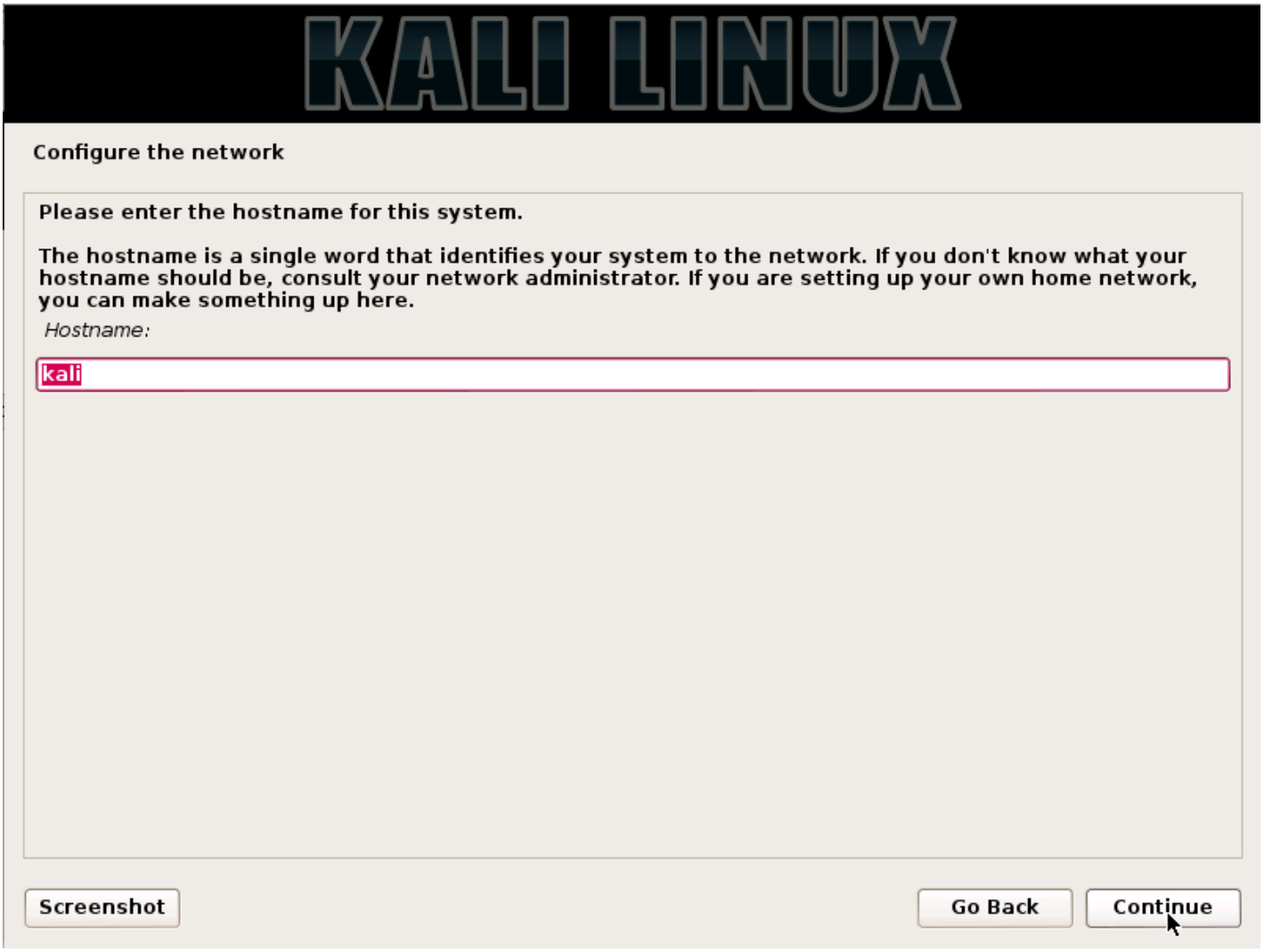
2. Select your preferred language and then your country location. You'll also be prompted to configure your keyboard with the appropriate keymap.



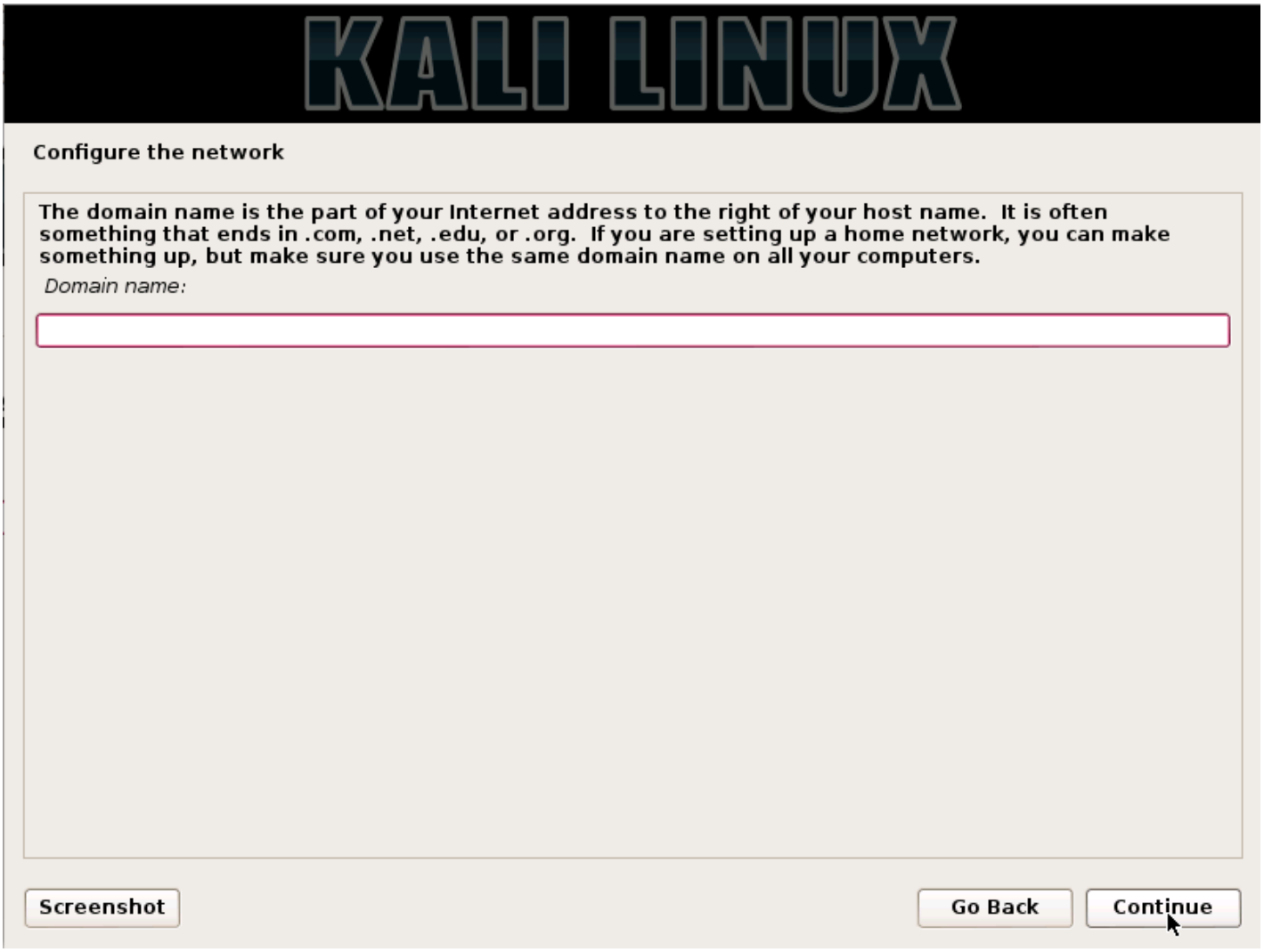
3. Specify your geographic location.



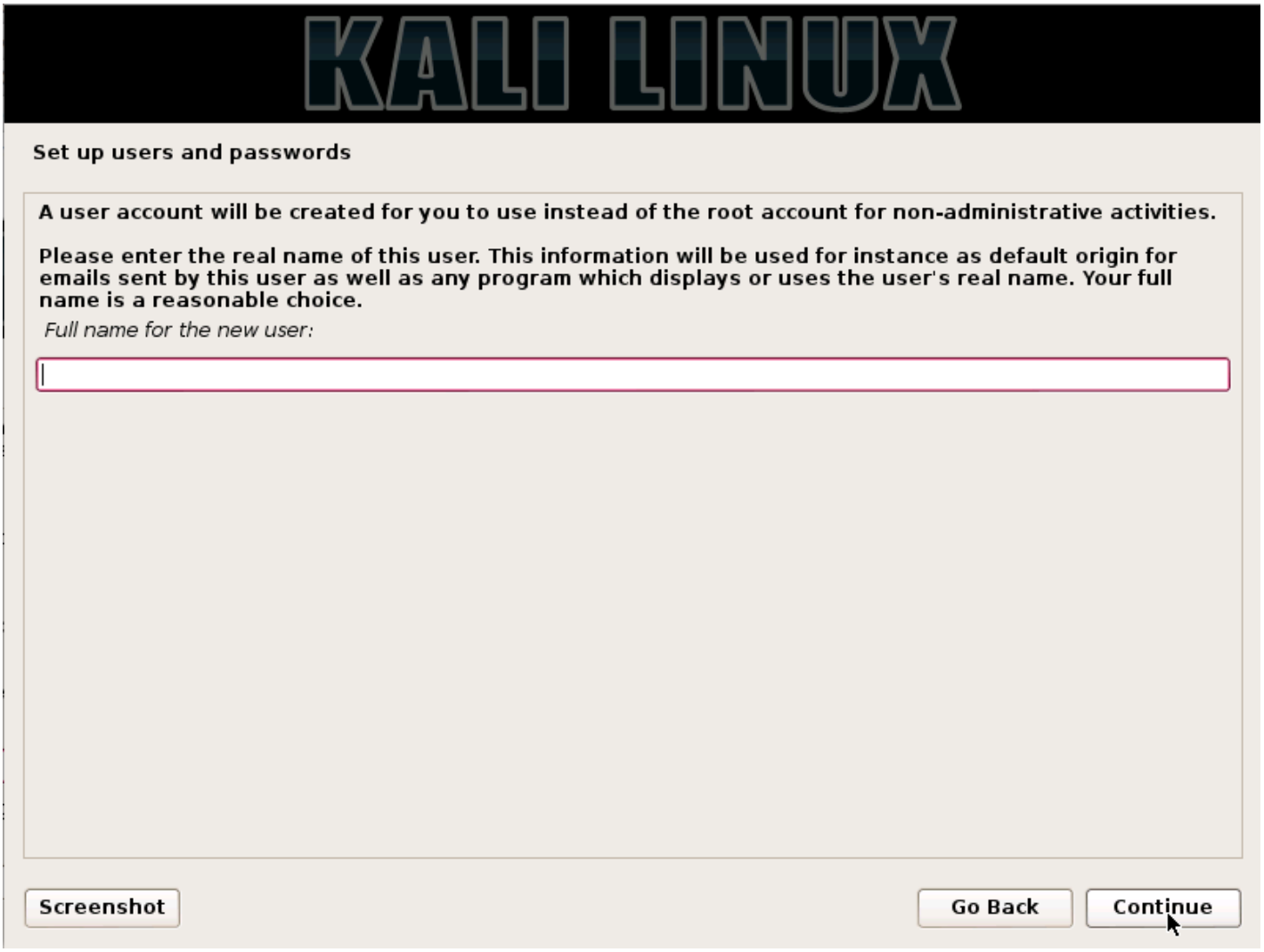
4. The installer will copy the image to your hard disk, probe your network interfaces, and then prompt you to enter a hostname for your system. In the example below, we've entered "kali" as our hostname.



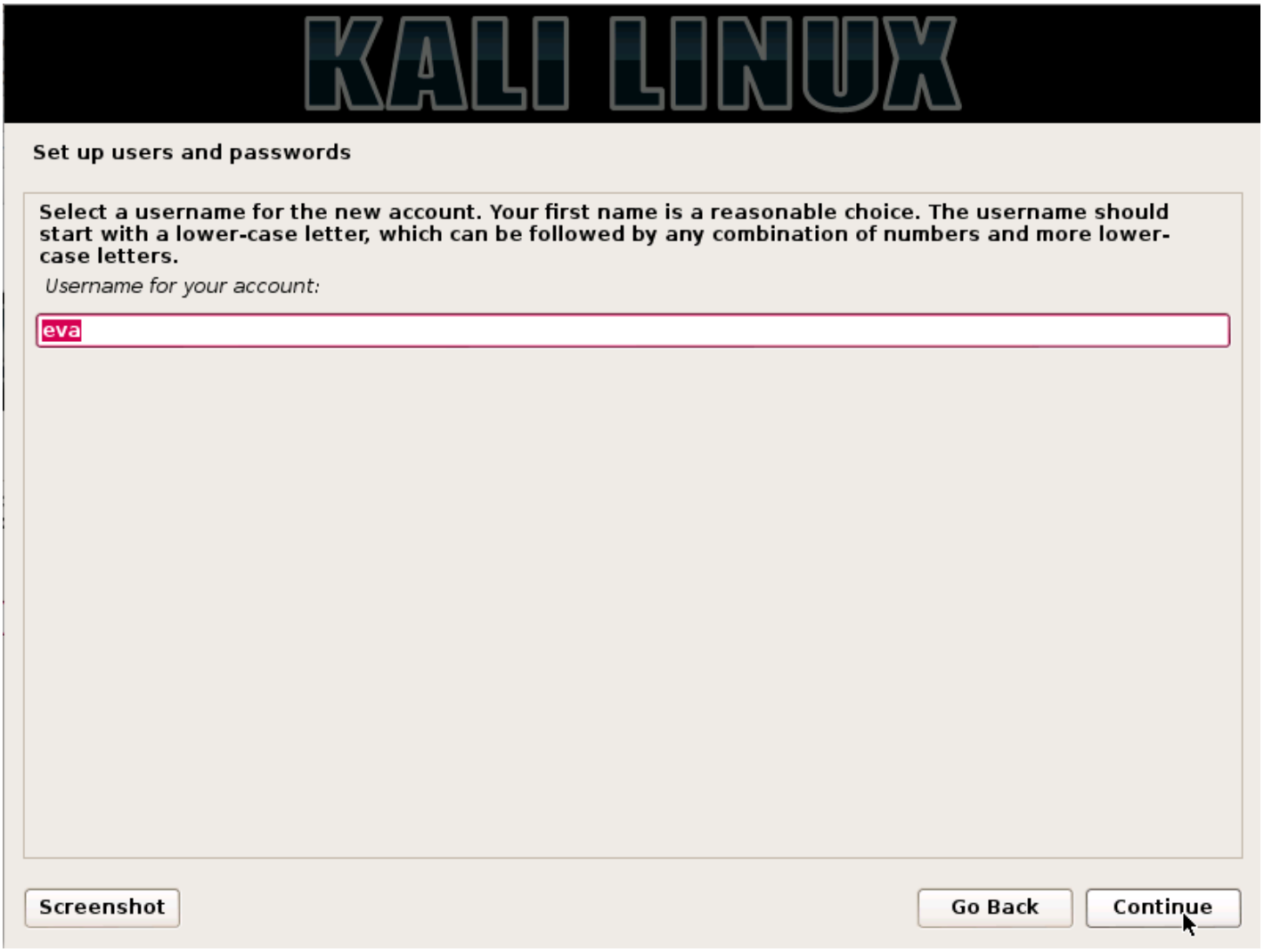
5. You may optionally provide a default domain name for this system to use.



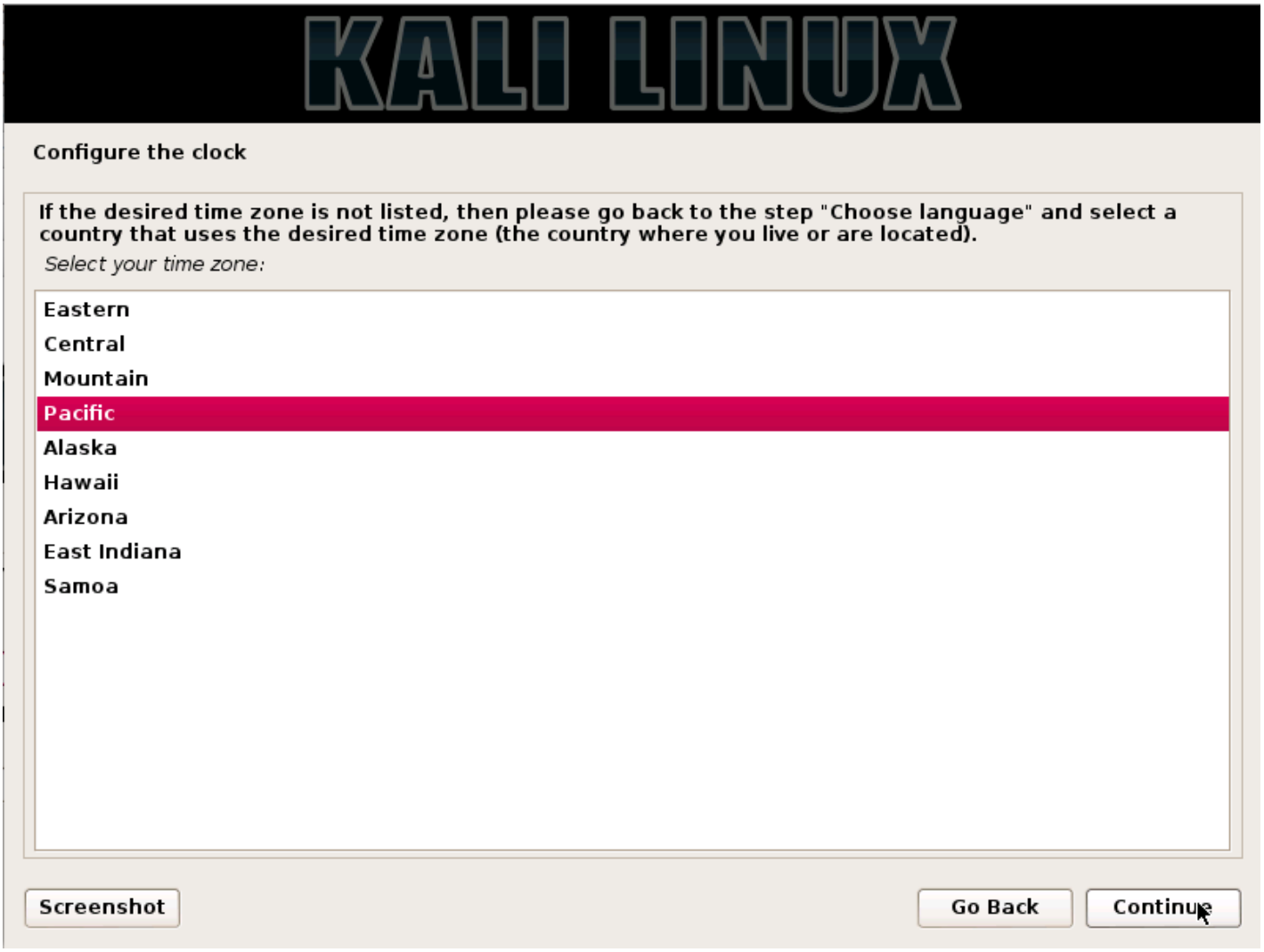
6. Next, provide a full name for a non-root user for the system.



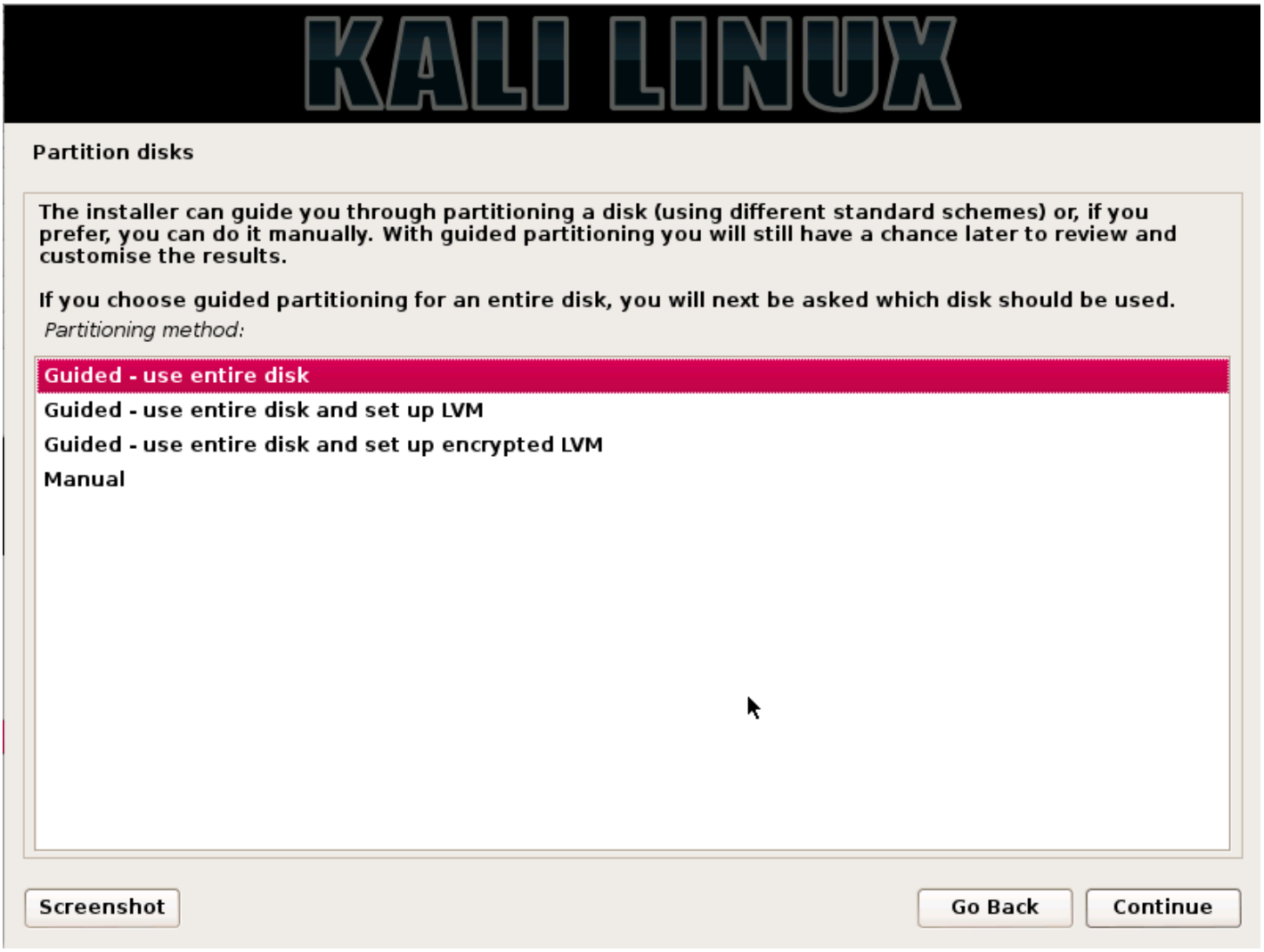
7. A default user ID will be created, based on the full name you provided. You can change this if you like.



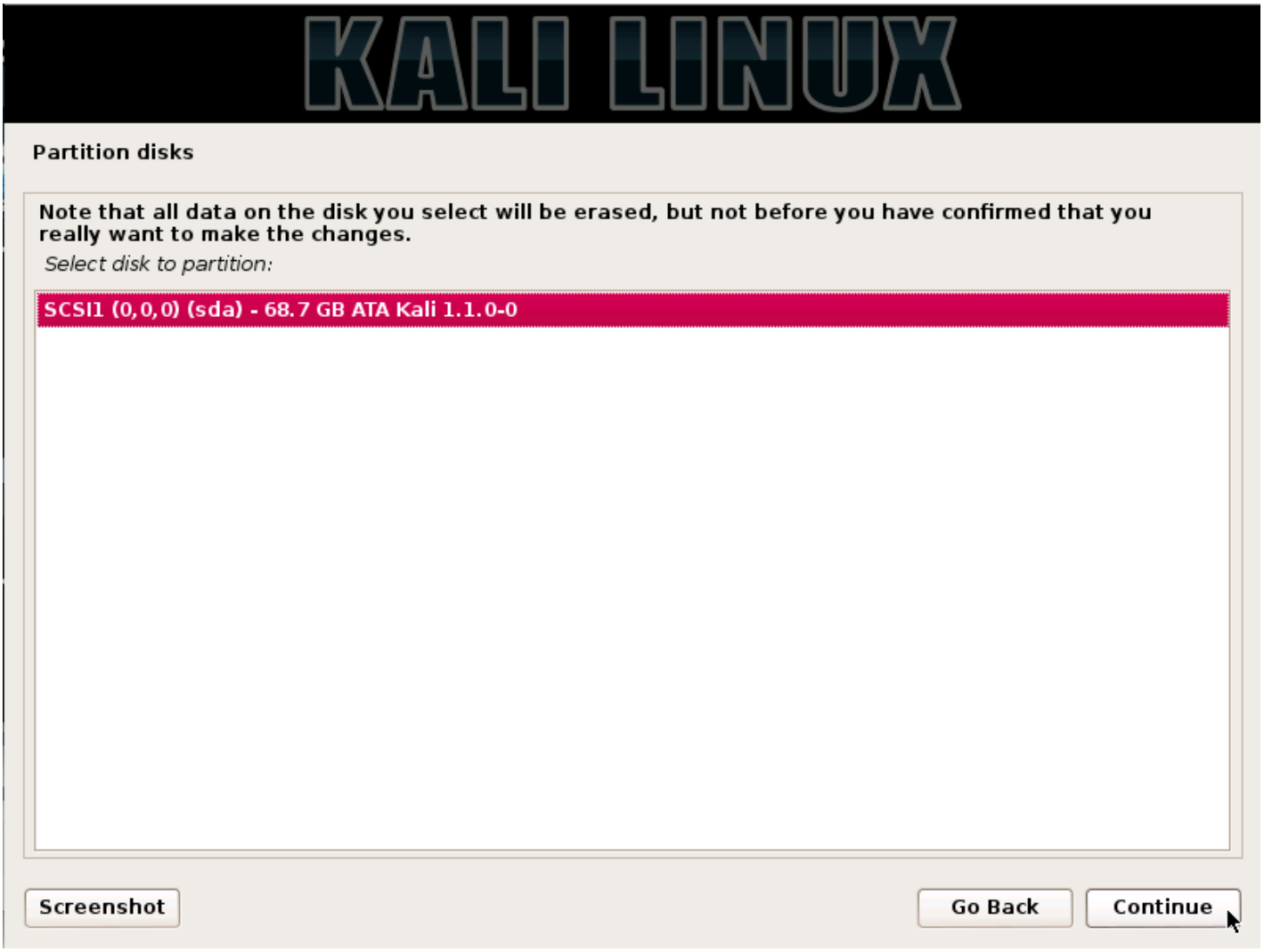
8. Next, set your time zone.



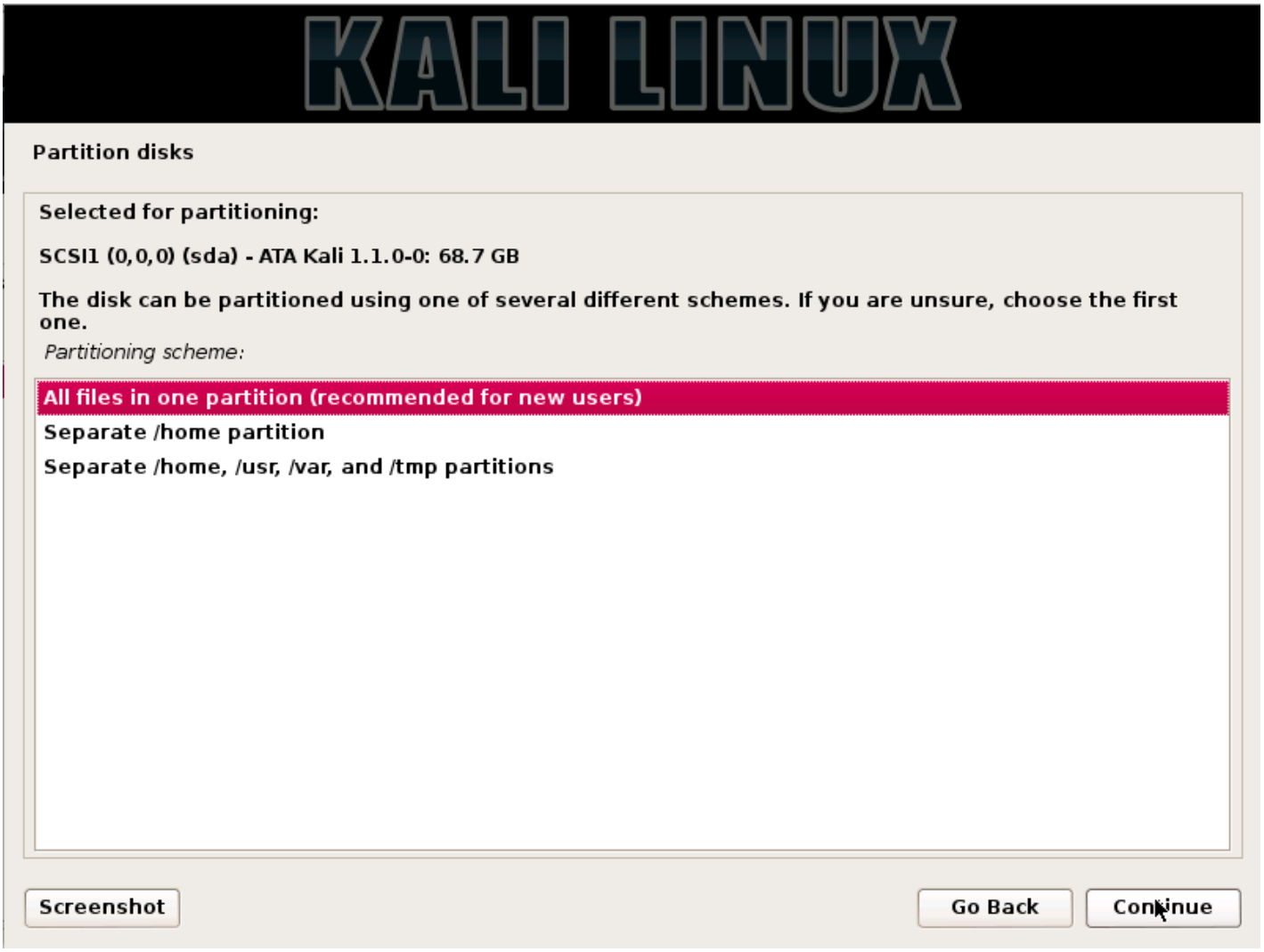
9. The installer will now probe your disks and offer you four choices. In our example, we're using the entire disk on our computer and not configuring LVM (logical volume manager). Experienced users can use the "Manual" partitioning method for more granular configuration options.



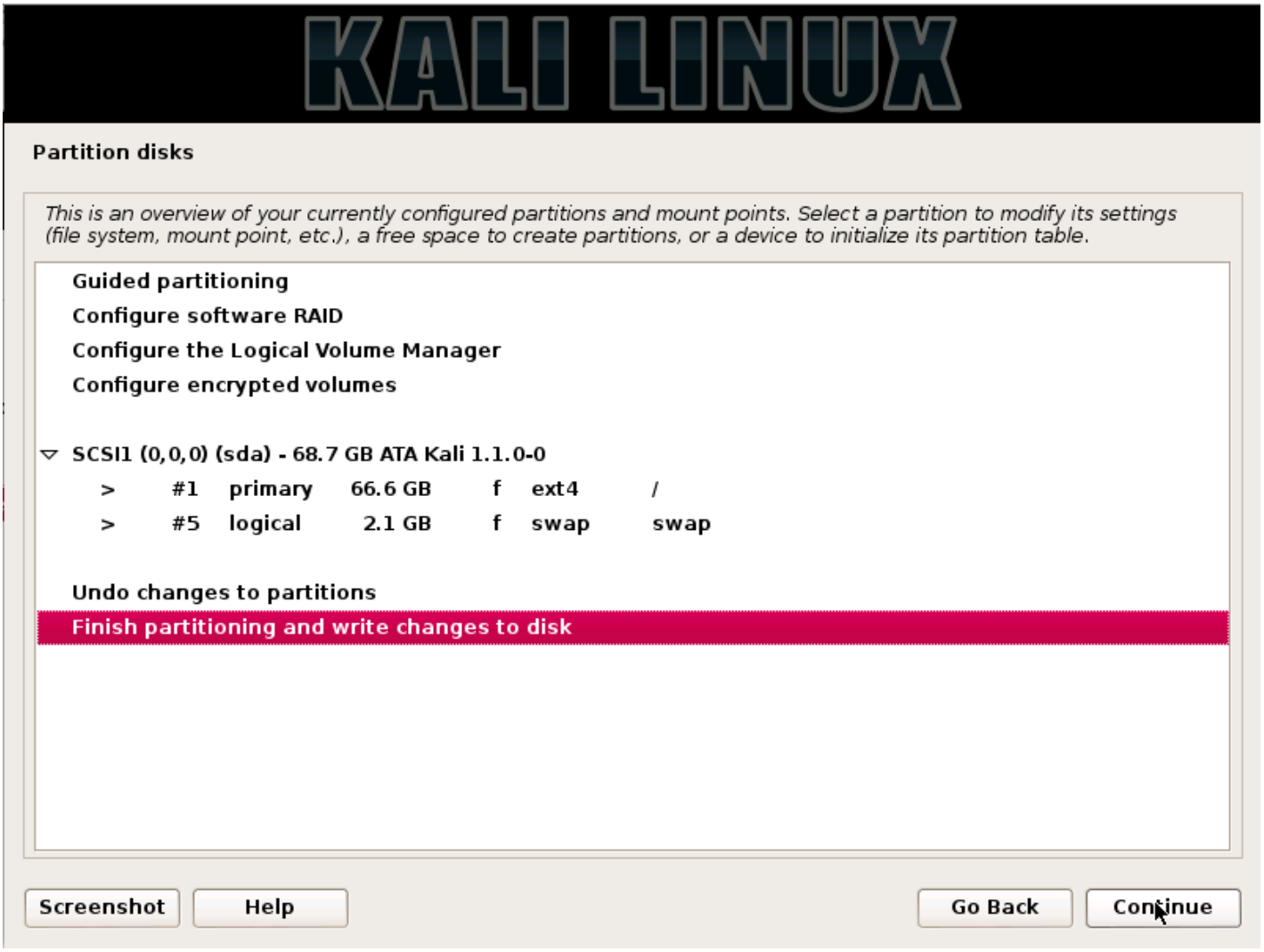
10. Select the disk to be partitioned.



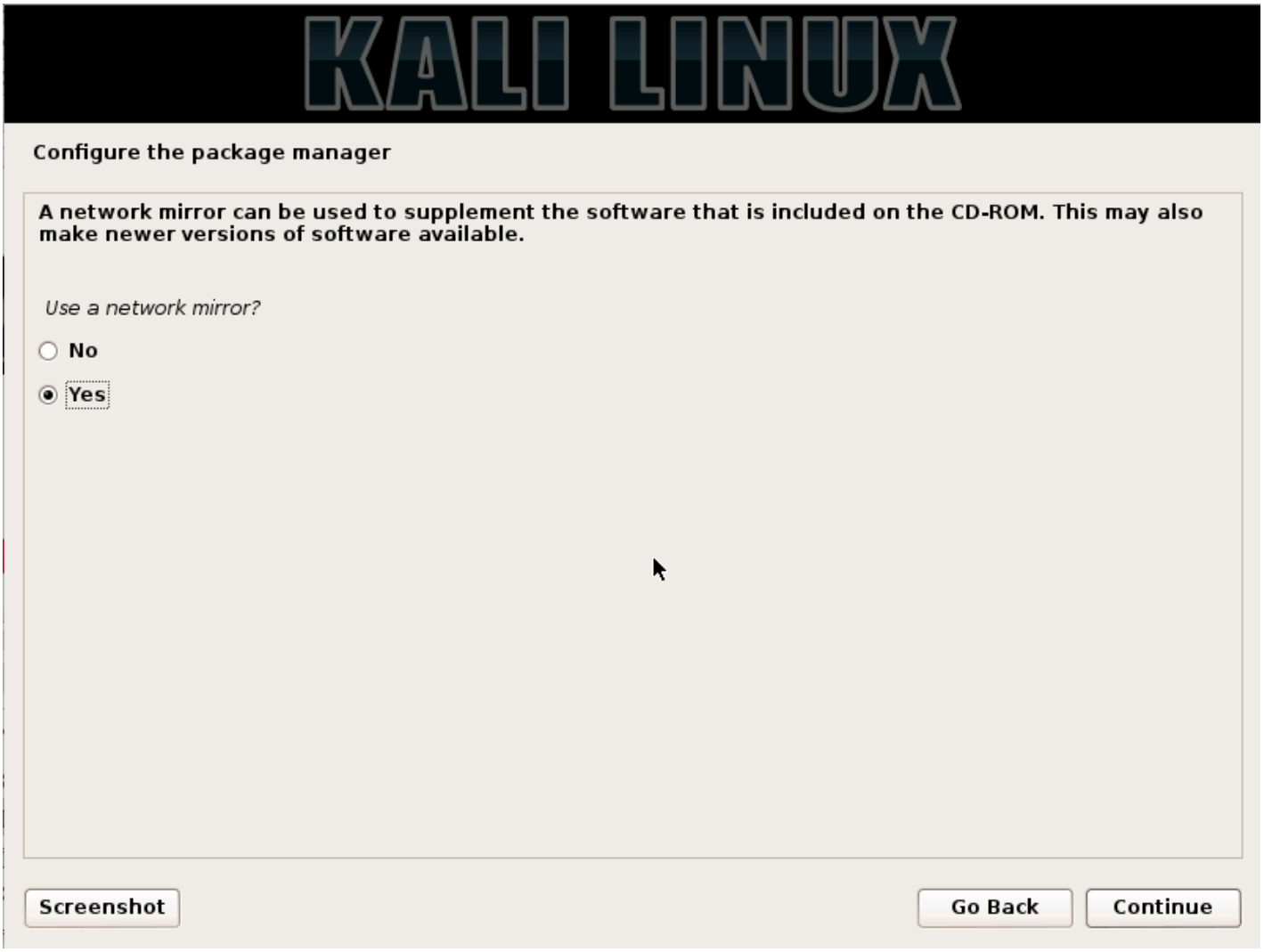
11. Depending on your needs, you can choose to keep all your files in a single partition — the default — or to have separate partitions for one or more of the top-level directories. If you're not sure which you want, you want "All files in one partition".



12. Next, you'll have one last chance to review your disk configuration before the installer makes irreversible changes. After you click *Continue*, the installer will go to work and you'll have an almost finished installation.

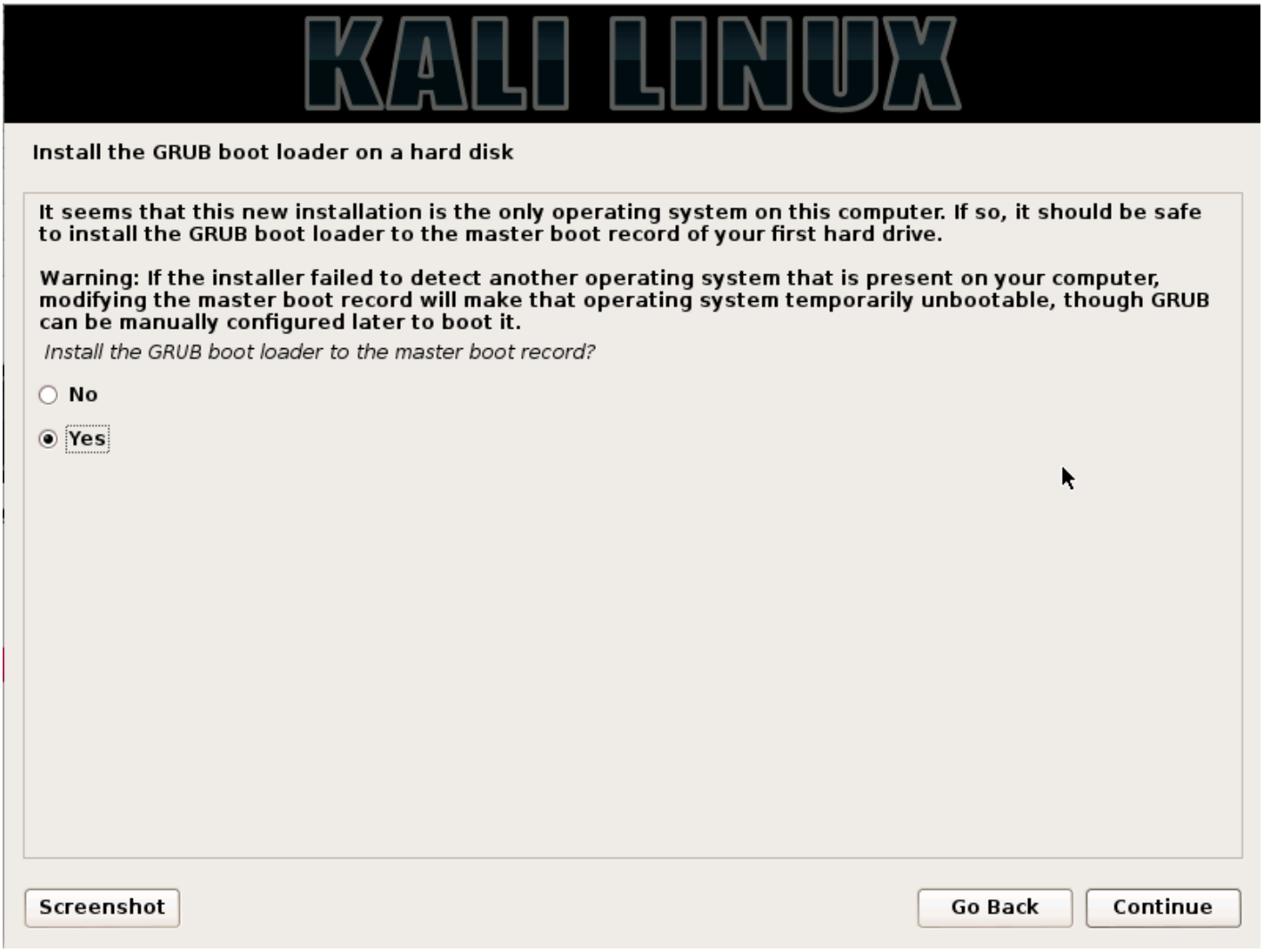


13. Configure network mirrors. Kali uses a central repository to distribute applications. You'll need to enter any appropriate proxy information as needed.

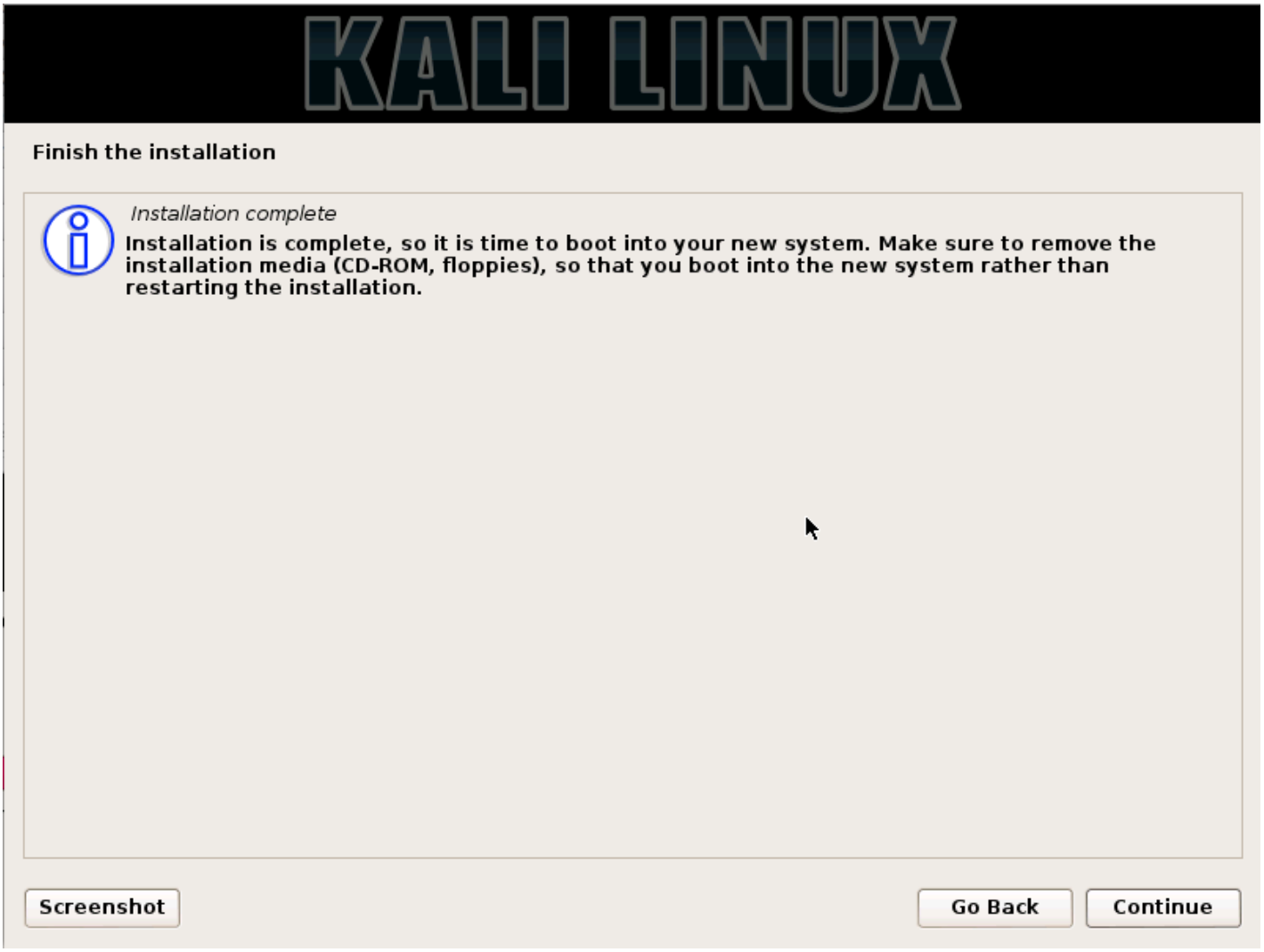


NOTE! If you select “NO” in this screen, you will **NOT** be able to install packages from Kali repositories.

14. Next, install GRUB.



15. Finally, click Continue to reboot into your new Kali installation.



Post Installation

Now that you've completed installing Kali Linux, it's time to customize your system. The Kali General Use section of our site has more information and you can also find tips on how to get the most out of Kali in our User Forums.

Dual Boot Kali with Windows

Kali Linux Dual Boot with Windows

Installing Kali alongside a Windows installation can be quite useful. However, you need to exercise caution during the setup process. First, make sure that you've backed up any important data on your Windows installation. Since you'll be modifying your hard drive, you'll want to store this backup on external media. Once you've completed the backup, we recommend you peruse Kali Linux Hard Disk Install, which explains the normal procedure for a basic Kali install.

In our example, we will be installing Kali Linux alongside an installation of Windows 7, which is currently taking up 100% of the disk space in our computer. We will start by resizing our current Windows partition to occupy less space and then proceed to install Kali Linux in the newly-created empty partition.

Download Kali Linux and either burn the ISO to DVD, or prepare a USB stick with Kali linux Live as the installation medium. If you do not have a DVD or USB port on your computer, check out the Kali Linux Network Install. Ensure you have:

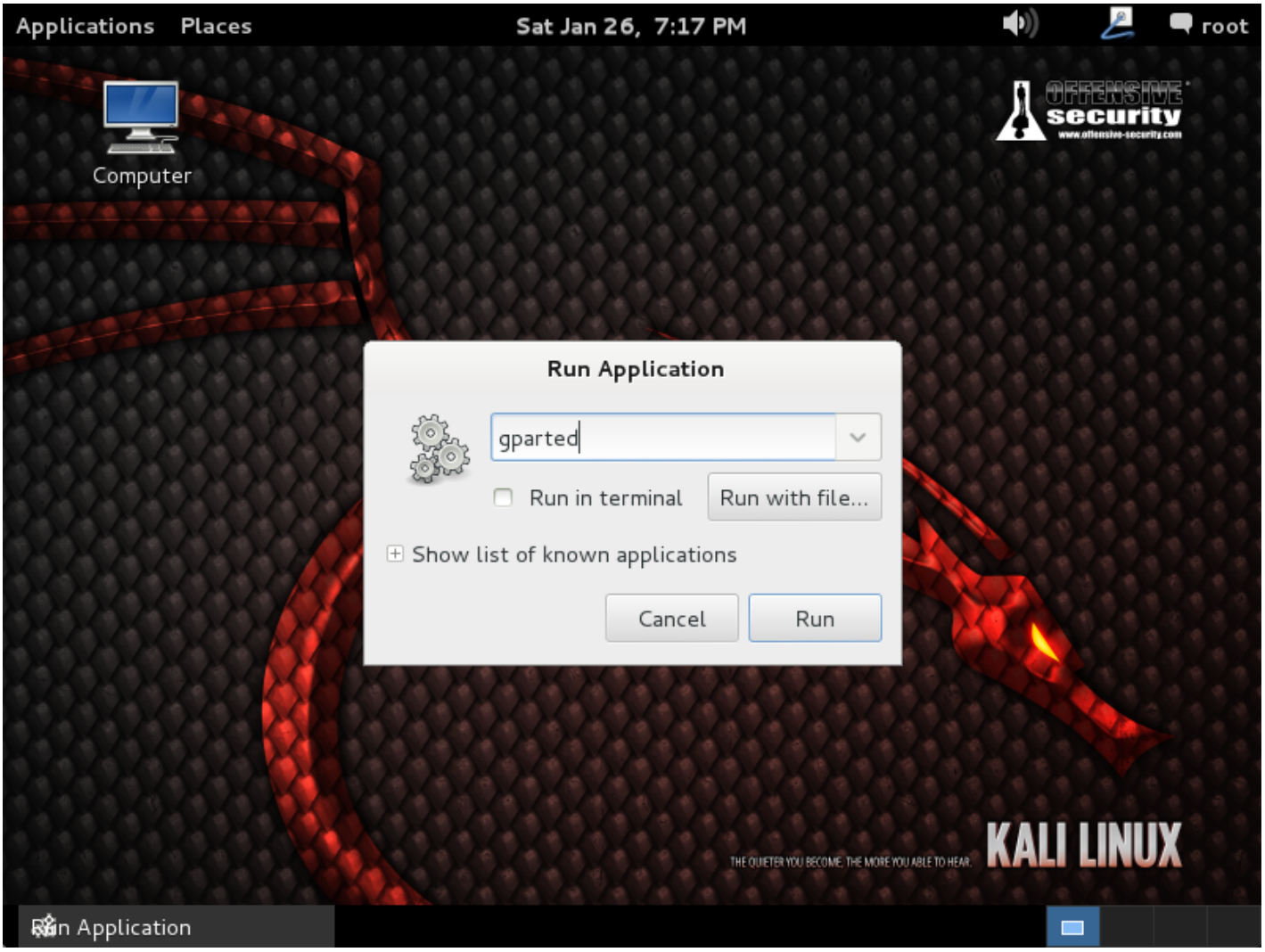
- Minimum of 20 GB free disk space on Windows
- CD-DVD / USB boot support

Preparing for the Installation

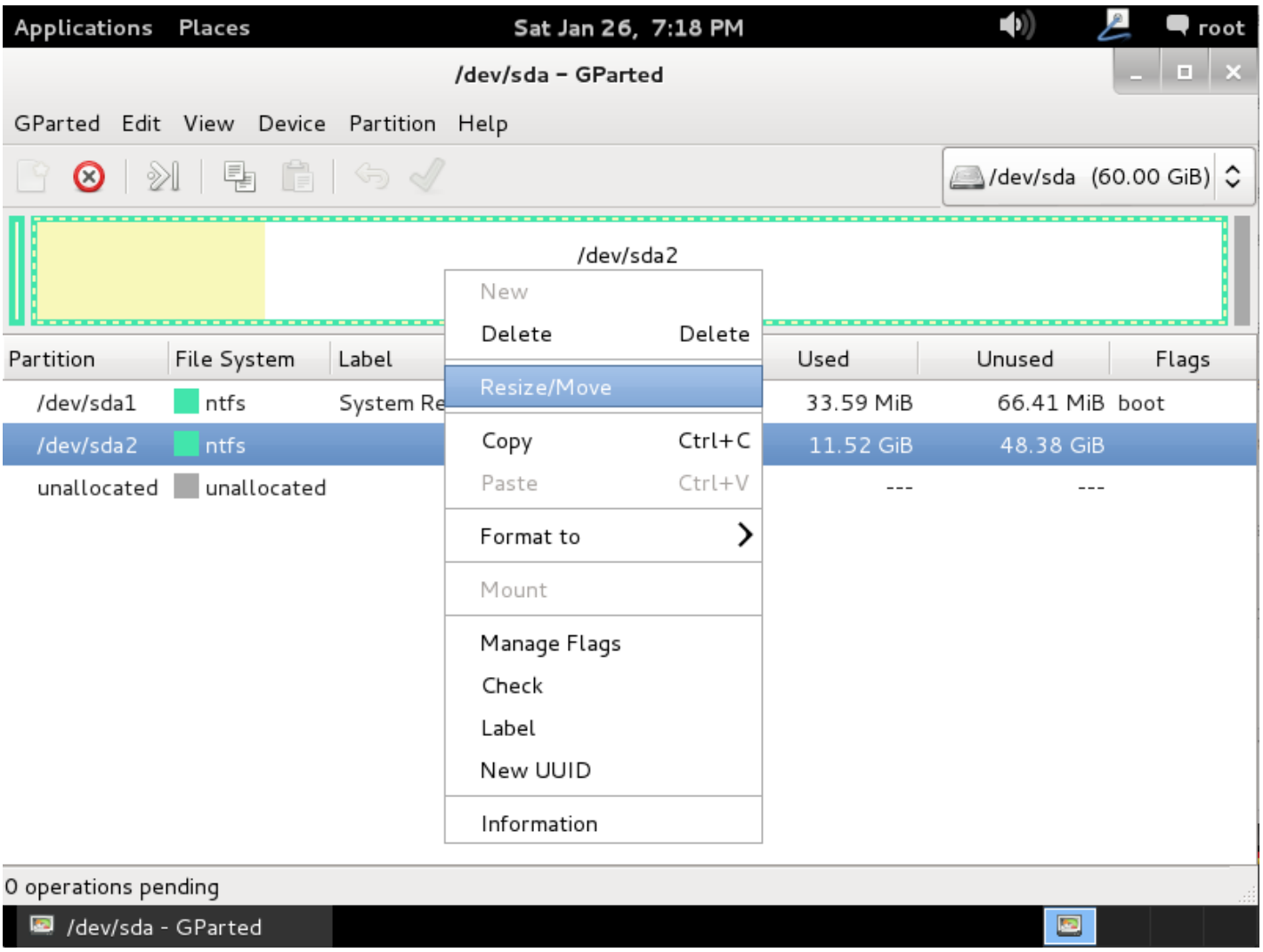
1. Download Kali Linux.
2. Burn The Kali Linux ISO to DVD or copy Kali Linux Live to USB.
3. Ensure that your computer is set to boot from CD / USB in your BIOS.

Dual Boot Installation Procedure

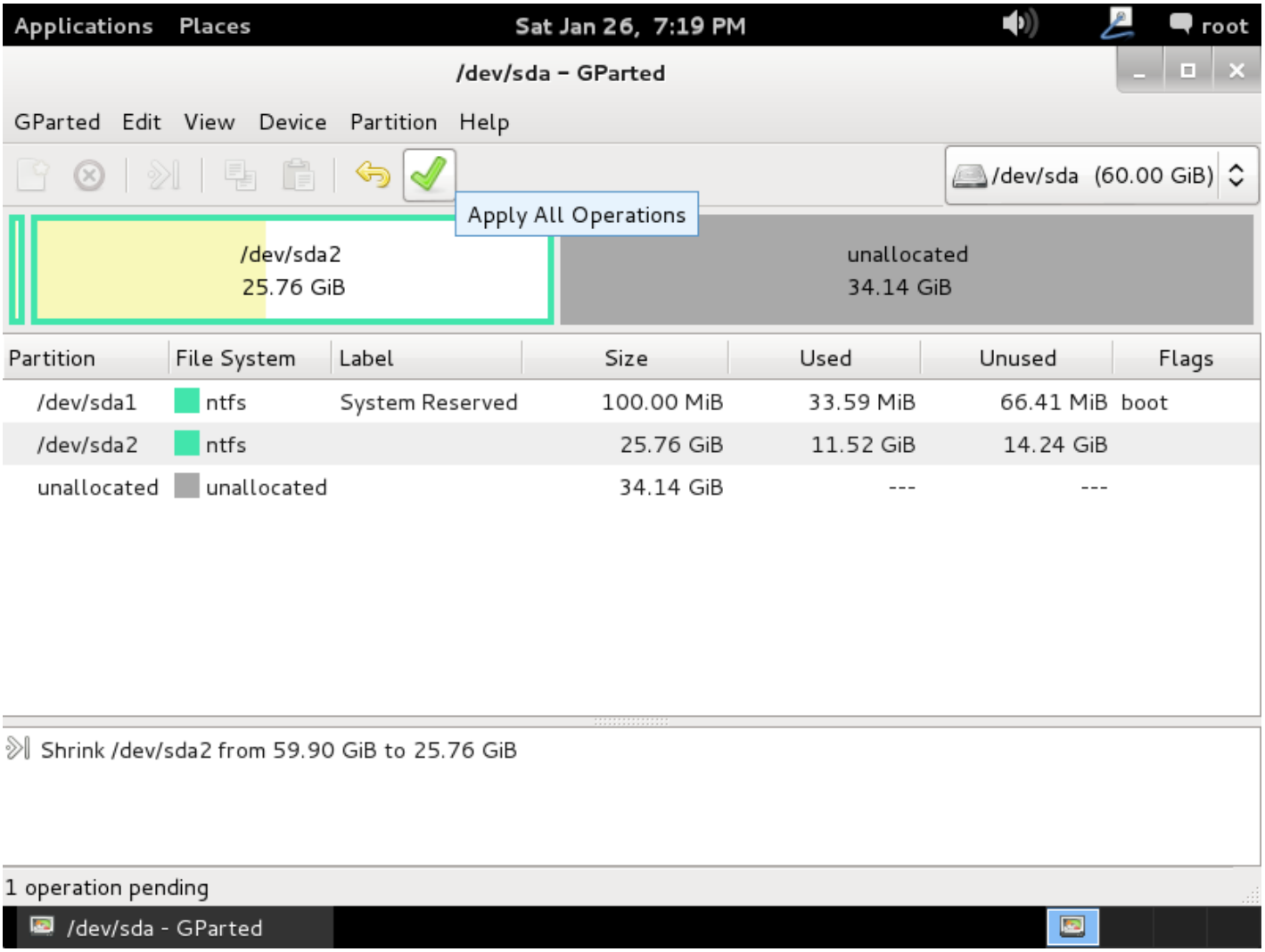
1. To start your installation, boot with your chosen installation medium. You should be greeted with the Kali Boot screen. Select *Live*, and you should be booted into the Kali Linux default desktop.
2. Now launch the **gparted** program. We'll use **gparted** to shrink the existing Windows partition to give us enough room to install Kali Linux.



3. Select your Windows partition. Depending on your system, it will usually be the second, larger partition. In our example, there are two partitions; the first is the System Recovery partition, and Windows is actually installed in /dev/sda2. Resize your Windows partition and leave enough space (20 GB minimum) for the Kali installation.

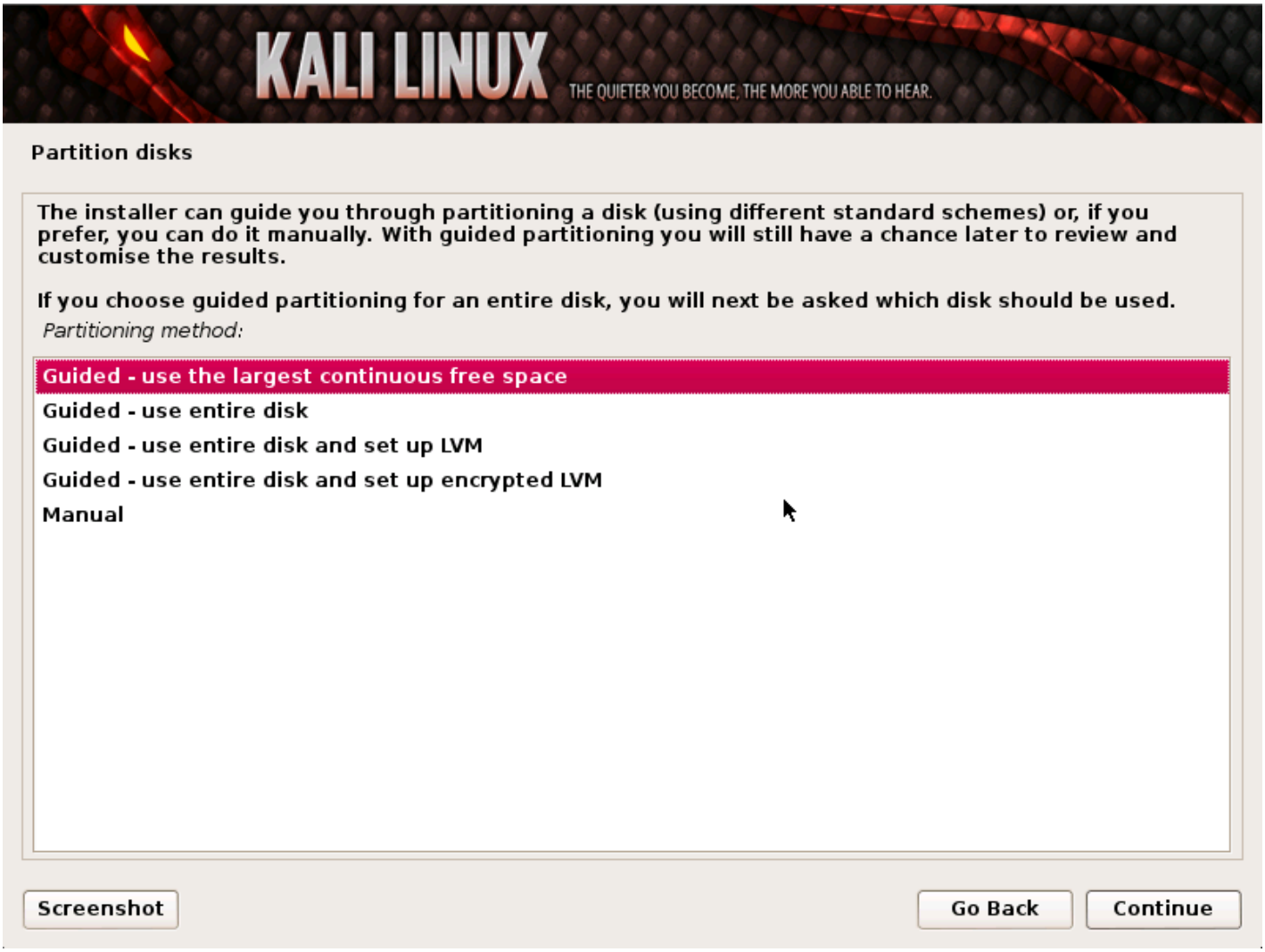


4. Once you have resized your Windows partition, ensure you “Apply All Operations” on the hard disk. Exit **gparted** and reboot.



Kali Linux Installation Procedure

1. The installation procedure from this point onwards is similar to a Kali Linux Hard Disk install, until the point of the partitioning, where you need to select “Guided – use the largest continuous free space” that you created earlier with **gparted**.



2. Once the installation is done, reboot. You should be greeted with a GRUB boot menu, which will allow you to boot either into Kali or Windows.

Post Installation

Now that you've completed installing Kali Linux, it's time to customize your system. The Kali General Use section of our site has more information and you can also find tips on how to get the most out of Kali in our User Forums.

admin

Dual Boot Kali on Mac Hardware

Kali Linux Installation Requirements

Since the release of Kali Linux 1.0.8, Kali Linux supports EFI out of the box. This added feature simplifies the process of getting Kali installed and running on various Apple MacBook Air, Pro, and Retina models.

> The make/model/year of the device will determine how successful your experience will be, with newer devices having a better chance of working. Pre-installing rEFInd may also increase the odds of success on older devices.

This guide will show you to dual-boot OSX with Kali Linux using rEFInd, with the option of encrypting the Kali Linux partition. If you wish to replace OSX completely, please refer to our Single Boot Kali on Mac Hardware guide.

By using using the 3rd party software rEFInd (a fork of rEFIt) we are able to open up the boot menu used in Apple's OSX OS, which is perfect for dual booting. It also has the advantage of helping older devices boot from USB that would not be able to otherwise. Once Kali Linux has been installed, rEFInd can be customized to be hidden or removed completely.

Installation Prerequisites

- A minimum of 20 GB disk space for the Kali Linux install.
- A minimum of 1 GB RAM. 2 GB or more recommended.
- Devices **older** than '**late 2012**', may require a blank DVD. **USB booting may not work without rEFInd** pre-installed.
- For devices **newer** than '**late 2012**', you'll need a blank DVD **or** a USB drive.
- OSX 10.7 or higher

Preparing for the Installation

1. Download Kali linux.
2. Burn the Kali Linux ISO image to a DVD or copy the image to USB drive.
3. Backup any important information on the device to external media.

Preparing OSX (Installing rEFInd)

1. At the time of this writing, the latest version of rEFInd is 0.8.3.
Boot into OSX and download a local copy.

```
osx:~ mbp$ curl -s -L http://sourceforge.net/projects/refind/files/0.8.3/refind-bin-0.8.3.zip -o
refind.zip
```

2. After downloading rEFInd, extract the contents of the zip file and run the install shell script with sudo.

```
osx:~ mbp$ unzip -q refind.zip
osx:~ mbp$ cd refind-bin-*/
osx:refind-bin-0.8.3 mbp$ sudo bash install.sh

WARNING: Improper use of the sudo command could lead to data loss
or the deletion of important system files. Please double-check your
typing when using sudo. Type "man sudo" for more information.

To proceed, enter your password, or type Ctrl-C to abort.

Password:
Installing rEFInd on OS X....
Installing rEFInd to the partition mounted at //
Copied rEFInd binary files

Copying sample configuration file as refind.conf; edit this file to configure
rEFInd.

WARNING: If you have an Advanced Format disk, *DO NOT* attempt to check the
bless status with 'bless --info', since this is known to cause disk corruption
on some systems!!

Installation has completed successfully.

osx:refind-bin-0.8.3 mbp$
```

Kali Linux Partitioning Procedure

1. Before we can install Kali Linux, there needs to be room on the hard disk. By booting into a live Kali session, we can resize the partition to our desired size. To do so, power on the device and immediately press and hold the **Option** key until you see the rEFInd boot menu.

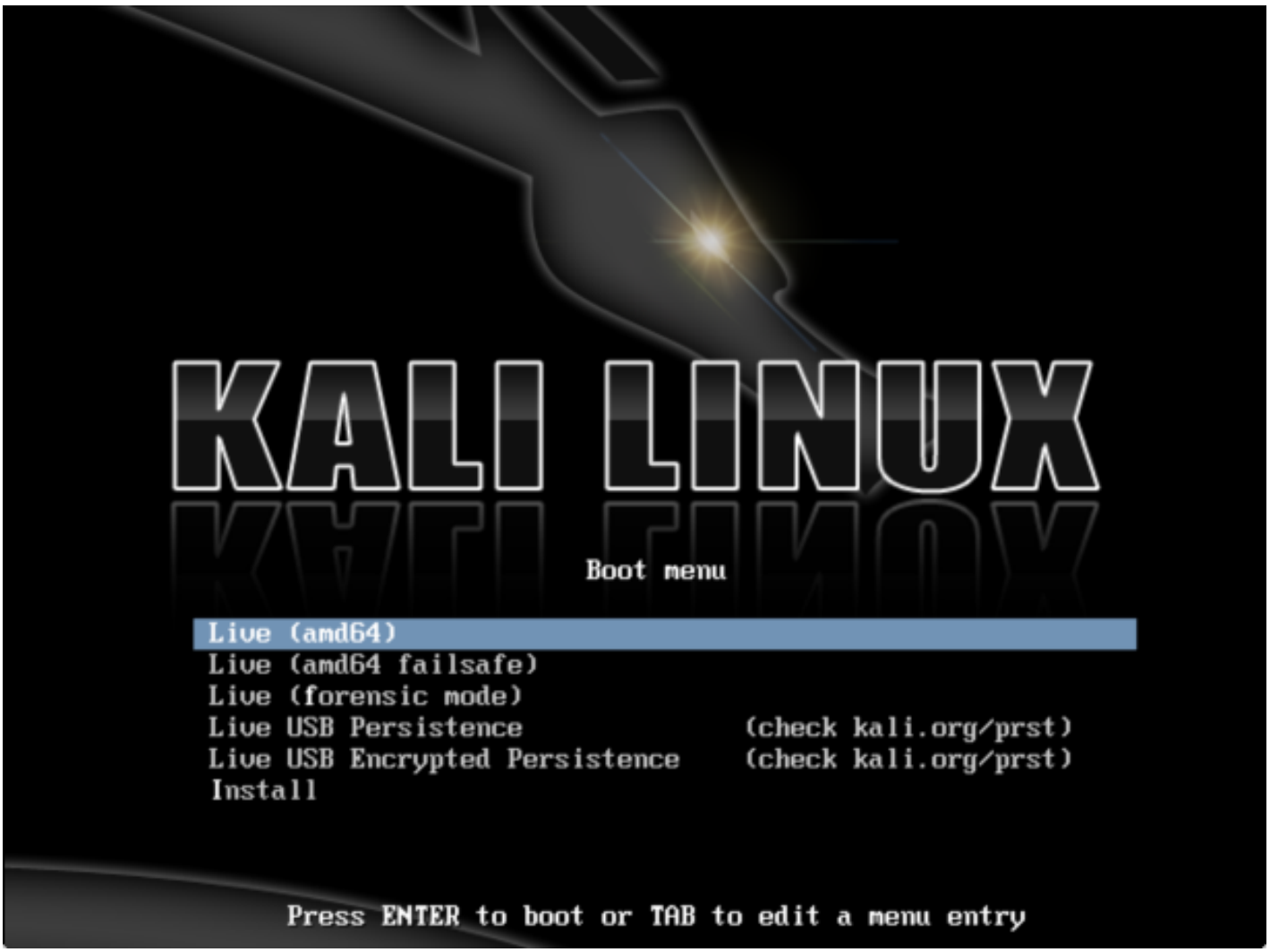
2. When the boot menu appears, insert your chosen installation medium. If everything works as expected, you will see **two** volumes:
    - EFI – EFI\BOOT\syslinux.efi from 61 MiB FAT volume
    - Windows – Legacy OS from FAT volume

   Although Kali Linux is based on Debian, Apple/rEFInd detects it as Windows.
   Select the Windows volume to continue.

    - If you are using a DVD, you may need to refresh the menu by pressing **ESC** once the disk if fully spinning.

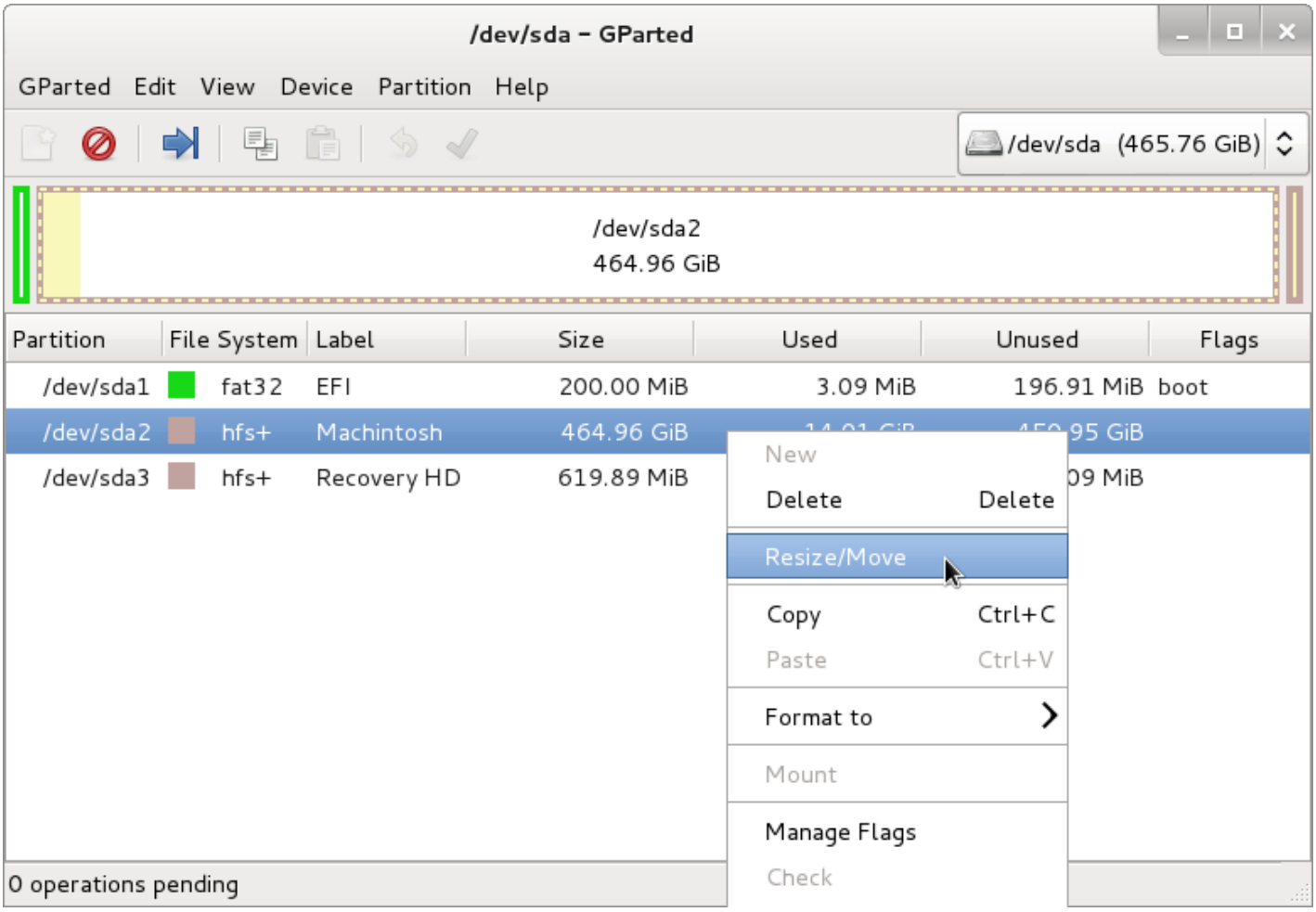
- If you still only see **one volume** (EFI), then the installation medium **is not supported** for your Apple device. If you haven't already done so, you may wish to install **rEFInd** and try again.
- If you select the EFI volume, the booting will hang at this point and you will **not** be able to continue.

3. You should be greeted with the Kali Boot screen. Select **Live** and you should be booted into the Kali Linux default desktop.



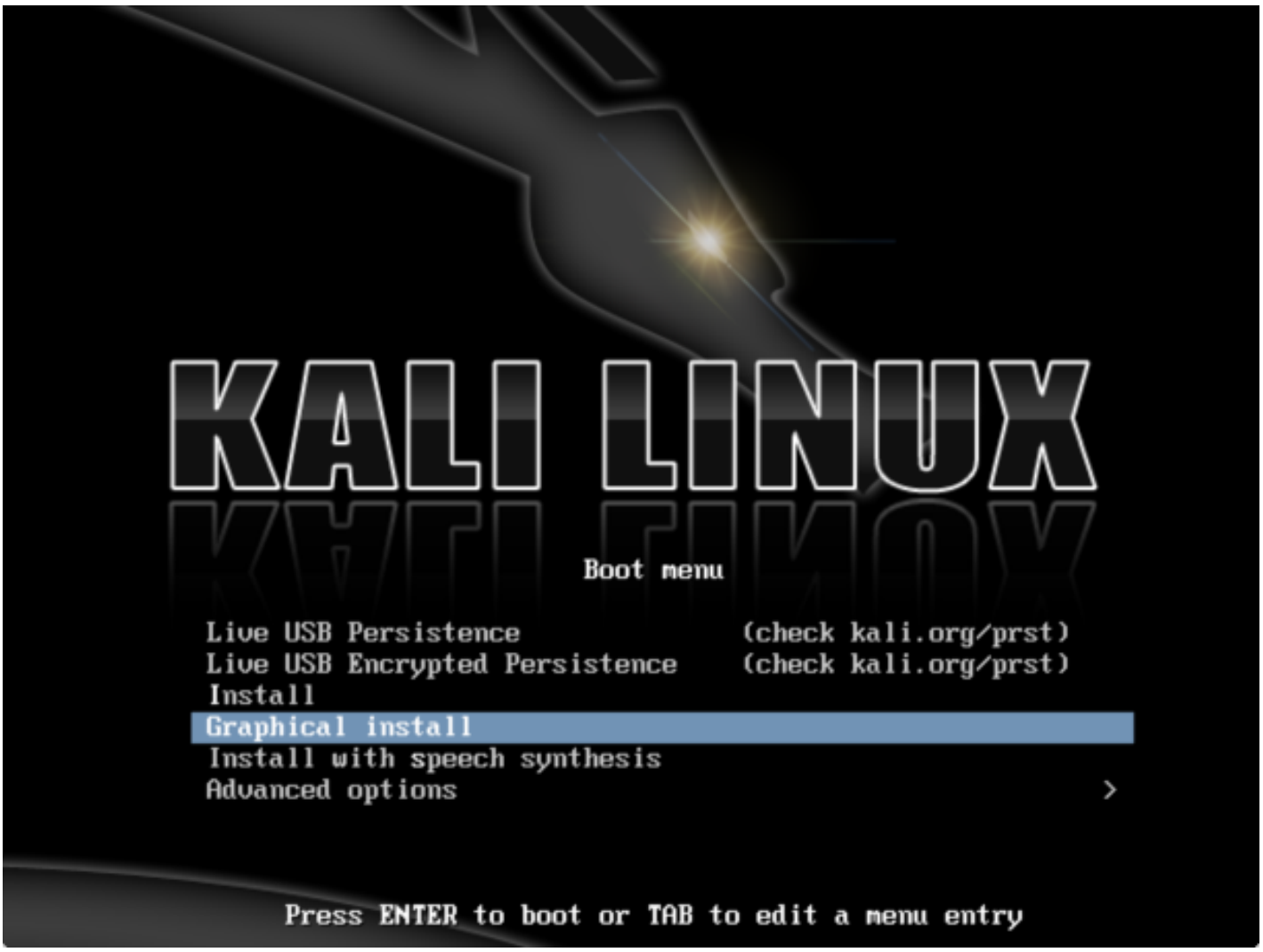
4. We can use GParted to shrink the existing OSX partition (HFS+), allowing us to install Kali in the free space. You can find GParted in the Kali menu by navigating to: Applications -> System Tools -> GParted Partition Editor

5. Once GParted has opened, select your OSX partition. Depending on your system, it will usually be the second, larger partition. In our example, there are three partitions: the EFI upgrade partition (/dev/sda1), OSX (/dev/sda2), and System Recovery (/dev/sda3). Resize your OSX partition and leave enough space (20 GB minimum) for the Kali installation.

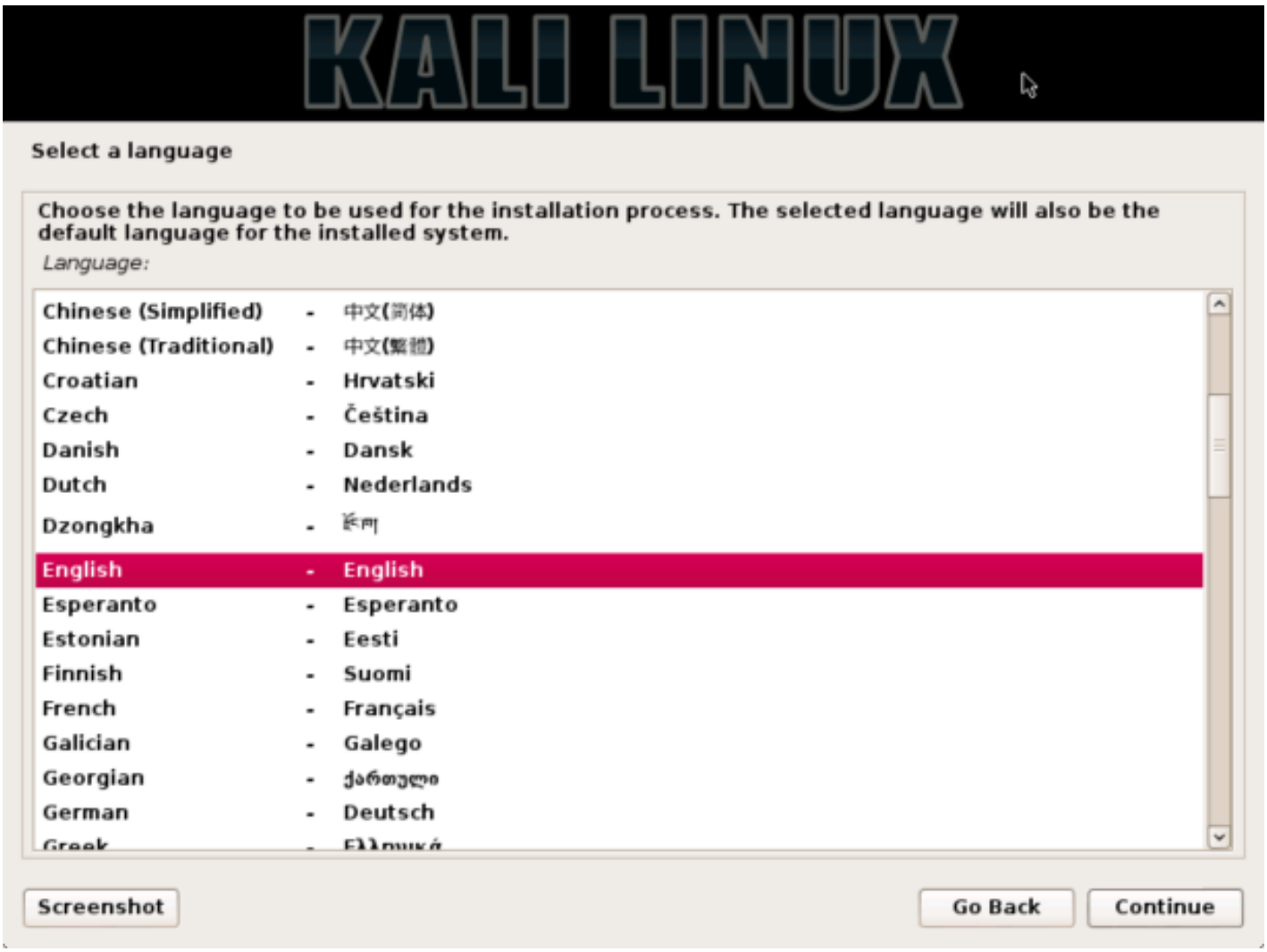


Kali Linux Installation Procedure

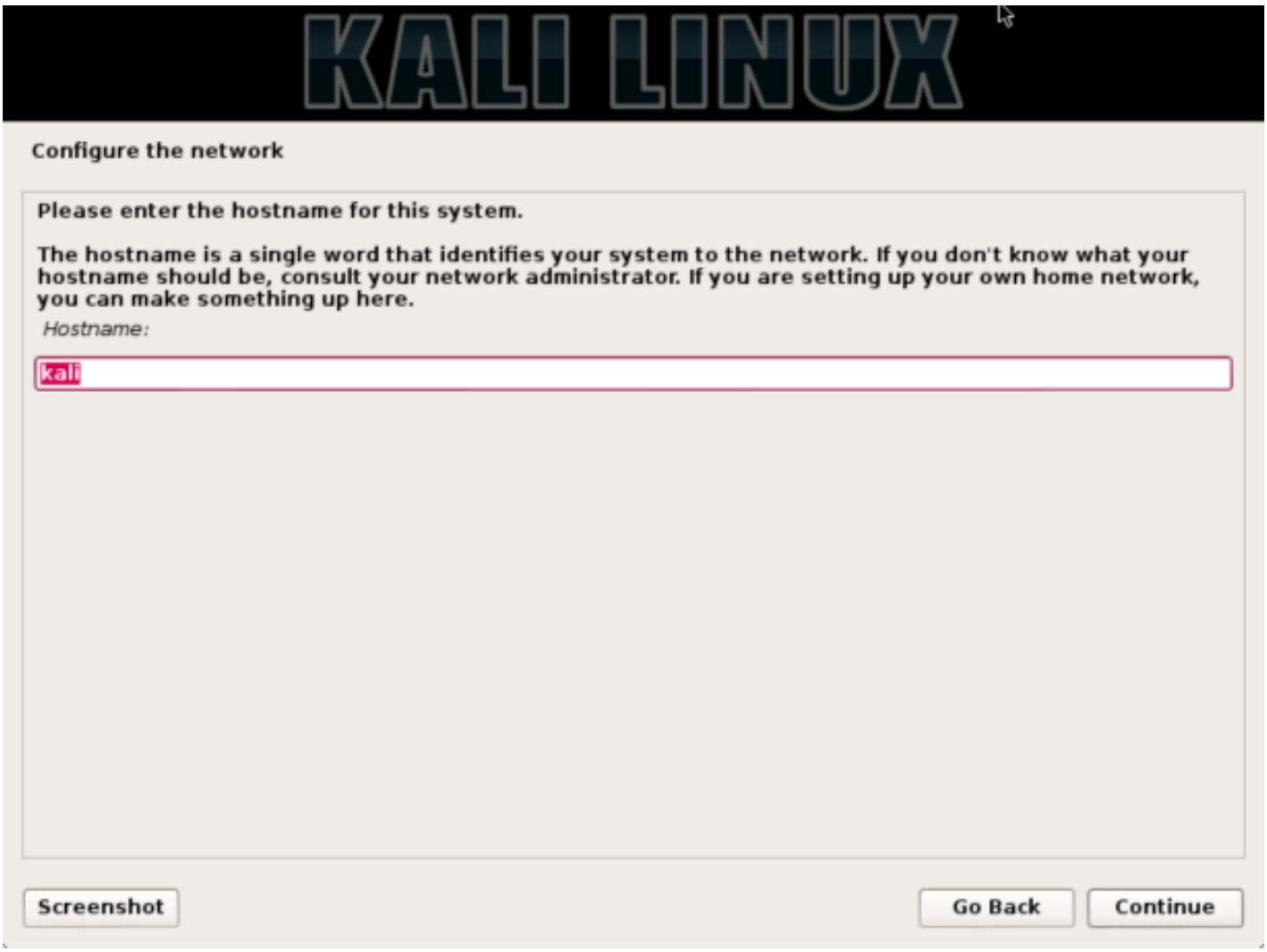
1. To start the Kali Linux installation, repeat steps 1 and 2 above to boot to the Kali Linux boot screen. Once you can see the the boot screen, choose ‘Live’, ‘Graphical Install’ or ‘(Text-Mode) Install’ to begin the setup. In this guide, we chose ‘Graphical Install’.



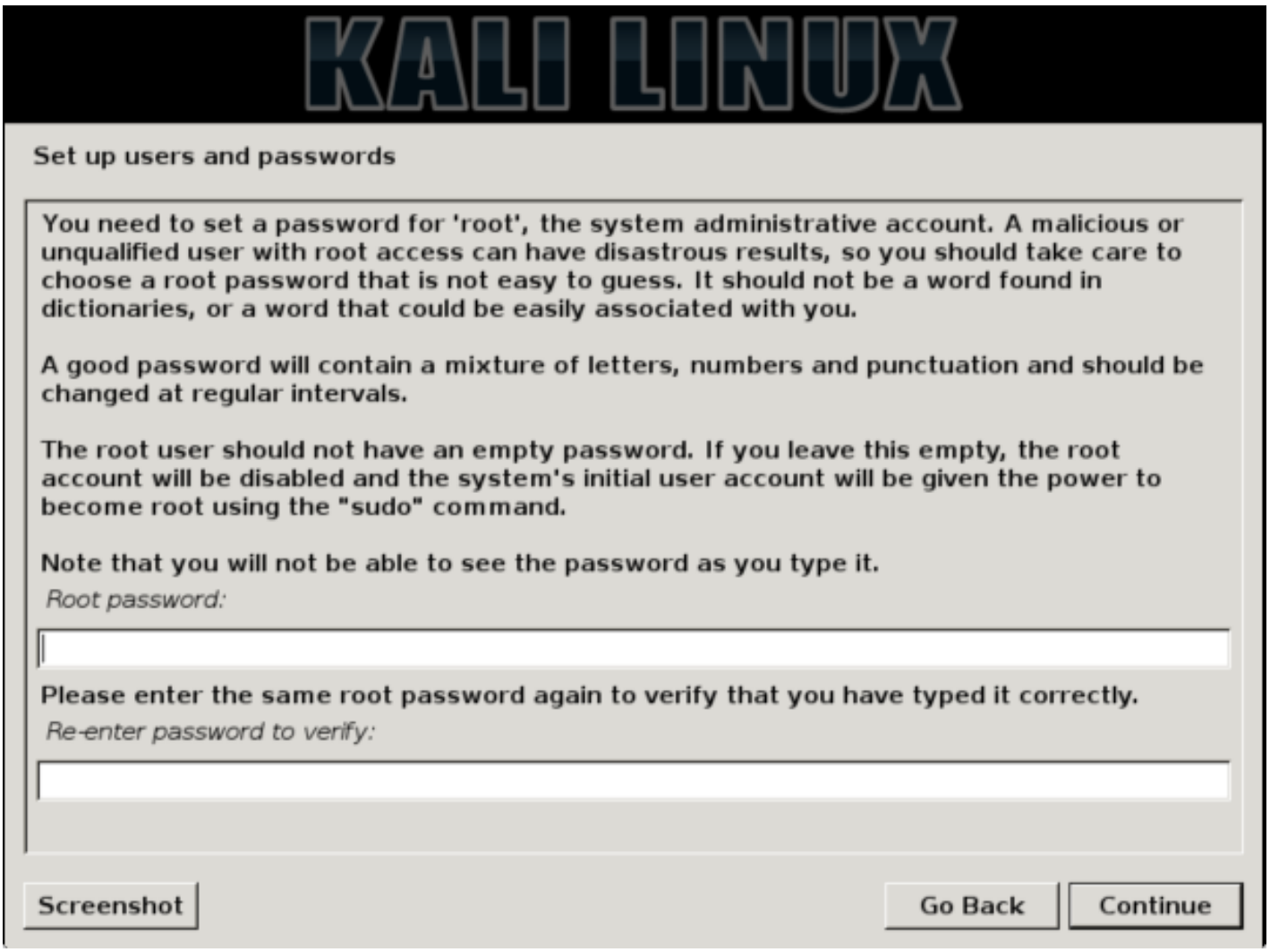
2. Select your preferred language and then your country location. You'll also be prompted to configure your keyboard with the appropriate keymap.



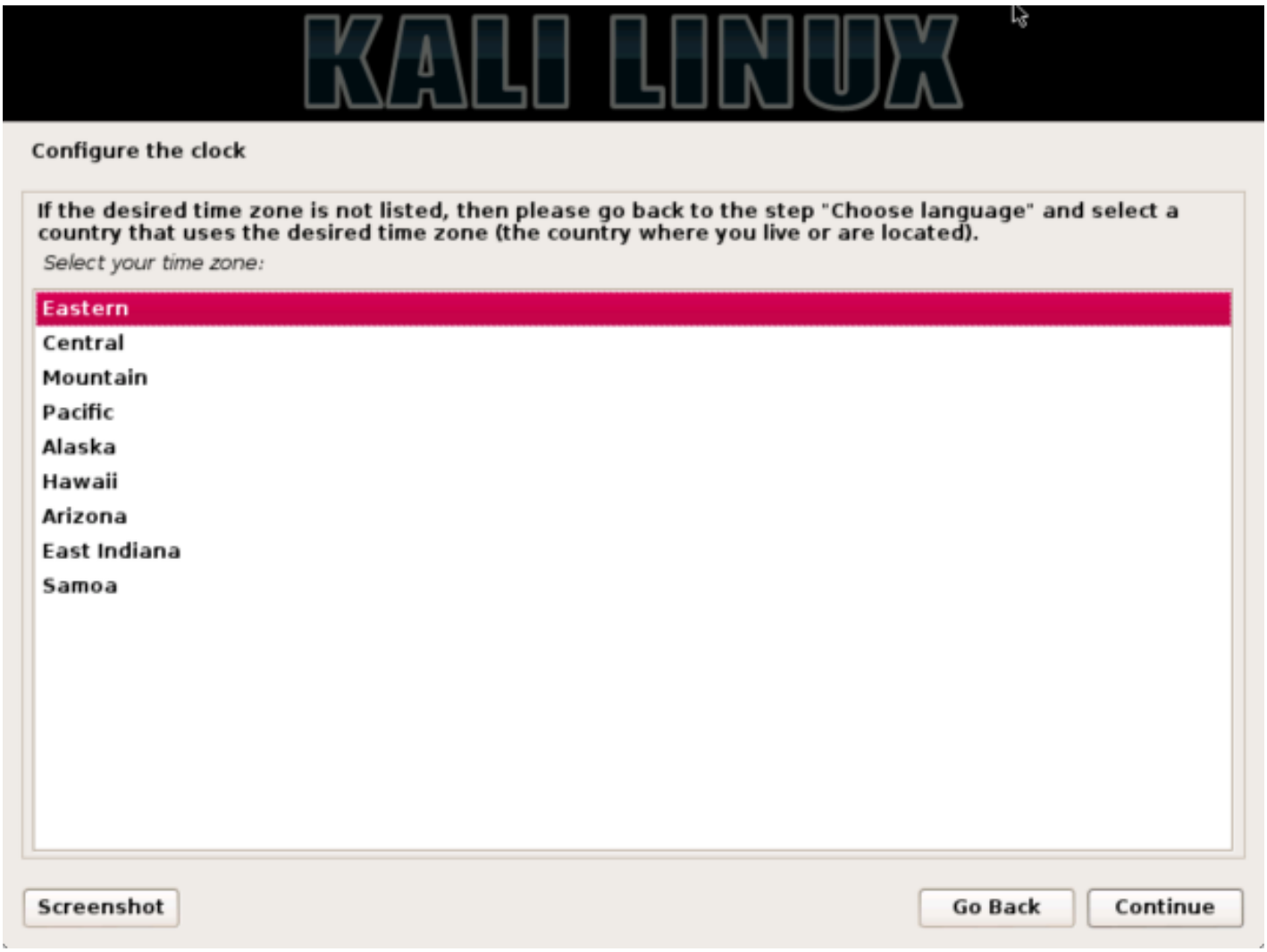
3. The installer will copy the image to your hard disk, probe your network interfaces, and then prompt you to enter a hostname then domain name for your system. In the example below, we've entered 'kali' as our hostname.
    - If the setup detected multiple NICs, it may prompt you which one to use for installation.
    - If the chosen NIC is 802.11 based, it will ask for wireless network information to collect, before prompting for a hostname.
    - If there isn't a DHCP service running on the network, it will ask you to manually enter the network information after probing for network interfaces.
    - If Kali Linux doesn't detect your NIC, you either need to include the drivers for it when prompted, or generate a custom Kali Linux ISO with them pre-included.



4. Enter a robust password for the root account.

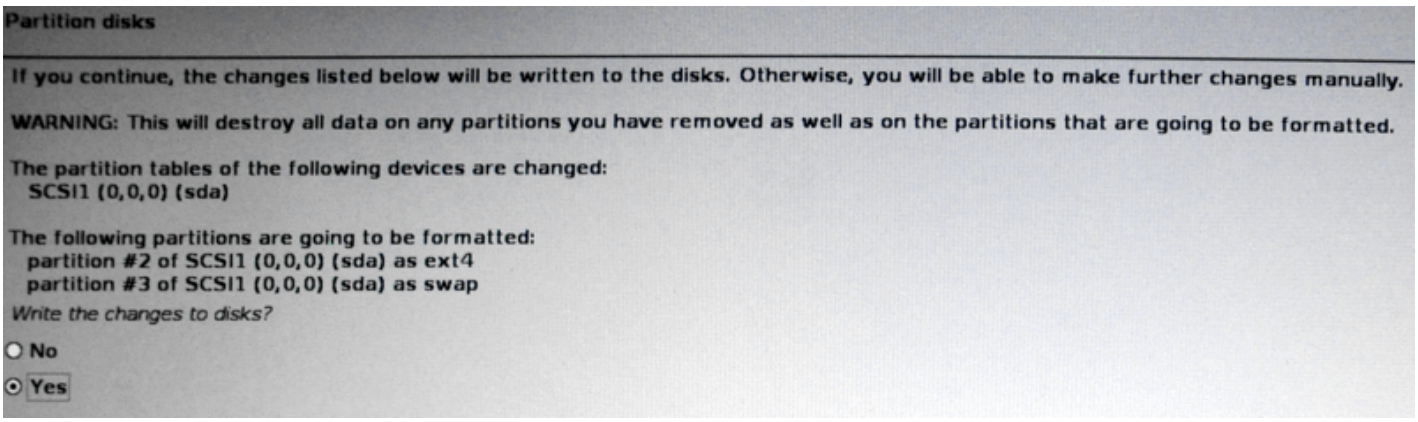


5. Next, set your time zone.

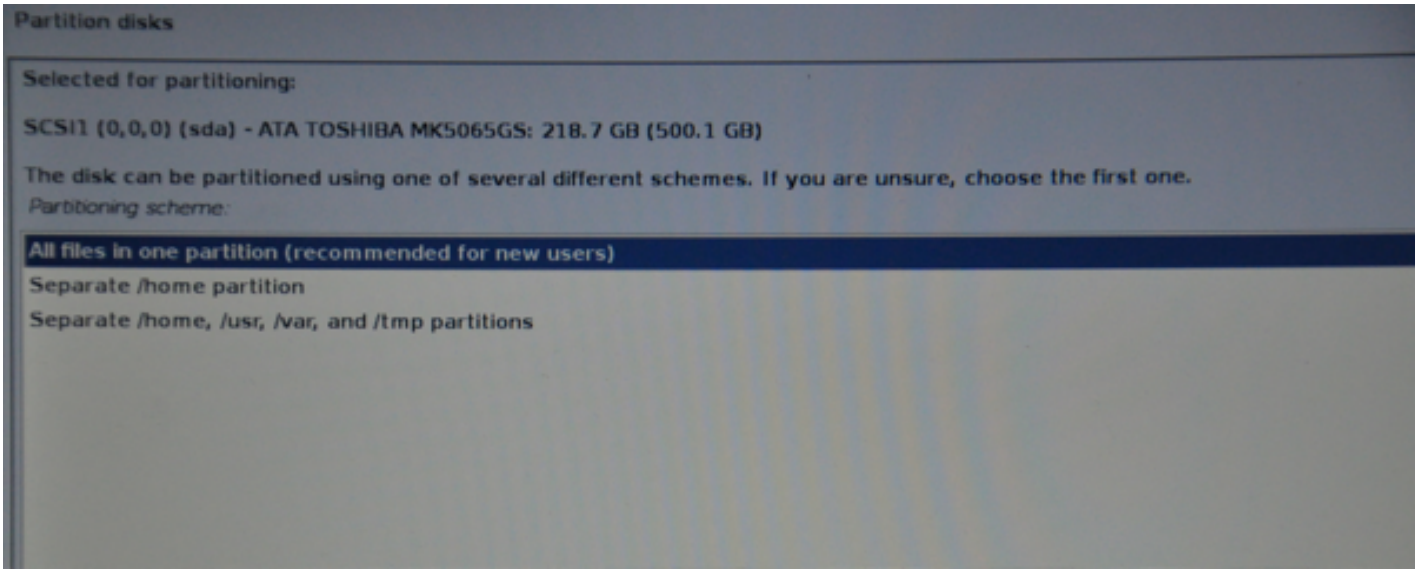


6. The installer will now probe your disks and offer you five choices. In our example, we're using the spare partition that we made during live mode, so we select 'Guided – use the largest continuous free space'.
    - Experienced users can use the 'Manual' option for more granular configuration options. This option will also allow you to set up encrypted LVM, so Kali Linux would be fully encrypted. The screen afterwards will prompt you for the password. You will have to enter the same password every time you start up Kali Linux.

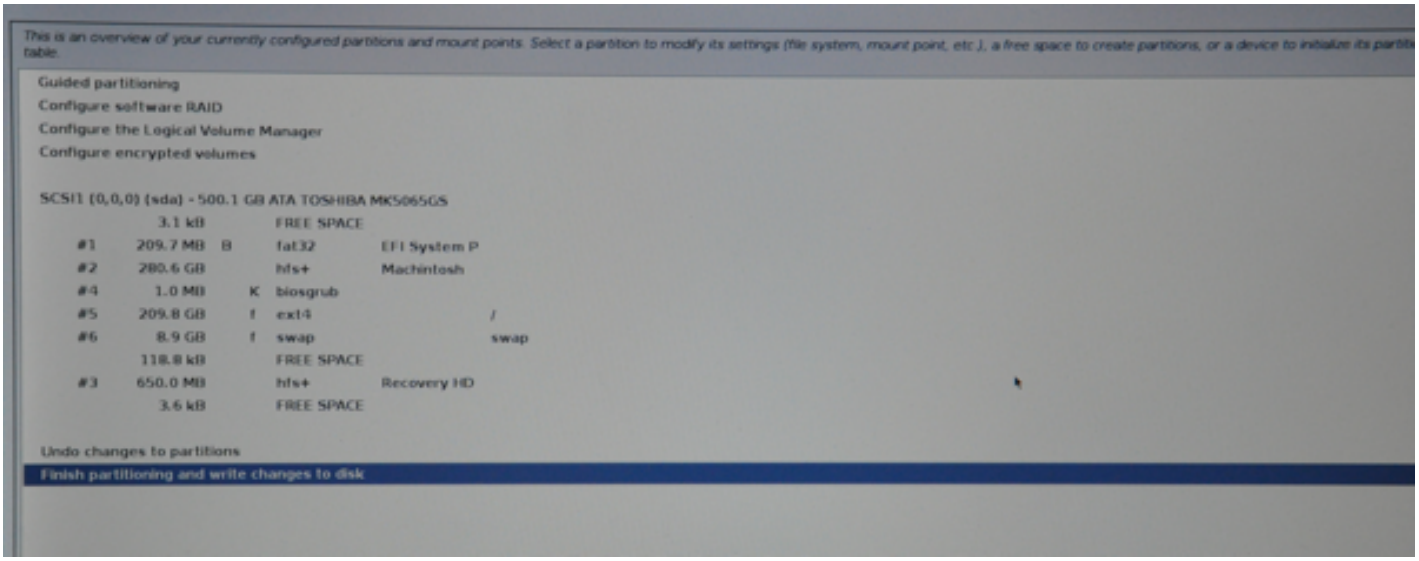
Kali will automatically securely wipe the hard disk before asking for the password. This may take 'a while' (hours) depending on size and speed of the drive. If you wish to risk it, you can skip it.



7. The next stage is to select the partition structure you want to use. We will go ahead and use the default option and have everything on one partition. Afterwards it will display an overview. If you agree to what it suggests, press the continue button.

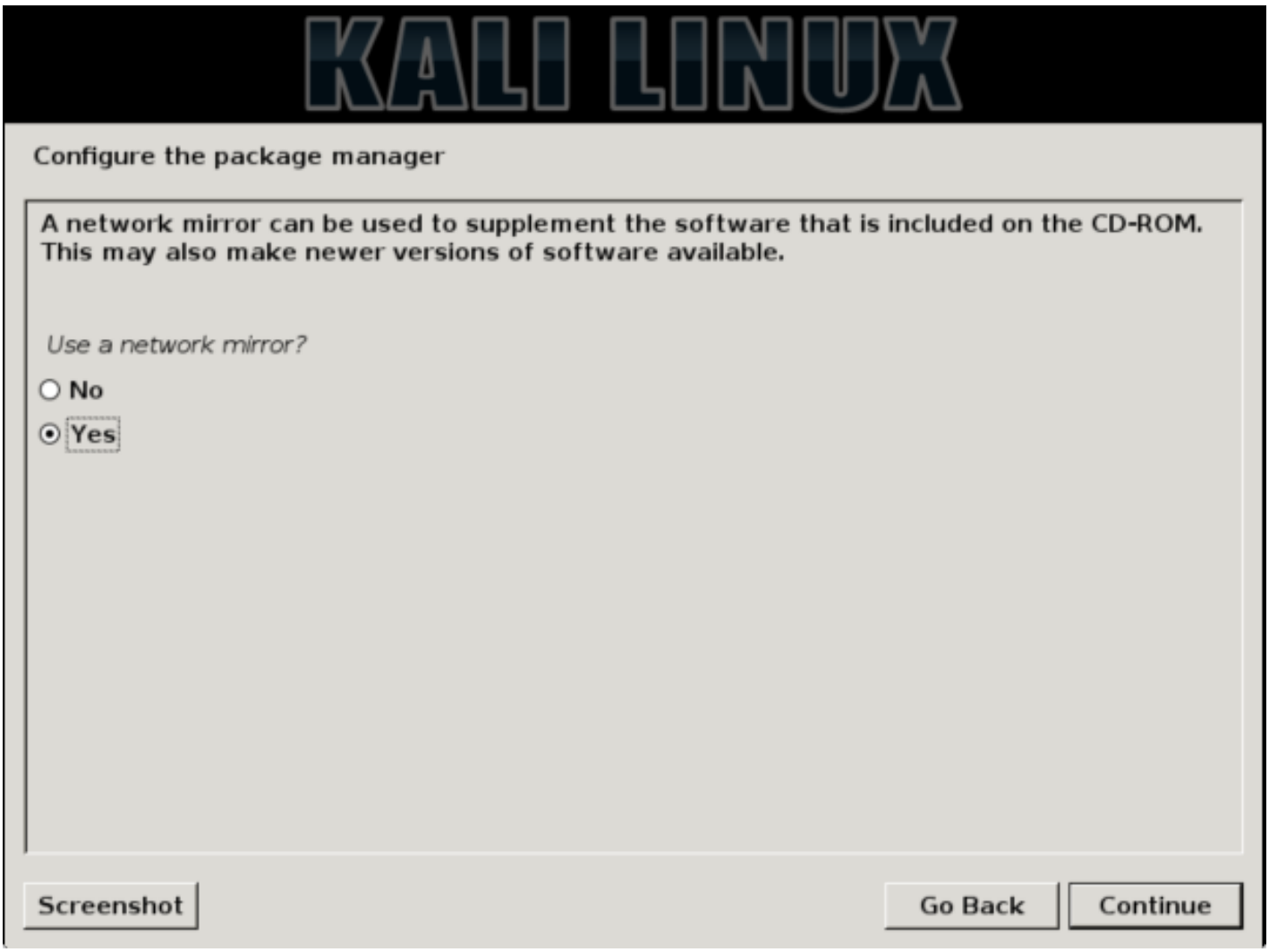


8. Next, you'll have one last chance to review your disk configuration before the installer makes irreversible changes. After you click Continue, the installer will go to work and you'll have an almost finished installation.

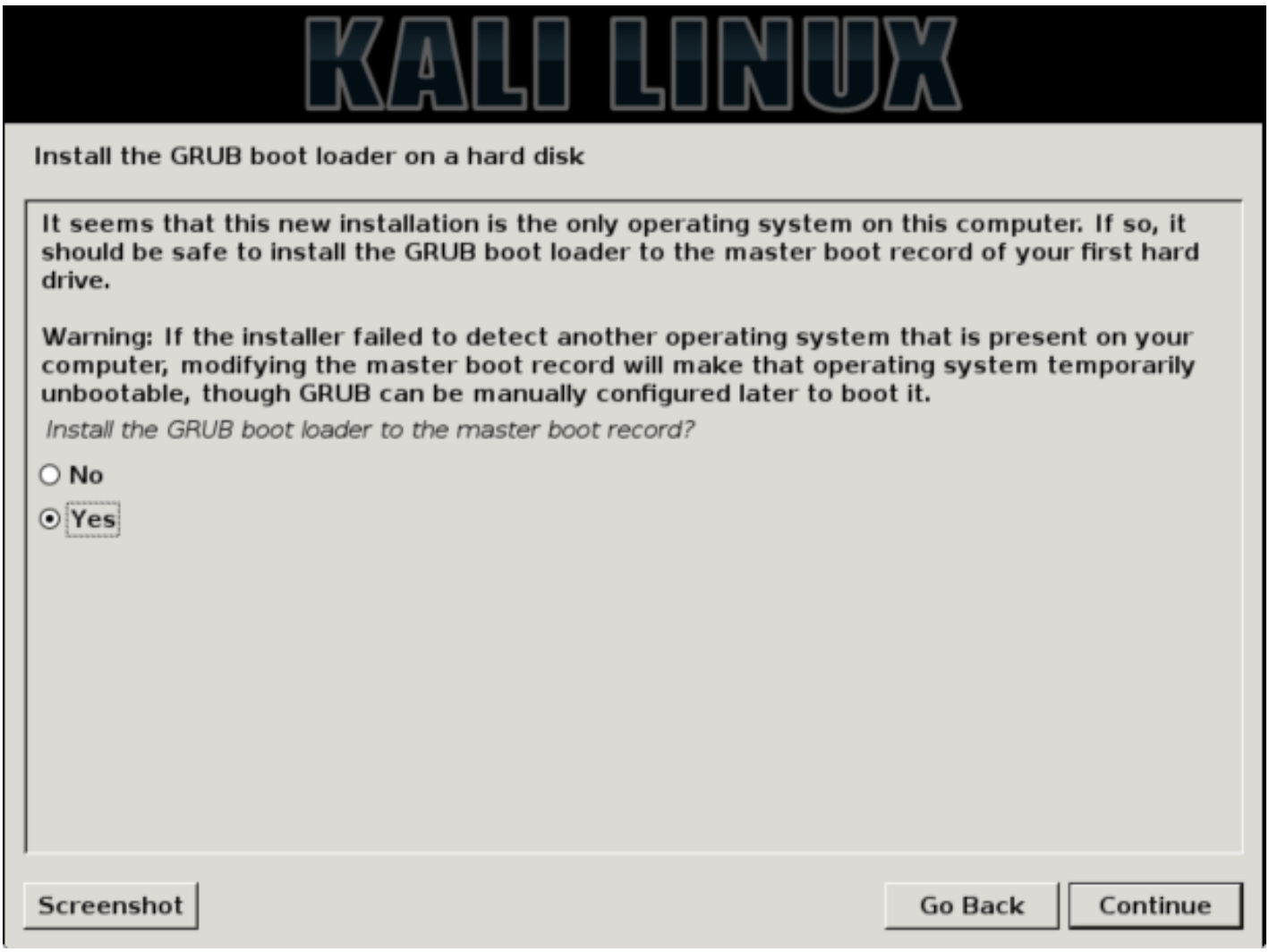


9. This screen configures the use of our Internet network mirrors. Kali can use our online central repository to distribute applications to keep packages up-to-date and allow for additional programs to be installed more easily. Should you need to enter any appropriate proxy information, the next screen will allow you to enter the required details.

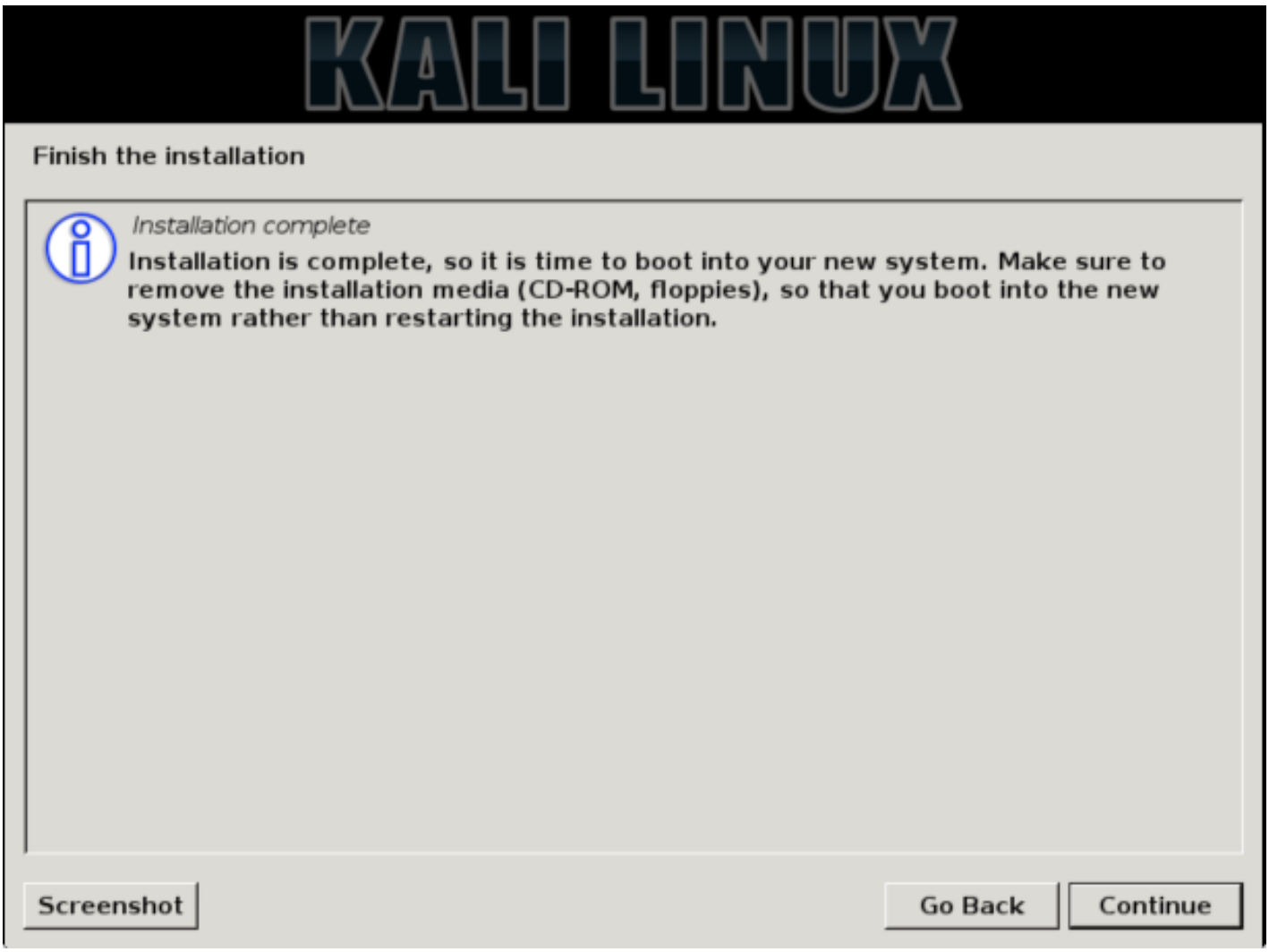
> If you select "NO" in this screen, you will NOT be able to install packages from Kali repositories until you alter your sources.



10. Next, install GRUB bootloader.



11. Finally, click ‘Continue’ to finish installing Kali Linux. It is highly recommend that you restart your machine at this stage.
    Once complete, repeat the first 2 steps again to boot into ‘Live mode’ once more.



12. If the gdisk package isn't included in your Kali Linux ISO, you will first need to install it.
If you enabled the network repository during the setup, this can easily be done:

```
apt-get update
apt-get install gdisk
```

13. We are now going to convert the Master Boot Record (MBR) to a hybrid, which will allow for Apple's EFI to detect and boot using GRUB.
Once complete, power off the device and remove any installation media when prompted.

```
root@kali:~# gdisk /dev/sda
GPT fdisk (gdisk) version 0.8.5

Partition table scan:
MBR: protective
```

```
BSD: not present
APM: not present
GPT: present

Found valid GPT with protective MBR; using GPT.

Command (? for help): p
Disk /dev/sda: 976773168 sectors, 465.8 GiB
Logical sector size: 512 bytes
Disk identifier (GUID): 1B3DB3D4-ECFD-47A1-9435-F2FF318C2F55
Partition table holds up to 128 entries
First usable sector is 34, last usable sector is 976773134
Partitions will be aligned on 8-sector boundaries
Total free space is 245 sectors (122.5 KiB)

Number Start (sector) End (sector) Size Code Name
1 40 409639 200.0 MiB EF00 EFI System Partition
2 409640 548413439 261.3 GiB AF00 Macintosh
3 975503592 976773127 619.9 MiB AB00 Recovery HD
4 548413440 548415487 1024.0 KiB EF02
5 548415488 958138367 195.4 GiB 0700
6 958138368 975503359 8.3 GiB 8200

Command (? for help): r

Recovery/transformation command (? for help): h

WARNING! Hybrid MBRs are flaky and dangerous! If you decide not to use one,
just hit the Enter key at the below prompt and your MBR partition table will
be untouched.

Type from one to three GPT partition numbers, separated by spaces, to be
added to the hybrid MBR, in sequence: 5
Place EFI GPT (0xEE) partition first in MBR (good for GRUB)? (Y/N): y

Creating entry for GPT partition #5 (MBR partition #2)
Enter an MBR hex code (default 07): 83
Set the bootable flag? (Y/N): y

Unused partition space(s) found. Use one to protect more partitions? (Y/N): n
```

```
Recovery/transformation command (? for help): w

Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): y
OK; writing new GUID partition table (GPT) to /dev/sda.
The operation has completed successfully.
root@kali:~#
```

14. At this stage, we are able to use both Kali Linux and OSX and select which one we want to use at start up.

rEFInd Configuration

If you wish, you can alter rEFInd in various ways now, including:

- The default OS selection (by default it is OSX)
- Timeout value (by default it is 20 seconds)
- Direct boot into the default OS (Note, by pressing **Options** during boot, you will have a one time boot menu)
- Remove rEFInd, enabling the use of the traditional Apple menu (booting to OSX and Kali Linux will still work)

If you wish to make any of these alterations, boot into OSX, and alter the following file:

```
osx:~ mbp$ sudo nano /EFI/refind/refind.conf
```

- The timeout value controls how long you have to select an OS from the boot menu. By setting it to '-1', it will boot directly into the default OS.

```
# Timeout in seconds for the main menu screen. Setting the timeout to 0
# disables automatic booting (i.e., no timeout). Setting it to -1 causes
# an immediate boot to the default OS *UNLESS* a keypress is in the buffer
# when rEFInd launches, in which case that keypress is interpreted as a
# shortcut key. If no matching shortcut is found, rEFInd displays its
# menu with no timeout.
#
timeout -1
```

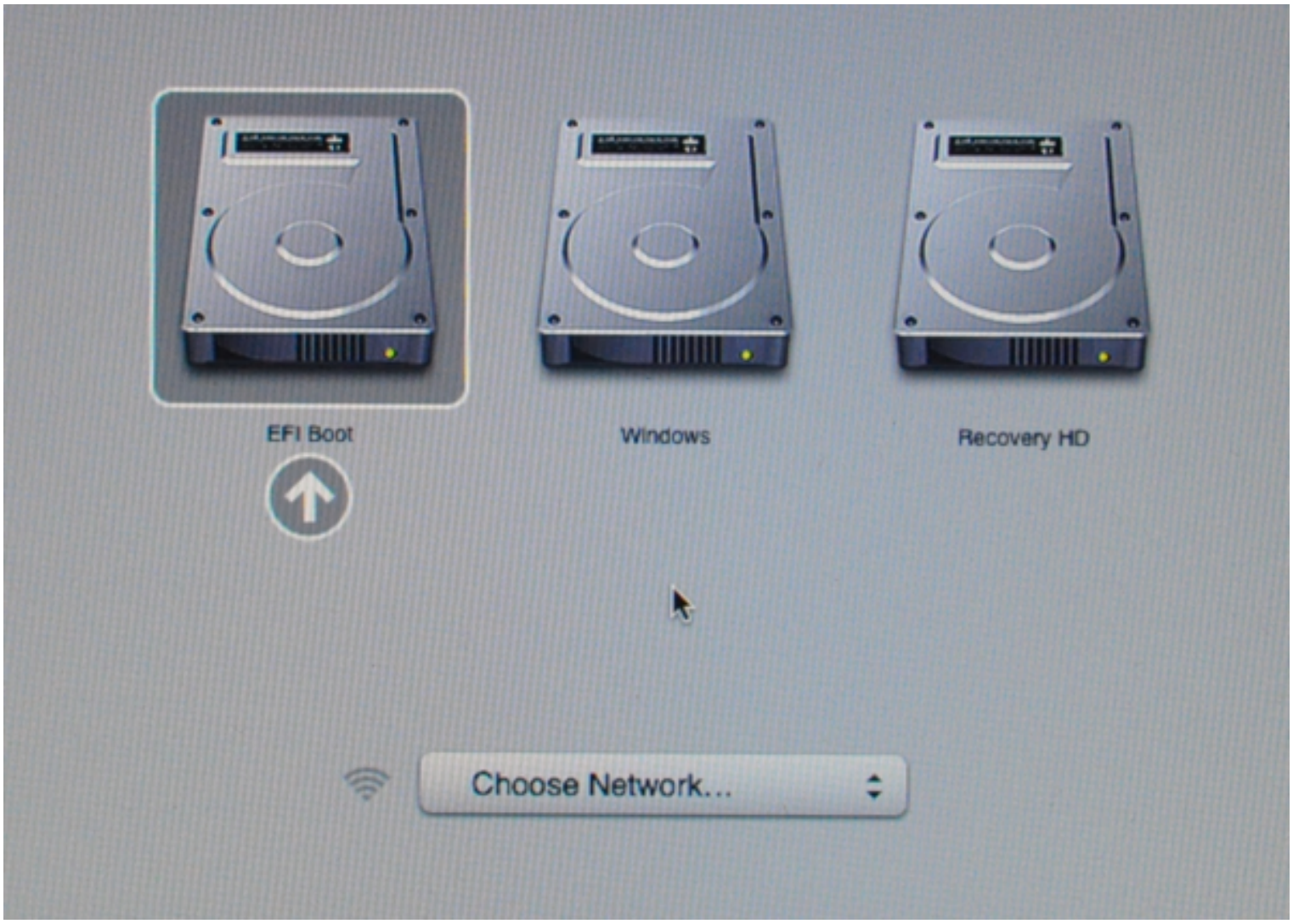
- The 'default_selection' value sets the default selection on startup. OSX will be at position '1' and Kali will be at '2'. In this example, we will use OSX as the default.

```
# Set the default menu selection.  The available arguments match the
# keyboard accelerators available within rEFInd.  You may select the
# default loader using:
#  - A digit between 1 and 9, in which case the Nth loader in the menu
#    will be the default.
#  - A "+" symbol at the start of the string, which refers to the most
#    recently booted loader.
#  - Any substring that corresponds to a portion of the loader's title
#    (usually the OS's name or boot loader's path).
# You may also specify multiple selectors by separating them with commas
# and enclosing the list in quotes. (The "+" option is only meaningful in
# this context.)
# If you follow the selector(s) with two times, in 24-hour format, the
# default will apply only between those times. The times are in the
# motherboard's time standard, whether that's UTC or local time, so if
# you use UTC, you'll need to adjust this from local time manually.
# Times may span midnight as in "23:30 00:30", which applies to 11:30 PM
# to 12:30 AM. You may specify multiple default_selection lines, in which
# case the last one to match takes precedence. Thus, you can set a main
# option without a time followed by one or more that include times to
# set different defaults for different times of day.
# The default behavior is to boot the previously-booted OS.
#
default_selection 1
```

- If we combine the two alterations and save our changes, the next time we reboot, it will appear that nothing has changed from before installing Kali Linux. However, if we hold down the 'Options' key for the

Apple boot menu, we will see the following:

- EFI Boot – OSX
- Windows – Kali Linux
- Recovery HD – OSX's Recovery Partition



Using Apple's boot menu, the value names cannot be altered. If you wish to customize these values, you will need to use rEFInd.

Single Boot Kali on Mac Hardware

Kali Linux Installation Requirements

Since the release of Kali Linux 1.0.8, Kali Linux supports EFI out of the box. This added feature simplifies the process of getting Kali installed and running on various Apple MacBook Air, Pro, and Retina models.

> The make/model/year of the device will determine how successful your experience will be, with newer devices having a better chance of working. Pre-installing rEFInd may also increase the odds of success on older devices.

This guide will show you to **replace** OSX with Kali Linux, with the option of encrypting the partition. However, if you wish to keep OSX, you will want to refer to our dual-boot guide instead.

Installation Prerequisites

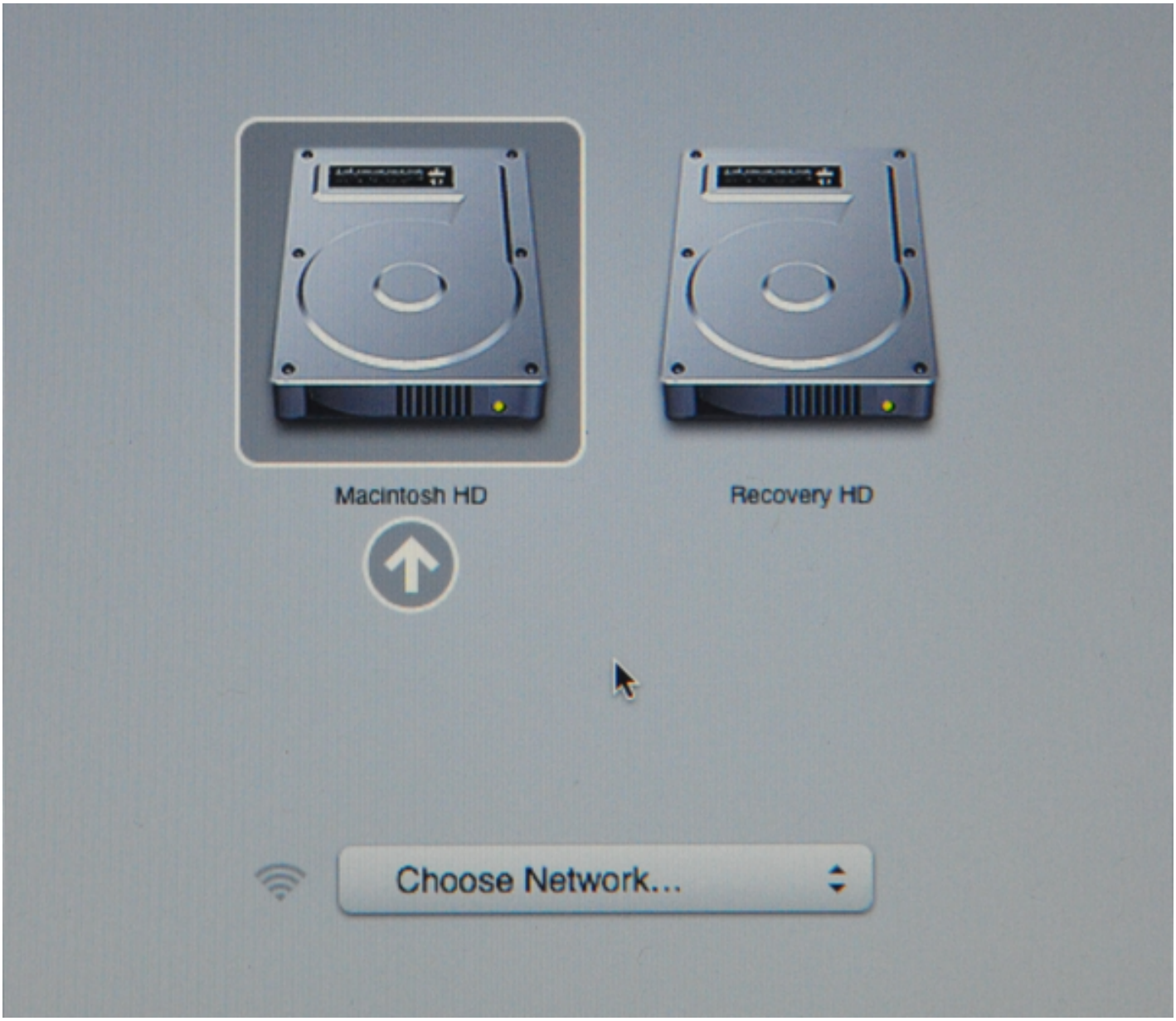
- A minimum of 20 GB disk space for the Kali Linux install.
- A minimum of 1 GB RAM. 2 GB or more recommended.
- For devices **older** than **late 2012**, you will need a blank DVD.
    - **USB booting may not work without rEFInd** installed.
- For devices **newer** than **late 2012**, you'll need a blank DVD **or** a USB drive.
- OSX 10.7 or higher.

Preparing for the Installation

1. Download Kali linux.
2. Burn the Kali Linux ISO image to a DVD or copy the image to USB drive.
3. Backup any important information on the device to an external media.

Kali Linux Installation Procedure

1. To start your installation, power on the device and immediately press and hold the **Option** key until you see the boot menu.

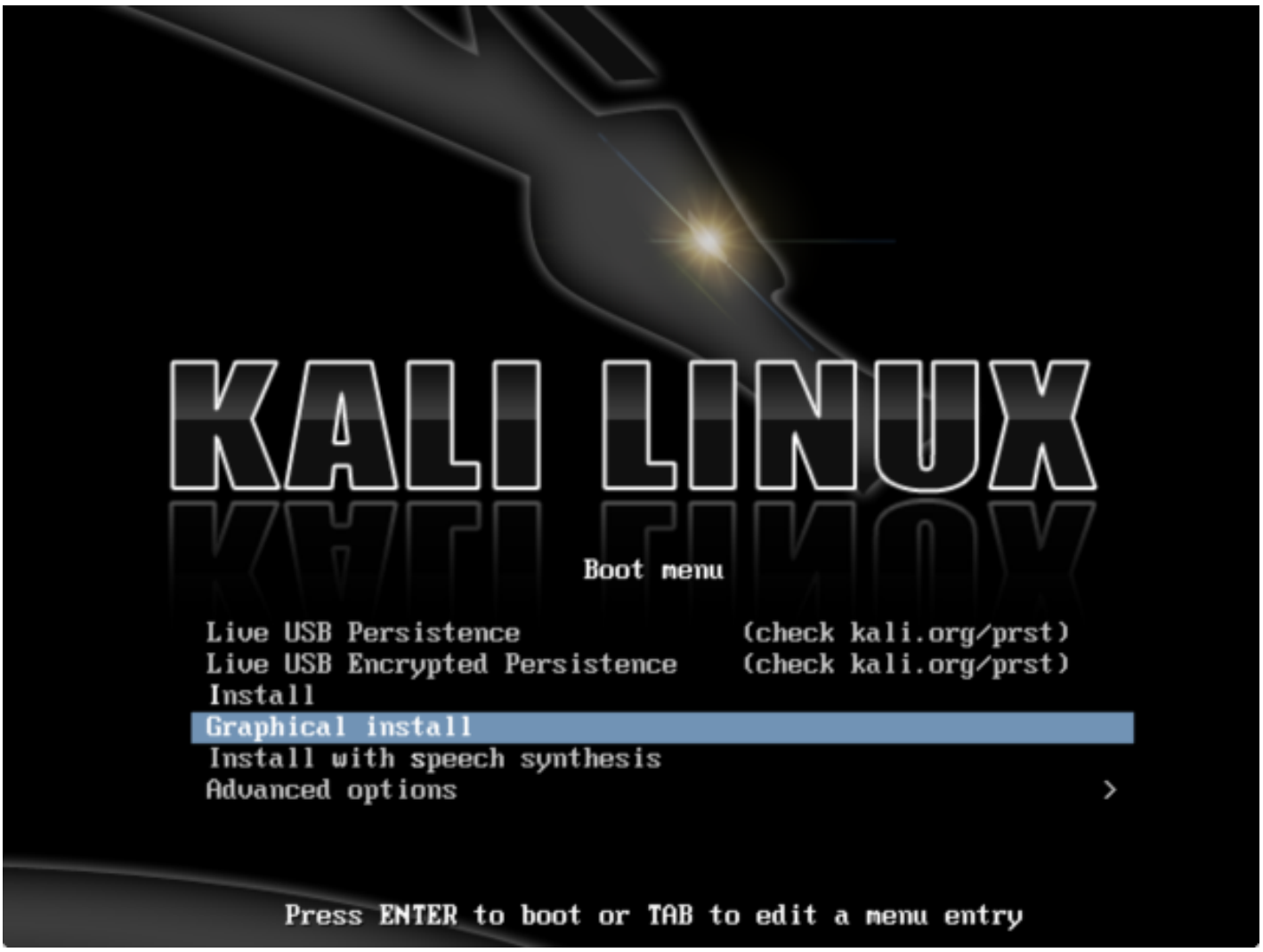


2. Now insert your chosen installation media. If everything was successful, you will see **two** volumes (EFI & Windows). Even though Kali Linux is based on Debian, Apple detects it as Windows.

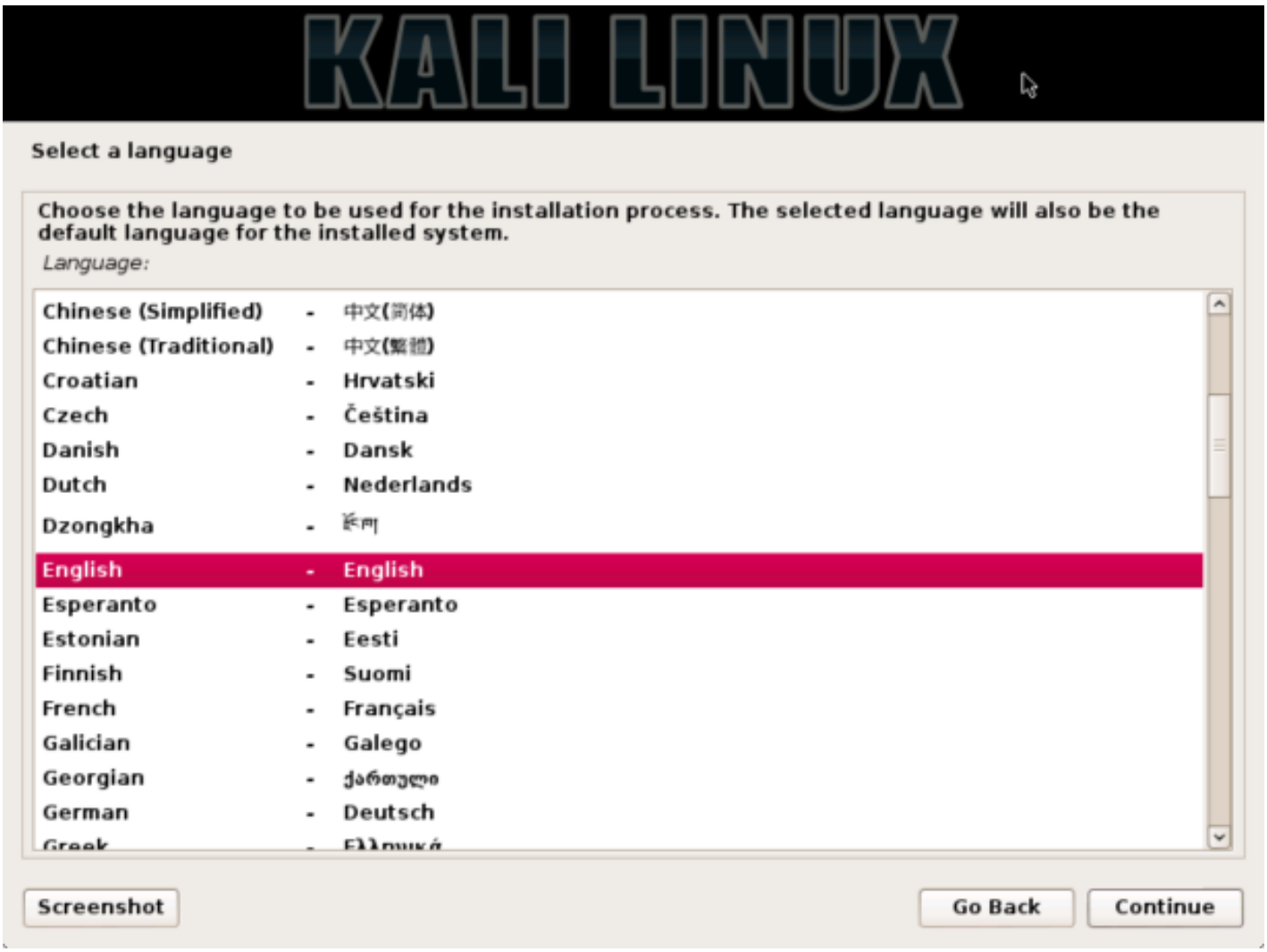
   Select the **Windows** volume to continue.
   - If you only see **one volume** (EFI), then the installation media **is not supported** for this device.

     You may wish to install **rEFInd** and try again.
   - If you select the EFI volume, the booting will hang at this point and you will **not** be able to continue.

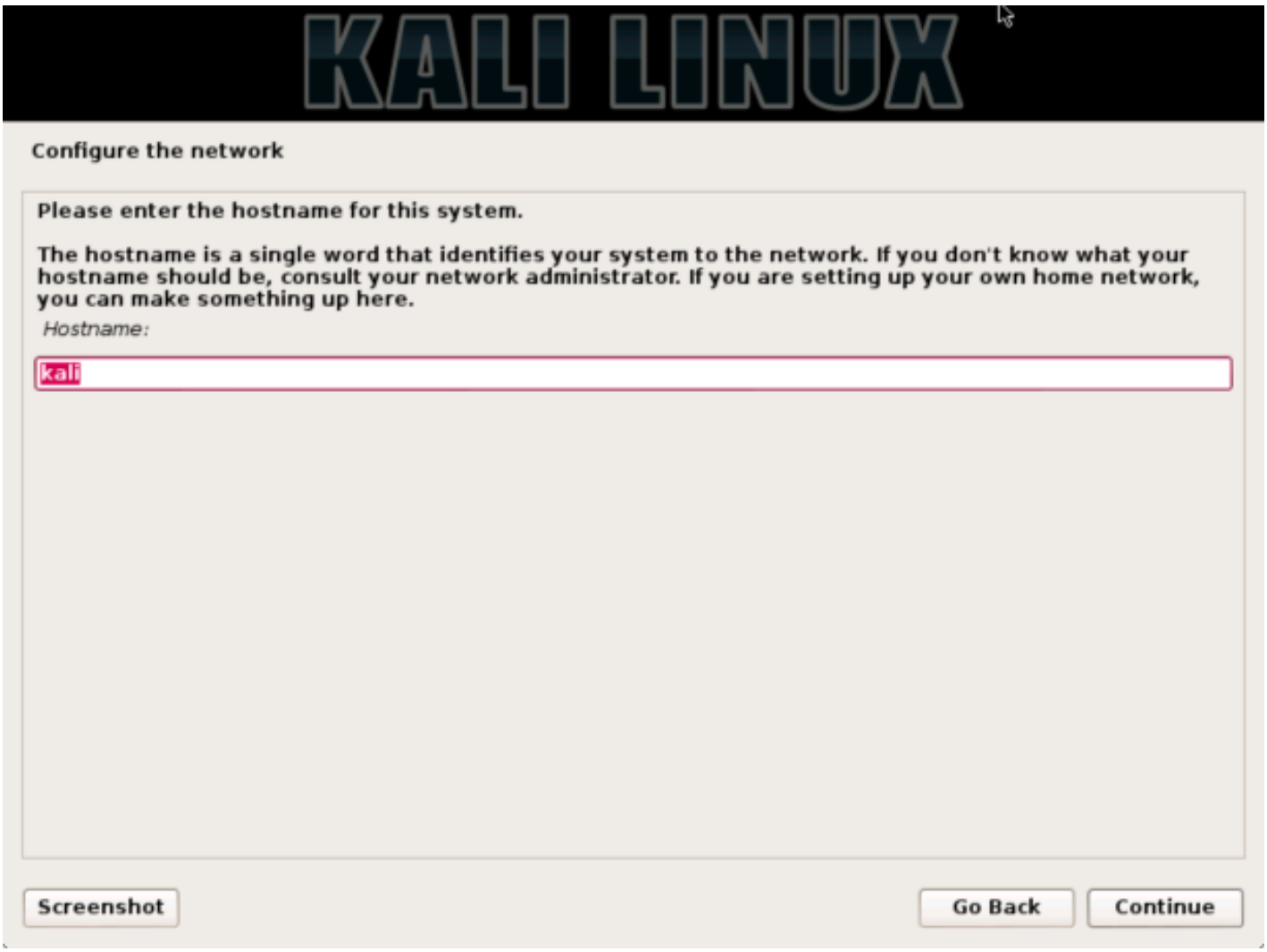
3. You should be greeted with the Kali boot screen. You are free to choose ‘Live’, ‘Graphical Install’, or ‘(Text-Mode) Install’ to install. In this example, we picked ‘Graphical install’.



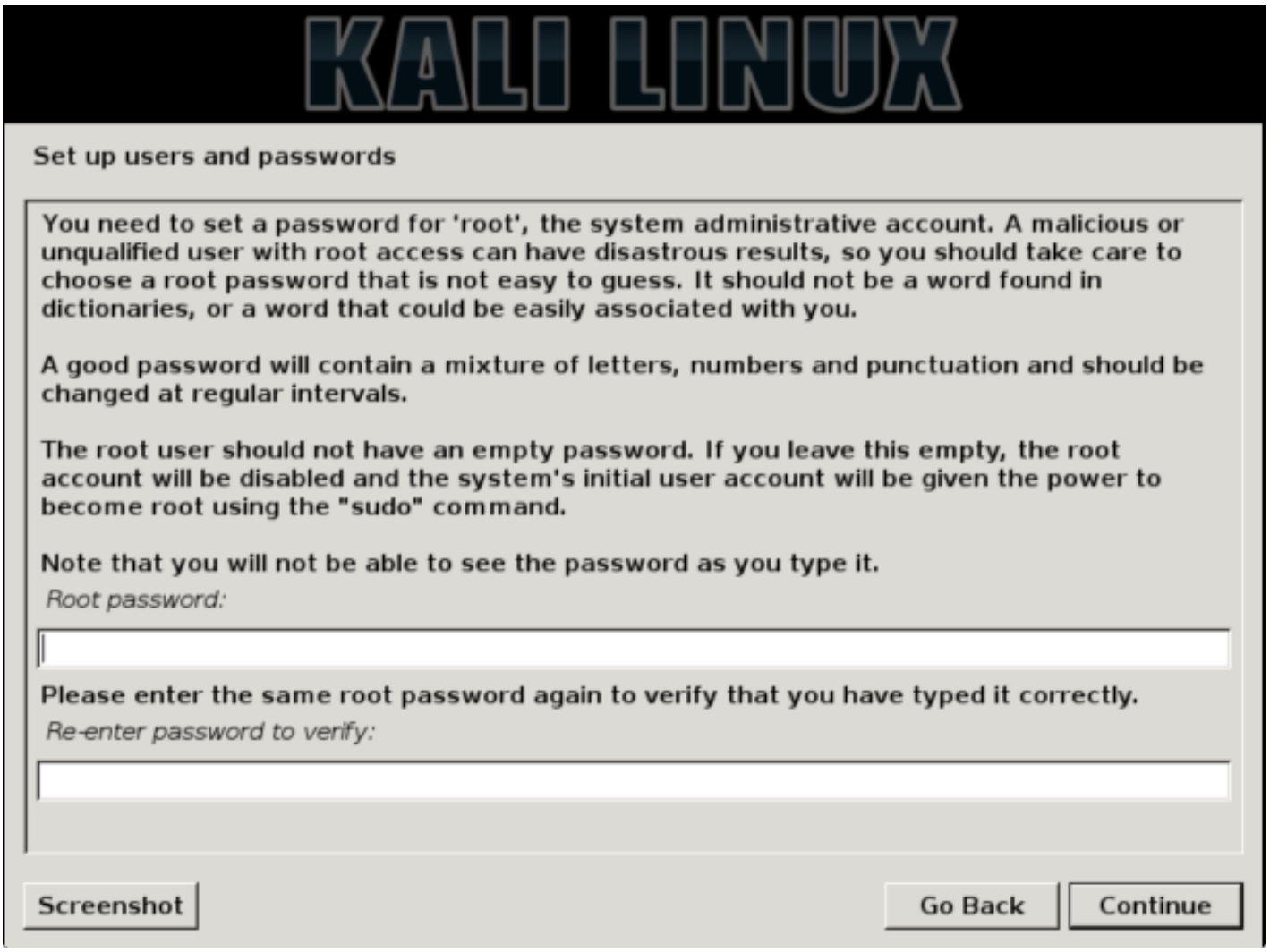
4. Select your preferred language and then your country location. You'll also be prompted to configure your keyboard with the appropriate keymap.



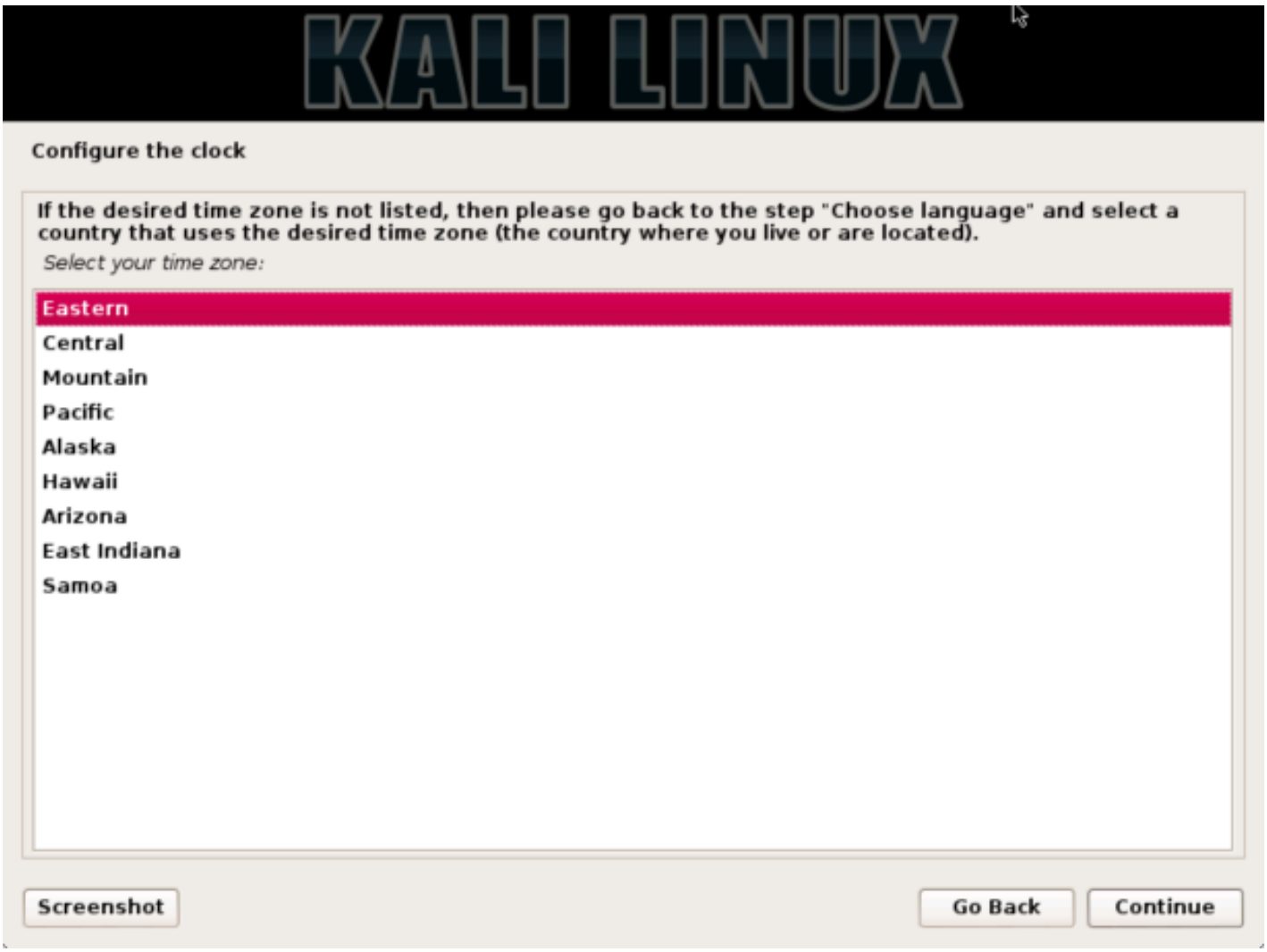
5. The installer will copy the image to your hard disk, probe your network interfaces, and then prompt you to enter a hostname and domain name for your system. In the example below, we've entered 'kali' as our hostname.
    - If the setup detects multiple NICs, it may prompt you which one to use for the install.
    - If the chosen NIC is 802.11 based, you will be asked for your wireless network information before being prompted for a hostname.
    - If there isn't a DHCP service running on the network, it will ask you to manually enter the network information after probing for network interfaces.
    - If Kali Linux doesn't detect your NIC, you either need to include the drivers for it when prompted, or generate a custom Kali Linux ISO with them pre-included.



6. Enter a robust password for the root account.

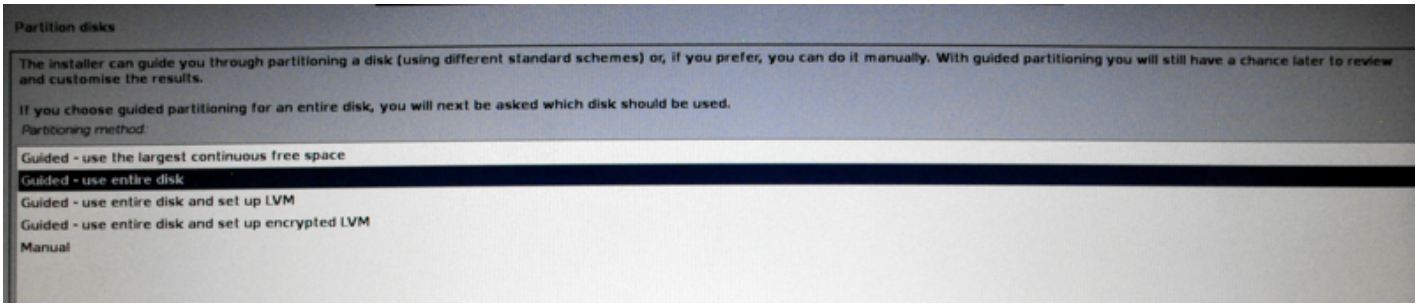


7. Next, set your time zone.

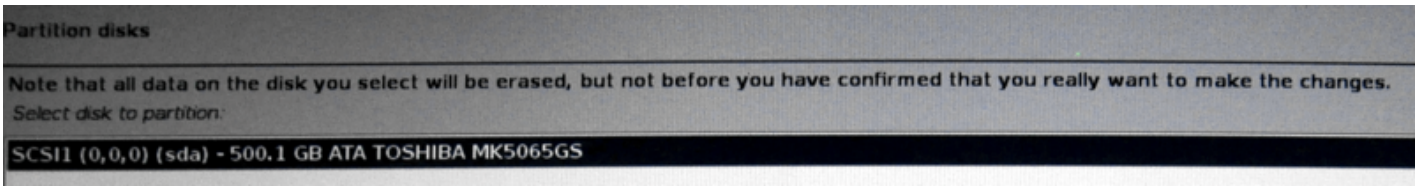


8. The installer will now probe your disks and offer you five choices. In our example, we're using the entire disk on our computer and not configuring LVM (logical volume manager), so we selected 'Guided – use the entire disk'.
    - Experienced users can use the 'Manual' option for more granular configuration options. This option will also allow you to set up encrypted LVM, so Kali Linux would be fully encrypted. The screen afterwards will prompt you for the password. You will have to enter the same password every time you start up Kali Linux.

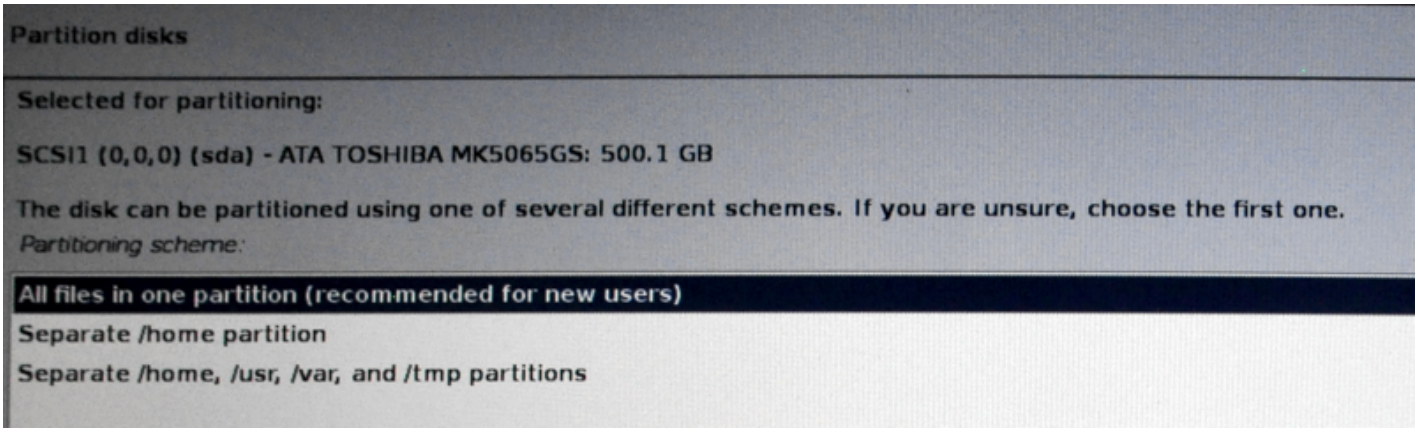
Kali will automatically securely wipe the hard disk before asking for the LVM password. This may take 'a while' (hours) depending on the size and speed of the drive. If you wish to risk it, you can skip it.



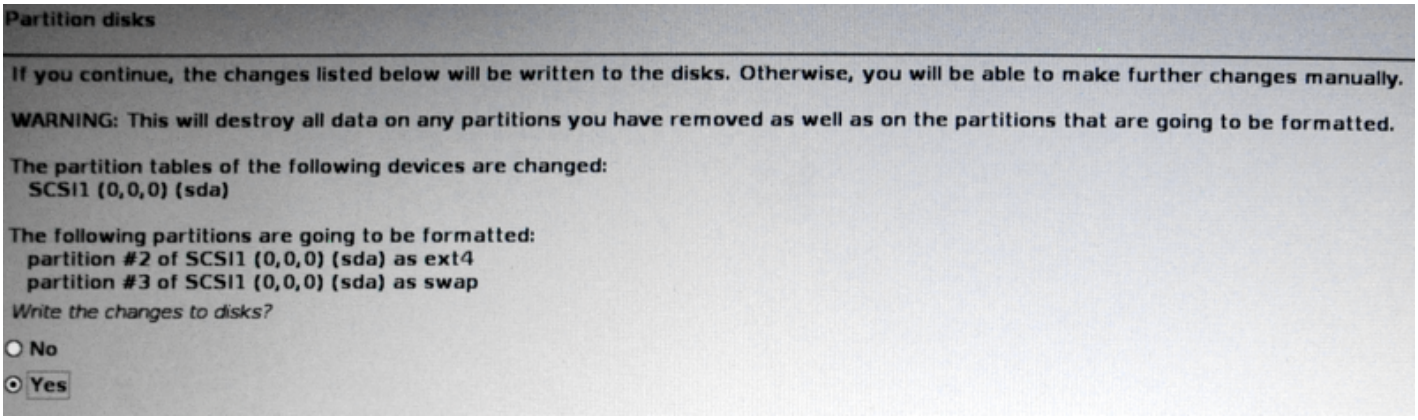
9. The installer will ask you to confirm which disk to erase. Double check then confirm the selection.



10. The next stage is to select the partition structure you want to use. We will go ahead and use the default option and have everything on one partition. Afterwards, the installer will display an overview. If you agree with what it suggests, press the continue button.

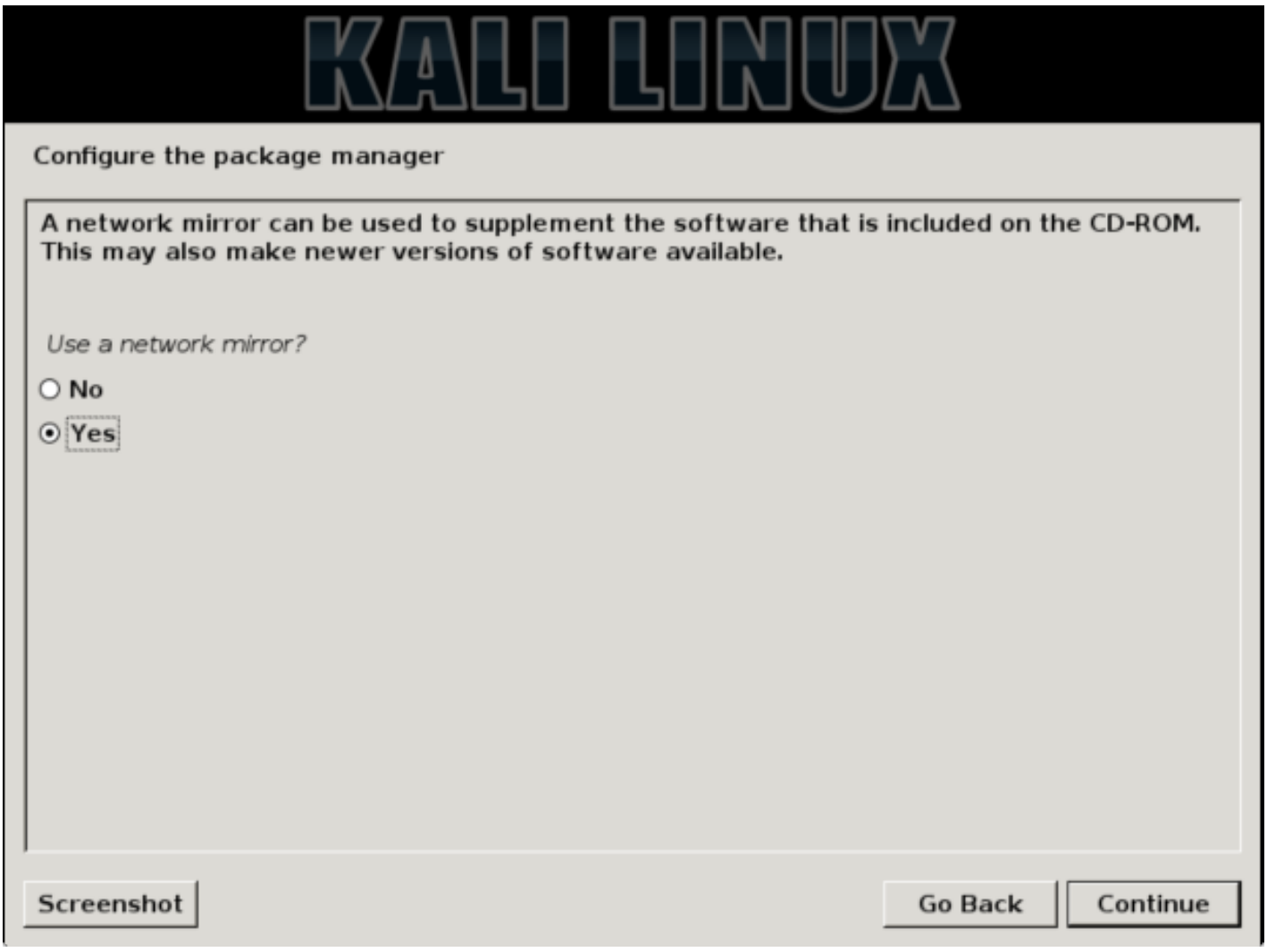


11. Next, you'll have one last chance to review your disk configuration before the installer makes irreversible changes. After you click **Continue**, the installer will go to work and you'll have an almost finished installation.

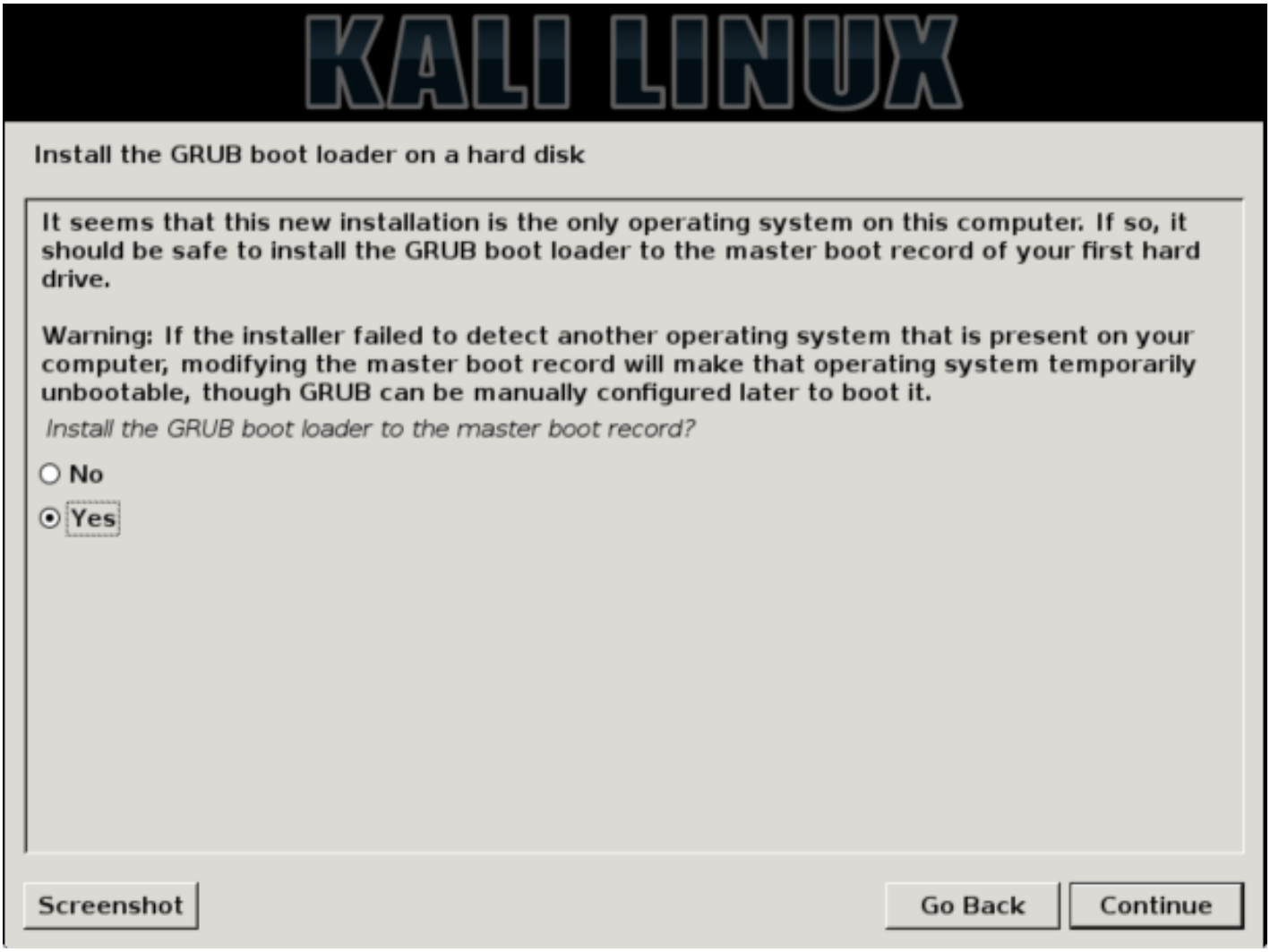


12. This screen configures the use of our Internet network mirrors. Kali can use our online central repository to distribute applications to keep packages up-to-date and allow for additional programs to be installed more easily. Should you need to enter any appropriate proxy information, the next screen will allow you to enter details.

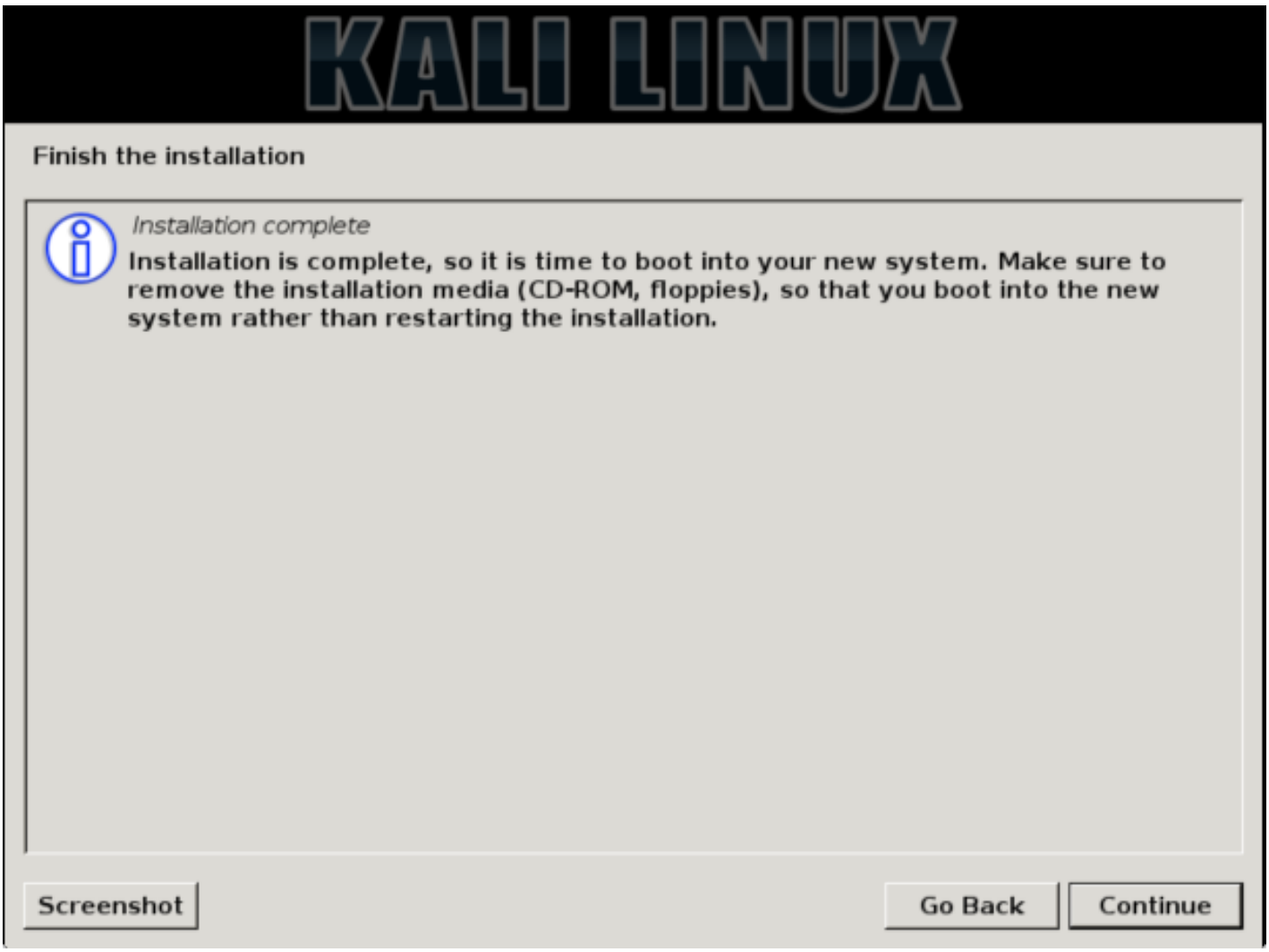
> If you select 'NO' in this screen, you will NOT be able to install packages from Kali repositories until you alter your sources.



13. Next, install the GRUB bootloader.



14. Finally, click **Continue** to finish installing Kali Linux. It is highly recommend that you restart your machine at this stage.
    Once the install has finished, repeat the first few steps again to boot into ‘Live mode’ once more.



15. If the gdisk package isn't included your Kali Linux ISO, you need to install it.
If you enabled the network repository during the setup, this can easily be done as follows:

```
apt-get update
apt-get install gdisk
```

16. We are now going to convert the MBR to a hybrid, which will allow for Apple's EFI to detect and boot to GRUB.

```
root@kali:~# gdisk /dev/sda
zsh: correct 'gdisk' to 'fdisk' [nyae]? n
GPT fdisk (gdisk) version 0.8.5

Partition table scan:
  MBR: protective
```

```
 BSD: not present
 APM: not present
 GPT: present

Found valid GPT with protective MBR; using GPT.



Command (? for help): p
Disk /dev/sda: 976773168 sectors, 465.8 GiB
Logical sector size: 512 bytes
Disk identifier (GUID): B6A4398E-3590-4BB7-AA57-D64EF74860D0
Partition table holds up to 128 entries
First usable sector is 34, last usable sector is 976773134
Partitions will be aligned on 2048-sector boundaries
Total free space is 4077 sectors (2.0 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1      2048      4095   1024.0 KiB  EF02
   2      4096       943585279   449.9 GiB   0700
   3       943585280       976771071   15.8 GiB    8200

Command (? for help): r

Recovery/transformation command (? for help): h

WARNING! Hybrid MBRs are flaky and dangerous! If you decide not to use one,
just hit the Enter key at the below prompt and your MBR partition table will
be untouched.

Type from one to three GPT partition numbers, separated by spaces, to be
added to the hybrid MBR, in sequence: 2
Place EFI GPT (0xEE) partition first in MBR (good for GRUB)? (Y/N): y

Creating entry for GPT partition #2 (MBR partition #2)
Enter an MBR hex code (default 07): 83
Set the bootable flag? (Y/N): y

Unused partition space(s) found. Use one to protect more partitions? (Y/N): n

Recovery/transformation command (? for help): w
```

```
Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): y
OK; writing new GUID partition table (GPT) to /dev/sda.
Warning: The kernel is still using the old partition table.
The new table will be used at the next reboot.
The operation has completed successfully.
root@kali:~#
```

17. After that is complete, all that is left is to reboot, take out the installation media, and enjoy Kali.

Kali Linux remote install via rescue system

Installing Kali Linux on a rented or colocated server can be useful for freelancers or enthusiasts alike. It provides a fast environment for performing network scans without having to go the extra mile to get a leased line at home.

Before starting the Installation, the server has to be booted into rescue mode and it’s network configuration saved. The extent of tool availability in the different rescue systems is taken care of where it matters.

First after connecting to the server via SSH, the disk has to be formatted. This is best achieved using `parted`, if available.

This is some test code.

> Some totally irrelevant quote from Ram Dass on the topic of encryption.

Kali Linux Encrypted Disk Install

At times, we have sensitive data we would prefer to encrypt using full disk encryption. With the Kali Installer, you can initiate an LVM encrypted install on either Hard Disk or USB drives. The installation procedure is very similar to a “normal Kali Linux Install”, with the exception of choosing an Encrypted LVM partition during the installation process.

Kali Linux Encrypted Installation Requirements

Installing Kali Linux on your computer is an easy process. First, you’ll need compatible computer hardware. The hardware requirements are minimal as listed below, though better hardware will naturally provide better performance. The i386 images have a default PAE kernel, so you can run them on systems with over 4GB of RAM. Download Kali Linux and either burn the ISO to DVD, or prepare a USB stick with Kali Linux Live as the installation medium.

Installation Prerequisites

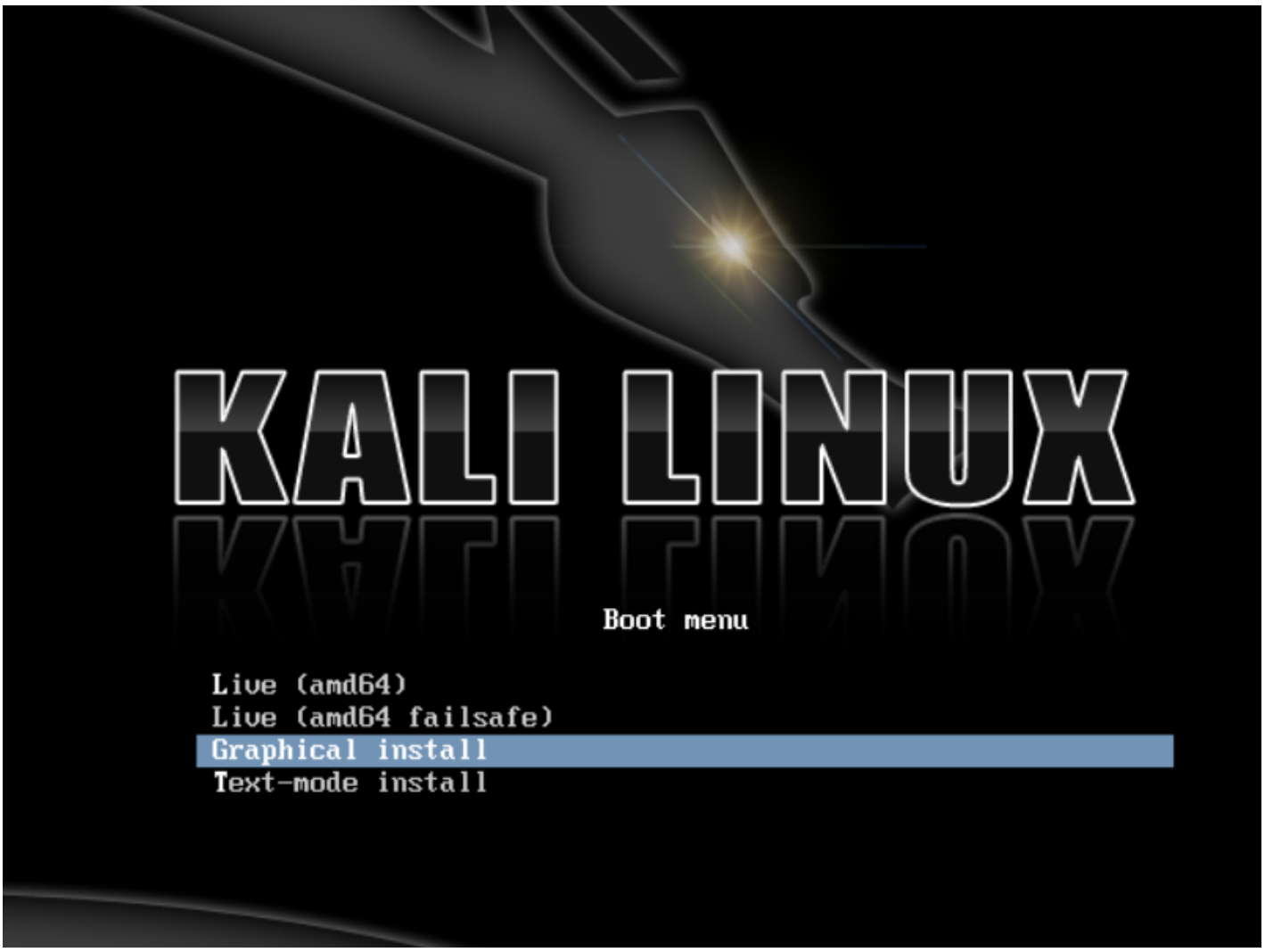
- A minimum of 20 GB disk space for the Kali Linux install.
- RAM for i386 and amd64 architectures, minimum: 1GB, recommended: 2GB or more.
- CD-DVD Drive / USB boot support

Preparing for the Installation

1. Download Kali linux.
2. Burn The Kali linux ISO to DVD or Image Kali Linux Live to USB.
3. Ensure that your computer is set to boot from CD / USB in your BIOS.

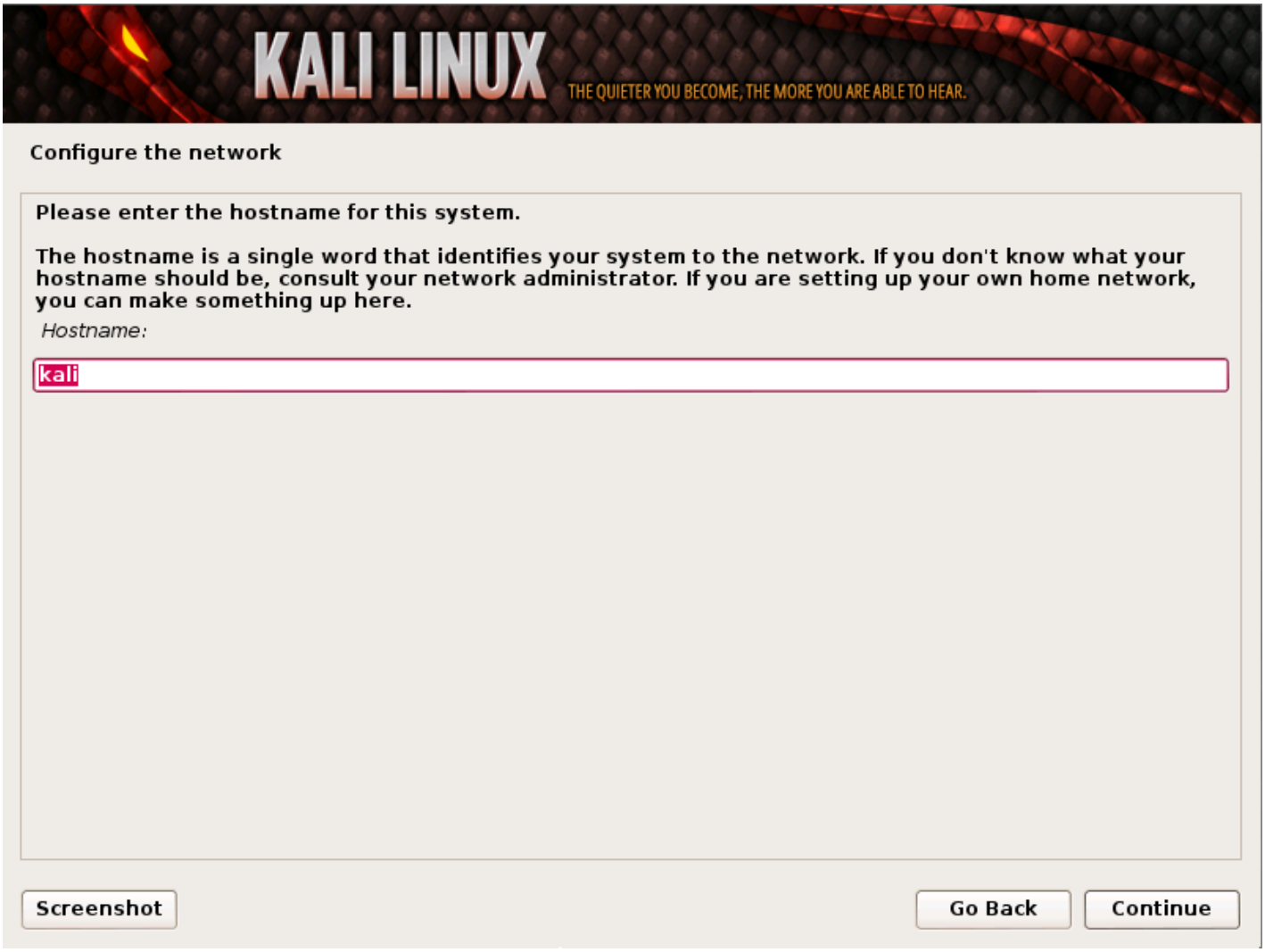
Kali Linux Installation Procedure

1. To start your installation, boot with your chosen installation medium. You should be greeted with the Kali Linux boot menu. Choose a *Graphical* or a *Text-Mode* install. In this example, we chose a GUI install.



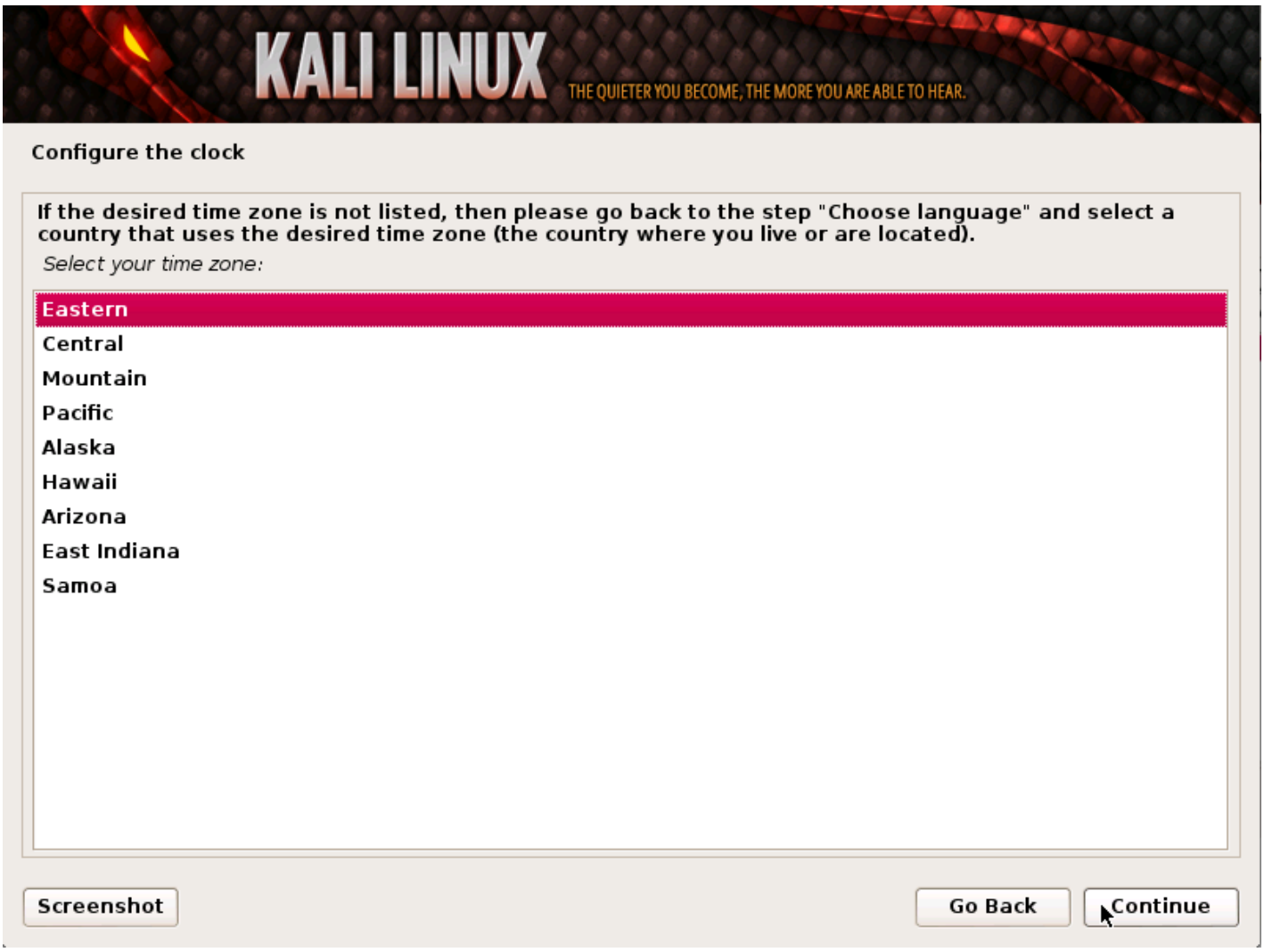
2. Select your preferred language and then your country location. You'll also be prompted to configure your keyboard with the appropriate keymap.

3. The installer will copy the image to your hard disk, probe your network interfaces, and then prompt you to enter a hostname for your system. In the example below, we've entered "kali" as the hostname.

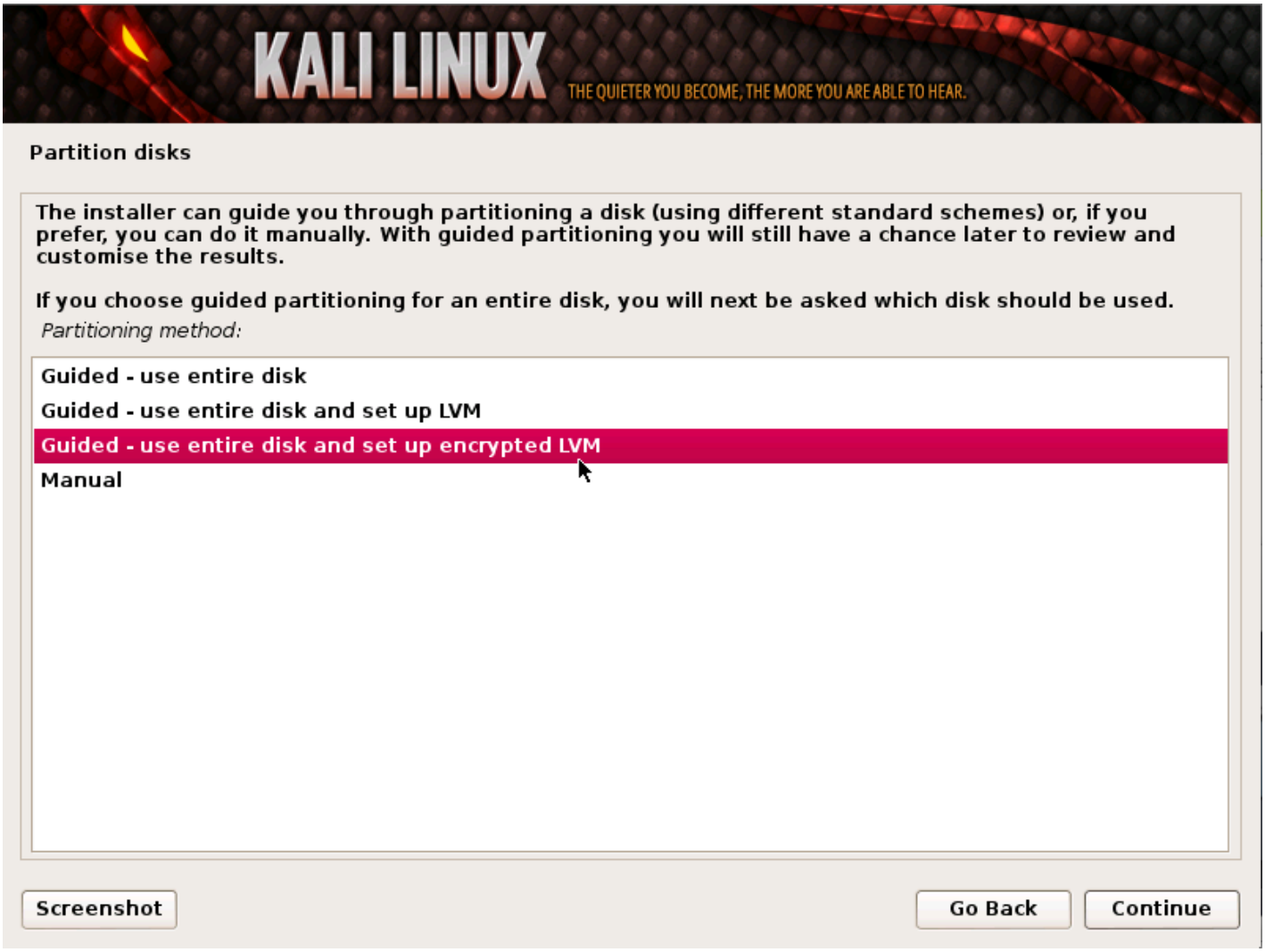


4. Enter a robust password for the root account.

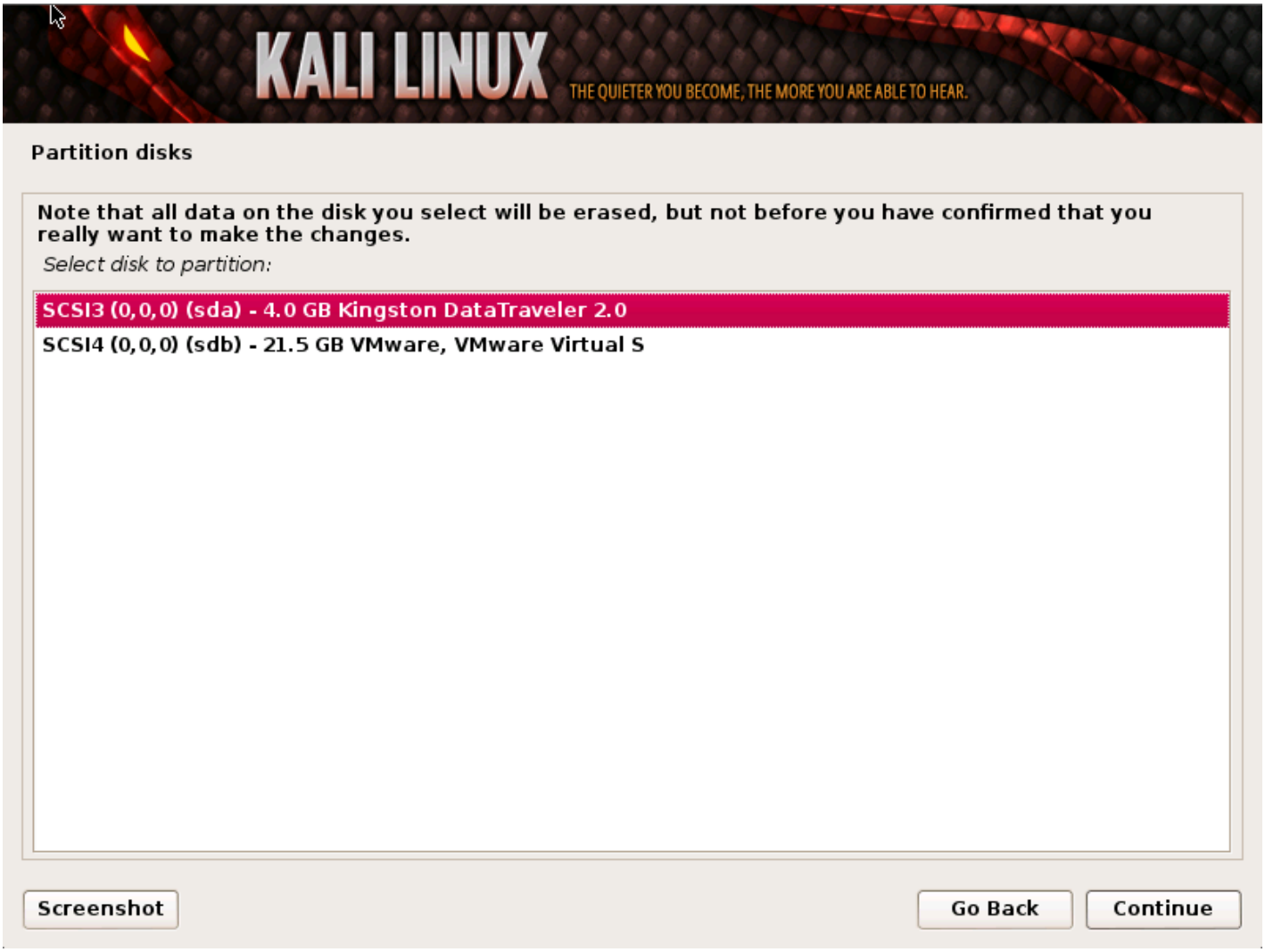
5. Next, set your time zone.



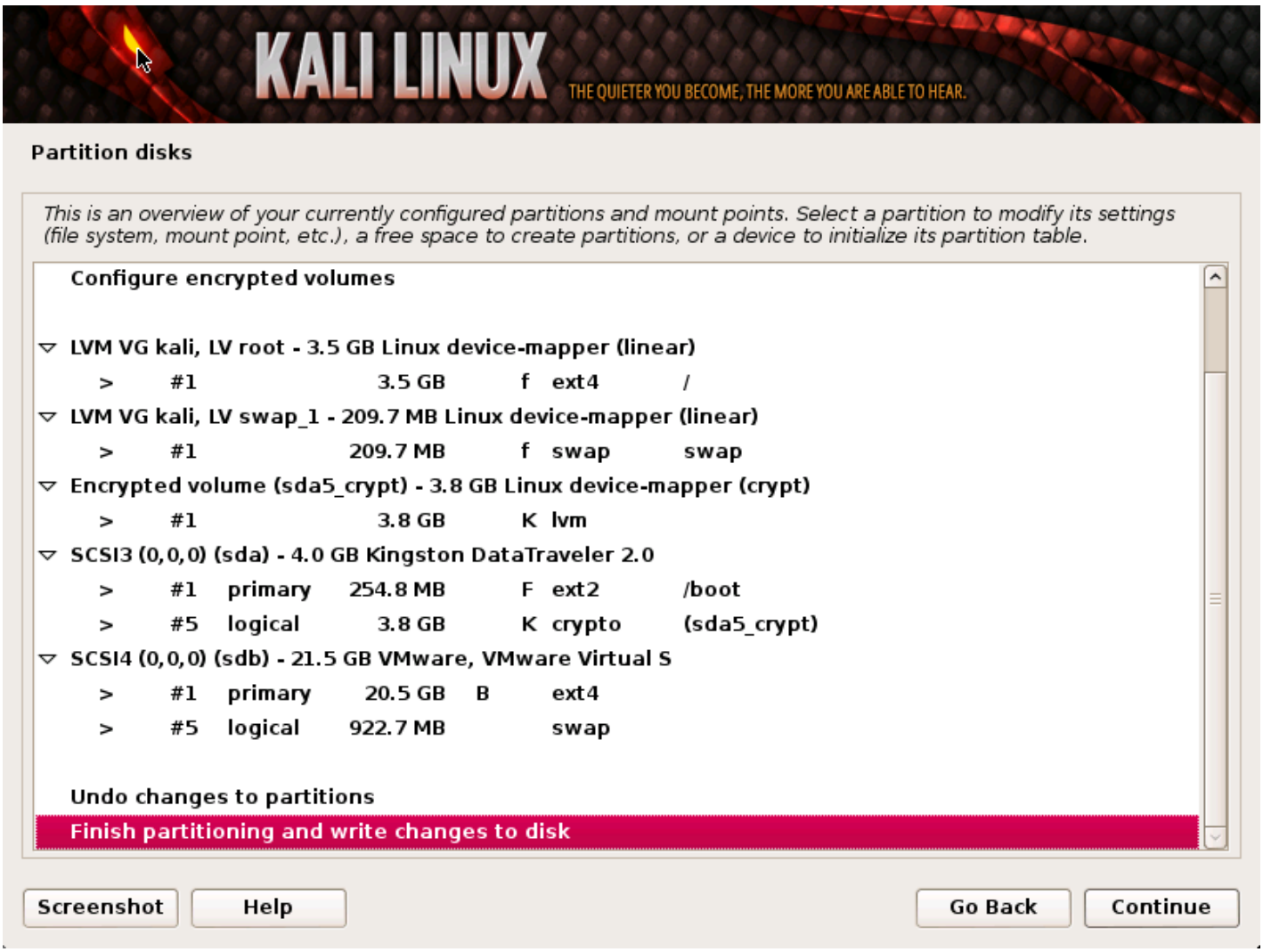
6. The installer will now probe your disks and offer you four choices. For an Encrypted LVM install, choose the “**Guided – use entire disk and set up encrypted LVM**” option as shown below.



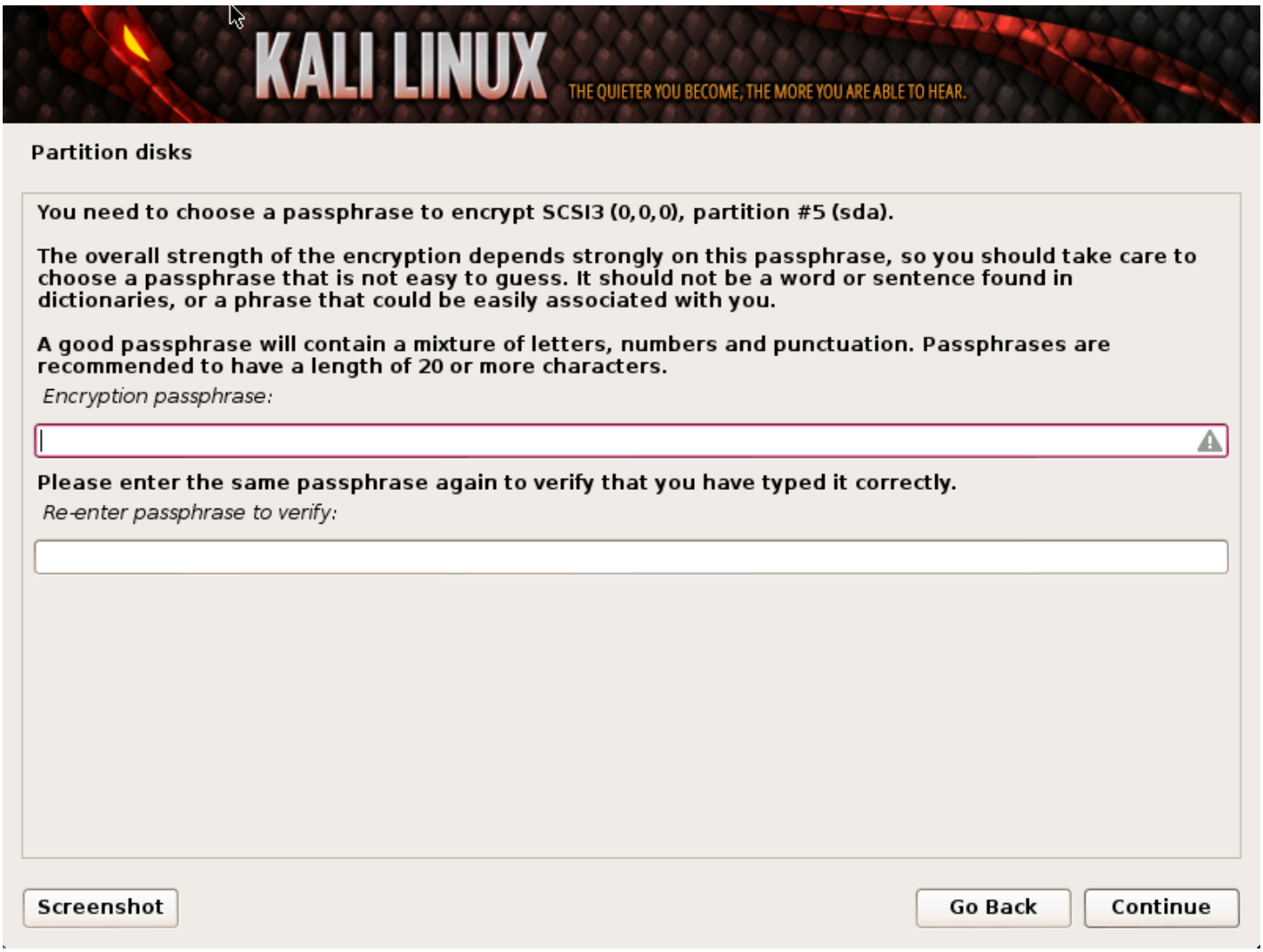
7. Choose the destination drive to install Kali. In this case, we chose a USB drive destination. We will use this USB drive to boot an encrypted instance of Kali.



8. Confirm your partitioning scheme and continue the installation.

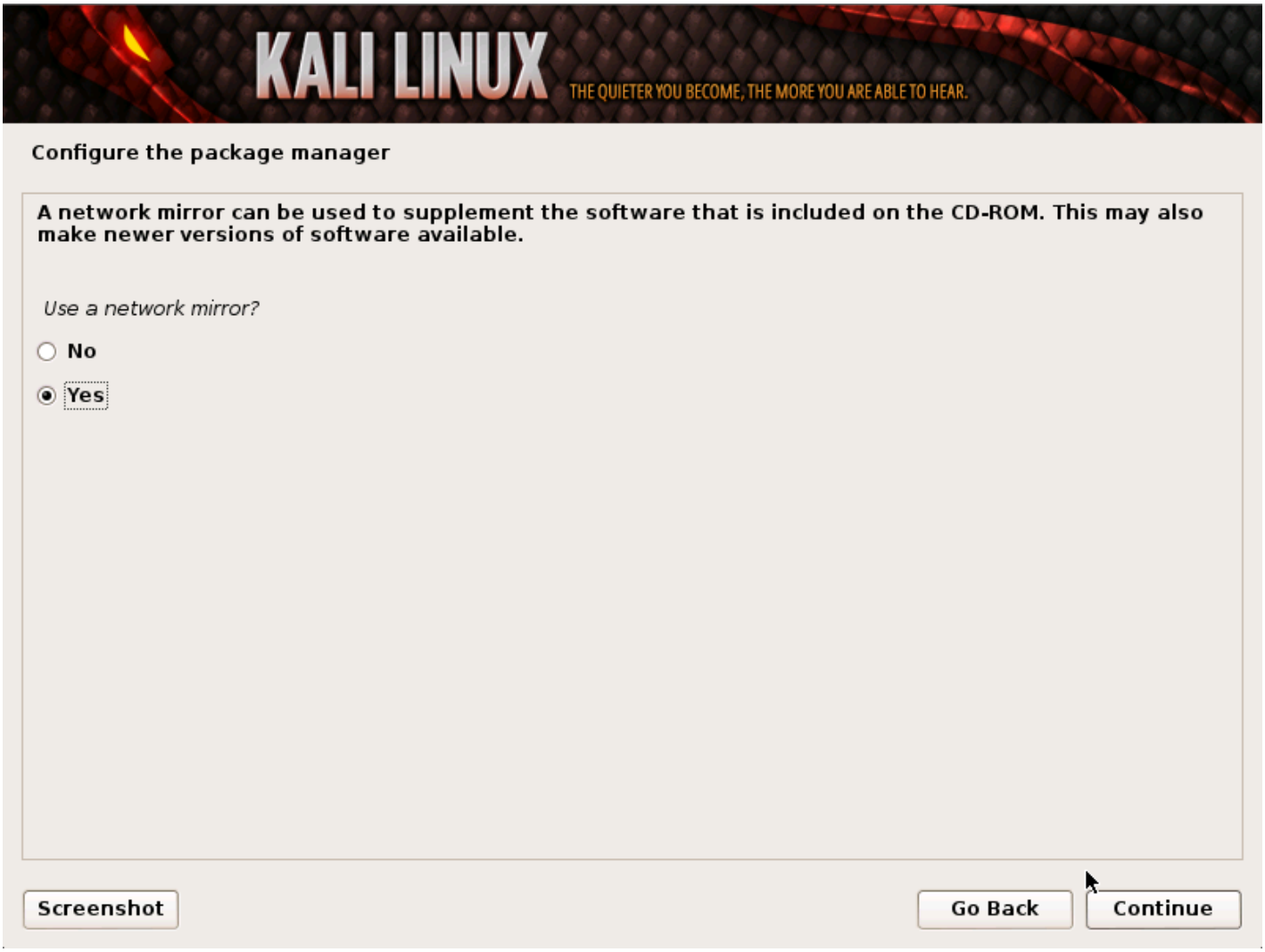


9. Next, you will be asked for an encryption password. You will need to remember this password and use it each time to boot the encrypted instance of Kali Linux.



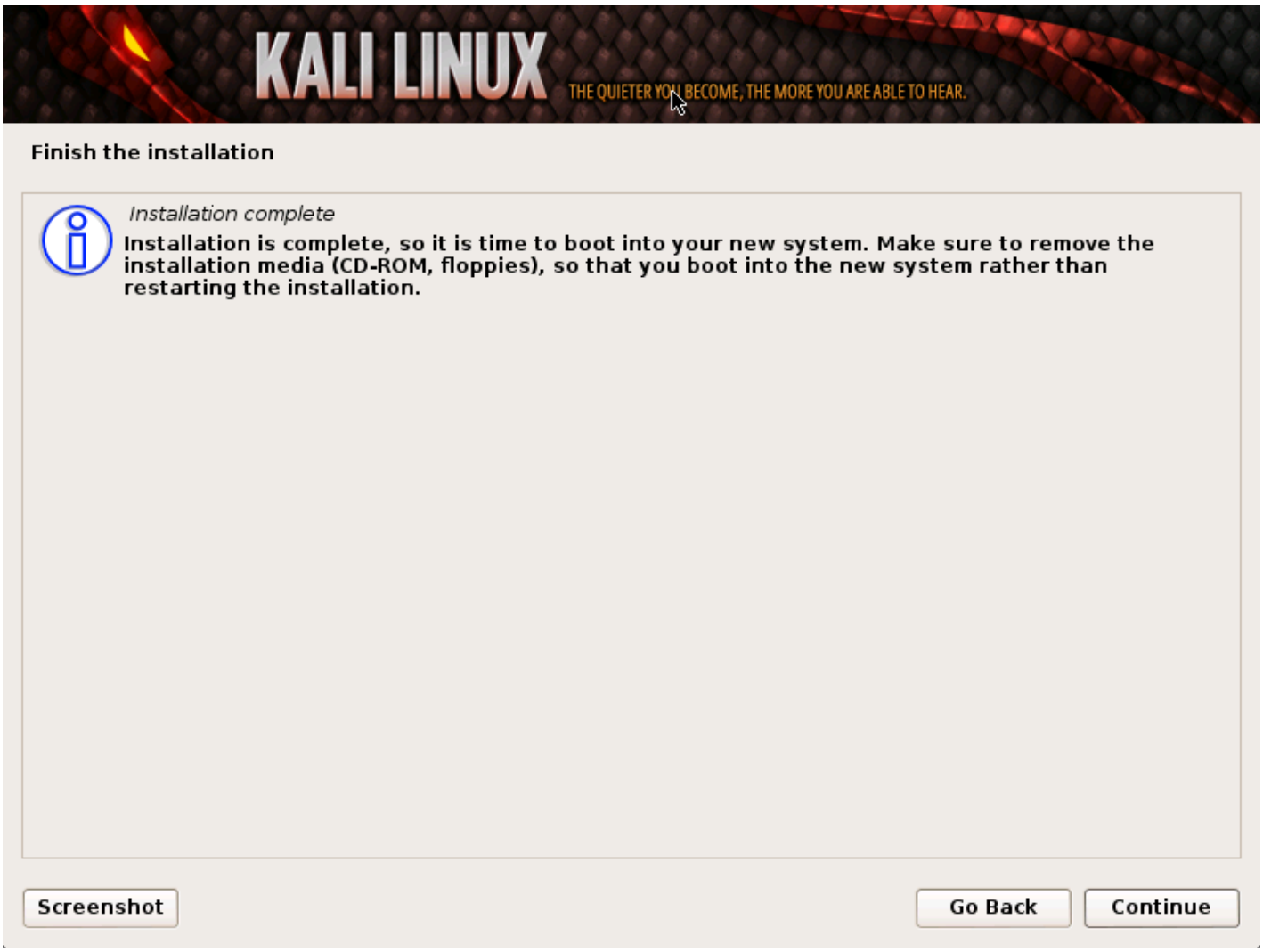
10. Configure network mirrors. Kali uses a central repository to distribute applications. You'll need to enter any appropriate proxy information as needed.

> **NOTE!** If you select "NO" in this screen, you will **NOT** be able to install packages from the Kali repositories.



11. Next, install GRUB.

12. Finally, click *Continue* to reboot into your new Kali installation. If you used a USB device as a destination drive, make sure you enable booting from USB devices in your BIOS. You will be asked for the encryption password you set earlier on every boot.



Post Installation

Now that you've completed installing Kali Linux, it's time to customize your system. The Kali General Use section of our site has more information and you can also find tips on how to get the most out of Kali in our User Forums.

Kali Linux Mini ISO Install

Kali Mini ISO Install

The Kali mini ISO is a convenient way to install a minimal Kali system and install it “from scratch”. The mini install ISO will download all required packages from our repositories, meaning you need to have a fast Internet connection to use this installation method.

Installation Prerequisites

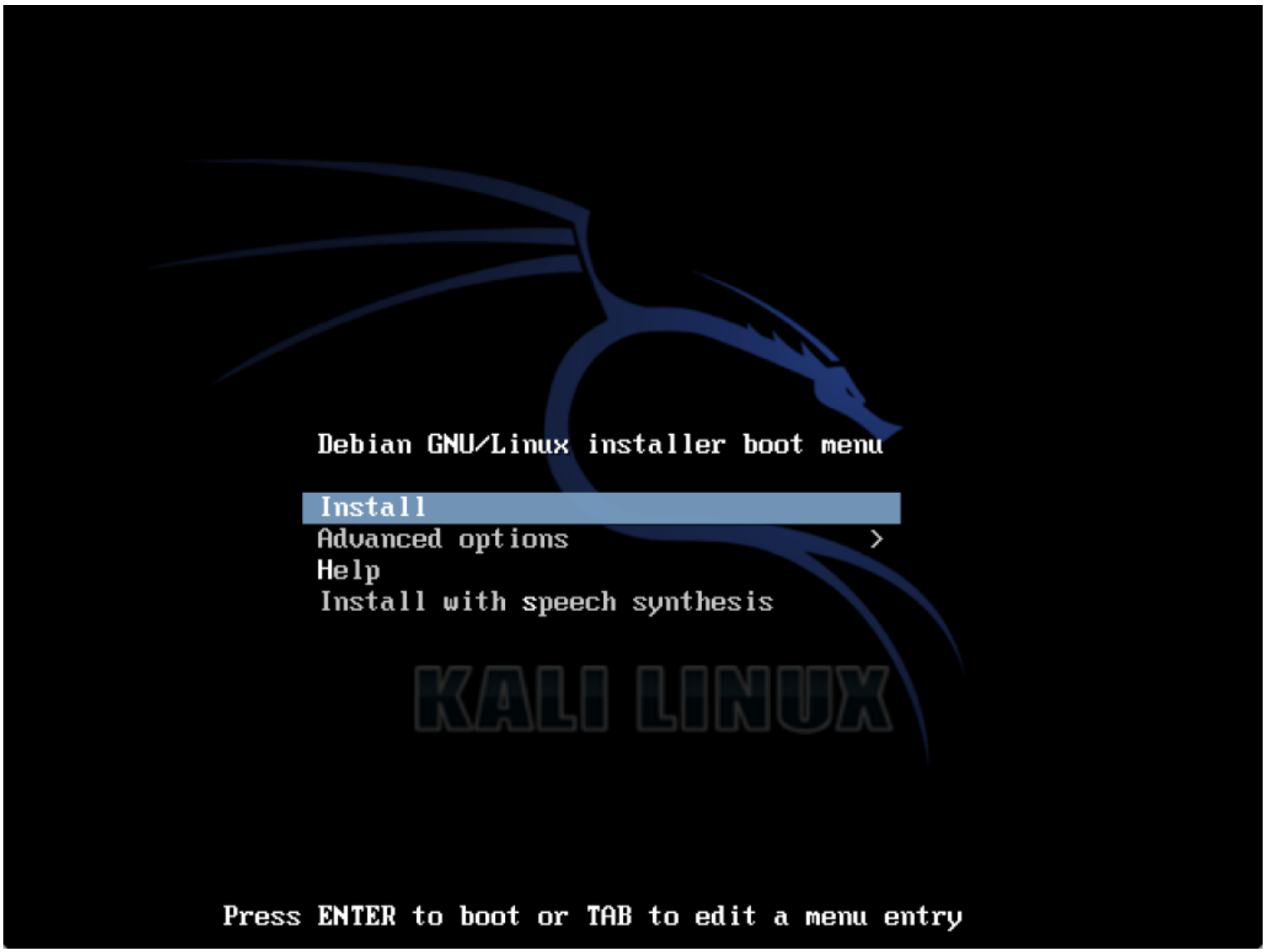
- A minimum of 8 GB disk space for the Kali Linux install.
- For i386 and amd64 architectures, a minimum of 512MB RAM.
- CD-DVD Drive / USB boot support

Preparing for the Installation

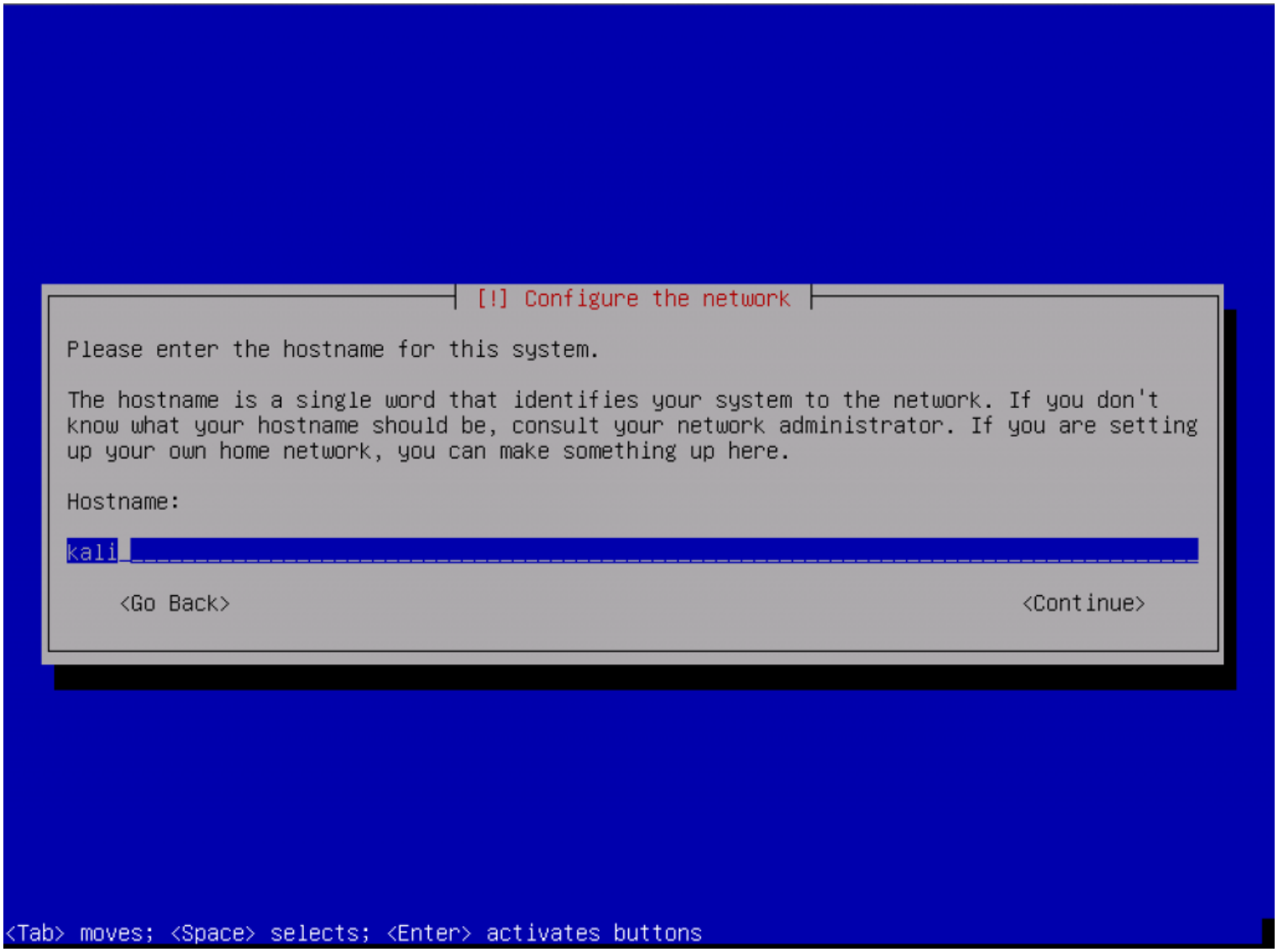
1. Download the Kali mini ISO
    - Text Installer
    - Graphical Installer
2. Burn The Kali Linux ISO to DVD or Image Kali Linux Live to USB.
3. Ensure that your computer is set to boot from CD / USB in your BIOS.

Kali Linux Installation Procedure

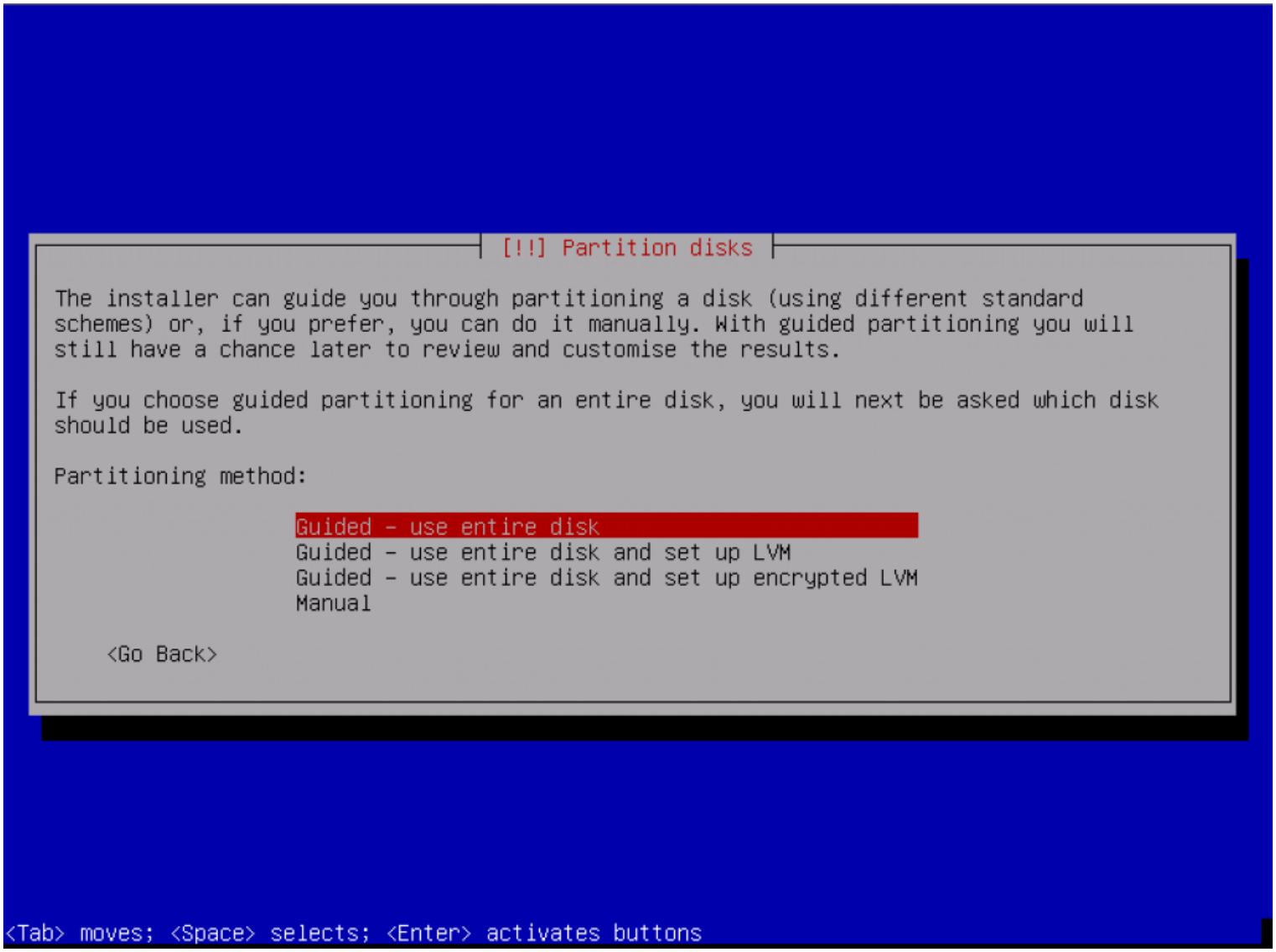
When you first boot the mini ISO, you will be presented with a small boot menu with various options. For this article, we will simply be doing a basic install.



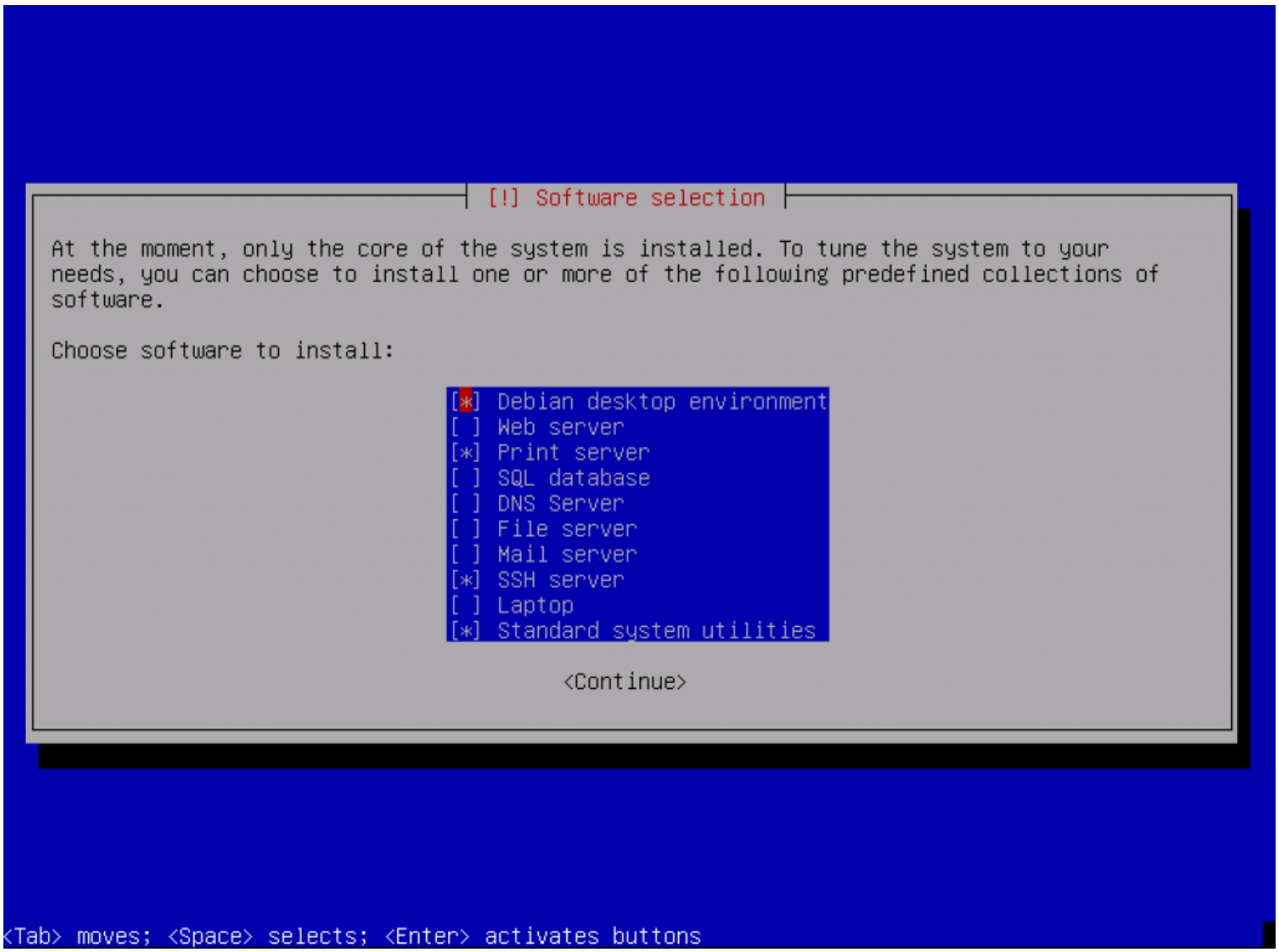
You will next be prompted for various things such as your language and keyboard type, then you will need to select a hostname for your installation. We will stick with the default of *kali*.



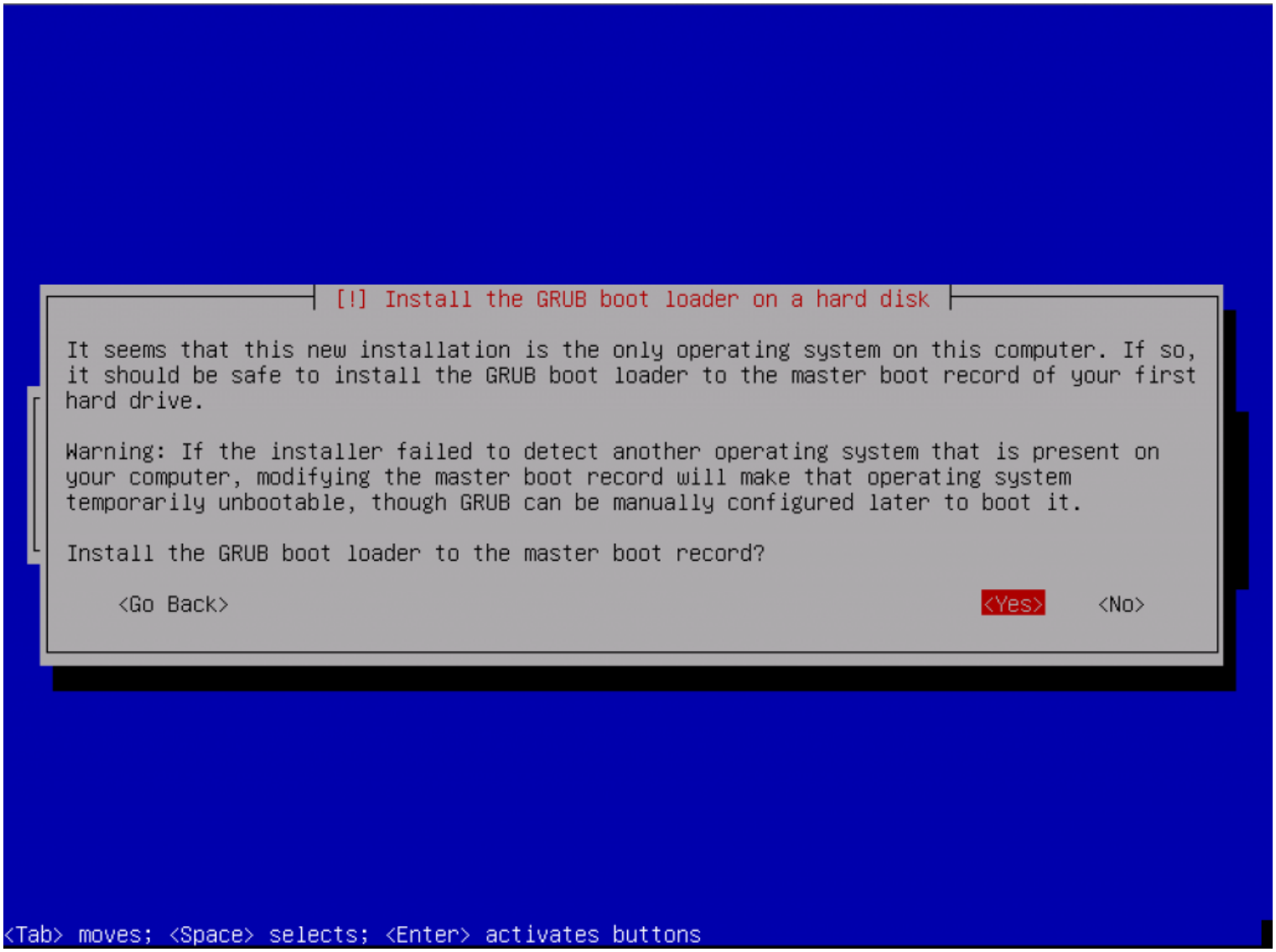
Next, you will need to select your time zone, then you'll be shown the partition options. To get up and running quickly, we will use 'Guided – use entire disk' and follow the prompts all the way through to create the new partitioning setup.



In order to reduce network bandwidth, a small subset of packages will be selected by default. If you wish to add different services or features, this is the area you would make your selections.



At this point, the installer will download all of the packages it requires and install them on the system. Depending on your Internet connectivity speed, this could take some time. Eventually, you will finally be prompted to install GRUB to finish the installation.



Post Installation

Now that you've completed installing Kali Linux, it's time to customize your system. The Kali General Use section of our site has more information and you can also find tips on how to get the most out of Kali in our User Forums.

Kali Linux Network PXE Install

Setup a PXE Server

Booting and installing Kali over the network (PXE) can be useful from a single laptop install with no CDROM or USB ports, to enterprise deployments supporting pre-seeding of the Kali installation.

First, we need to install *dnsmasq* to provide the DHCP/TFTP server and then edit the *dnsmasq.conf* file.

```
apt-get install dnsmasq
nano /etc/dnsmasq.conf
```

In *dnsmasq.conf*, enable DHCP, TFTP and PXE booting and set the *dhcp-range* to match your environment. If needed you can also define your gateway and DNS servers with the *dhcp-option* directive as shown below:

```
interface=eth0
dhcp-range=192.168.101.100,192.168.101.200,12h
dhcp-boot=pxelinux.0
enable-tftp
tftp-root=/tftpboot/
dhcp-option=3,192.168.101.1
dhcp-option=6,8.8.8.8,8.8.4.4
```

With the edits in place, the dnsmasq service needs to be restarted in order for the changes to take effect.

```
service dnsmasq restart
```

Download Kali PXE Netboot Images

Now, we need to create a directory to hold the Kali Netboot image and download the image we wish to serve from the Kali repos.

```
mkdir -p /tftpboot
cd /tftpboot
# for 64 bit systems:
wget http://http.kali.org/kali/dists/kali-rolling/main/installer-amd64/current/images/netboot/netboot.tar.gz
# for 32 bit systems:
wget http://http.kali.org/kali/dists/kali-rolling/main/installer-i386/current/images/netboot/netboot.tar.gz
tar zxpf netboot.tar.gz
rm netboot.tar.gz
```

Configure Target to Boot From Network

With everything configured, you can now boot your target system and configure it to boot from the network. It should get an IP address from your PXE server and begin booting Kali.

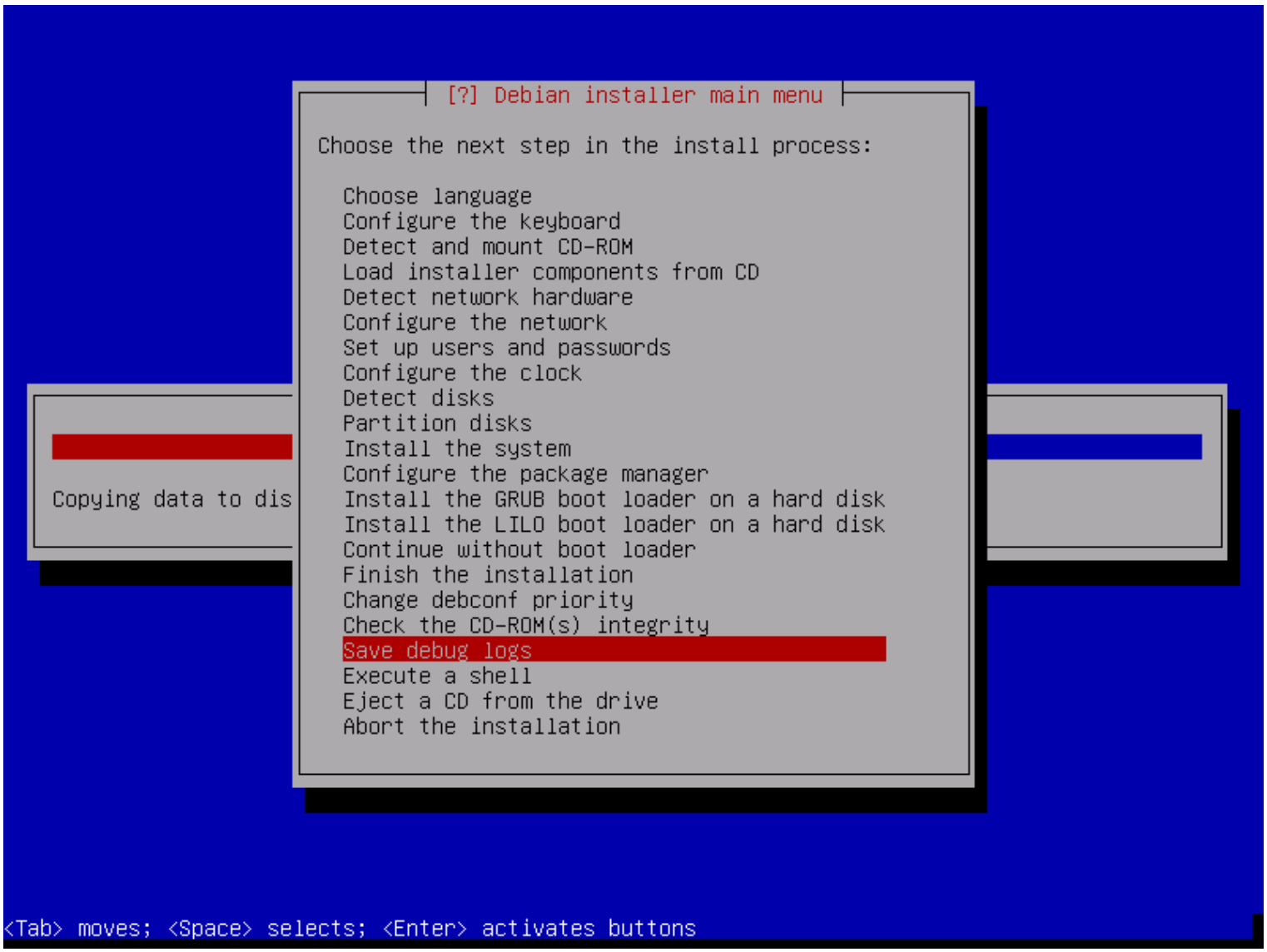
Troubleshooting Installations

Kali Linux installation failures

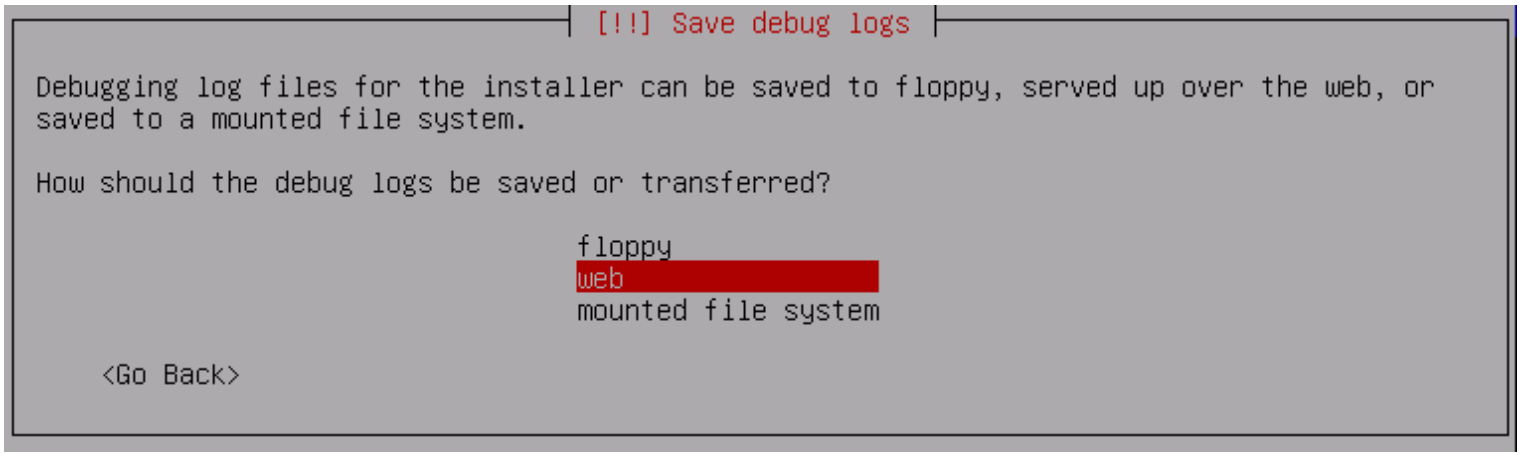
There can be a wide variety of reasons for a Kali Linux installation to fail. This could include issues such as a corrupt or incomplete ISO download, not enough disk space on the target machine, etc. The following article will give you some pointers on what to look for when your Kali Linux installation has failed. The following is an example of the dreaded “Red Screen”, indicating the installation encountered a problem.


[!!] Install the system
Installation step failed
An installation step failed. You can try to run the failing item again from the menu, or skip it and choose something else. The failing step is: Install the system
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons

Hitting the **continue** button should take you to the **Debian installer main menu**. From that main menu, browse to the "**save debug logs**":


[?] Debian installer main menu
Choose the next step in the install process:
Choose language
Configure the keyboard
Detect and mount CD-ROM
Load installer components from CD
Detect network hardware
Configure the network
Set up users and passwords
Configure the clock
Detect disks
Partition disks
Install the system
Configure the package manager
Install the GRUB boot loader on a hard disk
Install the LILO boot loader on a hard disk
Continue without boot loader
Finish the installation
Change debconf priority
Check the CD-ROM(s) integrity
Save debug logs
Execute a shell
Eject a CD from the drive
Abort the installation
Copying data to dis
<Tab> moves; <Space> selects; <Enter> activates buttons

Going into the debug logs, you are presented with several ways of transferring the installation log files away from the failed installation. The most convenient way is usually to start a web server on the machine undergoing the installation.



Once you choose this option, a web server is started from which you are able to download or view several installation log files.

Look over the logs files for anything irregular, or any error messages which look like they might be the cause of your failed installation. In this case, the target machine did not have enough disk space to install Kali Linux, as was seen towards the end of the **syslinux** log file

```
Aug 19 23:45:05 base-installer: error: The tar process copying the live system failed (only 152937 out of
```

286496 files have been copied, last file was).
Aug 19 23:45:05 main-menu[927]: (process:7553): tar: write error: No space left on device
Aug 19 23:45:05 main-menu[927]: WARNING **: Configuring 'live-installer' failed with error code 1
Aug 19 23:45:05 main-menu[927]: WARNING **: Menu item 'live-installer' failed.
Aug 19 23:50:23 main-menu[927]: INFO: Modifying debconf priority limit from 'high' to 'medium'
Aug 19 23:50:23 debconf: Setting debconf/priority to medium
Aug 19 23:56:49 main-menu[927]: INFO: Menu item 'save-logs' selected

04. Kali Linux on ARM

Kali Linux - Gem PDA

Gem Installation Guide

The Gemini PDA is a multi-boot Android smartphone with a keyboard that supports the installation of up to three operating systems side-by-side (currently any three of the following: Android, Sailfish, Debian, Kali Linux).

Operating systems aren’t installed but flashed using the Smart Phone Flash tool provided by Mediatek. To flash a new Gemini PDA with rooted Android and Kali Linux requires only four steps:

1. Download and extract the Kali-Gem firmware archive, which includes everything to setup the Gemini with the following partition table:
    1. Android (rooted), 16 GB
    2. Kali Linux, 40 GB

3. Empty

2. Download, install, and run the SP Flash tool
3. Backup the current NVRAM partition
4. Flash the Kali-Gem firmware

1. Download and Extract the Firmware

The Kali Linux Gemini image can be downloaded from the Offensive Security ARM images page. The folder contains all files required for a complete re-flash of the Gemini PDA with bootloader, a rooted Android partition and Kali-Linux.

> If you would like a different partition layout, follow the official flashing guide. Make sure that the Linux partition is at least 6GB.With a linux partition created, you can write the image file ***linux_root.img*** straight into it. Don't forget to flash the corresponding Linux kernel image "linux_boot.img".

The rest of the steps in this guide detail a complete re-flash of a Gemini x27 with a Windows PC.

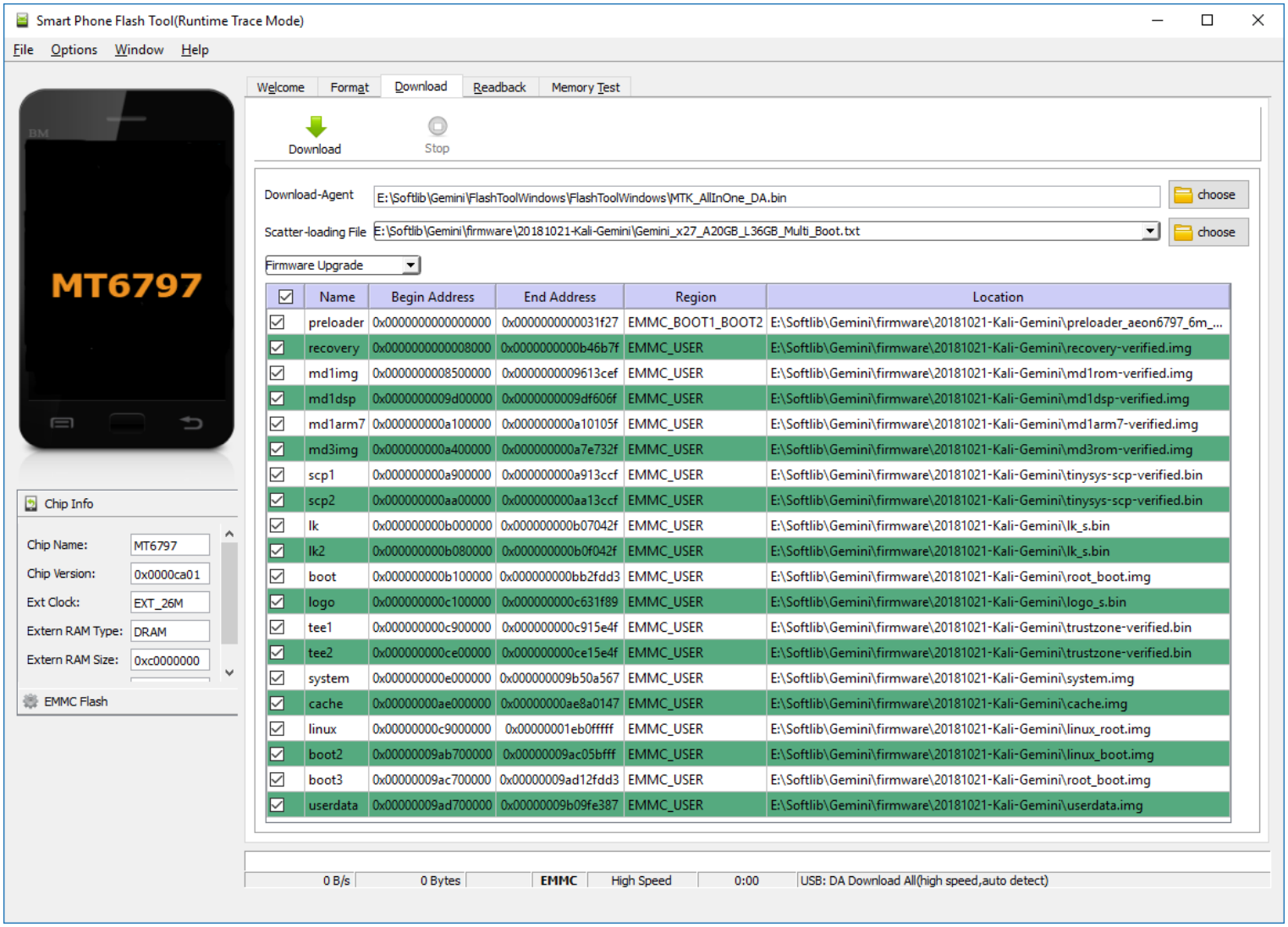
2. Download and Install the Drivers and Flash Tool

1. You can find the latest drivers here: Windows Flash Tool Drivers
2. Once downloaded, unzip the archive
3. You will find a folder called ***FlashToolDrivers***. Open the folder and double click on the Install (***install.bat***) file
4. Let the installation run and choose "Yes" when asked to make changes
5. Now that the drivers have been installed, you can download the latest Windows flash tool: Windows Flash Tool
6. Unzip the downloaded zip file containing the Windows FlashTool directory
7. Next, run 'flash_tool.exe' in the ***FlashToolWindows*** folder

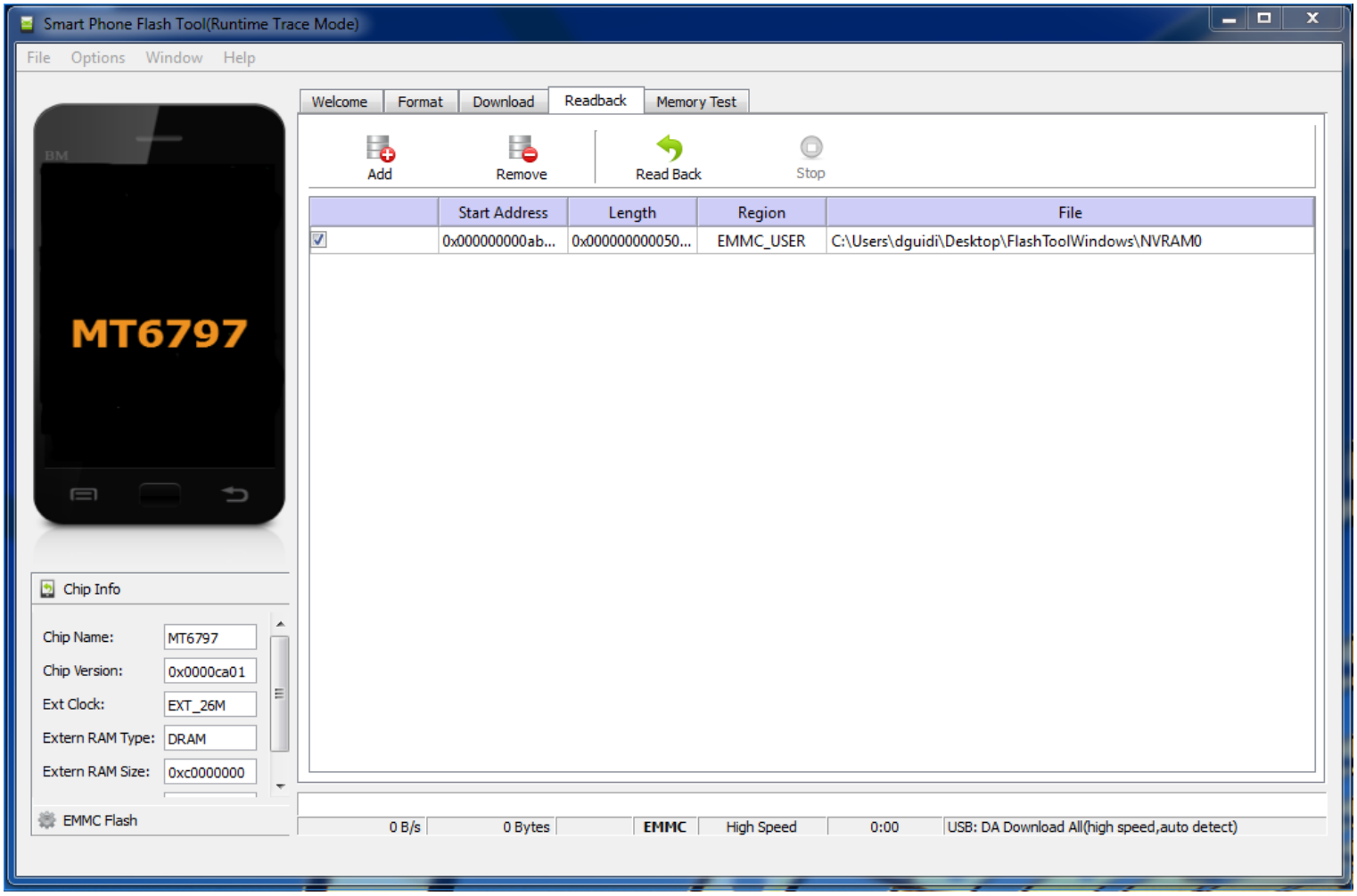
3. Backup the Current NVRAM Partition

Before flashing the device with a different firmware, it is a good idea to backup the current NVRAM partition. This partition stores key information for your Gemini, including the IMEI number. If it gets lost or damaged, your Gemini will not be able to make or receive calls.

To create a backup of your NVRAM partition, first select the "Scatter-loading file" by pressing the "choose" button and select ***Gemini_x27_A20GB_L36GB_Multi_Boot.txt*** inside the downloaded and extracted firmware folder. You should now see a partition table similar to the following:



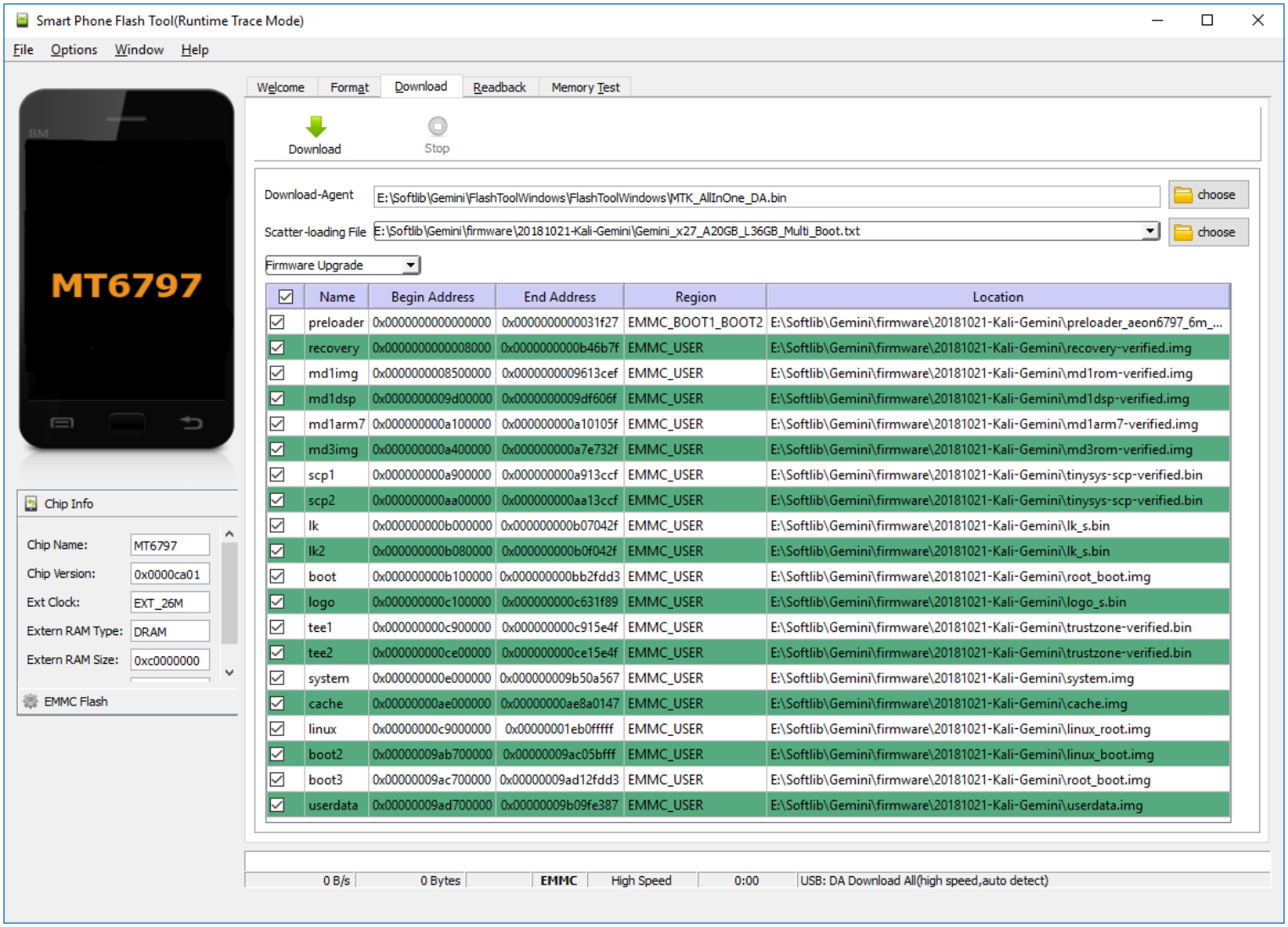
Next go to the “Readback” tab and click on the “Add” button. A row will appear in the table as in the following screenshot:



To back up the NVRAM partition, click the “Read Back” button, connect your Gemini to your PC, and power on the Gemini by pressing the “Esc” button for about a second or two. The flash tool will detect the unit and back up the partition.

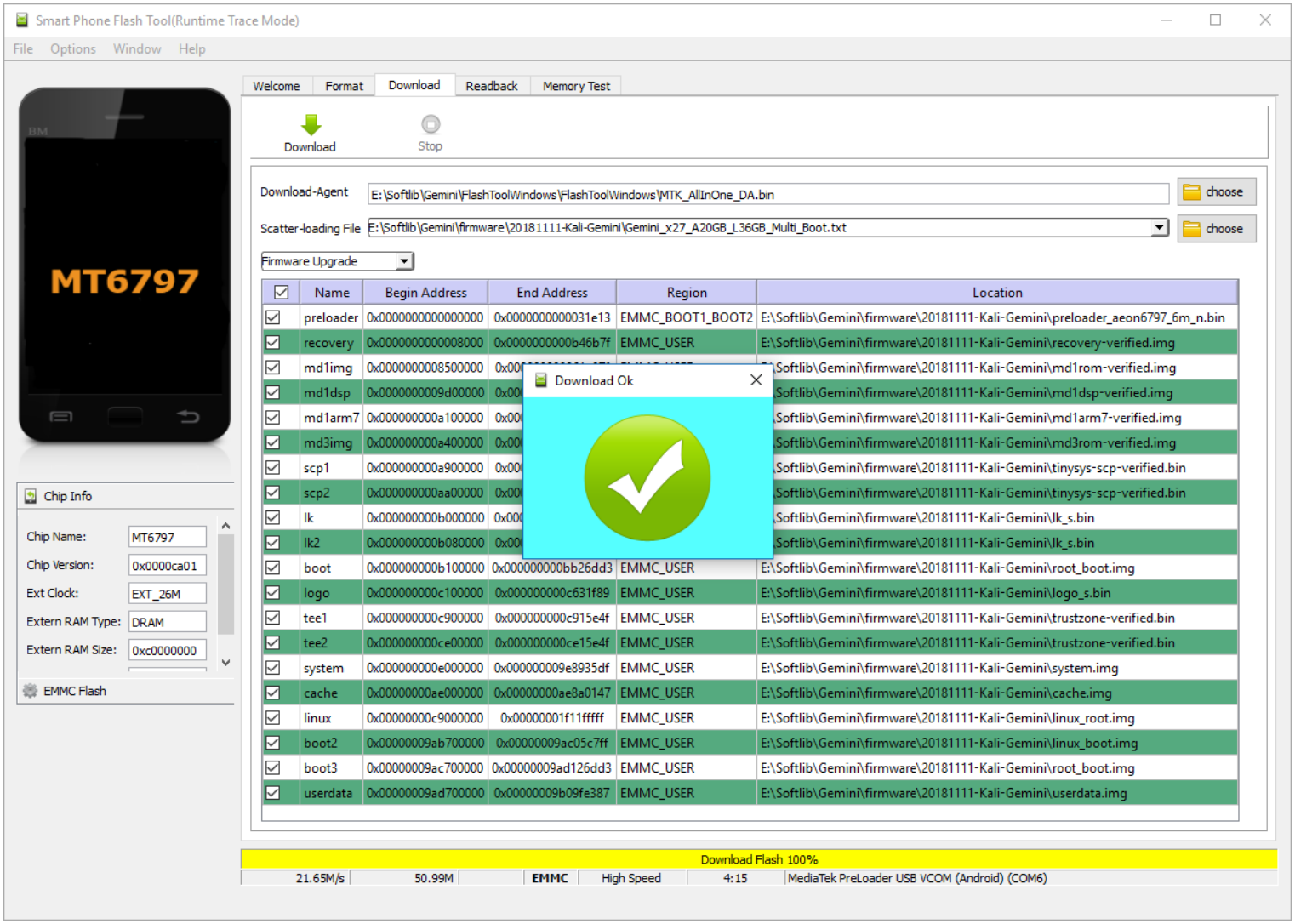
4. Flash the Kali Linux Firmware

- Click on the “Download” tab
- Choose the “Scatter-loading file” by pressing the “choose” button as in the following screenshot and select ***Gemini_x27_A20GB_L36GB_Multi_Boot.txt*** inside the downloaded and extracted firmware folder. Ensure that the other settings match the following:
    - “Download-Agent” should be set to the file ***MTK_AllInOne_DA.bin***, which is located in the ***FlashToolWindows*** or ***FlashToolLinux*** folder
    - “Scatter-loading file” should be set to the specific scatter file of the firmware that was customized for the Kali-Gem built, which is located in the firmware folder
- Select the “Firmware Upgrade” option from the drop down menu. This will automatically select all the partitions in the table
- Your screen should look like the following:



To start the flashing process, just click on the big “Download” button, connect your Gemini to your PC, and power on the Gemini by pressing the “Esc” button for about a second or two.

Once booting, the flash tool will detect the unit and will start flashing the device with the selected firmware. The following screenshot shows a successfully completed flashing process:



Boot Notes

The multi-boot mechanism works as follows. Starting from a switched off Gemini, press the “Esc” (On) key to start the unit until the Gemini vibrates. Once you feel the vibration, you can choose the boot mode by pressing the following key combination:

- Boot 1 (Android): Default booting option when no keys or buttons are pressed
- Recovery Mode: Esc (On) is pressed. This will always boot into recovery mode
- Boot 2 (Kali Linux): Silver button on the right hand side of the device is pressed
- Boot 3 (N/A): Both Esc(On) key and silver button on the right hand side of the device are pressed at the same time. Keep the keys/buttons pressed until the screen turns ON

Logging in for the First Time

The default usernames/passwords configured for the device are:

root / toorkali / kali

After logging in for the first time, we recommend the following steps:

- Open terminal, change passwords, and run ‘sudo dpkg-reconfigure locales’ to adjust the settings according to your region
- LXQT Regionals: Applications -> Preferences -> LXQT settings -> Locale
- Default Applications: Applications -> Preferences -> LXQT settings -> Session Settings
- Set “Turn off monitor(s) when lid is closed”: Applications -> Preferences -> LXQT settings -> Power Management
- Wifi Setup: Applications -> Usual applications -> Internet -> Conman UI Setup

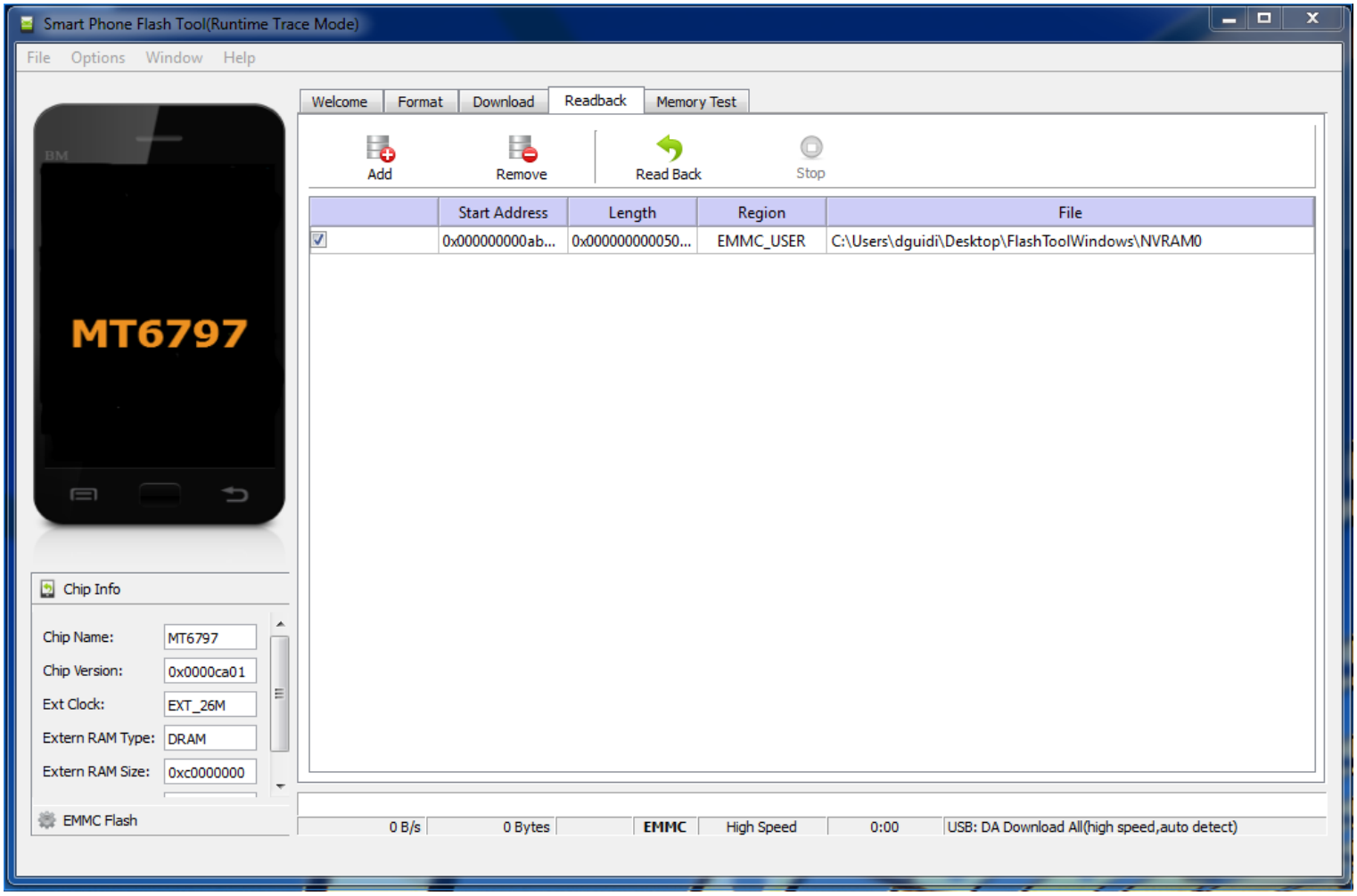
- Run ‘bluetoothctl’ on the command line to setup a mouse
- This image comes with Kali Linux Top10 pre-installed, run ‘apt update && apt install kali-linux-full’

Finish Android Rooting Process

Run the pre-installed “Magisk Manager” to complete the rooting process.

Backing up the Kali Linux root Partition

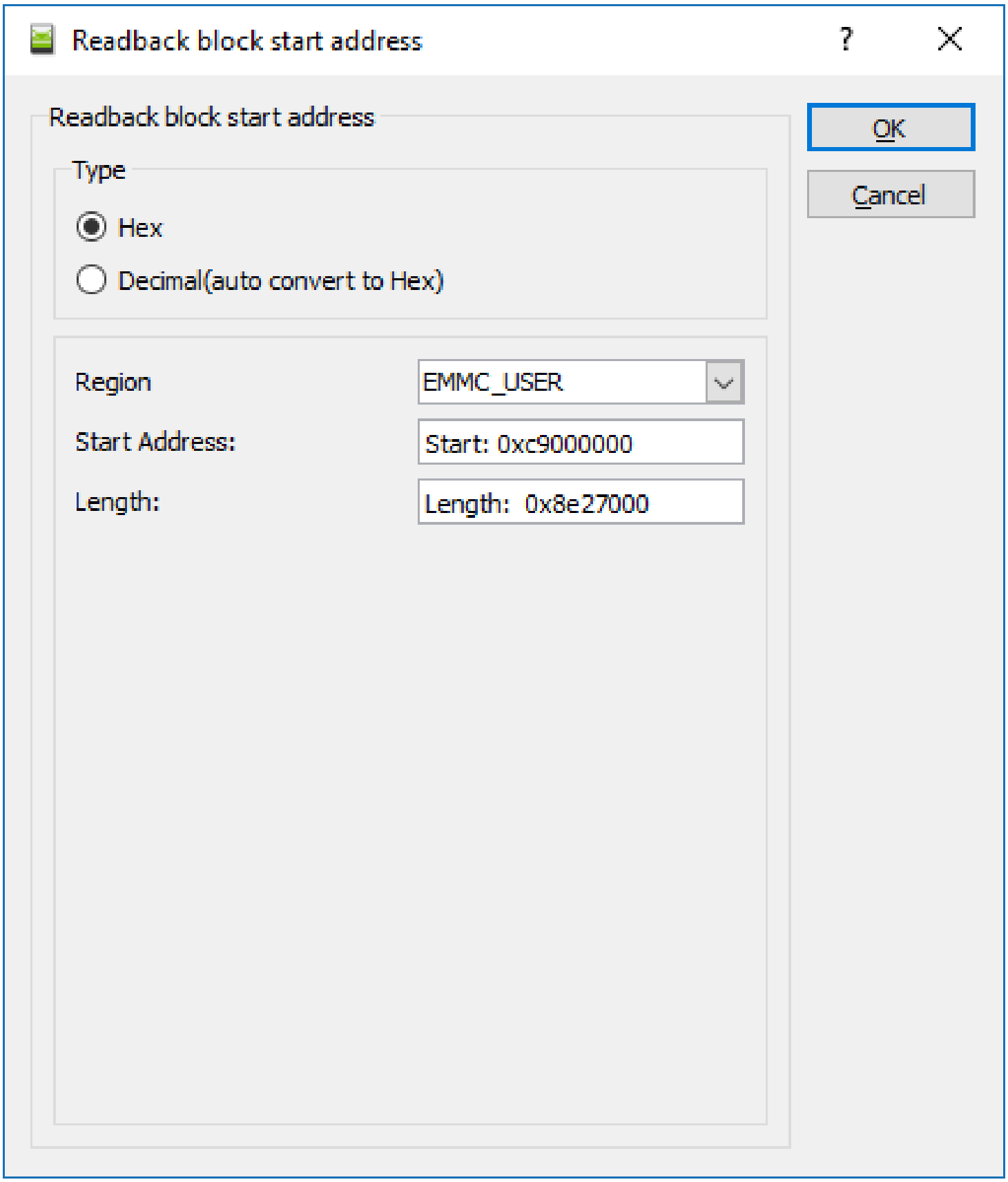
To create a backup of your rootfs partition, click on the “Readback” tab in the flash tool and then on the “Add” button. A row will appear in the table as in the following screenshot:



Double-click on the file name and enter the name and location of the resulting image file. In the next screen, change the start address and length to the following values:

- Start Address: 0xc9000000
- Length: 0x8e2700000

It should look like this screenshot:



Click ”OK” and then click “Readback”. Connect the Gemini and turn it on by pressing “OK”.

That’s All

Please keep in touch by joining us in the Kali Forums.

Information:

Kali: www.kali.org
Gemini: https://geminiplanet.com
Planet Computers: planetcom.co.uk
Planet Computers Developers Forum: https://developer.planetcom.co.uk/forumdisplay.php?fid=1
Gemian: http://gemian.thinkglobally.org
Gemian Wiki: https://github.com/gemian/gemini-keyboard-apps/wiki
OESF Forum: https://www.oesf.org/forum
IRC Logs: http://logs.nslu2-linux.org/livelogs/gemini-pda/
Halium: https://halium.org

Guides:

Linux Flashing Guide
Android Flashing Guide

Downloads:

Windows Flash Tool Drivers
SP-Flashing Tool for Windows

Kali Linux – ASUS Chromebook Flip

The ASUS Chromebook Flip is a quad core 1.8GHz, with 2GB or 4GB of RAM Chromebook with a 10.1″ 10 point multitouch touchscreen. Kali Linux fits on an external micro SD card or USB key.

Kali on ASUS Chromebook Flip – User Instructions

If all you want to do is install Kali on your ASUS Chromebook Flip, follow these instructions:

1. Get a nice fast 8 GB micro SD card or USB key.
2. Put your Chromebook in developer mode, and enable USB boot. You can ignore legacy boot on that page since these devices do not have SeaBIOS.
3. Download the Kali ASUS Chromebook Flip image from our downloads area.
4. Use the **dd** utility to image this file to your microSD card or USB key. In our example, we use a microSD which is located at */dev/sdb*. ***Change this as needed.***

Alert! This process will wipe out your SD card/USB key. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-veyron.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the ASUS Chromebook Flip with the microSD/USB key plugged in. Log in to Kali (***root*** / ***toor***), that's it, you're done!

Kali on ASUS Chromebook Flip – Developer Instructions

If you are a developer and want to tinker with the Kali ASUS Chromebook Flip image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **chromebook-arm-veyron.sh**

Kali Linux – MiniX

The Mini-X is a dual core 1GHz, with 1GB of RAM. Kali Linux fits on an external micro SD card.

Kali on Mini-X – User Instructions

If all you want to do is install Kali on your Mini-X, follow these instructions:

1. Get a nice fast 8 GB micro SD card.
2. Download the Kali Mini-X image from our downloads area.
3. Use the **dd** utility to image this file to your microSD card. In our example, we use a microSD which is located at ***/dev/sdb***. ***Change this as needed.***

Alert! This process will wipe out your SD card. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-mini-x.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the Mini-X with the microSD plugged in. Log in to Kali (***root*** / ***toor***), that's it, you're done!

Kali on Mini-X – Developer Instructions

If you are a developer and want to tinker with the Kali Mini-X image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **mini-x.sh**

Kali Linux – Cubietruck

The Cubietruck is a dual core 1GHz, with 2GB of RAM. Kali Linux fits on an external micro SD card.

Kali on Cubietruck – User Instructions

If all you want to do is install Kali on your Cubietruck, follow these instructions:

1. Get a nice fast 8 GB micro SD card.
2. Download the Kali Cubietruck image from our downloads area.
3. Use the **dd** utility to image this file to your microSD card. In our example, we use a microSD which is located at ***/dev/sdb***. ***Change this as needed.***

Alert! This process will wipe out your SD card. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-cubietruck.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the Cubietruck with the microSD plugged in. Log in to Kali (***root*** / ***toor***), that's it, you're done!

Kali on Cubietruck – Developer Instructions

If you are a developer and want to tinker with the Kali Cubietruck image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **cubietruck.sh**

Kali Linux – Raspberry Pi2

The Raspberry Pi2 is a quad core 900MHz, with 1GB of RAM. Kali Linux fits on an external micro SD card.

Kali on Raspberry Pi2 – User Instructions

If all you want to do is install Kali on your Raspberry Pi2, follow these instructions:

1. Get a nice fast 8 GB micro SD card or eMMC.
2. Download the Kali Raspberry Pi2 image from our downloads area.
3. Use the **dd** utility to image this file to your microSD card. In our example, we use a microSD which is located at ***/dev/sdb***. ***Change this as needed.***

This process will wipe out your SD card. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-rpi2.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the Raspberry Pi2 with the microSD plugged in. Log in to Kali (***root*** / ***toor***), that's it, you're done!

Kali on Raspberry Pi2 – Developer Instructions

If you are a developer and want to tinker with the Kali Raspberry Pi2 image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **rpi2.sh**

Kali Linux – Trimslice

The Trimslice is a dual core 1GHz, with 1GB of RAM. Kali Linux fits on an external micro SD card.

Kali on Trimslice – User Instructions

If all you want to do is install Kali on your Trimslice, follow these instructions:

1. Get a nice fast 8 GB micro SD card or eMMC.
2. Download the Kali Trimslice image from our downloads area.
3. Use the **dd** utility to image this file to your microSD card. In our example, we use a microSD which is located at ***/dev/sdb***. ***Change this as needed.***

This process will wipe out your SD card. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-trimslice.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the Trimslice with the microSD plugged in. Log in to Kali (***toor*** / ***toor***), that's it, you're done!

Kali on Trimslice – Developer Instructions

If you are a developer and want to tinker with the Kali Trimslice image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **trimslice.sh**

Kali Linux – Cubieboard2

The Cubieboard2 is a dual core 1.4GHz, with 1GB of RAM. Kali Linux fits on an external micro SD card.

Kali on Cubieboard2 – User Instructions

If all you want to do is install Kali on your Cubieboard2, follow these instructions:

1. Get a nice fast 8 GB micro SD card or eMMC.
2. Download the Kali Cubieboard2 image from our downloads area.
3. Use the **dd** utility to image this file to your microSD card. In our example, we use a microSD, which is located at ***/dev/sdb***. ***Change this as needed.***

This process will wipe out your SD card. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-cubieboard2.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the Cubieboard2 with the microSD plugged in. Log in to Kali (***root*** / ***toor***), that's it, you're done!

Kali on Cubieboard2 – Developer Instructions

If you are a developer and want to tinker with the Kali Cubieboard2 image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **cubieboard2.sh**

Kali Linux – RIoTboard

The RIoTboard is a Cortex A9 1GHz, with 1GB of RAM. Kali Linux fits on an external micro SD card.

Kali on RIoTboard – User Instructions

If all you want to do is install Kali on your RIoTboard, follow these instructions:

1. Get a nice fast 8 GB micro SD card or eMMC.
2. Download the Kali RIoTboard image from our downloads area.
3. Use the **dd** utility to image this file to your microSD card. In our example, we use a microSD which is located at ***/dev/sdb***. ***Change this as needed.***

This process will wipe out your SD card. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-riot.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the RIoTboard with the microSD plugged in. Log in to Kali (***toot*** / ***toor***), that's it, you're done!

Kali on RIoTboard – Developer Instructions

If you are a developer and want to tinker with the Kali RIoTboard image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **riot.sh**

Kali Linux – NanoPi2

The NanoPi2 is a quad core 1.9GHz, with 1GB of RAM. Kali Linux fits on an external micro SD card.

Kali on NanoPi2 – User Instructions

If all you want to do is install Kali on your NanoPi2, follow these instructions:

1. Get a nice fast 8 GB micro SD card or eMMC.
2. Download the Kali NanoPi2 image from our downloads area.
3. Use the **dd** utility to image this file to your microSD card. In our example, we use a microSD which is located at ***/dev/sdb***. ***Change this as needed.***

This process will wipe out your SD card. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-nanopi2.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the NanoPi2 with the microSD plugged in. Log in to Kali (***root*** / ***toor***), that's it, you're done!

Kali on NanoPi2 – Developer Instructions

If you are a developer and want to tinker with the Kali NanoPi2 image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **nanopi2.sh**

Kali Linux - Utilite Pro

The Utilite Pro is a quad core 1.2GHz Cortex A9, with 2GB of RAM. Kali Linux fits on an external micro SD card.

Kali on Utilite Pro – User Instructions

If all you want to do is install Kali on your Utilite Pro, follow these instructions:

1. Get a nice fast 8 GB micro SD card or eMMC.
2. Download the Kali Utilite image from our downloads area.
3. Use the **dd** utility to image this file to your microSD card. In our example, we use a microSD which is located at ***/dev/sdb***. ***Change this as needed.***

Alert! This process will wipe out your SD card. If you choose the wrong storage device, you may wipe out your computers hard disk.

```
xzcat kali-$version-utilite.img.xz | dd of=/dev/sdb bs=512k
```

This process can take awhile depending on your device speed and image size.

Once the *dd* operation is complete, boot up the Utilite Pro with the microSD plugged in. Log in to Kali (***toor*** / ***toor***), that's it, you're done!

Kali on Utilite – Developer Instructions

If you are a developer and want to tinker with the Kali Utilite Pro image, including changing the kernel configuration and generally being adventurous, check out the kali-arm-build-scripts repository on GitHub, and follow the *README.md* file's instructions. The script to use is **utilite.sh**

05. Using Kali Linux

Kali on ARM - A bit of history

When BackTrack ARM first came out, it was one image, for a Motorola Xoom. The work was done on the Xoom itself by muts. He started from an ubuntu image for it, built all of the packages for Backtrack on it, then installed them. I then took the work and expanded it to support 3 or 4 different ARM devices I had, following a similar proceedure. I showed muts the work I'd done and he was as excited about it as I was.

When Kali came about, we retooled everything, including build servers for armel, armhf, and arm64. No more building packages manually on the ARM devices themselves. So everything was in place, but the images for ARM devices were still being built manually. Putting out an updated image meant downloading the last release, writing it to an sdcard, booting the device, running updates, building the kernel, installing the new kernel, cleaning up the logs and apt cache, then powering the system off, plugging the sdcard back into my other system, and creating a dd image of the sdcard, putting it on to a server. This was very error prone due to the nature of sd cards from different manufacturers having different actual sizes.

We wanted to make it so anyone could, starting from a Kali amd64 installation, build an image that would work on any of our supported ARM devices, end up with exactly what we put out, and most importantly, customize it for their needs. So I created the kali-arm-build-scripts – they aren't fancy, but they're easy to read, follow and modify.

Configuring Yubikeys for SSH Authentication

This document explains how to configure a Yubikey for SSH authentication

Prerequisites

Install Yubikey Personalization Tool and Smart Card Daemon

```
apt install -y yubikey-personalization scdaemon
```

Detect Yubikey

First, you'll need to ensure that your system is fully upgraded.

```
root@kali:~# pcsc_scan
Scanning present readers...
Reader 0: Yubico Yubikey 4 OTP+U2F+CCID 00 00
  Card state: Card inserted,
Possibly identified card (using /usr/share/pcsc/smartcard_list.txt):
    Yubico Yubikey 4 OTP+CCID
```

Configuration

In order for our Yubikey to be detected as a smart card, we'll need to set our Yubikey to CCID mode.

```
root@kali:~# ykpersonalize -m 86
The USB mode will be set to: 0x86

Commit? (y/n) [n]: y
```

After this modification, GPG should now be able to recognize our Yubikey as a smart card.

```
root@kali:~# gpg --card-status
Reader ...........: Yubico Yubikey 4 OTP U2F CCID 00 00
Version ..........: 2.1
Manufacturer .....: Yubico
Key attributes ...: rsa2048 rsa2048 rsa2048
Max. PIN lengths .: 127 127 127
PIN retry counter : 3 0 3
```

Now we will need to change the default PIN that is configured.
Note: The default PIN is 123456 and default admin PIN is 12345678.

```
root@kali:~# gpg --change-pin
gpg: OpenPGP card no. F84822122020100060415878500000 detected

1 - change PIN
2 - unblock PIN
3 - change Admin PIN
4 - set the Reset Code
Q - quit

Your selection? 1 # Enter a new PIN
PIN changed.

1 - change PIN

Your selection? 3 # Enter a new admin PIN
PIN changed.

Your selection? q
```

Kali in the Browser

In certain environments, such as a headless installation, or AMAZON EC2 / Azure instances, we are often limited to SSH access. However, there might be times when we need to use a graphical application in Kali, or we just want access to the Kali GUI – this is where a noVNC stack becomes useful.

Kali sources.list Repositories

The single most common causes of a broken Kali Linux installation are following unofficial advice, and particularly arbitrarily populating the system's **sources.list** file with unofficial repositories. The following post aims to clarify what repositories should exist in **sources.list**, and when they should be used.

Any additional repositories added to the Kali sources.list file will most likely *BREAK YOUR KALI LINUX INSTALL*.

Regular repositories

On a standard, clean install of Kali Linux, you should have the following entry present in **/etc/apt/sources.list**:

```
deb http://http.kali.org/kali kali-rolling main non-free contrib
```

You can find a list of official Kali Linux mirrors here.

Source repositories

In case you require source packages, you might also want to add the following repositories as well:

```
deb-src http://http.kali.org/kali kali-rolling main non-free contrib
```

The kali-dev repository

WARNING: While kali-dev is publicly accessible to everybody on all Kali mirrors, this distribution should not be used by end-users as it will regularly break.

This repository is actually Debian's Testing distribution with all the kali-specific packages (available in the kali-dev-only repository) force-injected. Kali packages take precedence over the Debian packages. Sometimes when Testing changes, some Kali packages must be updated and this will not happen immediately. During this time, kali-dev is likely to be broken. This repository is where Kali developers push updated packages and is the basis

used to create kali-rolling.

The kali-rolling repository

Contrary to kali-dev, kali-rolling is expected to be of better quality because it's managed by a tool that ensures installability of all the package it contains. That tool picks updated packages from kali-dev and copies them to kali-rolling only when they have been verified to be installable. Note however that those checks do not include any functional testing. It might still contain broken software due to other problems that are not covered by the package dependencies. **Kali Rolling is the primary repository that most users should be using.** They can also report any issue they have with Kali specific packages on bugs.kali.org. Make sure to select the "kali-dev" version in "Product version".

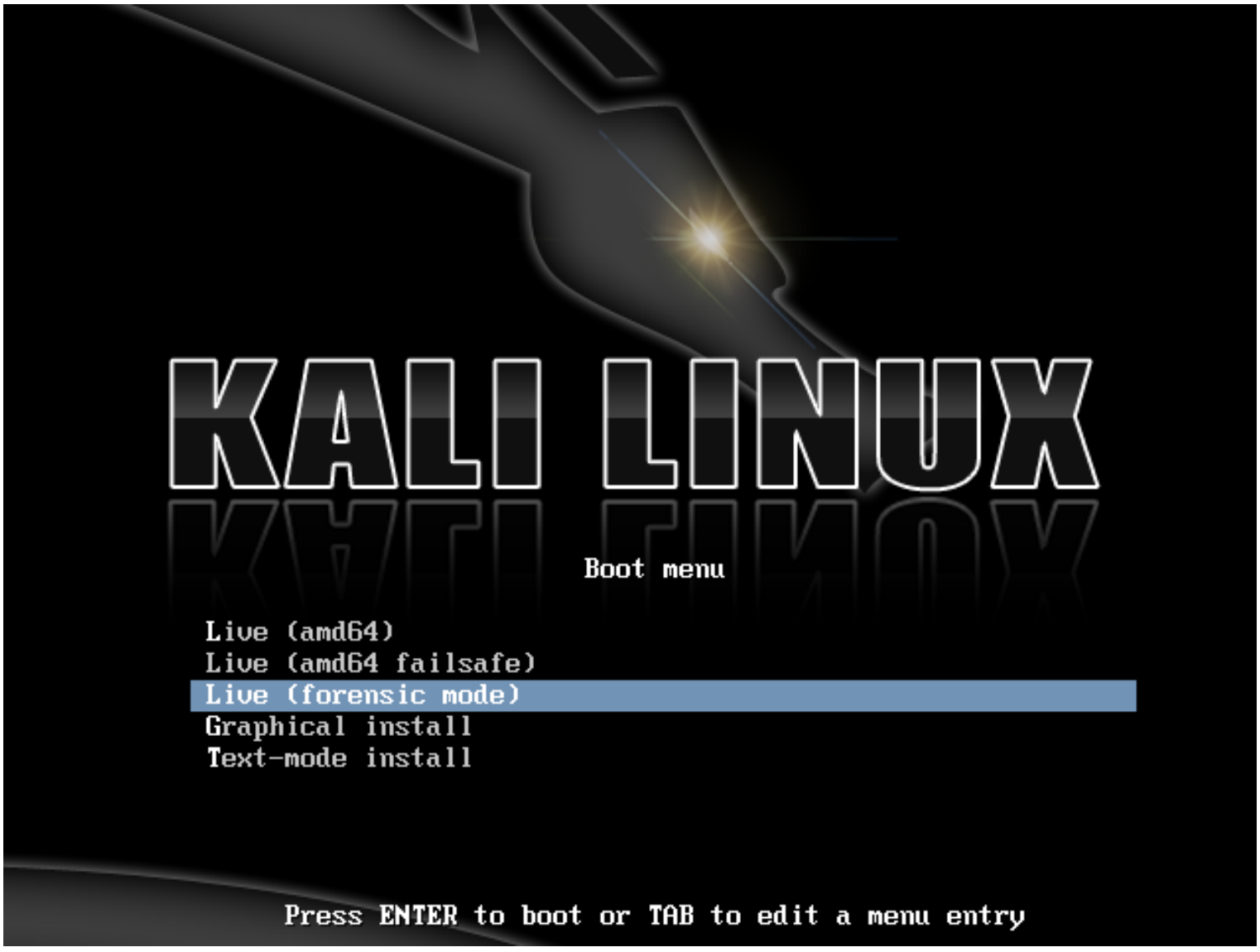
Kali Rolling users are expected to have the following entries in their sources.list:

```
deb http://http.kali.org/kali kali-rolling main non-free contrib
```

Kali Linux Forensics Mode

Kali Linux “Live” provides a “forensic mode”, a feature first introduced in BackTrack Linux. The “Forensic mode live boot” option has proven to be very popular for several reasons:

- Kali Linux is widely and easily available, many potential users already have Kali ISOs or bootable USB drives.
- When a forensic need comes up, Kali Linux “Live” makes it quick and easy to put Kali Linux on the job.
- Kali Linux comes pre-loaded with the most popular open source forensic software, a handy toolkit when you need to do forensic work.



When booted into the forensic boot mode, there are a few *very important changes* to the regular operation of the system:

1. First, the internal hard disk is *never* touched. If there is a swap partition it will *not* be used and no

internal disk will be auto mounted. We verified this by first taking a standard system and removing the hard drive. A hash was taken of the drive using a commercial forensic package. We then reattached the drive to the computer and booted Kali Linux “Live” in forensic mode. After using Kali for a period of time, we then shut the system down, removed the hard drive, and took the hash again. These hashes matched, indicating that at no point was anything changed on the drive in any way.

2. The other, equally important, change is that auto-mounting of removable media is ***disabled***. USB thumb drives, CDs, and the like will ***not*** be auto-mounted when inserted. The idea behind this is simple: in forensic mode, ***nothing*** should happen to ***any*** media without ***direct user action***. *Anything* that you do as a user is *on you*.

If you plan on using Kali for real world forensics of any type, we recommend that you don’t just take our word for any of this. All forensic tools should *always* be validated to ensure that you know how they will behave in any circumstance in which you are going to be using them.

Finally, while Kali continues to focus on providing the best collection of open source penetration testing tools available, it is always possible that we may have missed *your* favorite open source forensic tool. If so, let us know! We are always on the lookout of high quality open source tools that we can add to Kali to make it even better.

Metasploit Framework

In keeping with the Kali Linux Network Services Policy, no network services, *including* database services, run on boot as a default, so there are a couple of steps that need to be taken in order to get Metasploit up and running with database support.

Start the Kali PostgreSQL Service

Metasploit uses PostgreSQL as its database so it needs to be launched first.

```
service postgresql start
```

You can verify that **PostgreSQL** is running by checking the output of **ss -ant** and making sure that port 5432 is listening.

```
State Recv-Q Send-Q Local Address:Port Peer Address:Port
LISTEN 0 128 :::22 :::*
LISTEN 0 128 *:22 *:*
LISTEN 0 128 127.0.0.1:5432 *:*
LISTEN 0 128 ::1:5432 :::*
```

Initialise the Metasploit PostgreSQL Database

With **PostgreSQL** up and running, we next need to create and initialize the **msf** database.

```
msfdb init
```

Launch msfconsole in Kali

Now that the **PostgreSQL** service is up and running and the database is initialized, you can launch **msfconsole** and verify database connectivity with the **db_status** command as shown below.

```
msfconsole
```

```
msf > db_status
[*] postgresql connected to msf3
msf >
```

Official Kali Linux Documentation eBook

We have created an offline version of the official Kali Linux documentation (this site), in a PDF format in your chosen language for easy viewing.

These eBooks will be updated when any new content has been added to the site.

All the translations have been done to the best of our abilities; however if you are able to make it more accurate, please get in touch.

eBook Download

-  English – https://docs.kali.org/pdf/kali-book-en.pdf
-  □□□□ – https://docs.kali.org/pdf/kali-book-zh-hans.pdf
-  Français – https://docs.kali.org/pdf/kali-book-fr.pdf
-  Deutsch – https://docs.kali.org/pdf/kali-book-de.pdf
-  – العربية https://docs.kali.org/pdf/kali-book-ar.pdf
-  Português – https://docs.kali.org/pdf/kali-book-pt-br.pdf
-  Nederlands – https://docs.kali.org/pdf/kali-book-nl.pdf
-  Italiano – https://docs.kali.org/pdf/kali-book-it.pdf
-  □□□ – https://docs.kali.org/pdf/kali-book-ja.pdf
-  Русский – https://docs.kali.org/pdf/kali-book-ru.pdf
-  Español – https://docs.kali.org/pdf/kali-book-es.pdf
-  Indonesia – https://docs.kali.org/pdf/kali-book-id.pdf

Last updated: 2013-Dec-06

Install NVIDIA GPU Drivers on Kali Linux

This document explains how to install NVIDIA GPU drivers and CUDA support, allowing integration with popular penetration testing tools.

Prerequisites

First, you'll need to ensure that your system is fully upgraded and that your card supports CUDA.
Note: GPUs with a CUDA compute capability > 5.0 are recommended, but GPUs with less will still work.

```
apt update && apt dist-upgrade -y && reboot
```

Let's determine the exact GPU installed, and check the kernel modules it's using.

```
root@kali:~# lspci -v
01:00.0 VGA compatible controller: NVIDIA Corporation GM204 [GeForce GTX 970] (rev a1) (prog-if 00 [VGA controller])
    Subsystem: ZOTAC International (MCO) Ltd. GM204 [GeForce GTX 970]
    Region 1: Memory at e0000000 (64-bit, prefetchable) [size=256M]
    Capabilities: [60] Power Management version 3
    Capabilities: [68] MSI: Enable+ Count=1/1 Maskable- 64bit+
    Capabilities: [78] Express (v2) Legacy Endpoint, MSI 00
    Capabilities: [600 v1] Vendor Specific Information: ID=0001 Rev=1 Len=024
    Kernel driver in use: nouveau
    Kernel modules: nouveau
```

Installation

Once the system has rebooted, we will proceed to install the **OpenCL ICD Loader**, **Drivers**, and the **CUDA toolkit**.

```
apt install -y ocl-icd-libopencl1 nvidia-driver nvidia-cuda-toolkit
```

During installation of the drivers the system created new kernel modules, so another reboot is required.

Verify Driver Installation

Now that our system should be ready to go, we need to verify the drivers have been loaded correctly. We can quickly verify this by running the nvidia-smi tool.

```
root@kali:~# nvidia-smi
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 375.26                 Driver Version: 375.26                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce GTX 970     Off  | 0000:01:00.0      On |                  N/A |
| 36%   46C    P0    47W / 325W |    200MiB /  4036MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID  Type  Process name                               Usage      |
|=============================================================================|
|    0       692    G   /usr/lib/xorg/Xorg                           198MiB |
+-----------------------------------------------------------------------------+
```

With the output displaying our driver and GPU correctly, we can now dive into benchmarking. Before we get too far ahead, let’s double check to make sure hashcat and CUDA are working together.

```
root@kali:~# hashcat -I
OpenCL Info:
```

```
Platform ID #1
  Vendor  : NVIDIA Corporation
  Name    : NVIDIA CUDA
  Version : OpenCL 1.2 CUDA 8.0.0
  Device ID #1
    Type          : GPU
    Vendor ID     : 32
    Vendor        : NVIDIA Corporation
    Name          : GeForce GTX 970
    Version       : OpenCL 1.2 CUDA
    Processor(s)  : 13
    Clock         : 1240
    Memory        : 1009/4036 MB allocatable
    OpenCL Version : OpenCL C 1.2
    Driver Version : 375.26
```

It appears everything is working, let’s go ahead and run a benchmark test.

Benchmarking

```
root@kali:~# hashcat -b
OpenCL Platform #1: NVIDIA Corporation
======================================
* Device #1: Geforce GTX 970, 1009/4095 MB allocatable, 13MCU

Hashtype: MD5
Speed.Dev.#1.....: 10443.1 MH/s
Hashtype: SHA1
Speed.Dev.#1.....: 3349.8 MH/s
Hashtype: SHA256
Speed.Dev.#1.....: 1321.8 MH/s
```

There are a multitude of configurations to improve cracking speed, not mentioned in this guide. However, we encourage you to take a look at the hashcat documentation for your specific cases.

Troubleshooting

In the event setup isn't going as planned, we'll install clinfo for detailed troubleshooting information.

```
apt install -y clinfo
```

OpenCL Loaders

It may be necessary to check for additional packages that may be conflicting with our setup. Let's first check to see what **OpenCL Loader** we have installed. The NVIDIA OpenCL Loader and the generic OpenCL Loader will both work for our system.

```
root@kali:~# dpkg -l |grep -i icd
ii  nvidia-egl-icd:amd64                375.26-2                amd64        NVIDIA EGL installable client
driver (ICD)
ii  nvidia-opencl-icd:amd64              375.26-2                amd64        NVIDIA OpenCL installable
client driver (ICD)
ii  nvidia-vulkan-icd:amd64              375.26-2                amd64        NVIDIA Vulkan installable
client driver (ICD)
ii  ocl-icd-libopencl1:amd64             2.2.11-1                amd64        Generic OpenCL ICD Loader
```

If **mesa-opencl-icd** is installed run:

```
apt remove mesa-opencl-icd
```

Since we have determined that we have a compatible ICD loader installed, we can easily determine which loader is currently being used.

```
root@kali:~# clinfo | grep -i "icd loader"
```

```
ICD loader properties
  ICD loader Name                 OpenCL ICD Loader
  ICD loader Vendor               OCL Icd free software
  ICD loader Version              2.2.11
  ICD loader Profile              OpenCL 2.1
```

As expected, our setup is using the open source loader that was installed earlier. Now, let’s get some detailed information about the system.

Querying GPU Information

We’ll use nvidia-smi once again, but with a much more verbose output.

```
root@kali:~# nvidia-smi -i 0 -q
Driver Version                  : 375.26
Attached GPUs                   : 1
GPU 0000:01:00.0
    Product Name                : GeForce GTX 970
    Product Brand               : GeForce
    Display Mode                : Enabled
    Display Active              : Enabled
    Persistence Mode            : Disabled
    Accounting Mode             : Disabled
    Accounting Mode Buffer Size     : 1920
Temperature
        GPU Current Temp        : 47 C
        GPU Shutdown Temp       : 96 C
        GPU Slowdown Temp       : 91 C
    Clocks
        Graphics                : 1101 MHz
        SM                      : 1101 MHz
        Memory                  : 3523 MHz
        Video                   : 1012 MHz
    Processes
        Process ID              : 692
            Type                : G
            Name                : /usr/lib/xorg/Xorg
```

```
        Used GPU Memory       : 198 MiB
```

It looks like our GPU is being recognized correctly, so let's use glxinfo to determine if 3D Rendering is enabled.

```
root@kali:~# glxinfo | grep -i "direct rendering"
direct rendering: Yes
```

The combination of these tools should assist the troubleshooting process greatly. If you still experience issues, we recommend searching for similar setups and any nuances that may affect your specific system.

Kali Linux VirtualBox Guest

If you run Kali Linux as a “guest” within VirtualBox, this article will help you to successfully install the “Guest Addition” tools.

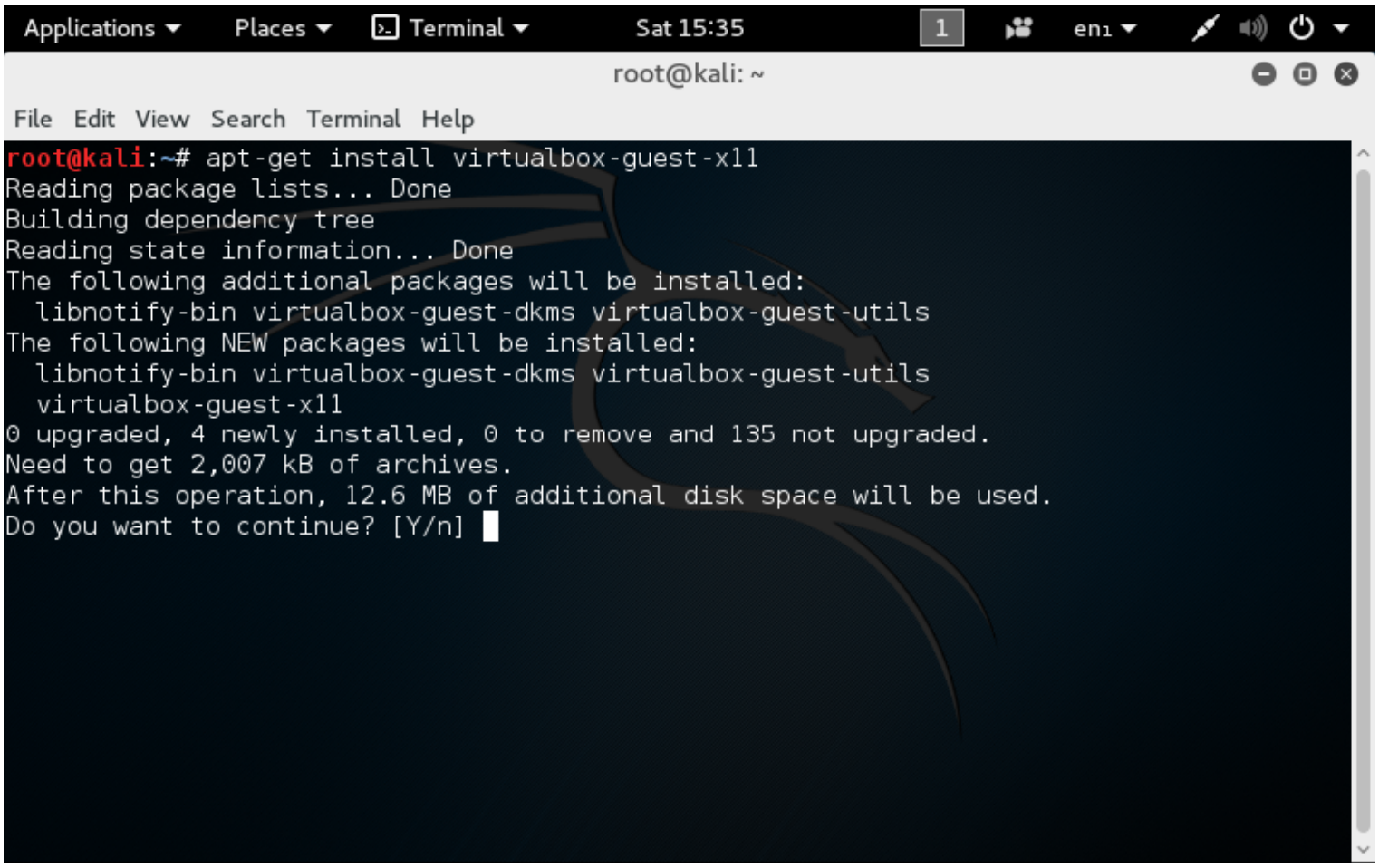
You must use version **4.2.xx or higher** of VirtualBox in order to take advantage of the improvements, including compatibility updates, and enhanced stability of both the core application and the Guest Additions.

Installing VirtualBox Guest Additions in Kali Linux

The VirtualBox Guest Additions provide proper mouse and screen integration, as well as folder sharing, with your host operating system. To install them, proceed as follows.

Start up your Kali Linux virtual machine, open a terminal window and issue the following commands.

```
apt-get update
apt-get install -y virtualbox-guest-x11
reboot
```

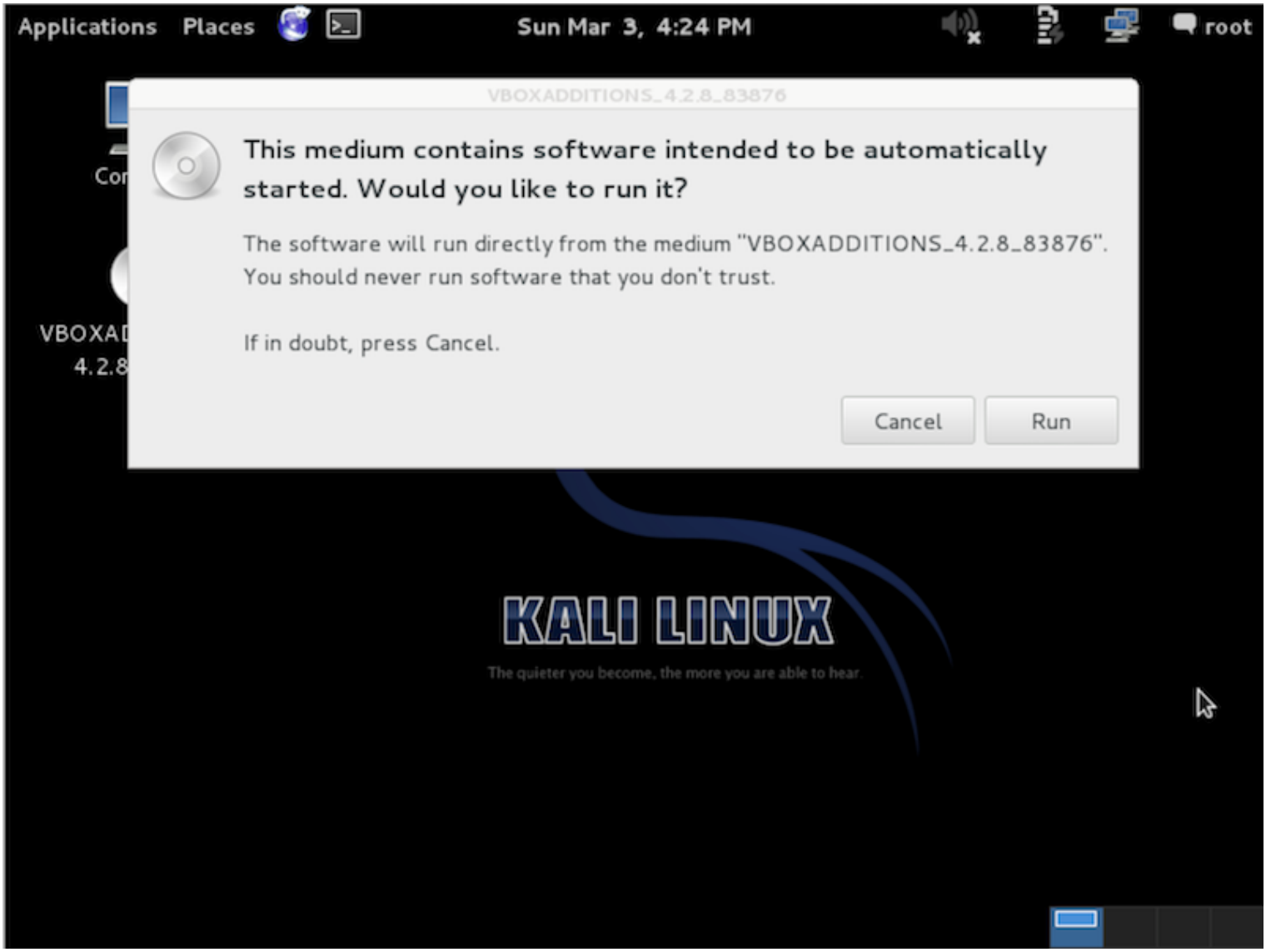


Installing VirtualBox Guest Additions in Older Kali Versions

Start up your Kali Linux virtual machine, open a terminal window and issue the following command to install the Linux kernel headers.

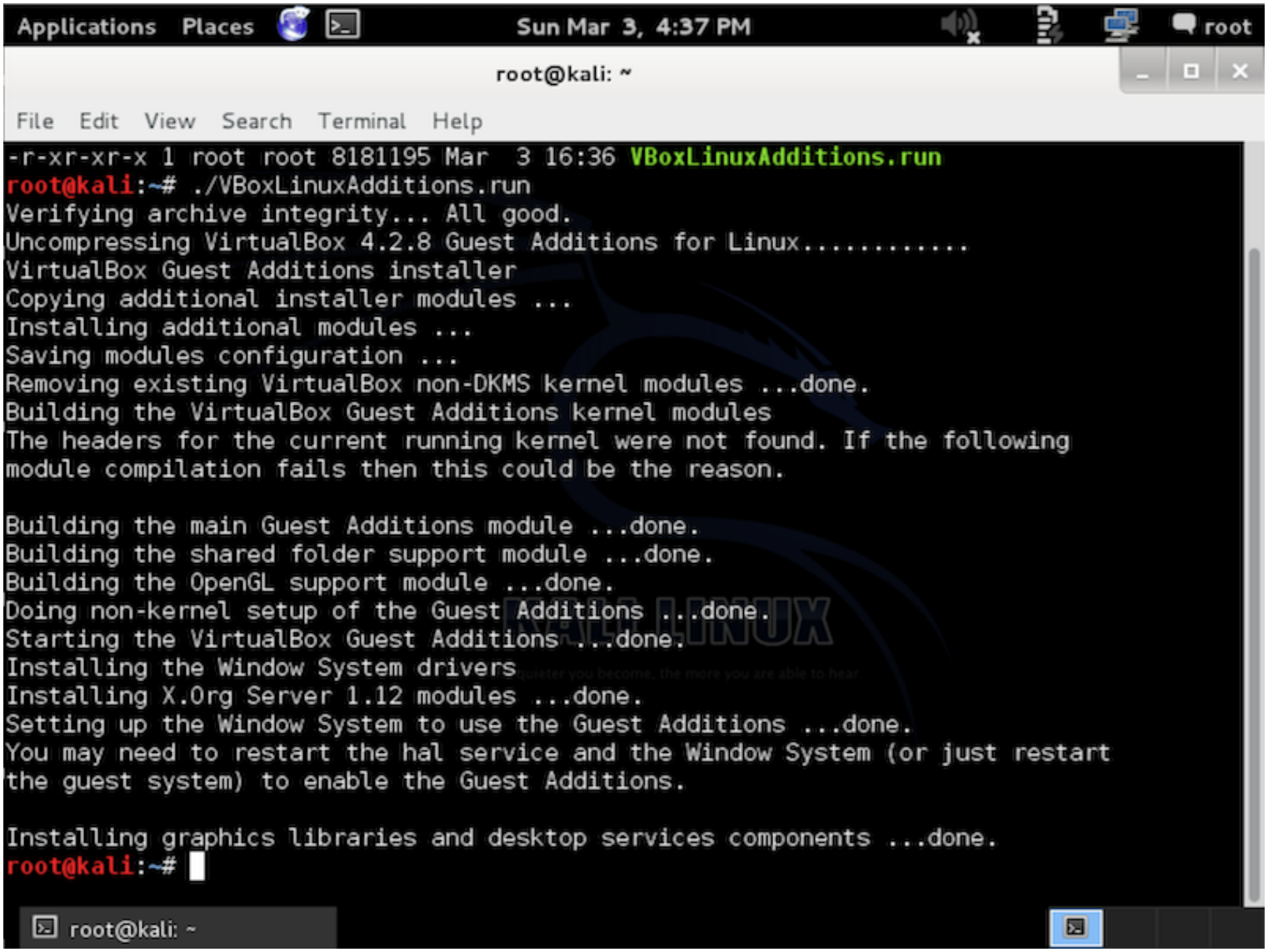
```
apt-get update && apt-get install -y linux-headers-$(uname -r)
```

Once this is complete you can now attach the “Guest Additions” CD-ROM image. Select “Devices” from the VirtualBox menu and then select “Install Guest Additions”. This will mount the Guest Additions ISO in the virtual CD drive in your Kali Linux virtual machine. When prompted to autorun the CD, click the Cancel button.



From a terminal window, copy the VboxLinuxAdditions.run file from the Guest Additions CD-ROM to a path on your local system. Ensure it is executable and run the file to begin the installation.

```
cp /media/cd-rom/VBoxLinuxAdditions.run /root/
chmod 755 /root/VBoxLinuxAdditions.run
cd /root
./VBoxLinuxAdditions.run
```



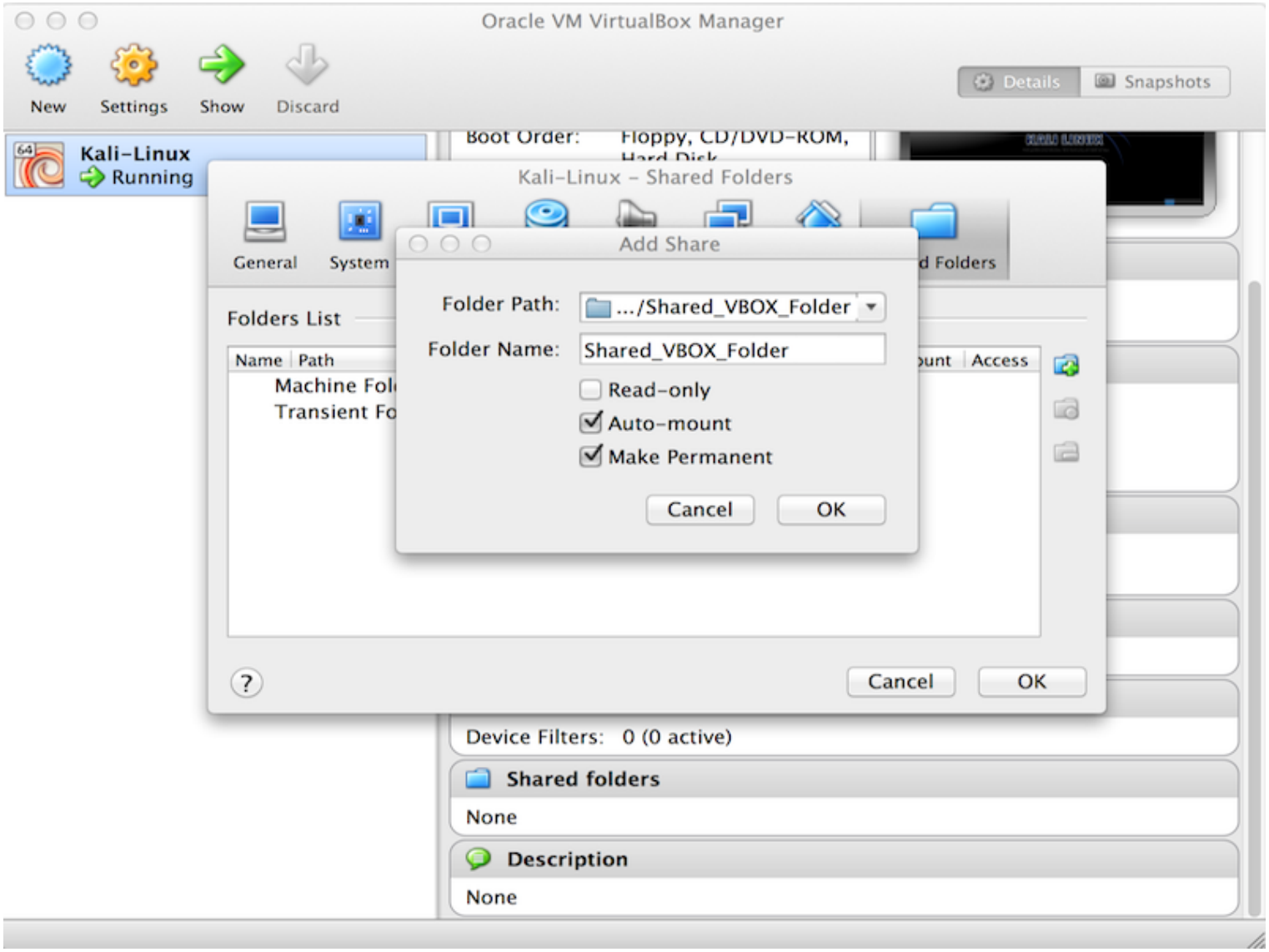
Reboot the Kali Linux VM to complete the Guest Additions installation. You should now have full mouse and screen integration as well as the ability to share folders with the host system.

Creating Shared Folders with the Host System

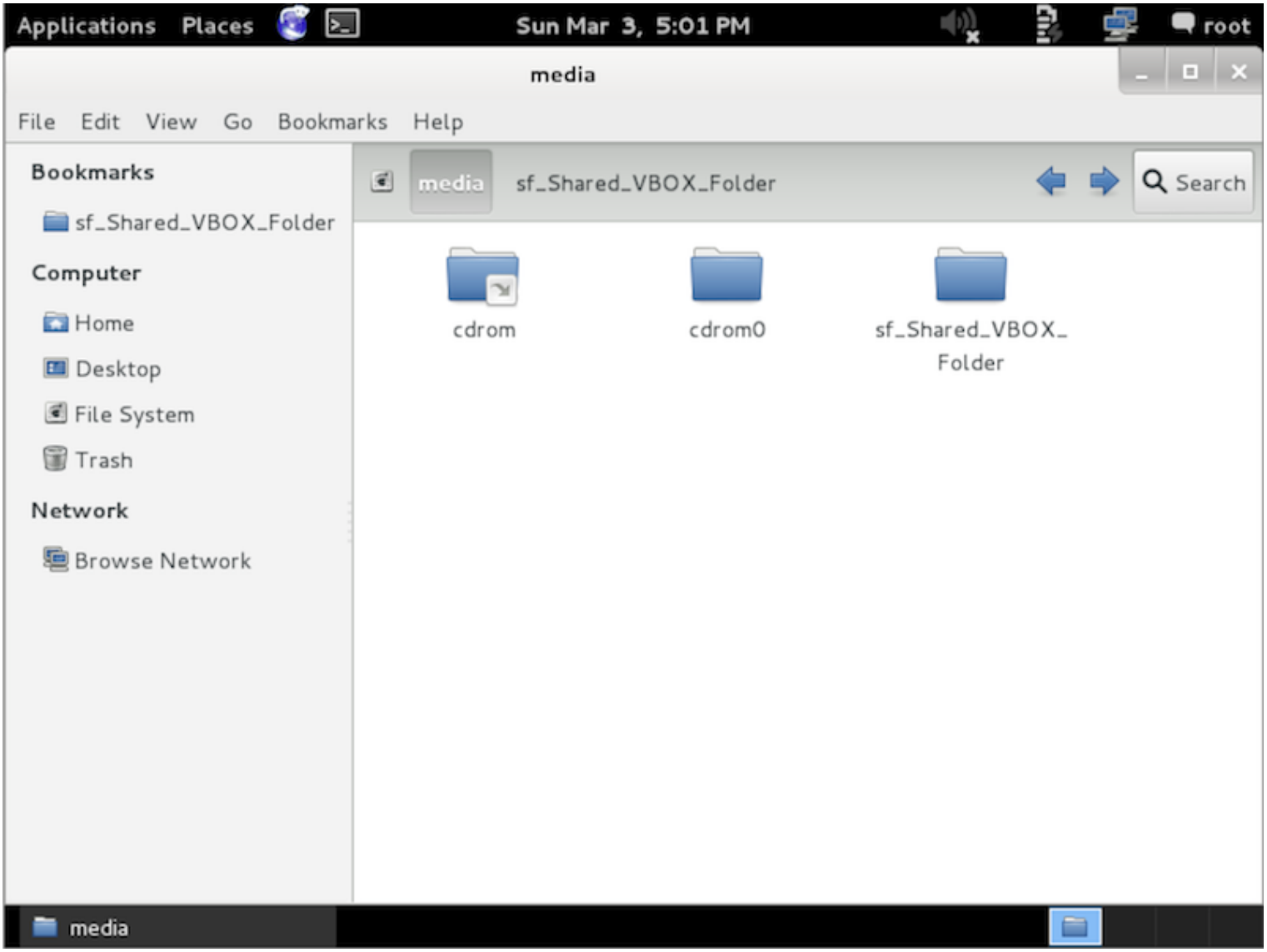
This section explains how to share folders on your host system with your Kali Linux VirtualBox “guest”.

From the VirtualBox Manager, select your Kali Linux VM instance and click on the “Shared Folders” link in the right window pane. This will launch a pop up window for adding shared folders. Within this window click the “Add Folders” icon.

In the Folder Path text box, provide the path to the folder you would like to share, or click the drop-down arrow to browse your host system for the path to the folder. Select the check boxes that allow for ‘Auto-mount’ and ‘Make Permanent’ and click the “OK” button both times when prompted.



Your shared folders will now be available in the media directory. You can create a bookmark or link for easier access to the directory.


Applications
Places
Sun Mar 3, 5:01 PM
root
media
File
Edit
View
Go
Bookmarks
Help
Bookmarks
sf_Shared_VBOX_Folder
Computer
Home
Desktop
File System
Trash
Network
Browse Network
media
sf_Shared_VBOX_Folder
Search
cdrom
cdrom0
sf_Shared_VBOX_
Folder
media

VMware Tools in a Kali Guest

Should you decide to create your own VMware installation of Kali Linux rather than using our pre-made VMware images, you will need to follow the instructions below in order to successfully install VMware Tools in your Kali installation.

Installing VMware Tools in Kali Linux Rolling

As of Sept 2015, **VMware** recommends **using the distribution-specific open-vm-tools (OVT)** instead of the VMware Tools package for guest machines. To install open-vm-tools in Kali, first make sure you are fully updated, and then enter the following:

```
apt update && apt -y full-upgrade

# Reboot now in case you have updated to a new kernel. Once rebooted:
apt -y --reinstall install open-vm-tools-desktop fuse
reboot
```

Adding Support for Shared Folders When Using OVT

Unfortunately, shared folders will not work out of the box. To enable this feature for your current session, you will need to execute the following script after logging in:

```
cat <<EOF > /usr/local/sbin/mount-shared-folders
#!/bin/bash
vmware-hgfsclient | while read folder; do
  vmwpath="/mnt/hgfs/\${folder}"
  echo "[i] Mounting \${folder}   (\${vmwpath})"
  mkdir -p "\${vmwpath}"
  umount -f "\${vmwpath}" 2>/dev/null
  vmhgfs-fuse -o allow_other -o auto_unmount ".host:/\${folder}" "\${vmwpath}"
done
sleep 2s
EOF
chmod +x /usr/local/sbin/mount-shared-folders
```

If you wish to make it a little easier, you can add a shortcut to the desktop (and allow the script to be executed upon double clicking if you are you are using GNOME):

```
ln -sf /usr/local/sbin/mount-shared-folders /root/Desktop/mount-shared-folders
gsettings set org.gnome.nautilus.preferences executable-text-activation 'ask'
```

Restarting OVT

If OVT stops functioning correctly, such as Copy/Paste between host and guest, the following script may help out:

```
cat <<EOF > /usr/local/sbin/restart-vm-tools
#!/bin/bash
systemctl stop run-vmblock\\\\x2dfuse.mount
killall -q -w vmtoolsd
systemctl start run-vmblock\\\\x2dfuse.mount
systemctl enable run-vmblock\\\\x2dfuse.mount
vmware-user-suid-wrapper vmtoolsd -n vmusr 2>/dev/null
vmtoolsd -b /var/run/vmroot 2>/dev/null
EOF
chmod +x /usr/local/sbin/restart-vm-tools
ln -sf /usr/local/sbin/restart-vm-tools /root/Desktop/restart-vm-tools
gsettings set org.gnome.nautilus.preferences executable-text-activation 'ask'
```

Installing VMware Tools in Older Kali Versions

The latest version of vmware-tools at this date compiles against our kernel, albeit with several warnings. We utilise a set of vmware-tool patches to facilitate the installation.

```
cd ~/
apt-get install git gcc make linux-headers-$(uname -r)
git clone https://github.com/rasa/vmware-tools-patches.git
cd vmware-tools-patches/
```

Next, mount the VMware tools ISO by clicking “Install VMware Tools” from the appropriate menu. Once the VMware Tools ISO has been attached to the virtual machine, copy the installer to the *downloads* directory and then run the installer script :

```
cd ~/vmware-tools-patches/
cp /media/cdrom/VMwareTools-9.9.0-2304977.tar.gz downloads/
./untar-and-patch-and-compile.sh
```

06. Customizing Kali Linux

Rebuilding a Source Package

Kali Linux is easy to customize at a per-package level, and it's equally simple to make modifications to individual packages and rebuild them from their source code for inclusion in your custom ISO or on your desktop install.

Accomplishing this is a simple three-step process:

- use **apt** to pull down the package sources
- modify them as needed
- rebuild the package using the Debian tools.

In this example, we will rebuild the libfreefare package in order to add some extra hardcoded Mifare access keys into the mifare-format tool.

Downloading the Package Source

```
# Get the source package
apt update
apt-get source libfreefare
cd libfreefare-0.4.0/
```

Edit the Package Source Code

Make the changes needed to the source code of the package. In our case, we modify an example file, mifare-classic-format.c.

```
nano examples/mifare-classic-format.c
```

Check for Build Dependencies

Check for any build dependencies the package may have. These need to be installed before you can build the package.

```
dpkg-checkbuilddeps
```

The output should be similar to the following, depending on what packages you already have installed. If **dpkg-checkbuilddeps** returns no output, that means you can proceed with the build, all of the dependencies are already satisfied.

```
dpkg-checkbuilddeps: Unmet build dependencies: dh-autoreconf libnfc-dev libssl-dev
```

Install Build Dependencies

Install any build dependencies if needed, as shown in the output of **dpkg-checkbuilddeps**:

```
apt install dh-autoreconf libnfc-dev libssl-dev
```

Build the Modified Package

With all of the dependencies installed, the **dpkg-buildpackage** command is all it takes to build your new version.

```
dpkg-buildpackage
```

Install the New Package

If the build completes without errors, you'll be able to install your newly-created package with **dpkg**.

```
dpkg -i ../libfreefare*.deb
```

Live Build a Custom Kali ISO

An Introduction to Building Your Own Kali ISO

Building a customized Kali ISO is easy, fun, and rewarding. You can configure virtually any aspect of your Kali ISO build using the Debian live-build scripts. These scripts allow developers to easily build live system images by providing a framework that uses a configuration set to automate and customize all aspects of building the image. The Kali Linux development team has adopted these scripts and they're used to produce the official Kali ISO releases.

Where Should You Build Your ISO?

Ideally, you should build your custom Kali ISO from **within a pre-existing Kali environment**.

Getting Ready — Setting up the live-build system

We first need to prepare the Kali ISO build environment by installing and setting up live-build and its requirements with the following commands:

```
apt install -y curl git live-build cdebootstrap
git clone git://gitlab.com/kalilinux/build-scripts/live-build-config.git
```

Now you can simply build an updated Kali ISO by entering the "live-build-config" directory and running our **build.sh** wrapper script, as follows:

```
cd live-build-config/
./build.sh --verbose
```

The "build.sh" script will take a while to complete, as it downloads all of the required packages needed to create your ISO. Good time for a coffee.

Configuring the Kali ISO Build (Optional)

If you want to customize your Kali Linux ISO, this section will explain some of the details. Through the **kali-config** directory, the Kali Linux live build supports a wide range of customization options, which are well-documented on the Debian live build 4.x page. However, for the impatient, here are some of the highlights.

Building Kali with Different Desktop Environments

Since Kali 2.0, we now support built in configurations for various desktop environments, including KDE, Gnome, E17, I3WM, LXDE, MATE and XFCE. To build any of these, you would use syntax similar to the following:

```
# These are the different Desktop Environment build options:
#./build.sh --variant {gnome,kde,xfce,mate,e17,lxde,i3wm} --verbose

# To build a KDE ISO:
./build.sh --variant kde --verbose
# To build a MATE ISO:
./build.sh --variant mate --verbose

#...and so on.
```

Controlling the packages included in your build

The list of packages included in your build will be present in the the respective kali-$variant directory. For example, if you're building a default Gnome ISO, you would use the following package lists file – **kali-config/variant-gnome/package-lists/kali.list.chroot**. By default, this list includes the "kali-linux-full" metapackage, as well as some others. These can be commented out and replaced with a manual list of packages to include in the ISO for greater granularity.

Build hooks, binary and chroot

Live-build hooks allows us to hook scripts in various stages of the Kali ISO live build. For more detailed information about hooks and how to use them, refer to the live build manual. As an example, we recommend you check out the existing hooks in **kali-config/common/hooks/**.

Overlaying files in your build

You have the option to include additional files or scripts in your build by overlaying them on the existing

filesystem, inside the **includes.{chroot,binary,installer}** directories, respectively. For example, if we wanted to include our own custom script into the **/root/** directory of the ISO (this would correspond to the "chroot" stage), then we would drop this script file in the **kali-config/common/includes.chroot/** directory before building the ISO.

Building a Kali Linux ISO for older i386 architectures

The Kali Linux i386 ISO has PAE enabled. If you require a default kernel for older hardware with PAE disabled, you will need to rebuild a Kali Linux ISO. The rebuilding process is much the same as described above, except that the **686-pae** parameter that needs to be changed to **586** in **auto/config** as follows. First, install the prerequisites.

```
apt install -y git live-build cdebootstrap debootstrap
git clone git://gitlab.com/kalilinux/build-scripts/live-build-config.git
```

Next, make the change in auto/config for the appropriate architecture:

```
cd live-build-config/
sed -i 's/686-pae/686/g' auto/config
```

Finally, run your build.

```
./build.sh --arch i386 --verbose
```

Building Kali on Non-Kali Debian Based Systems

You can easily run live-build on Debian based systems other than Kali. The instructions below have been tested to work with both Debian and Ubuntu.

First, we prep the system by ensuring it is fully updated, then proceed to download the Kali archive keyring and live-build packages.

```
sudo apt update
sudo apt upgrade
cd /root/

wget http://http.kali.org/pool/main/k/kali-archive-keyring/kali-archive-keyring_2018.1_all.deb
wget https://archive.kali.org/kali/pool/main/l/live-build/live-build_20180618kali1_all.deb
```

With that completed, we install some additional dependencies and the previously downloaded files.

```
sudo apt install -y git live-build cdebootstrap debootstrap curl
sudo dpkg -i kali-archive-keyring_2018.1_all.deb
sudo dpkg -i live-build_20180618kali1_all.deb
```

With the environment all prepared, we start the live-build process by setting up the build script and checking out the build config.

```
cd /usr/share/debootstrap/scripts/
echo "default_mirror http://http.kali.org/kali"; sed -e
"s/debian-archive-keyring.gpg/kali-archive-keyring.gpg/g" sid > /tmp/kali
sudo mv /tmp/kali .
sudo ln -s kali kali-rolling

cd ~
git clone git://gitlab.com/kalilinux/build-scripts/live-build-config.git

cd live-build-config/
```

At this point, we have to edit the `build.sh` script to bypass a version check. We do this by commenting out the “exit 1” below.

```
# Check we have a good debootstrap
ver_debootstrap=$(dpkg-query -f '${Version}' -W debootstrap)
if dpkg --compare-versions "$ver_debootstrap" lt "1.0.97"; then
if ! echo "$ver_debootstrap" | grep -q kali; then
echo "ERROR: You need debootstrap >= 1.0.97 (or a Kali patched debootstrap). Your current version:
$ver_debootstrap" >&2
exit 1
fi
fi
```

With that change made, the script should like as follows:

```
# Check we have a good debootstrap
ver_debootstrap=$(dpkg-query -f '${Version}' -W debootstrap)
if dpkg --compare-versions "$ver_debootstrap" lt "1.0.97"; then
if ! echo "$ver_debootstrap" | grep -q kali; then
echo "ERROR: You need debootstrap >= 1.0.97 (or a Kali patched debootstrap). Your current version:
$ver_debootstrap" >&2
# exit 1
fi
fi
```

At this point, we can build our ISO as normal

```
sudo ./build.sh --variant light --verbose
```

Recompiling the Kali Linux Kernel

The customizability of Kali Linux extends all the way down into the Linux kernel.

Depending on your requirements, you might want to add drivers, patches, or kernel features that are not included in the stock Kali Linux kernel. The following guide will describe how the Kali Linux kernel can be quickly modified and recompiled for your needs. Note that global wireless injection patches are already present by default in the Kali Linux kernel.

Install Build Dependencies

Start by installing all the build dependencies for recompiling the kernel.

```
apt install build-essential libncurses5-dev fakeroot unxz
```

Download the Kali Linux Kernel Source Code

The remainder of this section focuses on the 4.9 version of the Linux kernel, but the examples can, of course, be adapted to the particular version of the kernel that you want. We assume that the linux-source-4.9 binary package has been installed. Note that we install a binary package containing the upstream sources, we do not retrieve the Kali source package named linux.

```
apt install linux-source-4.9
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bc libreadline7
Suggested packages:
  libncurses-dev | ncurses-dev libqt4-dev
The following NEW packages will be installed:
  bc libreadline7 linux-source-4.9
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 95.4 MB of archives.
After this operation, 95.8 MB of additional disk space will be used.
```

```
Do you want to continue? [Y/n] y
[...]
ls /usr/src
linux-config-4.9  linux-patch-4.9-rt.patch.xz  linux-source-4.9.tar.xz
```

Notice that the package contains */usr/src/linux-source-4.9.tar.xz*, a compressed archive of the kernel sources. You must extract these files in a new directory (not directly under /usr/src/, since there is no need for special permissions to compile a Linux kernel). Instead, *~/kernel/* is more appropriate.

```
mkdir ~/kernel; cd ~/kernel
tar -xaf /usr/src/linux-source-4.9.tar.xz
```

Configure Your Kernel

When recompiling a more recent version of the kernel (possibly with an additional patch), the configuration will most likely be kept as close as possible to that proposed by Kali. In this case, and rather than reconfiguring everything from scratch, it is sufficient to copy the */boot/config-version* file (the version is that of the kernel currently used, which can be found with the **uname -r** command) into a *.config* file in the directory containing the kernel sources.

```
cp /boot/config-4.9.0-kali1-amd64 ~/kernel/linux-source-4.9/.config
```

If you need to make changes or if you decide to reconfigure everything from scratch, you must take the time to configure your kernel. This can be done by calling the **make menuconfig** command.

```
make menuconfig
```

The details of using **menuconfig** to set up a kernel build are beyond the scope of this guide. There is a detailed tutorial on configuring a kernel build on Linux.org.

Build the Kernel

Once the kernel configuration is ready, a simple **make deb-pkg** will generate up to 5 Debian packages: *linux-image-**version*** that contains the kernel image and the associated modules, *linux-headers-**version***, which contains the header files required to build external modules, *linux-firmware-image-**version***, which contains the firmware files needed by some drivers (this package might be missing when you build from the kernel sources provided by Debian or Kali), *linux-image-**version**-dbg*, which contains the debugging symbols for the kernel image and its modules, and *linux-libc-dev*, which contains headers relevant to some user-space libraries like GNU glibc. The Linux kernel image is a big build, expect it to take a while to complete.

```
make clean
make deb-pkg LOCALVERSION=-custom KDEB_PKGVERSION=$(make kernelversion)-1
[...]
ls ../*.deb
../linux-headers-4.9.0-kali1-custom_4.9.2-1_amd64.deb
../linux-image-4.9.0-kali1-custom_4.9.2-1_amd64.deb
../linux-image-4.9.0-kali1-custom-dbg_4.9.2-1_amd64.deb
../linux-libc-dev_4.9.2-1_amd64.deb
```

Install the Modified Kernel

When the build has successfully completed, you can go ahead and install the new custom kernel and reboot your system. Please note that the specific kernel version numbers will vary — in our example, done on a Kali 2016.2 system, it was 4.9.2. Depending on the current kernel version you're building, you will need to adjust your commands accordingly.

```
dpkg -i ../linux-image-4.9.0-kali1-custom_4.9.2-1_amd64.deb
reboot
```

Once your system has rebooted, your new kernel should be running. If things go wrong and your kernel fails to boot successfully, you can still use the GrUB menu to boot from the original stock Kali kernel and fix your issues.

Custom EfikaMX Image

The following document describes our own method of creating a **custom Kali Linux EfikaMX ARM image** and is targeted at developers. If you would like to install a pre-made Kali image, check out our Install Kali on an EfikaMX article.

01. Create a Kali rootfs

Build a Kali rootfs as described in our Kali documentation, using an **armhf** architecture. By the end of this process, you should have a populated rootfs directory in **~/arm-stuff/rootfs/kali-armhf**.

02. Create the Image File

Next, we create the physical image file, which will hold our EfikaMX rootfs and boot images.

```
apt-get install kpartx xz-utils sharutils
cd ~
mkdir -p arm-stuff
cd arm-stuff/
mkdir -p images
cd images
dd if=/dev/zero of=kali-custom-efikamx.img bs=1MB count=7000
```

03. Partition and Mount the Image File

```
parted kali-custom-efikamx.img --script -- mklabel msdos
parted kali-custom-efikamx.img --script -- mkpart primary ext2 4096s 266239s
parted kali-custom-efikamx.img --script -- mkpart primary ext4 266240s 100%
```

```
loopdevice=`losetup -f --show kali-custom-efikamx.img`
device=`kpartx -va $loopdevice| sed -E 's/.*(loop[0-9])p.*/\1/g' | head -1`
```

```
device="/dev/mapper/${device}"
bootp=${device}p1
rootp=${device}p2

mkfs.ext2 $bootp
mkfs.ext4 $rootp
mkdir -p boot
mkdir -p root
mount $bootp boot
mount $rootp root
```

04. Copy and Modify the Kali rootfs

```
rsync -HPavz /root/arm-stuff/rootfs/kali-armhf/ root
echo nameserver 8.8.8.8 > root/etc/resolv.conf
sed 's/0-1/0//g' root/etc/init.d/udev
```

05. Compile the EfikaMX Kernel and Modules

If you're not using ARM hardware as the development environment, you will need to set up an ARM cross-compilation environment to build an ARM kernel and modules. Once that's done, proceed with the following instructions.

```
cd ~/arm-stuff
mkdir -p kernel
cd kernel
git clone --depth 1 https://github.com/genesi/linux-legacy.git
cd linux-legacy
export ARCH=arm
export CROSS_COMPILE=~/arm-stuff/kernel/toolchains/arm-eabi-linaro-4.6.2/bin/arm-eabi-
make efikamx_defconfig
# configure your kernel !
```

```
make menuconfig
make -j$(cat /proc/cpuinfo|grep processor|wc -l)
make modules_install INSTALL_MOD_PATH=~/arm-stuff/images/root
make uImage
cp arch/arm/boot/uImage ~/arm-stuff/images/boot

cat << EOF > ~/arm-stuff/images/boot/boot.script
setenv ramdisk uInitrd;
setenv kernel uImage;
setenv bootargs console=tty1 root=/dev/mmcblk0p2 rootwait rootfstype=ext4 rw quiet;
${loadcmd} ${ramdiskaddr} ${ramdisk};
if imi ${ramdiskaddr}; then; else
setenv bootargs ${bootargs} noinitrd;
setenv ramdiskaddr "";
fi;
${loadcmd} ${kerneladdr} ${kernel}
if imi ${kerneladdr}; then
bootm ${kerneladdr} ${ramdiskaddr}
fi;
EOF

mkimage -A arm -T script -C none -n "Boot.scr for EfikaMX" -d ~/arm-stuff/images/boot/boot.script
~/arm-stuff/images/boot/boot.scr
```

```
umount $bootp
umount $rootp
kpartx -dv $loopdevice
losetup -d $loopdevice
```

Use the **dd** utility to image this file to your SD card. In our example, we assume the storage device is located at /dev/sdb. **Change this as needed.**

```
```

```
dd if=kali-custom-efikamx.img of=/dev/sdb bs=1M
```

Once the dd operation is complete, unmount and eject the SD card and boot your EfikaMX into Kali Linux

Custom Beaglebone Black Image

The following document describes our own method of creating a**custom Kali Linux Beaglebone Black ARM image** and is targeted at developers. If you would like to install a pre-made Kali image, check out our Install Kali on Beaglebone Black article.

01. Create a Kali rootfs

Build a Kali rootfs as described in our Kali documentation, using an **armhf** architecture. By the end of this process, you should have a populated rootfs directory in **~/arm-stuff/rootfs/kali-armhf**.

02. Create the Image File

Next, we create the physical image file, which will hold our Beaglebone Black rootfs and boot images.

```
apt-get install kpartx xz-utils sharutils
cd ~
mkdir -p arm-stuff
cd arm-stuff/
mkdir -p images
cd images
dd if=/dev/zero of=kali-custom-bbb.img bs=1MB count=7000
```

03. Partition and Mount the Image File

```
parted --script kali-custom-bbb.img mklabel msdos
fdisk kali-custom-bbb.img << __EOF__
n
p
1

+64M
t
e
p
```

```
w
__EOF__
parted --script kali-custom-bbb.img set 1 boot on
fdisk kali-custom-bbb.img << __EOF__
n
p
2

w
__EOF__
```

```
loopdevice=`losetup -f --show kali-custom-bbb.img`
device=`kpartx -va $loopdevice| sed -E 's/.*(loop[0-9])p.*/\1/g' | head -1`
device="/dev/mapper/${device}"
bootp=${device}p1
rootp=${device}p2

mkfs.vfat -F 16 $bootp -n boot
mkfs.ext4 $rootp -L kaliroot
mkdir -p boot
mkdir -p root
mount $bootp boot
mount $rootp root
```

04. Copy and Modify the Kali rootfs

```
rsync -HPavz /root/arm-stuff/rootfs/kali-armhf/ root
echo nameserver 8.8.8.8 > root/etc/resolv.conf
```

05. Compile the Beaglebone Black Kernel and Modules

If you're not using ARM hardware as the development environment, you will need to set up an ARM cross-compilation environment to build an ARM kernel and modules. Once that's done, proceed with the following instructions.

```
cd ~/arm-stuff
wget
https://launchpad.net/linaro-toolchain-binaries/trunk/2013.03/+download/
gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313_linux.tar.bz2
tar xjf gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313_linux.tar.bz2
export CC=`pwd`/gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313_linux/bin/arm-linux-gnueabihf-

git clone git://git.denx.de/u-boot.git
cd u-boot/
git checkout v2013.04 -b beaglebone-black
wget
https://raw.github.com/eewiki/u-boot-patches/master/v2013.04/0001-am335x_evm-uEnv.txt-bootz-n-
fixes.patch
patch -p1 < 0001-am335x_evm-uEnv.txt-bootz-n-fixes.patch
make ARCH=arm CROSS_COMPILE=${CC} distclean
make ARCH=arm CROSS_COMPILE=${CC} am335x_evm_config
make ARCH=arm CROSS_COMPILE=${CC}
cd ..

mkdir -p kernel
cd kernel
git clone git://github.com/RobertCNelson/linux-dev.git
cd linux-dev/
git checkout origin/am33x-v3.8 -b tmp
./build_kernel.sh
mkdir -p ../patches
wget http://patches.aircrack-ng.org/mac80211.compat08082009.wl_frag+ack_v1.patch -O
../patches/mac80211.patch
cd KERNEL
patch -p1 --no-backup-if-mismatch < ../../patches/mac80211.patch
cd ..
./tools/rebuild.sh
cd ..
```

```
cat << EOF > boot/uEnv.txt
mmcroot=/dev/mmcblk0p2 ro
mmcrootfstype=ext4 rootwait fixrtc
uenvcmd=run loaduimage; run loadfdt; run mmcargs; bootz 0x80200000 - 0x80F80000
EOF

cp -v kernel/linux-dev/deploy/3.8.13-bone20.zImage boot/zImage
mkdir -p boot/dtbs
tar -xovf kernel/linux-dev/deploy/3.8.13-bone20-dtbs.tar.gz -C boot/dtbs/

tar -xovf kernel/linux-dev/deploy/3.8.13-bone20-modules.tar.gz -C root/
tar -xovf kernel/linux-dev/deploy/3.8.13-bone20-firmware.tar.gz -C root/lib/firmware/

cat << EOF > root/etc/fstab
/dev/mmcblk0p2 / auto errors=remount-ro 0 1
/dev/mmcblk0p1 /boot/uboot auto defaults 0 0
EOF
```

```
umount $rootp
kpartx -dv $loopdevice
losetup -d $loopdevice
```

Use the **dd** utility to image this file to your SD card. In our example, we assume the storage device is located at /dev/sdb. **Change this as needed.**

```
dd if=kali-custom-bbb.img of=/dev/sdb bs=1M
```

Once the dd operation is complete, unmount and eject the SD card and boot your Beaglebone Black into Kali

Linux. When booting you will need to press and hold the “BOOT” button, it’s the one closest to the microSD card.

Custom CuBox Image

The following document describes our own method of creating a **custom Kali Linux CuBox ARM image** and is targeted at developers. If you would like to install a pre-made Kali image, check out our Install Kali on CuBox article.

01. Create a Kali rootfs

Build a Kali rootfs as described in our Kali documentation, using an **armhf** architecture. By the end of this process, you should have a populated rootfs directory in **~/arm-stuff/rootfs/kali-armhf**.

02. Create the Image File

Next, we create the physical image file, which will hold our CuBox rootfs and boot images.

```
apt-get install kpartx xz-utils sharutils
cd ~
mkdir -p arm-stuff
cd arm-stuff/
mkdir -p images
cd images
dd if=/dev/zero of=kali-custom-cubox.img bs=1MB count=7000
```

03. Partition and Mount the Image File

```
parted kali-custom-cubox.img --script -- mklabel msdos
parted kali-custom-cubox.img --script -- mkpart primary ext4 0 -1
```

```
loopdevice=`losetup -f --show kali-custom-cubox.img`
device=`kpartx -va $loopdevice| sed -E 's/.*(loop[0-9])p.*/\1/g' | head -1`
device="/dev/mapper/${device}"
```

```
rootp=${device}p1

mkfs.ext4 $rootp
mkdir -p root
mount $rootp root
```

04. Copy and Modify the Kali rootfs

```
rsync -HPavz /root/arm-stuff/rootfs/kali-armhf/ root
echo nameserver 8.8.8.8 > root/etc/resolv.conf
```

05. Compile the CuBox Kernel and Modules

If you're not using ARM hardware as the development environment, you will need to set up an ARM cross-compilation environment to build an ARM kernel and modules. Once that's done, proceed with the following instructions.

```
cd ~/arm-stuff
mkdir -p kernel
cd kernel
git clone --depth 1 https://github.com/rabeeh/linux.git
cd linux
touch .scmversion
mkdir -p ../patches
wget http://patches.aircrack-ng.org/mac80211.compat08082009.wl_frag+ack_v1.patch -O
../patches/mac80211.patch
patch -p1 --no-backup-if-mismatch < ../patches/mac80211.patch
export ARCH=arm
export CROSS_COMPILE=~/arm-stuff/kernel/toolchains/arm-eabi-linaro-4.6.2/bin/arm-eabi-
make cubox_defconfig
# configure your kernel !
make menuconfig
```

```
make -j$(cat /proc/cpuinfo|grep processor|wc -l)
make modules_install INSTALL_MOD_PATH=~/arm-stuff/images/root
make uImage
cp arch/arm/boot/uImage ~/arm-stuff/images/root/boot

cat << EOF > ~/arm-stuff/images/root/boot/boot.txt
echo "== Executing ${directory}${bootscript} on ${device_name} partition ${partition} =="
setenv unit_no 0
setenv root_device ?

if itest.s ${device_name} -eq usb; then
itest.s $root_device -eq ? && ext4ls usb 0:1 /dev && setenv root_device /dev/sda1 && setenv unit_no 0
itest.s $root_device -eq ? && ext4ls usb 1:1 /dev && setenv root_device /dev/sda1 && setenv unit_no 1
fi

if itest.s ${device_name} -eq mmc; then
itest.s $root_device -eq ? && ext4ls mmc 0:2 /dev && setenv root_device /dev/mmcblk0p2
itest.s $root_device -eq ? && ext4ls mmc 0:1 /dev && setenv root_device /dev/mmcblk0p1
fi

if itest.s ${device_name} -eq ide; then
itest.s $root_device -eq ? && ext4ls ide 0:1 /dev && setenv root_device /dev/sda1
fi

if itest.s $root_device -ne ?; then
setenv bootargs "console=ttyS0,115200n8 vmalloc=448M video=dovefb:lcd0:1920x1080-32@60-edid
clcd.lcd0_enable=1 clcd.lcd1_enable=0 root=${root_device} rootfstype=ext4"
setenv loadimage "${fstype}load ${device_name} ${unit_no}:${partition} 0x00200000
${directory}${image_name}"
$loadimage && bootm 0x00200000
echo "!! Unable to load ${directory}${image_name} from ${device_name} ${unit_no}:${partition} !!"
exit
fi

echo "!! Unable to locate root partition on ${device_name} !!"
EOF

mkimage -A arm -T script -C none -n "Boot.scr for CuBox" -d ~/arm-stuff/images/root/boot/boot.txt
~/arm-stuff/images/root/boot/boot.scr
```

```
umount $rootp
kpartx -dv $loopdevice
losetup -d $loopdevice
```

Use the **dd** utility to image this file to your SD card. In our example, we assume the storage device is located at /dev/sdb. **Change this as needed.**

```
dd if=kali-custom-cubox.img of=/dev/sdb bs=1M
```

Once the dd operation is complete, unmount and eject the SD card and boot your CuBox into Kali Linux

Custom Raspberry Pi Image

The following document describes our own method of creating a **custom Kali Linux Raspberry Pi ARM image** and is targeted at developers. If you would like to install a pre-made Kali image, check out our Install Kali on Raspberry Pi article.

01. Create a Kali rootfs

Build a Kali rootfs as described in our Kali documentation, using an **armel** architecture. By the end of this process, you should have a populated rootfs directory in **~/arm-stuff/rootfs/kali-armel**.

02. Create the Image File

Next, we create the physical image file, which will hold our Raspberry Pi rootfs and boot images.

```
apt-get install kpartx xz-utils sharutils
cd ~
mkdir -p arm-stuff
cd arm-stuff/
mkdir -p images
cd images
dd if=/dev/zero of=kali-custom-rpi.img bs=1MB count=7000
```

03. Partition and Mount the Image File

```
parted kali-custom-rpi.img --script -- mklabel msdos
parted kali-custom-rpi.img --script -- mkpart primary fat32 0 64
parted kali-custom-rpi.img --script -- mkpart primary ext4 64 -1
```

```
loopdevice=`losetup -f --show kali-custom-rpi.img`
device=`kpartx -va $loopdevice| sed -E 's/.*(loop[0-9])p.*/\1/g' | head -1`
```

```
device="/dev/mapper/${device}"
bootp=${device}p1
rootp=${device}p2

mkfs.vfat $bootp
mkfs.ext4 $rootp
mkdir -p root
mkdir -p boot
mount $rootp root
mount $bootp boot
```

04. Copy and Modify the Kali rootfs

```
rsync -HPavz /root/arm-stuff/rootfs/kali-armel/ root
echo nameserver 8.8.8.8 > root/etc/resolv.conf
```

05. Compile the Raspberry Pi Kernel and Modules

If you're not using ARM hardware as the development environment, you will need to set up an ARM cross-compilation environment to build an ARM kernel and modules. Once that's done, proceed with the following instructions.

```
cd ~/arm-stuff
mkdir -p kernel
cd kernel
git clone https://github.com/raspberrypi/tools.git
git clone https://github.com/raspberrypi/linux.git raspberrypi
cd raspberrypi
touch .scmversion
export ARCH=arm
export CROSS_COMPILE=~/arm-stuff/kernel/toolchains/arm-eabi-linaro-4.6.2/bin/arm-eabi-
make bcmrpi_cutdown_defconfig
```

```
# configure your kernel !
make menuconfig
make -j$(cat /proc/cpuinfo|grep processor|wc -l)
make modules_install INSTALL_MOD_PATH=~/arm-stuff/images/root
cd ../tools/mkimage/
python imagetool-uncompressed.py ../../raspberrypi/arch/arm/boot/Image
```

```
cd ~/arm-stuff/images
git clone git://github.com/raspberrypi/firmware.git rpi-firmware
cp -rf rpi-firmware/boot/* boot/
rm -rf rpi-firmware

cp ~/arm-stuff/kernel/tools/mkimage/kernel.img boot/
echo "dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1
root=/dev/mmcblk0p2 rootfstype=ext4 rootwait" > boot/cmdline.txt
```

```
umount $rootp
umount $bootp
kpartx -dv $loopdevice
losetup -d $loopdevice
```

Use the **dd** utility to image this file to your SD card. In our example, we assume the storage device is located at /dev/sdb. **Change this as needed.**

```
dd if=kali-pi.img of=/dev/sdb bs=1M
```

Once the dd operation is complete, unmount and eject the SD card and boot your Pi into Kali Linux

Custom Chromebook Image

The following document describes our own method of creating a **custom Kali Linux Samsung Chromebook ARM image** and is targeted at developers. If you would like to install a pre-made Kali image, check out our Install Kali on Samsung Chromebook article.

In this guide, we create an image with two boot partitions – one containing a kernel hard-coded to boot from the SD card and the other containing a kernel hard-coded to boot from USB. Depending on your USB storage media type, make sure to mark the relevant boot partition with higher priority after you dd the image to your USB device as instructed in the last stages of this guide.

01. Create a Kali rootfs

Start by building a Kali rootfs as described in our Kali documentation, using an **armhf** architecture. By the end of this process, you should have a populated rootfs directory in **~/arm-stuff/rootfs/kali-armhf**.

02. Create the Image File

Next, we create the physical image file that will hold our Chromebook rootfs and boot images.

```
apt-get install kpartx xz-utils gdisk uboot-mkimage u-boot-tools vboot-kernel-utils vboot-utils cgpt
cd ~
mkdir -p arm-stuff
cd arm-stuff/
mkdir -p images
cd images
dd if=/dev/zero of=kali-custom-chrome.img bs=1MB count=7000
```

03. Partition and Mount the Image File

```
parted kali-custom-chrome.img --script -- mklabel msdos
parted kali-custom-chrome.img --script -- mktable gpt
gdisk kali-custom-chrome.img << EOF
x
```

```
l
8192
m
n
1

+16M
7f00
n
2

+16M
7f00
n
3

w
y
EOF
```

```
loopdevice=`losetup -f --show kali-custom-chrome.img`
device=`kpartx -va $loopdevice| sed -E 's/.*(loop[0-9])p.*/\1/g' | head -1`
device="/dev/mapper/${device}"
bootp1=${device}p1
bootp2=${device}p2
rootp=${device}p3

mkfs.ext4 $rootp
mkdir -p root
mount $rootp root
```

04. Copy and Modify the Kali rootfs

Copy over the Kali rootfs you bootstrapped earlier using **rsync** to the mounted image.

```
cd ~/arm-stuff/images/
rsync -HPavz ~/arm-stuff/rootfs/kali-armhf/ root

echo nameserver 8.8.8.8 > root/etc/resolv.conf

mkdir -p root/etc/X11/xorg.conf.d/
cat << EOF > root/etc/X11/xorg.conf.d/50-touchpad.conf
Section "InputClass"
Identifier "touchpad"
MatchIsTouchpad "on"
Driver "synaptics"
Option "TapButton1" "1"
Option "TapButton2" "3"
Option "TapButton3" "2"
Option "FingerLow" "15"
Option "FingerHigh" "20"
Option "FingerPress" "256"
EndSection
EOF
```

05. Compile the Samsung Chromium Kernel and Modules

If you're not using ARM hardware as the development environment, you will need to set up an ARM cross-compilation environment to build an ARM kernel and modules. Once that's done, proceed with the following instructions.

Fetch the Chromium kernel sources and place them in our development tree structure:

```
cd ~/arm-stuff
mkdir -p kernel
cd kernel
git clone http://git.chromium.org/chromiumos/third_party/kernel.git -b chromeos-3.4 chromeos
cd chromeos
```

```
cat << EOF > kernel.its
/dts-v1/;

/ {
description = "Chrome OS kernel image with one or more FDT blobs";
#address-cells = ;
images {
kernel@1{
description = "kernel";
data = /incbin/("arch/arm/boot/zImage");
type = "kernel_noload";
arch = "arm";
os = "linux";
compression = "none";
load = ;
entry = ;
};
fdt@1{
description = "exynos5250-snow.dtb";
data = /incbin/("arch/arm/boot/exynos5250-snow.dtb");
type = "flat_dt";
arch = "arm";
compression = "none";
hash@1{
algo = "sha1";
};
};
};
configurations {
default = "conf@1";
conf@1{
kernel = "kernel@1";
fdt = "fdt@1";
};
};
};
EOF
```

Patch the kernel, in our case, with wireless injection patches.

```
mkdir -p ../patches
wget http://patches.aircrack-ng.org/mac80211.compat08082009.wl_frag+ack_v1.patch -O
../patches/mac80211.patch
wget http://patches.aircrack-ng.org/channel-negative-one-maxim.patch -O ../patches/negative.patch
patch -p1 < ../patches/negative.patch
patch -p1 < ../patches/mac80211.patch
```

Configure, then cross-compile the Chromium kernel as shown below.

```
export ARCH=arm
export CROSS_COMPILE=~/arm-stuff/kernel/toolchains/arm-eabi-linaro-4.6.2/bin/arm-eabi-

./chromeos/scripts/prepareconfig chromeos-exynos5
# Disable LSM
sed -i 's/CONFIG_SECURITY_CHROMIUMOS=y/# CONFIG_SECURITY_CHROMIUMOS is not set/g' .config
# If cross compiling, do this once:
sed -i 's/if defined(__linux__)/if defined(__linux__) ||defined(__KERNEL__) /g' include/drm/drm.h

make menuconfig
make -j$(cat /proc/cpuinfo|grep processor|wc -l)
make dtbs
cp ./scripts/dtc/dtc /usr/bin/
mkimage -f kernel.its kernel.itb
make modules_install INSTALL_MOD_PATH=~/arm-stuff/images/root/

# copy over firmware. Ideally use the original firmware (/lib/firmware) from the Chromebook.
git clone git://git.kernel.org/pub/scm/linux/kernel/git/dwmw2/linux-firmware.git
cp -rf linux-firmware/* ~/arm-stuff/images/root/lib/firmware/
rm -rf linux-firmware
```

```
echo "console=tty1 debug verbose root=/dev/mmcblk1p3 rootwait rw rootfstype=ext4" > /tmp/config-sd
echo "console=tty1 debug verbose root=/dev/sda3 rootwait rw rootfstype=ext4" > /tmp/config-usb

vbutil_kernel --pack /tmp/newkern-sd --keyblock /usr/share/vboot/devkeys/kernel.keyblock --version 1
--signprivate /usr/share/vboot/devkeys/kernel_data_key.vbprivk --config=/tmp/config-sd --vmlinuz kernel.itb
--arch arm
vbutil_kernel --pack /tmp/newkern-usb --keyblock /usr/share/vboot/devkeys/kernel.keyblock --version 1
--signprivate /usr/share/vboot/devkeys/kernel_data_key.vbprivk --config=/tmp/config-usb --vmlinuz
kernel.itb --arch arm
```

06. Prepare the Boot Partition

```
dd if=/tmp/newkern-sd of=$bootp1 # first boot partition for SD
dd if=/tmp/newkern-usb of=$bootp2 # second boot partition for USB

umount $rootp

kpartx -dv $loopdevice
losetup -d $loopdevice
```

07. dd the Image and Mark the USB Drive Bootable

```
dd if=kali-custom-chrome.img of=/dev/sdb bs=512k
cgpt repair /dev/sdb
```

2

This is the point where you need to mark either boot partition 1 or 2 to have higher priority. The number

with the higher priority will boot first. The example below will give priority 10 to the first partition (-i) and will thus boot successfully from a SD card.

```
cgpt add -i 1 -S 1 -T 5 -P 10 -l KERN-A /dev/sdb
cgpt add -i 2 -S 1 -T 5 -P 5 -l KERN-B /dev/sdb
```

To see your partition list and order, use the command **cgpt show**.

```
root@kali:~# cgpt show /dev/sdb
start size part contents
0 1 PMBR
1 1 Pri GPT header
2 32 Pri GPT table
8192 32768 1 Label: "KERN-A"
Type: ChromeOS kernel
UUID: 63AD6EC9-AD94-4B42-80E4-798BBE6BE46C
Attr: priority=10 tries=5 successful=1
40960 32768 2 Label: "KERN-B"
Type: ChromeOS kernel
UUID: 37CE46C9-0A7A-4994-80FC-9C0FFCB4FDC1
Attr: priority=5 tries=5 successful=1
73728 3832490 3 Label: "Linux filesystem"
Type: 0FC63DAF-8483-4772-8E79-3D69D8477DE4
UUID: E9E67EE1-C02E-481C-BA3F-18E721515DBB
125045391 32 Sec GPT table
125045423 1 Sec GPT header
root@kali:~#
```

Once this operation is complete, boot up your Samsung Chromebook with the SD/USB device plugged in. At the developer mode boot screen, hit CTRL+u to boot from from your USB storage device. Log in to Kali (root / toor)

and startx.

Custom MK/SS808 Image

The following document describes our own method of creating a **custom Kali Linux MK/SS808 ARM image** and is targeted at developers. If you would like to install a pre-made Kali image, check out our Install Kali on MK/SS808 article.

01. Create a Kali rootfs

Build yourself a Kali rootfs as described in our Kali documentation, using an **armhf** architecture. By the end of this process, you should have a populated rootfs directory in **~/arm-stuff/rootfs/kali-armhf**.

02. Create the Image File

Next, we create the physical image file which will hold our MK/SS808 rootfs and boot images.

```
apt-get install kpartx xz-utils sharutils
cd ~
mkdir -p arm-stuff
cd arm-stuff/
mkdir -p images
cd images
dd if=/dev/zero of=kali-custom-ss808.img bs=1MB count=7000
```

03. Partition and Mount the Image File

```
parted kali-custom-ss808.img --script -- mklabel msdos
parted kali-custom-ss808.img --script -- mkpart primary ext4 1 -1
```

```
loopdevice=`losetup -f --show kali-custom-ss808.img`
device=`kpartx -va $loopdevice| sed -E 's/.*(loop[0-9])p.*/\1/g' | head -1`
device="/dev/mapper/${device}"
```

```
rootp=${device}p1

mkfs.ext4 $rootp
mkdir -p root
mount $rootp root
```

04. Copy and Modify the Kali rootfs

```
rsync -HPavz /root/arm-stuff/rootfs/kali-armhf-xfce4/ root
echo nameserver 8.8.8.8 > root/etc/resolv.conf
```

05. Compile the rk3066 Kernel and Modules

If you're not using ARM hardware as the development environment, you will need to set up an ARM cross-compilation environment to build an ARM kernel and modules. Once that's done, proceed with the following steps.

```
apt-get install xz-utils
cd ~/arm-stuff
mkdir -p kernel
cd kernel

git clone git://github.com/aloksinha2001/picuntu-3.0.8-alok.git rk3066-kernel
cd rk3066-kernel
sed -i "/vpu_service/d" arch/arm/plat-rk/Makefile
```

```
export ARCH=arm
export CROSS_COMPILE=~/arm-stuff/kernel/toolchains/arm-eabi-linaro-4.6.2/bin/arm-eabi-
```

```
# A basic configuration for the UG802 and MK802 III
# make rk30_hotdog_ti_defconfig
# A basic configuration for the MK808
make rk30_hotdog_defconfig

# configure your kernel !
make menuconfig
# Configure the kernel as per http://www.armtvtech.com/armtvtechforum/viewtopic.php?f=66&t=835
mkdir ../initramfs/
wget http://208.88.127.99/initramfs.cpio -O ../initramfs/initramfs.cpio

mkdir -p ../patches
wget http://patches.aircrack-ng.org/mac80211.compat08082009.wl_frag+ack_v1.patch -O
../patches/mac80211.patch
wget http://patches.aircrack-ng.org/channel-negative-one-maxim.patch- O ../patches/negative.patch
patch -p1 < ../patches/mac80211.patch
patch -p1 < ../patches/negative.patch

./make_kernel_ruikemei.sh
```

```
make modules -j$(cat /proc/cpuinfo|grep processor|wc -l)
make modules_install INSTALL_MOD_PATH=~/arm-stuff/images/root
git clone git://git.kernel.org/pub/scm/linux/kernel/git/dwmw2/linux-firmware.git firmware-git
mkdir -p ~/arm-stuff/images/root/lib/firmware
cp -rf firmware-git/* ~/arm-stuff/images/root/lib/firmware/
rm -rf firmware-git
```

```
umount $rootp
kpartx -dv $loopdevice
losetup -d $loopdevice
```

07. dd the Image to a USB device

Use the **dd** utility to image this file to your SD card. In our example, we assume the storage device is located at /dev/sdb. **Change this as needed.**

```
dd if=kali-custom-ss808.img of=/dev/sdb bs=512k
```

Once the dd operation is complete, unmount and eject the SD card and boot your MK/SS808 into Kali Linux

Custom ODROID X2 U2 Image

The following document describes our own method of creating a **custom Kali Linux ODROID image** and is targeted at developers. If you would like to install a pre-made Kali ODROID image, check our Install Kali on ODROID article.

01. Create a Kali rootfs

Start by building a Kali rootfs as described in our Kali documentation using an **armhf** architecture. By the end of this process, you should have a populated rootfs directory in **~/arm-stuff/rootfs/kali-armhf**.

02. Create the Image File

Next, we create the physical image file which will hold our ODROID rootfs and boot images.

```
apt-get install kpartx xz-utils uboot-mkimage
cd ~
mkdir -p arm-stuff
cd arm-stuff/
mkdir -p images
cd images
dd if=/dev/zero of=kali-custom-odroid.img bs=1MB count=7000
```

03. Partition and Mount the Image File

```
parted kali-custom-odroid.img --script -- mklabel msdos
parted kali-custom-odroid.img --script -- mkpart primary fat32 4096s 266239s
parted kali-custom-odroid.img --script -- mkpart primary ext4 266240s 100%

loopdevice=`losetup -f --show kali-custom-odroid.img`
device=`kpartx -va $loopdevice| sed -E 's/.*(loop[0-9])p.*/\1/g' | head -1`
device="/dev/mapper/${device}"
bootp=${device}p1
rootp=${device}p2
mkfs.vfat $bootp
mkfs.ext4 -L kaliroot $rootp
```

```
mkdir -p boot root
mount $bootp boot
mount $rootp root
```

04. Copy and Modify the Kali rootfs

Copy over the Kali rootfs you bootstrapped earlier using **rsync** to the mounted image.

```
cd ~/arm-stuff/images/
rsync -HPavz ~/arm-stuff/rootfs/kali-armhf/ root
echo nameserver 8.8.8.8 > root/etc/resolv.conf
```

Edit the **~/arm-stuff/images/root/etc/inittab** file and locate the “Example how to put a getty on a serial line”.

```
nano root/etc/inittab
```

Add the following line to the end of that section.

```
T1:12345:respawn:/sbin/agetty 115200 ttySAC1 vt100
```

If you want the serial console to autologin as root, use the following line instead:

```
T1:12345:respawn:/bin/login -f root ttySAC1 /dev/ttySAC1 >&1
```

Now, make sure there is a *ttySAC1* entry in the **~/arm-stuff/images/root/etc/udev/links.conf** file.

```
nano root/etc/udev/links.conf
```

If an entry for *ttySAC1* doesn't already exist, add it to the file so it looks as follows:

```
M null c 1 3
M console c 5 1
M ttySAC1 c 5 1
```

Add *ttySAC* entries in the **~/arm-stuff/images/root/etc/udev/links.conf** file.

```
cat << EOF >> root/etc/securetty
ttySAC0
ttySAC1
ttySAC2
EOF
```

Place a basic xorg.conf file in the rootfs.

```
cat << EOF > root/etc/X11/xorg.conf
# X.Org X server configuration file for xfree86-video-mali

Section "Device"
Identifier "Mali-Fbdev"
```

```
# Driver "mali"
Option "fbdev" "/dev/fb1"
Option "DRI2" "true"
Option "DRI2_PAGE_FLIP" "true"
Option "DRI2_WAIT_VSYNC" "true"
Option "UMP_CACHED" "true"
Option "UMP_LOCK" "false"
EndSection

Section "Screen"
Identifier "Mali-Screen"
Device "Mali-Fbdev"
DefaultDepth 24
EndSection

Section "DRI"
Mode 0666
EndSection
EOF
```

Link **init** in the root, rootfs directory:

```
cd ~/arm-stuff/images/root
ln -s /sbin/init init
```

05. Compile the ODROID Kernel and Modules

If you're not using ARM hardware as the development environment, you will need to set up an ARM cross-compilation environment to build an ARM kernel and modules. Once that's done, proceed with the following instructions.

We next need to fetch the ODROID kernel sources and place them in our development tree structure:

```
cd ~/arm-stuff
mkdir -p kernel
cd kernel
git clone --depth 1 https://github.com/hardkernel/linux.git -b odroid-3.8.y odroid
cd odroid
touch .scmversion
```

Configure, then cross-compile the ODROID kernel.

```
export ARCH=arm
export CROSS_COMPILE=~/arm-stuff/kernel/toolchains/arm-eabi-linaro-4.6.2/bin/arm-eabi-

# for ODROID-X2
make odroidx2_defconfig
# for ODROID-U2
make odroidu2_defconfig
# configure your kernel !
make menuconfig
# and enable
CONFIG_HAVE_KERNEL_LZMA=y
CONFIG_RD_LZMA=y

# If cross compiling, run this once
sed -i 's/if defined(__linux__)/if defined(__linux__) ||defined(__KERNEL__) /g' include/uapi/drm/drm.h

make -j $(cat /proc/cpuinfo|grep processor|wc -l)
make modules_install INSTALL_MOD_PATH=~/arm-stuff/images/root/
```

Chroot into the rootfs and create an initrd. Make sure to use the correct kernel version/extraversion for the **mkinitramfs** command. In our case, it was “3.8.13”.

```
```

```
LANG=C chroot ~/arm-stuff/images/root/
apt-get install initramfs-tools uboot-mkimage
cd /
# Change the example "3.8.13" to your current odroid kernel revision
mkinitramfs -c lzma -o ./initramfs 3.8.13
mkimage -A arm -O linux -T ramdisk -C none -a 0 -e 0 -n initramfs -d ./initramfs ./uInitrd
rm initramfs
exit
```

06. Prepare the Boot Partition

Copy the kernel and generated initrd file to the mounted boot partition as shown below.

```
mv ~/arm-stuff/images/root/uInitrd ~/arm-stuff/images/boot/
cp arch/arm/boot/zImage ~/arm-stuff/images/boot/
```

Dump a **boot.txt** file, which contains required boot parameters for the ODROID in the boot partition.

```
cat << EOF > ~/arm-stuff/images/boot/boot.txt
setenv initrd_high "0xffffffff"
setenv fdt_high "0xffffffff"
setenv bootcmd "fatload mmc 0:1 0x40008000 zImage; fatload mmc 0:1 0x42000000 uInitrd; bootm
0x40008000 0x42000000"
setenv bootargs "console=tty1 console=ttySAC1,115200n8 root=LABEL=kaliroot rootwait ro
mem=2047M"
boot
EOF
```

Generate a **boot.scr** file, which is required to boot the ODROID.

```
mkimage -A arm -T script -C none -n "Boot.scr for ODROID" -d ~/arm-stuff/images/boot/boot.txt
~/arm-stuff/images/boot/boot.scr
```

Unmount the root and boot partitions, then umount the loop device.

```
cd ~/arm-stuff/images/
umount $bootp
umount $rootp
kpartx -dv $loopdevice

wget http://www.mdrjr.net/odroid/mirror/old-releases/BSPs/Alpha4/unpacked/boot.tar.gz
tar zxpf boot.tar.gz
cd boot
sh sd_fusing.sh $loopdevice
cd ..
losetup -d $loopdevice
```

Now, image the file onto your USB storage device. Our device is **/dev/sdb**. Change this as needed.

```
dd if=kali-custom-odroid.img of=/dev/sdb bs=1M
```

Once this operation is complete, connect your UART serial cable to the ODROID and boot it up with the microSD/SD card plugged in. Through the serial console, you will be able to log in to Kali (root / toor) and startx.

If everything works and you want the ODROID to start on boot, make sure to use the “autologin” line in the inittab given above and add the following to your bash_profile:

```
# If you don't have a .bash_profile, copy it from /etc/skel/.profile first
cat << EOF >> ~/.bash_profile
if [ -z "$DISPLAY" ] && [ $(tty) = /dev/ttySAC1 ]; then
startx
fi
EOF
```

08. Install Mali Graphic Drivers (Optional)

These steps are experimental and not fully tested yet. They should be preformed inside the Kali rootfs.

```
# http://malideveloper.arm.com/develop-for-mali/drivers/open-source-mali-gpus-linux-exadri2-and-x11-display-drivers/
apt-get install build-essential autoconf automake make libtool xorg xorg-dev xutils-dev libdrm-dev
wget http://malideveloper.arm.com/downloads/drivers/DX910/r3p2-01rel0/DX910-SW-99003-r3p2-01rel0.tgz
wget http://malideveloper.arm.com/downloads/drivers/DX910/r3p2-01rel0/DX910-SW-99006-r3p2-01rel0.tgz
wget --no-check-certificate https://dl.dropbox.com/u/65312725/mali_opengl_hf_lib.tgz

tar -xzvf mali_opengl_hf_lib.tgz
cp mali_opengl_hf_lib/* /usr/lib/

tar -xzvf DX910-SW-99003-r3p2-01rel0.tgz
tar -xzvf DX910-SW-99006-r3p2-01rel0.tgz
cd DX910-SW-99003-r3p2-01rel0/x11/xf86-video-mali-0.0.1/
./autogen.sh
chmod +x configure

CFLAGS="-O3 -Wall -W -Wextra -I/usr/include/libdrm
-IDX910-SW-99006-r3p2-01rel0/driver/src/ump/include" LDFLAGS="-L/usr/lib -lMali -lUMP -lpthread"
./configure --prefix=/usr --x-includes=/usr/include --x-libraries=/usr/lib
cp -rf ../../../DX910-SW-99006-r3p2-01rel0/driver/src/ump/include/ump src/
mkdir -p umplock
cd umplock
wget
```

```
http://service.i-onik.de/a10_source_1.5/lichee/linux-3.0/modules/mali/DX910-SW-99002
-r3p0-04rel0/driver/src/devicedrv/umplock/umplock_ioctl.h
cd ..

make
make install
```

07. Kali Community Support

Submitting Bugs for Kali Linux

Introduction

This article is a guide for putting together a bug report so that it gets addressed as quickly as possible.

First, Kali Linux is a labor of love, born out of a desire to give back to the community, a community that we're a part of. In our development roles, it's our goal to continually improve and evolve the project, making things better for the entire community of Kali Linux users. The developers who provide support to you are volunteers doing so out of altruism. Kali Linux is their gift to you. Please keep this in mind when making your comments.

Second, the goal of a successful bug report is to enable the Kali Linux developers to reproduce the issue and see the failure, if any. If the Kali development team can reproduce the reported failure, they can proceed to gather extra information until the root cause is determined. If the failure can't be reproduced, the development team will request additional information until they can reproduce the results reported by the submitter.

Please note, submissions are best read by our team in **English**.

Help us help you! To give us the best start in getting your issue resolved:

- Supply all the information you can. Try to stick to what's relevant, but if you're uncertain, too much is better than too little.
- Keep your bug report objective and try to stick to the facts at hand.
    - Be very clear about what *is* a fact — document these whenever possible, via logs, scrollback captures, etc. — and what is a hypothesis on your part.
    - Do not quote Wikipedia and other non-primary resources as "facts" in your submission. What's happening on your system is what's at issue, not what Wikipedia claims
- Do not stack multiple issues into a single report; submit additional reports as needed. One person should submit one report, for one bug, on one particular hardware combination. Trying to stack multiple variations into a single report makes any specific issue in there very difficult to tease out. What seem like similar bugs to you may in fact turn out to be unrelated.
- If one of the developers asks for additional information, please do your best to understand what's being asked for and provide it in a reasonable time. If you're not sure you understand what you're being asked for, ask for clarification, we'll do our best to provide more guidance. Do not post comments that are unhelpful such as "Me too!" or "+1".
- Do not complain about how long it takes to fix a bug. Remember: the developers are volunteers with day jobs, which are not fixing your bugs for you.

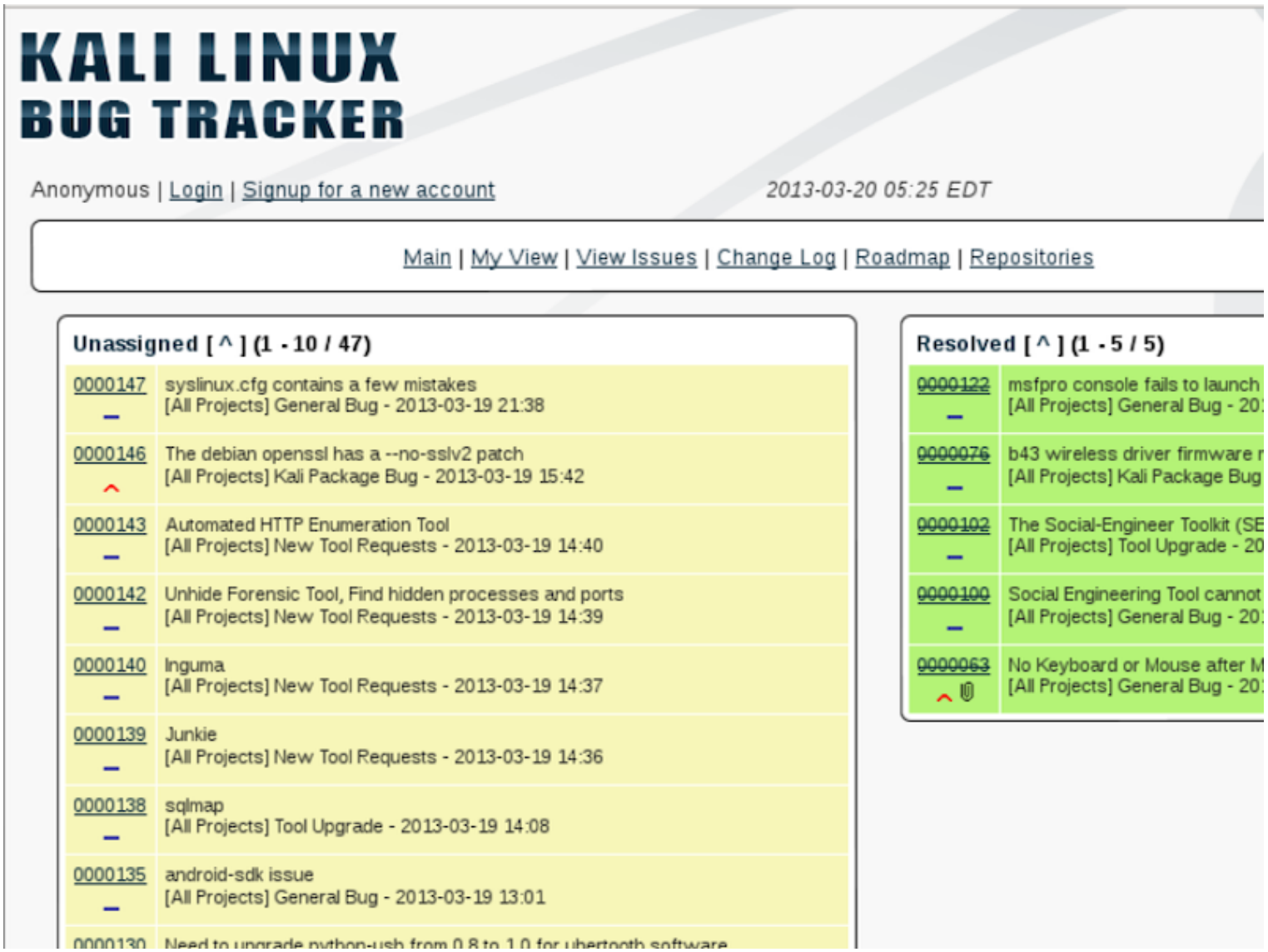
How to Report a Bug

The Kali Linux Bug Tracker can be found at https://bugs.kali.org. This section will guide you through signing up for a new account, creating a system profile, and creating a detailed bug report for submission to the Bug Tracker.

Signing Up For a Bug Tracker Account

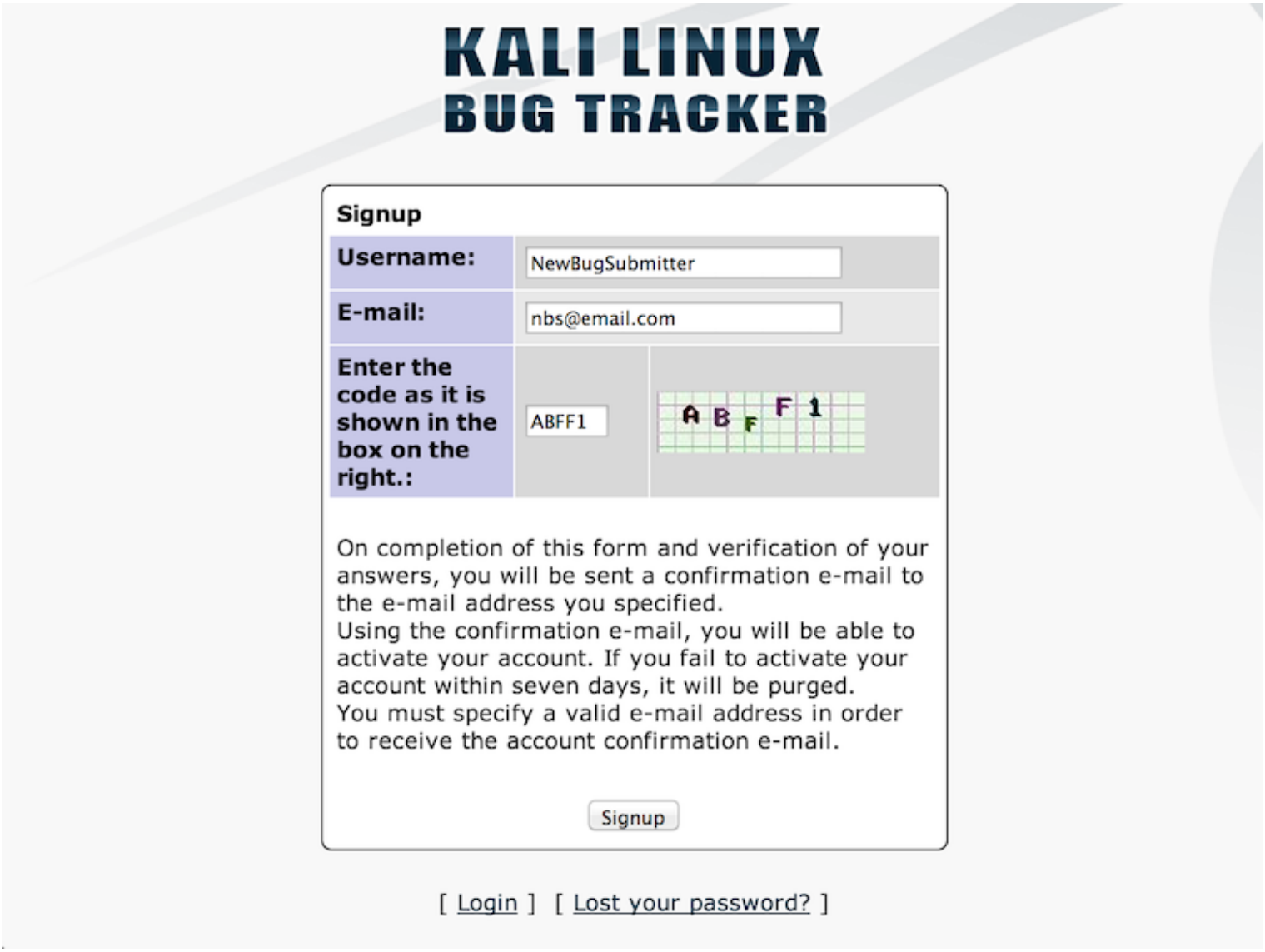
You'll have to create an account before the Bug Tracker will allow you to submit reports or comment on existing ones.

On the bug tracker website, click **Signup for new account**.



Provide a username, e-mail address, and respond to the CAPTCHA challenge. Click the **Signup** button to

proceed.



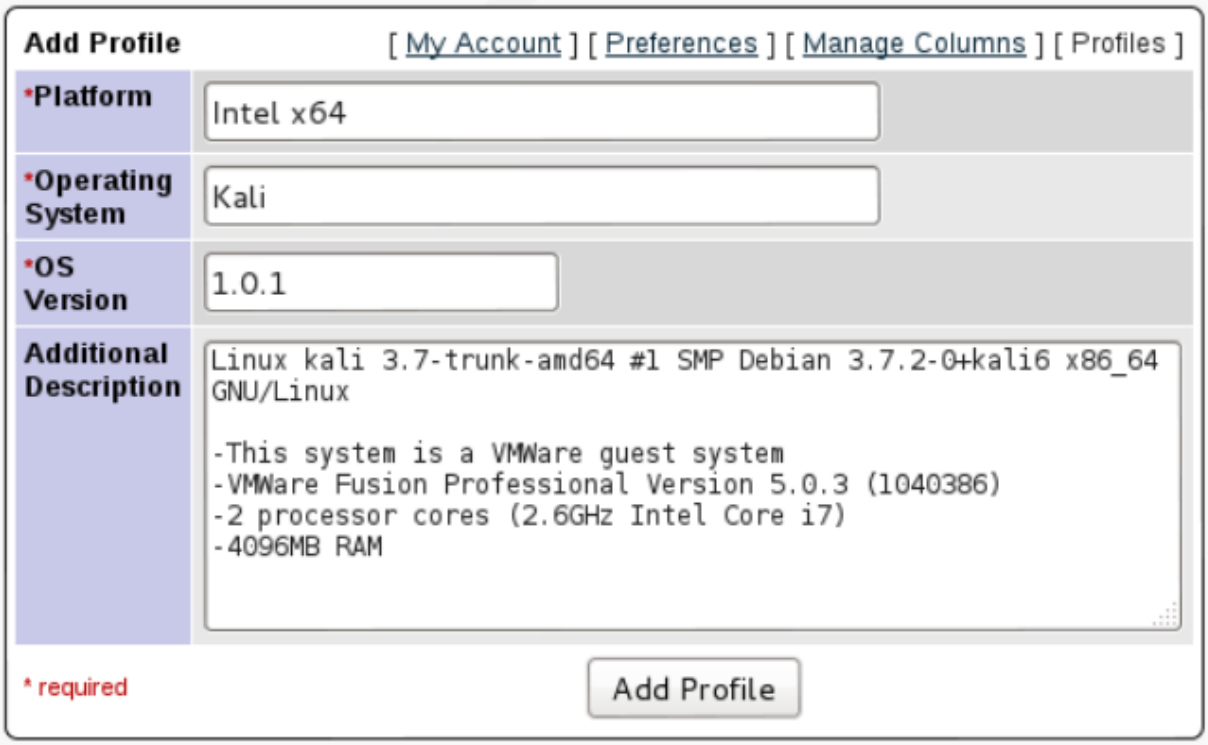
If successful, the next page will notify you that the account registration has been processed and the bug tracker system will send a confirmation email to the address you provided. You will need to visit the link in the email in order to activate your account.

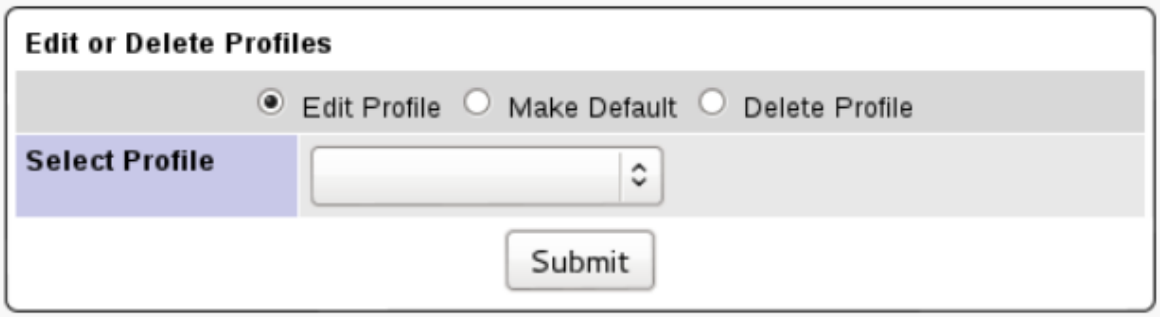
Once your account has been activated, click **Proceed** to continue to the Bug Tracker login page.

Creating a Profile in the Kali Linux Bug Tracker

Although not required, we recommend you create a unique profile as part of your Bug Tracker account. Profiles are shortcuts that predefine values for your CPU platform, operating system and version, as well as allowing you to provide some additional information, all of which is automatically submitted as part of your bug report. You can create a custom profile for each Kali system you're using or select from the default profiles provided.

To create or edit a custom profile, select **My Account** from the main page and then select **Profiles**. Add the specific information and description for your system and click the **Add Profile** button when done.





Once the profile has been added, it will appear in the **Select Profile** drop-down list when you report a new issue. Create as many different profiles as you need, just be sure to select the appropriate one when submitting your bug report, or a lot of confusion may result.

Be Sure You Are Not Duplicating a Previous Report

Before starting your report, search the site for keywords related to your issue. If there is already an existing bug not related to hardware, please do not duplicate the request or add notes that provide no new information or are otherwise unnecessary (e.g. “Me Too” or “+1”). If the bug has already been reported, you can view the status of any progress toward resolving the issue by clicking the ID link.

However, if you believe the issue to be hardware related, please submit a new report with your *specific* information, even if it appears similar. There is a strong chance that your hardware does not exactly match that of another reporter. Do not assume that just because you have the same desktop or laptop model that your issue is not unique.

Creating the Report

To begin your report, log into your account and click the **Report Issue** link on the landing page. You will need to provide as much information as you possibly can. If unsure, review the pointers at the beginning of this document.

The following fields are *mandatory* within the report:

- Category
- Summary
- Description

Even though the other fields are not mandatory, we recommend you try to include as much information as possible within each option while paying special attention to the following fields:

- Reproducibility — How reproducible is this bug? Always? Only sometimes? Only under specific circumstances?
- Select Profile — We need to know what you're running, and what you're running it *on.*
- Steps to Reproduce — Be very clear here, and provide as much concrete detail as you can.
- Additional Information
- Upload File (error logs, screenshot)

Decide the Proper Category

There are currently four (4) categories available in the Kali bug tracker. Before you begin your request, ensure it is properly designated for one of the following:

- General Bug
- Kali Package Bug
- New Tool Requests
- Tool Upgrade

Do not request support or ask questions within the bug tracker. Kali Linux offers several options for support including https://docs.kali.org , https://forums.kali.org, and our IRC chat room (***#kali-linux*** on freenode)

Providing a Descriptive Summary

The summary field is essentially the 'title' of the bug report and it will be the first thing Kali developers and other visitors see. Provide a short, yet descriptive, summary that describes the issue or request.

A good summary: Chromium Package installed from repo will not run as root user

A bad summary: Chromium doesn't work

The summary does not need to include everything, but it should convey your reason for submitting the report.

Using dpkg to Find the Package and Version for the Report

You can find which package is installed using a combination of dpkg flags. It is important to include relevant information from the output of these commands in your report. The output can also be placed in a text file and uploaded. (Discussed later within this document.)

- dpkg –search
- dpkg –list
- dpkg –status

Sample Output

```
root@kali:~# which chromium
/usr/bin/chromium
root@kali:~# type chromium
chromium is /usr/bin/chromium
root@kali:~# dpkg --search /usr/bin/chromium
chromium: /usr/bin/chromium
root@kali:~# dpkg --list chromium
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name           Version      Architecture Description
+++-==============-============-============-=================================
ii  chromium       55.0.2883.75-3 amd64        Google open source chromium web
root@kali:~# dpkg --status chromium
Package: chromium
Status: install ok installed
Priority: optional
Section: web
Installed-Size: 160895
Maintainer: Debian Chromium Maintainers <pkg-chromium-maint@lists.alioth.debian.org>
```

```
Architecture: amd64
Source: chromium-browser
Version: 55.0.2883.75-3
...Output Truncated...
```

Building the Description Scenario

This is your opportunity to to help us out and provide a well thought-out description of the problem you're experiencing. Please provide as many details and facts as possible.

Please ensure you include the following where applicable:

- *Exact and complete text* of any error messages (screen output or log files)
- *Exactly* what you typed or what actions you took to produce the issue
- A suggested fix, workaround, or patch if you are able to produce one
- The version of the package having the problem, and any information relating to dependent packages
- The kernel version, shared C library, and any other details that seem appropriate
- The output of the command `uname -a`
- The output of the command `dpkg -s libc6 | grep ^Version`
- If applicable, software version — (i.e. `python -V`, etc.)
- Details of your hardware
- If you are reporting an issue with a device driver, please provide full details on all hardware in your system — for a complete report on your system install `lshw` from the repos.
- Add any other details that seems relevant
- Do not worry about the report being "too long" — as long as the information is relevant, include it.

Here's an example of a good bug report, providing information that the development team can immediately use to reproduce and try to understand the bug:

Package: Chromium

Architecture: amd64

Maintainer: Debian Chromium Maintainers

Source: chromium-browser

Version: 55.0.2883.75-3

I installed the chromium web browser from the Kali Linux repos, using the command ‘apt install chromium’. I launched the program from the Kali menu by selecting Applications/Internet/Chromium Web Browser. Chromium did not launch as expected, instead it provided an error pop-up window.

The error message stated, “Chromium cannot be run as root. Please start Chromium as a normal user. To run as root, you must specify an alternate –user-data-dir for storage of profile information”.

I clicked the Close button to close the pop up window.

uname –a output: Linux kali 4.7.0-kali1-amd64 #1 SMP Debian 4.7.6-1kali1 (2016-10-17) x86_64 GNU/Linux

C Library Version: 2.24-8

The Importance of Reproducibility

The Kali Linux bug tracker allows you to provide the frequency of the issue being reported. If you are submitting a request for a new tool or an upgrade to an existing tool, simply select **N/A** from the drop down options. If submitting a bug, please provide the appropriate response.

Continuing the example above, by design, Chromium will not launch as root, so you would select ‘always’ from the drop-down menu.

It is extremely important you provide an accurate response. If the Kali developers attempt to reproduce the issue, they need to know the frequency. If the issue happens occasionally but you have marked ‘always’, the issue may be closed prematurely as the developer doing the testing may not experience the issue.

Selecting the Proper Profile

As discussed above, using a custom-defined profile is best for each issue reported. If custom profiles are not created, select the appropriate “standard” profile from the drop-down menu. At the time of this writing, the following options are available.

- armel Kali 1.0
- armhf Kali 1.0
- x64 Kali 1.0
- x86 Kali 1.0

Providing Steps to Reproduce the Issue

Although this may seem redundant when compared with the description section, this section should *only* include the steps taken to reproduce the issue. Some steps may seem unnecessary, but it is important to ensure you document the process as well as you can. The missing step may be the one that's key to reproducing the issue.

Here's an example of a good set of steps for reproducing our example Chromium bug.

1. Opened a terminal window by selecting Applications/Accessories/Terminal
2. Typed 'apt install chromium' in the terminal and hit enter to run the command
3. Attempted to run Chromium web browser by selecting Applications/Internet/Chromium Web Browser

Providing Additional Information

In this section, you can provide any additional information you believe is relevant to the issue. If you have a fix or workaround for the issue, please provide it in this section. Again, it is important to stick to the facts and document the steps clearly so the developers can reproduce the issue.

An example of some useful "Additional Information":

There is a simple fix that is well documented on several forums. I tried it and it fixed the issue for me.

- Using a text editor open /etc/chromium/default
- Add –user-data-dir flag
- i.e. CHROMIUM_FLAGS="–user-data-dir"

Can this be patched within the repo version of Chromium so adding this flag is not required for future releases?

Uploading Relevant Files

Sometimes it is important to provide information to the development team that can't easily be typed in as text. This section of the report allows you to add screenshots and log files. Be mindful of the size limitation in place.

You can add a file by clicking the 'Choose File' button. This will open the file manager for your system and allow you to select the file you want to attach to your report. Once you have selected the file, click the 'Open' button to return to your report and click the 'Upload File' button.

Submitting the Report

At this point, you are ready to submit the report. All that is left to do is click the ‘Submit Report’ button. Your report will be submitted and assigned a tracking ID. The report will show up on your ‘My View’ page under ‘Reported by Me’. This will allow you to track the issue to resolution.

Summary

Bug reports help the Kali Linux development team see the failure with their own eyes. Since they cannot be with you to experience the problem you’re having, you must provide instructions detailed enough that they can make their own systems fail themselves.

Describe everything in detail, stating the steps taken, what you saw, what you did, as well as the expected outcome.

Attempt to find an issue or fix through research, if at all possible — remember, open source development is a participatory process! If you are able to provide a solution to fix the issue for your system, provide the developers with the same level of detail as you did when reporting the bug. It is important that the developers know *exactly* what you did, so they can successfully repeat the process. This should not stop you from filing a full explanation of the symptom that caused the unexpected behavior.

Write accurately, be clear, precise, and concise to ensure the developers cannot misinterpret what you are trying to convey.

Be prepared to provide additional information; the developers will not ask if they don’t need the information.

Please be patient with your request, the developers want to fix your issue as much as you do. We love what we do and are proud to continue making Kali Linux the most advanced penetration testing distribution ever, and grateful for the assistance we get from you, our community of users, in doing so.

© Offensive Security 2019

Kali Linux Community Forums

The official community forums for the Kali Linux project are located at forums.kali.org.

It's our goal that everyone feel welcome in the Kali Linux community, and to ensure that everyone understands the expectations, we have outlined some simple rules below. Please take a few moments to review them before joining the forums.

Forum Rules

By registering with our forums you agree to be bound by the following rules.

- We do not condone any illegal activity at all.
- Any advice or information offered in the forums is to be used for legal informational, professional, or educational purposes.
- New registrants' posts will be moderated at first, causing a slight delay in the post appearing – please ***do not*** report problems with your post not appearing instantly during your first three days of membership.
- Please use sensible and descriptive titles for your posts – titles like "Please Help Me!!" or "Need Assistance" or "What Am I Doing Wrong?" etc., don't give forum members any idea what the issue might be and are unlikely to draw either interest or assistance.
- Do *not* cross-post across sub-forums, please – one post in the relevant area is enough!
- Before creating a new thread, please search the forums for similar or related previous postings. If you *do* create a new thread asking a question that has already been asked, don't be surprised if your thread gets deleted without notice.
- ***Do not post*** about breaking into networks that do not belong to you and for which you have no permissions. ***Do not post*** about illegal activities.
- Religious, political, or pornographic references will *not* be tolerated. Stay on topic.
- Posts about your success in "hacking" into your neighbours WiFi or queries about how to break into a network, etc., are not welcome and will vanish (quite possibly along with your user ID).
- Spam messages they will be summarily removed, and you will be summarily banned if you posted them.
- Members signatures may **not** contain URLs or web links, in *any* form.
- We *will not tolerate* abusive, sexist, racist, or any other derogatory remarks, nor members acting like self-appointed moderators. The forum staff are responsible for moderation, and they're here to help you. Please use their services.
- If ANY member has an issue with the content of ANY post within the forums, use the **REPORT THIS POST**" button – This is the red triangle icon when using the default forum theme (or the asterisk icon when using the Blackfire Razor forum theme) found in the *top right corner* of each post.
- Breaking the forum rules may incur infractions ranging from loss of posting privileges to a temporary or permanent ban, at the sole discretion of the forum staff.
- These rules are subject to alteration and/or addition. You are responsible for staying aware of any

changes.

Kali Linux IRC Channel

Kali Linux has an official IRC channel, **#kali-linux**, on the Freenode network. Please take a few moments to review the rules and guidelines below before joining the channel.

#kali-linux IRC Rules and Guidelines

We try to remain as informal as possible but there *are* some rules and we'd appreciate if you would follow them! Broadly, if you're friendly, tolerant, and reasonable, you'll probably go a long way without any specific knowledge of the rules – but for to avoid any doubt and prevent any misunderstandings, here they are.

How to Treat Other Users

In order to make the channel a pleasant place for all of our users, we expect all to remain as friendly and tolerant as possible. We request that you refrain from profanity and that you show respect to the other channel members and visitors. If you find that you're becoming frustrated with the channel or other users, we encourage you to take a moment to do something else. Try to ensure you don't make people feel like you're just taking advantage of them – help others out while you're waiting for a reply to your questions, and say thanks!

How to Argue

As mentioned above, we'd appreciate it if you'd strive to be friendly and tolerant. We also encourage debates and in-depth discussions about topical subjects. If you choose to participate in one, we expect you to remain as reasonable as possible and employ the skills of logic and critical thinking. These skills will serve you well in discussion, enable you to communicate more efficiently, and spot when others are being less than forthcoming with the truth!

Language

Our IRC channel is an English-speaking channel. There are no other official channels in any other language.

We are a *family-friendly* IRC channel and will not tolerate foul language. Save it for the playground.

Staying on Topic

We maintain no strict policy regarding off-topic chat in the channel however, the discussion of Kali Linux projects is the primary focus of the channel, so you may be asked to take discussions elsewhere, particularly if there are venues on **freenode** better suited to them (such as **#politics**), if there are other more relevant conversations going on, or if they're repetitive or otherwise seen by the channel staff as being detrimental to the good atmosphere of the channel.

Certain things are seen as being specifically off-topic. These topics include:

- **Support or encouragement of illegal activity** – it should go without saying, but we don't exist to help you break the law or do things you shouldn't be doing. Such queries are absolutely off-topic for the channel, for freenode as a whole, and may very well get you removed from the channel and/or network. Please don't ask. Laws vary from country to country and channel OPs may determine whether a specific discussion is appropriate for the channel or not.
- **Warez/cracks/pirated software** – these too are off-topic for the channel and network so again: please don't ask.
- **Political and religious matters** — Many people have very strong political/religious beliefs and we respect that. We also recognize that these are volatile and contentious topics which have nothing to do with Kali Linux, penetration testing, or anything related to those subjects, and are best discussed elsewhere.

Asking for Help

If you're asking for help, first off, thanks! – questions and the resulting discussion of the answer(s) in a collaborative environment are what make IRC great and by helping to add to the atmosphere, you benefit the entire Kali Linux community.

We often find that we learn a lot even from questions we already think we know the answers to – about people, alternative approaches, and cool new resources and tools. However, if you are intending to ask a question, we'd appreciate it if you'd follow a couple useful guidelines to help you, and us, make the best use of our time.

- **Do your research first** — It's very frustrating when people ask a question that can virtually be answered by punching the keywords into a Google search! We also have forums and a wiki that contain answers to many questions we see daily so it's to everyone's benefit if these assets are used before asking in IRC.
- **Give us the whole picture** — If you're asked for more information, please provide it accurately. The correct answer will depend on it. Looking at this from another angle: the more we learn about your problem, the more this independently benefits us too – a large part of the development of new releases comes out of helping others with issues discovered with specific setups; even if you're asking us questions, you can help teach us something too!
- **If you find the answer somewhere else, tell us** – it isn't compulsory, but if you don't get an answer to your question in the channel but you find it elsewhere, consider letting us know! That way, we can help out the next person with a similar question. It also lets people know that you already have an answer you're happy with, or that if anyone's researching the question for you, they can stop.
- **Wait for an answer** – not everyone in the channel is online all the time, and you may find you get an answer several minutes, or even hours, later. Feel free to stick around and chat, or even answer other peoples questions – you'll find it helps pass the time and makes others likely to help you! Help us build a

community of friendly security professionals and enthusiasts.

Spam, Flooding, and Various Other Forms of Disruptive Behaviour

Spam, flooding, disrespect or verbal attacks against other users, misleading links, intentionally wrong answers, and other forms of disruptive behaviour not otherwise specified are unwelcome. Disruptive behaviour includes (but is not restricted to) the operation of unauthorized bots in the channel, public logging of the channel, and scripts such as those that publicly announce what track your MP3 player is playing or your away status.

If you have more than 5 lines of text to paste, use pastebin for your data and then paste the URL into channel.

Dealing With the Channel Staff

From time to time, you may be asked to take conversations elsewhere, treat others reasonably, steer a conversation in a particular direction, or a variety of other things in order to preserve the ambiance and usefulness of the channel. If you're the target of such a request, please be as reasonable as you can and if you wish to take issue with it, do so in a private message with the channel staffer in question, rather than making noise in channel.

Discipline

Repeated breaking of the rules will cause channel staffers to mute (+q), ban (+b), kick, or otherwise remove you from the channel. This will particularly apply if you're seen to be willfully ignoring the rules after we've drawn your attention to them.

Many forms of disruptive behaviour, such as flooding or trolling, may result in discipline without a warning. We try and avoid the use of force wherever possible and we'd appreciate it if you'd help us in pursuing this goal!

If you're a bystander while a staffer is forced to use his or her powers for channel management, we'd appreciate your understanding and consideration in awaiting the end of the incident, and your assistance in keeping the situation as favourable as possible by not complaining, commentating, or gloating. This serves to make antisocial behaviour such as flooding less attractive (the smaller the reaction, the less the return on the malfeasance), and so benefits you as well as us!

Thanks for your cooperation and help in making the #kali-linux a more enjoyable, friendlier, and more productive experience for everyone involved.

Official Kali Linux Sites

The Kali Linux project uses several different subdomains of kali.org, each with a specific purpose. This article lists the official Kali sites and the purpose each one of them serves. *Note that these sites are the **only** official Kali Linux sites* and are *the only authoritative sources of information* available for the distribution.

The sites listed below are the **ONLY** official outlets for the Kali Linux Distribution.

Public Websites

- www.kali.org
- docs.kali.org
- forums.kali.org
- bugs.kali.org
- gitlab.com/kalilinux
- tools.kali.org
- pkg.kali.org

The main Kali Linux website is our primary means of communicating news about the Kali Linux project, general introductory information, and general updates about the project and its ongoing development.
Blog posts about new tools, features, Kali Linux tips, tricks, and tutorials can be found here. This should be *your one and only source* to download the official Kali Linux distribution.

Where you are right now. The Kali Linux documentation site contains a basic set of Kali Linux-related documentation and tutorials, which we continually work to update and improve.
For Backtrack users, the changes introduced in Kali are substantial and we have tried to address a broad range of common issues.
Sub-domains of docs.kali.org are also considered official — these are our document translation servers, e.g. fr.docs.kali.org is our French-language documentation site.

If you run into an issue or situation that isn't directly covered in the official Kali Linux documentation, there is a

good chance that a member of the Kali Linux Forums will know the answer. The Kali forum has members from all over the world, covering the entire range of skill levels, and are open and willing to help newcomers who are willing to learn.
More information on the Kali Linux Forums can be found in this article.

Despite our best efforts at making Kali Linux perfect, bugs and errors are inevitable. We are always open to improvement and can only effectively do so when issues or tools suggestions are reported to us. You are encouraged to submit bug reports at bugs.kali.org to help us make Kali Linux even better.
More information on submitting bug reports to the Kali Linux project can be found in this article.

This site is the Kali Linux project's official git repository and is publicly accessible. Most users will never need to interact with it directly, but users who wish to more closely monitor the development of Kali Linux, or people who want to know when they should run 'apt full-upgrade', can peruse or clone the repository.

The core of Kali Linux is the comprehensive toolset, drawn from many difference sources in the security and forensics software community. The tools site provides both an up-to-date listing of the tools which are available in Kali Linux, as well as provinding a quick reference to each of them. From here, the versions of the tools can be tracked against their upstream sources.

The Kali Linux Package Tracker site allows you to follow the evolution of Kali Linux both with email updates and a comprehensive web interface. The tracker can also help in identifying which versions of various tools and packages are in our repository at any given moment.

Social Media

We're on social media — follow us on Twitter, "like" our official Facebook page to keep up with important announcements and news.

- twitter
- facebook

We don't tweet a lot but when we do, it's important. Information on releases and blog posts will be pushed to our twitter account, @KaliLinux.

As with our Twitter account, we won't overwhelm you with information on our Kali Facebook page but when we do post, it will be worth it.

Official Kali Linux Mirrors

Using Official Repositories

The Kali Linux distribution has two repositories, which are mirrored world-wide:

- http.kali.org (mirrorlist): the main package repository;
- cdimage.kali.org (mirrorlist): the repository of pre-built Kali ISO images.

When using the default hosts listed above, you'll automatically be redirected to a mirror site which is geographically close to you, and which is guaranteed to be up-to-date. If you prefer to manually select a mirror, click on the ***mirrorlist*** link near each hostname above and select a mirror that suits you. You will then need to edit your `/etc/apt/sources.list` file accordingly with the chosen values.

> ***IMPORTANT!*** **Do not add** additional repositories to your **/etc/apt/sources.list** file.Doing so will most likely **break** your Kali installation.

How to Set Up a Kali Linux Mirror

Requirements

To be an official Kali Linux mirror, you will need a web-accessible server**(http required and https if possible too)** with lots of disk space, good bandwidth, rsync, and SSH access enabled. As of early 2015, the main package repository is about 450 GB and the ISO images repository is about 50 GB but you can expect those numbers to grow regularly. A mirror site is expected to make the files available over HTTP and RSYNC so those services will need to be enabled. FTP access is optional.

Note on "Push Mirroring" — The Kali Linux mirroring infrastructure uses SSH-based triggers to ping the mirrors when they need to be refreshed. This currently takes place 4 times a day.

Create a User Account for the Mirror

If you don't have yet an account dedicated for the mirrors, create such an account (here we call it "archvsync"):

```
$ sudo adduser --disabled-password archvsync
Adding user 'archvsync' ...
```

```
[...]
Is the information correct? [Y/n]
```

Create Directories for the Mirror

Create the directories that will contain the mirrors and change their owner to the dedicated user that you just created:

```
$ sudo mkdir /srv/mirrors/kali{,-images}
$ sudo chown archvsync:archvsync /srv/mirrors/kali{,-images}
```

Configure rsync

Next, configure the rsync daemon (enable it if needed) to export those directories:

```
$ sudo sed -i -e "s/RSYNC_ENABLE=false/RSYNC_ENABLE=true/" /etc/default/rsync
$ sudo vim /etc/rsyncd.conf
$ cat /etc/rsyncd.conf
uid = nobody
gid = nogroup
max connections = 25
socket options = SO_KEEPALIVE

[kali]
path = /srv/mirrors/kali
comment = The Kali Archive
read only = true

[kali-images]
path = /srv/mirrors/kali-images
comment = The Kali ISO images
read only = true
```

```
$ sudo service rsync start
Starting rsync daemon: rsync.
```

Configure Your Mirror

Configuration of your web server and FTP server are outside the scope of this article. Ideally, you should export the mirrors at http://yourmirror.net/kali and http://yourmirror.net/kali-images (and do the same for the FTP protocol, if you're supporting it).

Now comes interesting part: the configuration of the dedicated user that will handle the SSH trigger and the actual mirroring. You should first unpack ftpsync.tar.gz in the user's account:

```
$ sudo su - archvsync
$ wget http://archive.kali.org/ftpsync.tar.gz
$ tar zxf ftpsync.tar.gz
```

Now we need to create a configuration file. We start from a template and we edit at least the *MIRRORNAME*, *TO*, *RSYNC_PATH*, and *RSYNC_HOST* parameters:

```
$ cp etc/ftpsync.conf.sample etc/ftpsync-kali.conf
$ vim etc/ftpsync-kali.conf
$ grep -E '^[^#]' etc/ftpsync-kali.conf
MIRRORNAME=`hostname -f`
TO="/srv/mirrors/kali/"
RSYNC_PATH="kali"
RSYNC_HOST=archive.kali.org
```

Set Up the SSH Keys

The last step is to setup the .ssh/authorized_keys file so that archive.kali.org can trigger your mirror:

```
$ mkdir -p .ssh
$ wget -O - -q http://archive.kali.org/pushmirror.pub >>.ssh/authorized_keys
```

If you have not unpacked the ftpsync.tar.gz in the home directory, then you must adjust accordingly the "~/bin/ftpsync" path, which is hard-coded in .ssh/authorized_keys.

Now you must send an email to devel@kali.org with all the URLs of your mirrors so that you can be added in the main mirror list and to open up your rsync access on archive.kali.org. Please indicate clearly who should be contacted in case of problems (or if changes must be made/coordinated to the mirror setup).

Instead of waiting for the first push from archive.kali.org, you should run an initial rsync with a mirror close to you, using the mirror list linked above to select one. Assuming that you picked archive-4.kali.org, here's what you can run as your dedicated mirror user:

```
$ rsync -qaH archive-4.kali.org::kali /srv/mirrors/kali/ &
$ rsync -qaH archive-4.kali.org::kali-images /srv/mirrors/kali-images/ &
```

Set Up cron to Manually Mirror ISO Images

The ISO images repository does not use push mirroring so you must schedule a daily rsync run. We provide a bin/mirror-kali-images script, which is ready to use that you can add in the crontab of your dedicated user. You just have to configure etc/mirror-kali-images.conf.

```
$ sudo su - archvsync
$ cp etc/mirror-kali-images.conf.sample etc/mirror-kali-images.conf
$ vim etc/mirror-kali-images.conf
$ grep -E '^[^#]' etc/mirror-kali-images.conf
TO=/srv/mirrors/kali-images/
$ crontab -e
```

```
$ crontab -l
# m h dom mon dow command
39 3 * * * ~/bin/mirror-kali-images
```

Please adjust the precise time so that archive.kali.org doesn’t get overloaded by too many mirrors at the same time.

08. Kali Linux Policies

Kali Linux Root User Policy

Most Linux distributions, quite sensibly, encourage the use of a non-privileged account while running the system and use a utility like `sudo` when and if escalation of privileges in needed. This is sound security advice: this provides an extra layer of protection between the user and any potentially disruptive or destructive operating system commands or operations. This is especially true for multiple user systems, where user privilege separation is a requirement — misbehavior by one user can disrupt or destroy the work of many users.

Kali Linux, however, as a security and auditing platform, contains many which tools can only run with root privileges. Further, Kali Linux's nature makes its use in a multi-user environment highly unlikely.

For these reasons, the default Kali user is "root", and no non-privileged user is created as a part of the installation process. This is one reason that Kali Linux is not recommended for use by Linux beginners who might be more apt to make destructive mistakes while running with root privileges.

Kali Network Service Policies

Kali Linux is a penetration testing toolkit, and may potentially be used in “hostile” environments. Accordingly, Kali Linux deals with network services in a very different way than typical Linux distributions. Specifically, Kali *does not enable any externally-listening services by default* with the goal of minimizing exposure when in a default state.

Default Disallow Policy

Kali Linux, as a standard policy, will *disallow network services from persisting across reboots by default*.

The following example can be seen when attempting to install a tool which would by default would start a network proxy service on TCP port 3142:

```
root@kali:~# apt-get install apt-cacher-ng
...
Setting up apt-cacher-ng (0.7.11-1) ...
update-rc.d: We have no instructions for the apt-cacher-ng init script.
update-rc.d: It looks like a network service, we disable it.
...
root@kali:~#
```

Notice how the update-rc.d script disallowed persistence of the apt-cacher-ng daemon by default.

Overriding the Default Policy

In certain situations, you may actually want certain services to persist over reboots. To allow for this, you can enable a service to persist through reboots using the systemctl command as follows:

```
root@kali:~# systemctl enable apt-cacher-ng
Synchronizing state of apt-cacher-ng.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apt-cacher-ng
insserv: warning: current start runlevel(s) (empty) of script `apt-cacher-ng' overrides LSB defaults (2 3 4 5).
insserv: warning: current stop runlevel(s) (0 1 2 3 4 5 6) of script `apt-cacher-ng' overrides LSB defaults (0
```

```
1 6).
```

Service whitelists and blacklists

Service whitelists and blacklists can be found in the **/usr/sbin/update-rc.d** file. You can edit this file to explicitly allow or deny services the ability to automatically start up at boot time.

```
root@kali:~# tail -95 /usr/sbin/update-rc.d |more
...
__DATA__
#
# List of blacklisted init scripts
#
apache2 disabled
avahi-daemon disabled
bluetooth disabled
cups disabled
dictd disabled
ssh disabled
...
#
# List of whitelisted init scripts
#
acpid enabled
acpi-fakekey enabled
acpi-support enabled
alsa-utils enabled
anacron enabled
...
```

Kali Linux Update Policies

The majority of the packages comprising the Kali Linux distribution are drawn directly from the Debian repositories. For those packages which have been incorporated into Kali Linux “as-is” — i.e. the vast majority — security updates arrive at essentially the same time for Kali Linux as for the main Debian distribution.

Other packages are supported on a best-effort basis by the Kali Linux development team.

Penetration Testing Tools Policy

Kali Linux Tools Policy

One of the key tasks in transitioning from Backtrack Linux to Kali was combing through the packages and selecting the “best of breed” from what was available.

We realize that there are many tools or scripts that can do the same job. Some are clearly better than others in some respect, some are more a matter of personal preference. With this in mind, keeping an updated, useful penetration testing tool repository is a challenging task. The Kali Development team uses some of these questions to help decide whether a specific tool should be included in Kali Linux.

- Is the tool useful/functional in a Penetration Testing environment?
- Does the tool overlap functionality of other existing tools?
- Does the licensing of the tool allow for free redistribution?
- How much resources does the tool require? Will it work in a “standard” environment?

The answers to questions such as these, among other considerations, help us come to a decision whether the tool should be included in Kali.

Most of the members of the Kali development team are working penetration testers, and we rely on our combined experience and expertise to select the best tools to add the most value to the Kali distribution as we continue its development.

Tools which are specifically aimed at DOS, DDOS or anonymity are *rarely used in legitimate engagements*, and are therefore *not installed by default* in Kali Linux.

New Tool Requests

We are always open to adding new and better tools to our distribution, but we ask that a case be made for each tool. Please put some thought and effort into the tool submission, and please do *not* just send the developers a one line request. Submissions for new tool requests can be made through our Kali Linux bug tracker.

Kali's Relationship With Debian

The Kali Linux distribution is based on Debian Testing. Therefore, most of the Kali packages are imported, as-is, from the Debian repositories. In some cases, newer packages may be imported from Debian Unstable or Debian Experimental, either to improve user experience, or to incorporate needed bug fixes.

Forked Packages

In order to implement some of Kali's unique features, we had to fork some packages. The Kali development team strives to keep such packages to a minimum by improving the upstream packages whenever possible, either by integrating the feature directly, or by adding the required hooks so that it's straightforward to enable the desired features without further modifying the upstream packages themselves.

Each package forked by Kali is maintained in a Git repository with a "debian" branch so that updating a forked package can be easily done with a simple *git merge debian* in its master branch.

Additional Packages

Beyond this, Kali incorporates many additional packages which are specific to the penetration testing and security auditing field. The majority of these packages constitute "free software" according to Debian's Free Software Guidelines. Kali intends to contribute those packages back to Debian and to maintain them directly within Debian.

To facilitate this, Kali packaging strives to comply with the Debian Policy and follow the best practices in use in Debian.

Kali Linux Open Source Policy

Kali Linux is a Linux distribution that aggregates thousands of free software packages in its `main` section. As a Debian derivative, all of the core software in Kali Linux complies with the Debian Free Software Guidelines.

As the specific exception to the above, Kali Linux's `non-free` section contains several tools which are not open source, but which have been made available for redistribution by Offensive Security through default or specific licensing agreements with the vendors of those tools.

If you want to build a Kali derivative, you should *review the license* of each Kali-specific non-free package before including it in your distribution — but note that non-free packages which are imported from Debian are safe to redistribute.

More importantly, all of the specific developments in Kali Linux's infrastructure or its integration with the included software have been put under the GNU GPL.

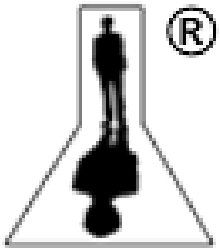
If you want more information about the license of any given piece of software, you can either check `debian/copyright` in the source package or `/usr/share/doc/`*`package`*`/copyright` for a package that you have already installed.

Kali Linux Trademark Policy

Kali Linux and Offensive Security want to promote the widespread recognition of our trademarks among the Internet community however, we also need to ensure our trademarks solely identify our company and our products. At the heart of our trademark policy is **trust** – we want to avoid the public from being confused into believing they are dealing with Kali Linux and/or Offensive Security when, in fact, they are not. This is of particular importance with regards to the development and distribution of trusted penetration testing distribution such as Kali Linux.

This document identifies and the describes our trademarks and provides guidance as to their fair use. We are generally quite accommodating when it comes to fair and honest use of our trademarks so if you are so inclined, feel free to contact us for further guidance.

Some of our Trademarks



Use in Print, Web, Media and Public Display

It is important to maintain the look and spelling of the trademarks. Please do not modify the marks. Examples of modifying the marks include abbreviating names, adding logos to the marks, or combining the marks with other words. We recommend you use the trademarks in the exact form as we use them.

The Offensive Security trademarks are to designate the source of our products and services. We encourage others to use the marks so long as they are used to identify the products and services of Offensive Security. We do not want to confuse the public into believing that they are dealing with us, when in fact, they are not.

The first mention of an Offensive Security trademark should be accompanied by a symbol indicating whether the mark is a registered trademark "®" or an unregistered trademark "™". Please refer to the above list for the appropriate symbol to use and if in doubt, use "™".

The use of an Offensive Security trademark should be set apart from surrounding text, either by capitalizing it or by italicizing, bolding or underlining it. The Offensive Security trademarks are to designate the source of our products and services.

When using an Offensive Security trademark in written materials, you should provide a statement indicating that the [trademark] is a trademark of Offensive Security. For example:

"KALI LINUX ™ is a trademark of Offensive Security." This statement can be provided directly in your text, or as a footnote or an endnote.

The use of Offensive Security trademarks in your domain names is prohibited because such use will lead to the confusion of customers. Any other use outside of the scope of the Trademark Policy is not permitted without express written permission of Offensive Security.

You may make t-shirts, desktop wallpaper, or other merchandise with Offensive Security Marks on them, though only for yourself and your friends (meaning people from whom you don't receive anything of value in return). You can't put the trademarks on anything that you produce commercially (whether or not you make a profit) — at least not without receiving written permission.

Contact

If you have any questions or comments, or wish to report misuse of the Offensive Security trademarks, please contact us.

09. The Kali Linux Dojo

01 - Meet the Kali Team

In this workshop, we will introduce the Kali Linux security auditing distribution, its history, development, architecture, and features. We will delve into how we got to where we are today, how penetration-testing tools are evaluated for inclusion in the distribution, and a look at the road ahead. If we have enough time, we'll also dabble with altering and patching Kali packages.

Many of our examples and exercises will involve pulling packages from Kali Linux repositories. To alleviate Internet network traffic, we have brought with us some local repositories. To force your Kali installation to use these repositories, modify your /etc/hosts file to include the local repository IP address with the following DNS entries.

192.168.1.201 archive.kali.org http.kali.org security.kali.org repo.kali.org

02 - Building Custom Kali ISOs

Building Custom Kali ISOs

One of the most powerful features of Kali Linux is the ability to create your own flavors of the distribution containing customized tools, desktop managers, and services. This workshop will show you how to create your own personalized Kali Linux ISO, customizing virtually every aspect using the live-build utility and making efficient use of the various meta-packages available in Kali.

The Awesomeness of Live Build

0x00 – Begin by updating the repos, installing the prerequisites, and checking out a fresh version of live-build-config from the Kali Git repositories:

```
apt update
apt install git live-build cdebootstrap devscripts -y
git clone git://gitlab.com/kalilinux/build-scripts/live-build-config.git
cd live-build-config
```

0x01 – Overwrite the default Kali package list, including only the packages you want. In the video, we simply edited the list and changed a few package names.

```
cat > kali-config/variant-default/package-lists/kali.list.chroot << EOF
kali-root-login
kali-defaults
kali-menu
kali-debtags
kali-archive-keyring
debian-installer-launcher
alsa-tools
locales-all
dconf-tools
openssh-server
EOF
```

0x02 – Add a customised syslinux boot entry which includes a boot parameter for a custom preseed file.

```
cat << EOF > kali-config/common/includes.binary/isolinux/install.cfg
label install
    menu label ^Install Automated
    linux /install/vmlinuz
    initrd /install/initrd.gz
    append vga=788 -- quiet file=/cdrom/install/preseed.cfg locale=en_US keymap=us hostname=kali
domain=local.lan
EOF
```

0x03 – Customise the ISO build. In this example, we'll have the SSH service start by default. To do this, we can use a chroot hook script which is placed in the "hooks" directory:

```
echo 'systemctl enable ssh' >>  kali-config/common/hooks/01-start-ssh.chroot
chmod +x kali-config/common/hooks/01-start-ssh.chroot
```

0x04 – Next, we download a wallpaper and overlay it. Notice how chroot overlayed files are placed in the *includes.chroot* directory.

```
mkdir -p kali-config/common/includes.chroot/usr/share/wallpapers/kali/contents/images
wget https://www.kali.org/dojo/bh2015/wp-blue.png
mv wp-blue.png kali-config/common/includes.chroot/usr/share/wallpapers/kali/contents/images
```

0x05 – Add a preseed file that will run through a default Kali installation with no input (unattended). We can include a ready made preseed configuration and alter it as needed:

```
mkdir -p kali-config/common/debian-installer
wget
https://raw.githubusercontent.com/offensive-security/kali-linux-preseed/master/kali-linux-full-
unattended.preseed -O kali-config/common/debian-installer/preseed.cfg
```

0x06 – Let's include a Nessus Debian package into the *packages* directory for inclusion into our final build. Since we used a 64 bit build, we're including a 64 bit Nessus Debian package. Download the Nessus .deb file and place it in the packages.chroot directory:

```
mkdir kali-config/common/packages.chroot
mv Nessus-*amd64.deb kali-config/common/packages.chroot/
```

0x07 – Now you can proceed to build your ISO , this process may take a while depending on your hardware and internet speeds. Once completed, your ISO can be found in the live-build root directory.

```
./build.sh -v
```

For more live-build implementations, refer to the following:

- http://www.offensive-security.com/kali-linux/kali-linux-recipes/
- https://gitlab.com/kalilinux/recipes/live-build-config-examples

03 - Kali Linux USB Persistence

USB Persistence & Encrypted Persistence

In this workshop, we will examine the various features available to us when booting Kali Linux from USB devices. We will explore features such as persistence, creating LUKS encrypted persistence stores, and even dabble in “LUKS Nuking” our USB drive. The default Kali Linux ISOs (from 1.0.7 onwards) support USB encrypted persistence.

0x01 – Start by imaging the Kali ISO onto your USB stick (ours was /dev/sdb). Once done, you can inspect the USB partition structure using *parted /dev/sdb print*.

```
dd if=kali-linux-2016.2-amd64.iso of=/dev/sdb bs=1M
```

0x02 – Create and format an additional partition on the USB stick. In our example, we create a persistent partition of about 7 GB in size:

```
root@kali:~# parted
GNU Parted 2.3
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.

(parted) print devices
/dev/sda (480GB)
/dev/sdb (31.6GB)

(parted) select /dev/sdb
Using /dev/sdb

(parted) print
Model: SanDisk SanDisk Ultra (scsi)
Disk /dev/sdb: 31.6GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
```

```
Number  Start   End     Size    Type     File system  Flags
 1      32.8kB  2988MB  2988MB  primary               boot, hidden
 2      2988MB  3050MB  64.9MB  primary  fat16

(parted) mkpart primary 3050 10000
(parted) quit
Information: You may need to update /etc/fstab.
```

0x04 – Encrypt the partition with LUKS:

```
cryptsetup --verbose --verify-passphrase luksFormat /dev/sdb3
```

0x05 – Open the encrypted partition:

```
cryptsetup luksOpen /dev/sdb3 my_usb
```

0x06 – Create an ext3 filesystem and label it.

```
mkfs.ext3 /dev/mapper/my_usb
e2label /dev/mapper/my_usb persistence
```

0x07 – Mount the partition and create your persistence.conf so changes persist across reboots:

```
mkdir -p /mnt/my_usb
```

```
mount /dev/mapper/my_usb /mnt/my_usb
echo "/ union" > /mnt/my_usb/persistence.conf
umount /dev/mapper/my_usb
cryptsetup luksClose /dev/mapper/my_usb
```

Now your USB stick is ready to plug in and reboot into Live USB Encrypted Persistence mode.

Multiple Persistence Stores

At this point we should have the following partition structure:

```
root@kali:~#  parted /dev/sdb print
```

We can add additional persistence stores to the USB drive, both encrypted or not...and choose which persistence store we want to load, at boot time. Let's create one more additional non-encrypted store. We'll label and call it "work".

0x01 – Create an additional, 4th partition which will hold the "work" data. We'll give it another 5GB of space.

```
root@kali:~# parted /dev/sdb
GNU Parted 2.3
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) print
Model: SanDisk SanDisk Ultra (scsi)
Disk /dev/sdb: 31.6GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos

Number  Start   End     Size    Type     File system  Flags
 1      32.8kB  2988MB  2988MB  primary               boot, hidden
```

```
 2     2988MB  3050MB  64.9MB  primary  fat16
 3     3050MB  10.0GB  6947MB  primary

(parted) mkpart primary 10000 15000
(parted) quit
Information: You may need to update /etc/fstab.
```

0x02 – Format the fourth partition, label it “work”.

```
mkfs.ext3 /dev/sdb4
e2label /dev/sdb4 work
```

0x03 – Mount this new partition and create a persistence.conf in it:

```
mkdir -p /mnt/usb
mount /dev/sdb4 /mnt/usb
echo "/ union" > /mnt/usb/persistence.conf
umount /mnt/usb
```

Boot the computer, and set it to boot from USB. When the boot menu appears, edit the persistence-label parameter to point to your preferred persistence store!

Emergency Self Destruction of Data in Kali

As penetration testers, we often need to travel with sensitive data stored on our laptops. Of course, we use full disk encryption wherever possible, including our Kali Linux machines, which tend to contain the most sensitive materials.

```
```

```
root@kali:~# cryptsetup luksAddNuke /dev/sdb3
Enter any existing passphrase:
Enter new passphrase for key slot:
```

Now dump the keyslots to see the changes:

```
root@kali:~# cryptsetup luksDump /dev/sda5
Device /dev/sda5 doesn't exist or access denied.
root@kali:~# cryptsetup luksDump /dev/sdb3
LUKS header information for /dev/sdb3

Version:        1
Cipher name:    aes
Cipher mode:    xts-plain64
Hash spec:      sha1
Payload offset: 4096
MK bits:        256
MK digest:      f7 17 b9 a7 9f 7f 9b 21 f2 b9 40 78 c2 97 f5 f0 c2 bb 28 8b
MK salt:        f5 a4 80 02 e7 21 0d 7e 5a 64 f4 96 78 a3 15 3c
                09 7b 3f 41 80 2b 5c bf c5 de 92 70 69 bb 34 b2
MK iterations:  64500
UUID:           96793acb-c2d3-45b7-aed9-1af952386556

Key Slot 0: ENABLED
  Iterations:             258064
  Salt:                   df 3c d6 03 4a 78 ce ef 62 fd f1 56 25 d4 c5 96
                          2a 12 bb 94 4b d7 cf c1 0a b5 27 47 09 ae 31 46
  Key material offset:    8
  AF stripes:             4000
Key Slot 1: ENABLED
  Iterations:             259108
  Salt:                   30 07 ff ef fc f5 74 65 04 f7 66 87 77 f1 74 4f
                          7d 2f 76 e2 71 e7 6a 9c 6d c1 c1 7b 80 53 cb c1
  Key material offset:    264
  AF stripes:             4000
Key Slot 2: DISABLED
```

```
Key Slot 3: DISABLED
Key Slot 4: DISABLED
Key Slot 5: DISABLED
Key Slot 6: DISABLED
Key Slot 7: DISABLED
root@kali:~#
```

Backup you LUKS keyslots and encrypt them:

```
cryptsetup luksHeaderBackup --header-backup-file luksheader.back /dev/sdb3
openssl enc -d -aes-256-cbc -in luksheader.back.enc -out luksheader.back
```

Now boot into your encrypted store, and give the Nuke password, rather than the real decryption password. This will render any info on the encrypted store useless. Once this is done, verify that the data is indeed inacessible.

Lets restore the data now. We'll decrypt our backup of the LUKS keyslots, and restore them to the encrypted partition:

```
openssl enc -d -aes-256-cbc -in luksheader.back.enc -out luksheader.back
cryptsetup luksHeaderRestore --header-backup-file luksheader.back /dev/sdb3
```

Our slots are now restored. All we have to do is simply reboot and provide our normal LUKS password and the system is back to its original state.

04 – Raspberry Pi Disk Encryption

With the advent of smaller, faster ARM hardware such as the new **Raspberry Pi 2 (or even 3!)** (which now has a Kali image built for it), we've been seeing more and more use of these small devices as "**throw-away hackboxes**". While this might be a new and novel technology, **there's one major drawback** to this concept – and that is the **confidentiality of the data** stored on the device itself. Most of the setups we've seen do little to protect the sensitive information saved on the SD cards of these little computers. This fact, together with a nudge from friends is what prompted us to create a LUKS encrypted, NUKE capable Kali Linux image for our Raspberry Pi devices. The following article describes the process, so you can repeat it and make your own shiny shiny.

Birds Eye View of the Disk Encryption Process

The process described below was tried and **tested successfully on a Raspberry Pi B+ and a Raspberry Pi 2/3** (henceforth collectively called "RPi"). but it should be trivial to port these instructions to any ARM device running Kali. Before we begin, let's take a minute to quickly describe what we'll be doing – as while this process is not complicated, it **is involved**. This is basically our spiel:

1. We download the required Kali RPi image and **dd** it to an SD card.
2. We chroot to the RPi image and install/update several files in preparation for our crypted boot.
3. We create an initramfs file which includes Dropbear and freshly generated SSH keys.
4. We rsync the modified rootfs to a temporary backup location and then delete the rootfs partition from the SD card.
5. We then recreate an encrypted partition to which we restore the root partition data. That's it!

If all goes well, the RPi will boot and then LUKS will kick in and ask for a password to decrypt the root drive, while simultaneously opening a Dropbear SSH session through which **you can SSH in and provide the boot decryption password**. Oh yeah, did we mention this image also has **LUKS NUKE capabilities**?

Getting Your Hands Dirty

As always, all our ARM dev is done on a Kali amd64 machine and we've made sure that we have all the dependencies we need. We download the latest Kali RPi3 image (2017.1), extract it, and **dd** it to our SD card, which in our case showed up as /dev/sdb2 – adapt as necessary!

```
dd if=/root/kali-2017.3-rpi3.img of=/dev/sdb bs=4M
```

Once dd'd, we mount the various partitions and chroot into the Kali RPi3 image:

```
mkdir -p /mnt/chroot/boot

mount /dev/sdb2 /mnt/chroot/
mount /dev/sdb1 /mnt/chroot/boot/

mount -t proc none /mnt/chroot/proc
mount -t sysfs none /mnt/chroot/sys
mount -o bind /dev /mnt/chroot/dev
mount -o bind /dev/pts /mnt/chroot/dev/pts
apt-get install qemu-user-static

cp /usr/bin/qemu-arm-static /mnt/chroot/usr/bin/
LANG=C chroot /mnt/chroot/
```

We then update our image and install some essential packages we will need for this process:

```
apt-get update
apt-get install busybox cryptsetup dropbear-initramfs
```

We create an initial initramfs file, which will trigger the dropbear SSH key generation. We first find out the modules directory version number as follows (this will change between different image versions and Kali releases):

```
root@kali:/# ls -l /lib/modules/ |awk -F" " '{print $9}'
4.9.59-Re4son-Kali-Pi+
```

We then use that version info to generate an initial initramfs file.

```
mkinitramfs -o /boot/initramfs.gz 4.9.59-Re4son-Kali-Pi+
```

We change the default root password.

```
passwd
```

Next, we modify the boot parameters in cmdline.txt and config.txt.

```
nano /boot/cmdline.txt
```

...and add / change the following parameters:

```
root=/dev/mapper/crypt_sdcard cryptdevice=/dev/mmcblk0p2:crypt_sdcard rootfstype=ext4
```

Next create or add to /boot/config.txt:

```
echo initramfs initramfs.gz >> /boot/config.txt
```

Now we deal with the Dropbear SSH access. We copy over SSH private key from our laptop, or, create one specifically for doing this:

Copying:

```
cp /root/.ssh/id_rsa.pub /etc/dropbear-initramfs/authorized_keys
chmod 0600 /etc/dropbear-initramfs/authorized_keys
```

Creating (on the host machine, **NOT** in the chroot:

```
ssh-keygen -N "" -f kali-luks-unlock
cat kali-luks-unlock.pub > /mnt/chroot/etc/dropbear-initramfs/authorized_keys
chmod 0600 /mnt/chroot/etc/dropbear-initramfs/authorized_keys
```

And limit the SSH connection to allow interaction with the cryptroot application only.

```
nano /etc/dropbear-initramfs/authorized_keys
```

We paste the following **before** the ssh public key begins.

```
command="/scripts/local-top/cryptroot && kill -9 `ps | grep -m 1 'cryptroot' | cut -d ' ' -f 3`"
```

Then to ensure we get cryptsetup in the initramfs, we edit the cryptsetup initramfs hook.

```
echo "CRYPTSETUP=y" >> /etc/cryptsetup-initramfs/conf-hook
```

We then edit fstab and crypttab with our configured boot device and exit the chroot:

```
cat > /etc/fstab <<EOF
proc /proc proc defaults 0 0
/dev/mmcblk0p1 /boot vfat defaults 0 2
/dev/mapper/crypt_sdcard / ext4 defaults,noatime 0 1
EOF

echo crypt_sdcard /dev/mmcblk0p2 none luks > /etc/crypttab
```

During our tests, we noticed that in some instances, the USB ports take a while to wake up, which can kill the initrd Dropbear network initialization. To fix this, we introduce a 5-second sleep in the configure_networking function located in the initrd itself:

```
nano /usr/share/initramfs-tools/scripts/functions
```

change:

```
configure_networking()
{
...
```

to:

```
configure_networking()
{

echo "Waiting 5 seconds for USB to wake"
sleep 5
...
```

Note: Do NOT add the "…" they are a placeholder to mean there is more stuff there, that we aren't editing.

Save the file then regenerate the initramfs and exit the chroot. You can ignore the cryptsetup and device-mapper warnings.

```
mkinitramfs -o /boot/initramfs.gz 4.9.59-Re4son-Kali-Pi+
exit
```

Now we proceed to tear down the chroot and backup our rootfs partition:

```
umount /mnt/chroot/boot
umount /mnt/chroot/sys
umount /mnt/chroot/proc
umount /mnt/chroot/dev/pts
umount /mnt/chroot/dev
mkdir -p /mnt/backup
rsync -avh /mnt/chroot/* /mnt/backup/
```

Once the backup is done, we unmount everything:

```
umount /mnt/chroot
```

Now we delete the existing 2nd partition on the SD card and recreate an empty one, which we will set up for LUKS encryption.

```
echo -e "d\n2\nw" | fdisk /dev/sdb
echo -e "n\np\n2\n\n\nw" | fdisk /dev/sdb
```

Unplug your SD card and plug it back in to have the new partitions register, then start setting up your encrypted partition.

```
cryptsetup -v -y --cipher aes-xts-plain64 --key-size 256 luksFormat /dev/sdb2
cryptsetup -v luksOpen /dev/sdb2 crypt_sdcard
mkfs.ext4 /dev/mapper/crypt_sdcard
```

Once ready, we restore the rootfs backup to the now encrypted partition.

```
mkdir -p /mnt/encrypted
mount /dev/mapper/crypt_sdcard /mnt/encrypted/
rsync -avh /mnt/backup/* /mnt/encrypted/
umount /mnt/encrypted/
rm -rf /mnt/backup
sync
```

Then we unmount and close the volume.

```
cryptsetup luksClose /dev/mapper/crypt_sdcard
```

That’s it!

Now all that remains is to boot up the RPi using the modified SD card. The initramfs will load Dropbear and get a DHCP address on your local LAN (you can also hardcode an IP), allowing you to SSH to the booting RPi and enter a decryption password. Once the password is accepted, Dropbear will exit and the RPi will continue to boot. You should see something like the following:

```
root@kali:~# ssh -i key 192.168.13.37
The authenticity of host '192.168.13.37 (192.168.13.37)' can't be established.
RSA key fingerprint is a6:a2:ad:7d:cb:d8:70:58:d1:ed:81:e8:4a:d5:23:3a.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.13.37' (RSA) to the list of known hosts.
Unlocking the disk /dev/mmcblk0p2 (crypt_sdcard)
Enter passphrase: cryptsetup: crypt_sdcard set up successfully
Connection to 192.168.13.37 closed.
root@kali:~#
```

Can I Have Some LUKS NUKE With That Pi?

If you are not familiar with the [Kali Linux LUKS NUKE] (https://www.kali.org/how-to/nuke-kali-linux-luks/) feature then you’re missing out. Although this stage is optional, it allows you to configure and apply an emergency self-destruct password to your LUKS encrypted drive. To do this, we simply define a Nuke password on our encrypted partition:

```
root@kali:~# cryptsetup luksAddNuke /dev/sdb2
Enter any existing passphrase: (existing passphrase)
```

Enter new passphrase for key slot: (new nuke passphrase)
root@kali:~#
With the Nuke password defined, you can now remotely wipe the LUKS decryption keyslots, making the data on the SD card inaccessible.

Raspberry Pi Disk Encryption Video

In order to give a bit more visual context to the process, we made a short video which shows the sequence of commands used to get LUKS disk encryption working on a Raspberry Pi B+. Enjoy!

Setting up LUKS disk encryption on a Raspberry Pi running Kali Linux. Also supports LUKS Nuke features!

Kali Dojo 04 – Kali on a Raspberry Pi with LUKS Disk Encryption from Offensive Security.

References

We came up with this procedure by cannibalising ideas and instructions from various sources on the net, most notably, the two below. Big thanks to the Raspberry Pi community!

1. https://www.ofthedeed.org/posts/Encrypted_Raspberry_Pi/
2. http://www.raspberrypi.org/forums/viewtopic.php?f=28&t=7626

04 - Deploying Kali over PXE/iPXE

Kali Linux supports several interesting installation and deployment options, which will be explored in depth. In this workshop, we'll show you how to deploy Kali over the network with PXE and iPXE technologies, pre-seed installations, deploy custom Kali setups.

05 - Kali Linux on Android

Kali Linux supports ARMEL and ARMHF architectures. This allows us to put Kali on a variety of interesting hardware platforms, as well as easily conduct chroot installs of Kali Linux on Android. In this workshop, we will show you how to install Kali Linux within a chroot environment on an Android device. We will also introduce and demonstrate Kali Linux NetHunter.

06 - Kali as a Hardware Backdoor

This will be the concluding session, where we'll use methods demonstrated throughout the day to show you how to create your very own "Raspberry Pi of Doom", based off Kali Linux and a Raspberry Pi.