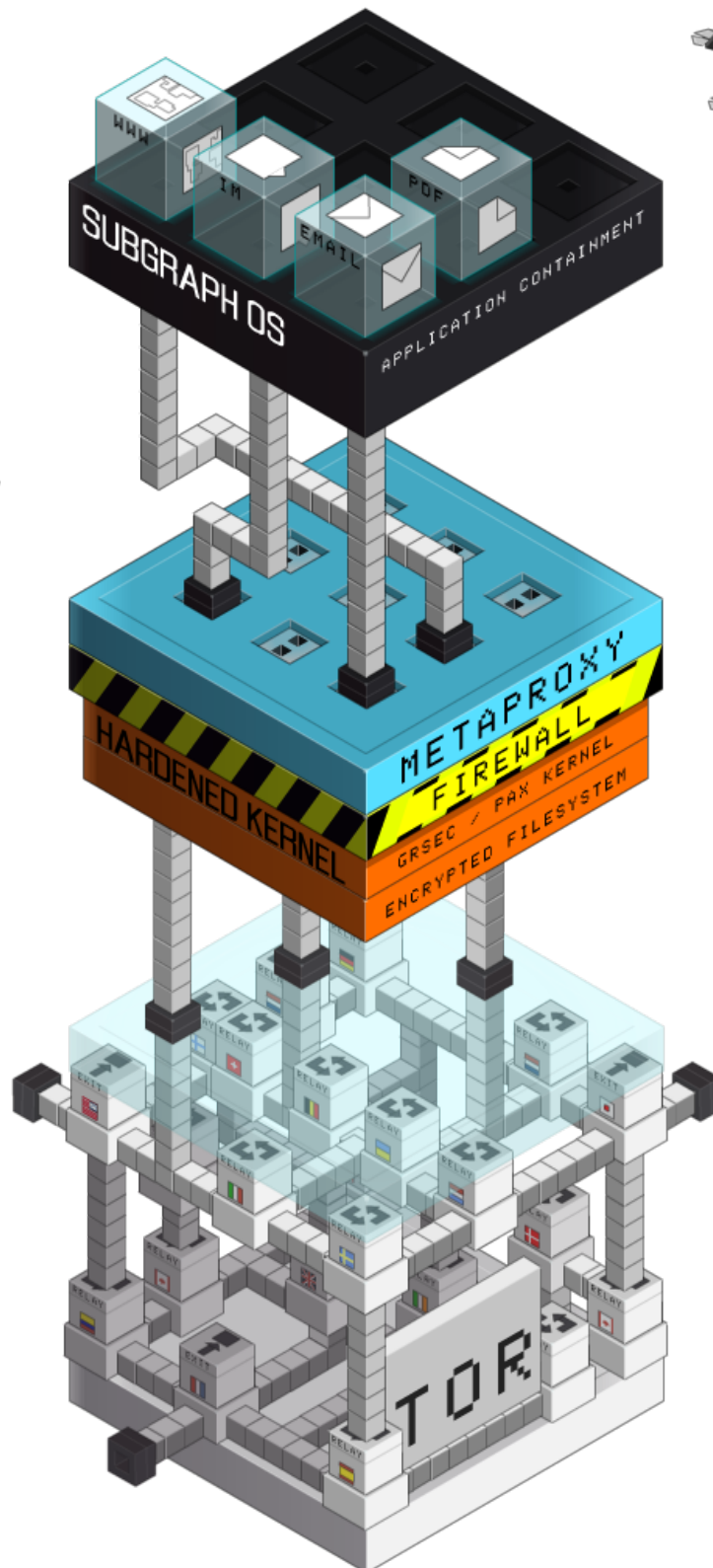


SUBGRAPH OS

H A N D B O O K



Subgraph OS Handbook

Contents

Preface	5
Subgraph OS	6
What is Subgraph OS?	7
What do we mean by security and privacy?	8
What is adversary resistant computing?	8
Installing Subgraph OS	10
Downloading and verifying the Subgraph OS ISO	11
Installation methods	12
Installing from a CD or DVD	13
Installing from a USB key	14
Booting from a USB key (Live mode)	15
Installing, step-by-step	17
After the First Boot	18
Everyday usage	19
Browsing the Web with Tor Browser	20
Configuring the Tor Browser security slider	20
Downloading and saving files in the Tor Browser	21
Uploading files in the Tor Browser	21
Viewing PDFs	24
Chatting with CoyIM	25
Adding an XMPP account to CoyIM	25
Chatting over Tor with Ricochet	26
Sharing files with OnionShare	29

Monitoring outgoing connections with Subgraph Firewall	30
Features and advanced usage	31
Sandboxing applications with Subgraph Oz	31
Securing sandboxed applications with Oz seccomp	32
Anonymizing communications with Tor	33
Securing the Tor control port with roflcoptor	34
Routing applications through Tor with Subgraph Metaproxy	35
Hardening the operating system and applications with Grsecurity	36
Configuring PaX flags with Paxrat	36
Anonymizing MAC addresses with Macouflage	37
Preventing unauthorized USB access with USB Lockout	38
Using virtual machines in Subgraph OS	39
Creating a virtual machine with Qemu.	39

Preface

We have created this handbook as an instructional manual on how to use the Subgraph operating system. This handbook also introduces various security and privacy enhancing technologies that we have developed at Subgraph.

This handbook is intended for new users of Subgraph OS, including users who are already familiar with other Linux-based operating systems. It also includes users who are just getting started with Linux. We hope that this handbook helps to ease the transition for both groups of users.

The first section of this book describes how to perform common tasks such as installing Subgraph OS and using the various applications that are included. Users who want to get up and running as quickly as possible should start here.

In the next section, we describe the various features of Subgraph OS that distinguish it from other operating systems. This is intended for users who are interested in learning in-depth about the various security and privacy enhancements that we have implemented. Advanced users will find this section useful for configuring operating system features and Subgraph applications.

Subgraph OS

What is Subgraph OS?

Subgraph OS is an adversary resistant computing platform. The main purpose of Subgraph OS is to empower people to communicate, share, and collaborate without fear of surveillance and interference. What this means in practical terms is that users of Subgraph OS can safely perform their day-to-day tasks securely and privately.

In some ways, Subgraph OS is like other operating systems – it is derived from Debian GNU/Linux and uses the GNOME desktop environment as its graphical user interface. Many applications found in other Linux distributions are also available in Subgraph OS. Therefore, users who are already familiar to Linux and particularly the GNOME desktop environment will find Subgraph OS easy to use.

Subgraph OS also has key differences from conventional Linux operating systems. In particular:

1. Subgraph OS anonymizes Internet traffic by sending it through the Tor network
2. Subgraph OS is hardened against common security vulnerabilities
3. Subgraph runs many desktop applications in a security sandbox to limit their risk in case of compromise

What do we mean by security and privacy?

Security and privacy can have many different meanings. In computer security, a secure system is one that assures the **confidentiality, integrity, and availability** of information it stores, processes, or communicates.

Confidentiality assures that information is not revealed to anybody who is not authorized

Integrity assures that information cannot be modified or tampered with by anybody who is not authorized

Availability assures that information can be reliably accessed by those who are authorized

Privacy is similar to confidentiality. Privacy also relies heavily on the integrity of communications. But we think about it in a broader sense as well.

We believe it as a fundamental right for people to communicate with each other privately. We also believe that people should not be required to reveal information about themselves or their social network without explicit consent. This is an ongoing challenge because our devices and the applications we use reveal a great deal of information about us. Additionally, they communicate this private information over networks that are untrustworthy and hostile.

We designed Subgraph OS with these privacy concerns in mind.

What is adversary resistant computing?

Subgraph OS has been designed from the ground up to defend against threats to security and privacy. We aim to provide our users with a computing platform that is **adversary resistant**.

When we use the term **adversary**, we are referring to an actual or hypothetical threat to the confidentiality, integrity, and availability of information.

A hacker exploiting a vulnerability in an application is one type of adversary. This is an actual and often active threat to security and privacy.

Passive or indirect threats also exist. A passive or indirect threat may be an adversary conducting surveillance by intercepting and analyzing network traffic.

Lastly, adversaries may present a theoretical or impractical threat. For example, a cryptography algorithm may have a theoretical weakness. As technology and attack methods improve, the weakness is no longer theoretical and may be practically exploited.

We use the term **adversary** to cover all of the above possibilities.

Secure systems should be resistant to all of these types of threats.

While no computing platform can anticipate and defend against all possible threats by all possible adversaries, we aspire to make such attacks extremely difficult for adversaries. And in making these attacks difficult, they also become more expensive for adversaries to scale.

Some of our users have critical security and privacy needs. Subgraph OS affords them strong security and privacy to conduct their activities safely. Casual users are afforded the same security and privacy without having to sacrifice the usability and maintainability afforded by other operating systems.

This is the adversary resistant computing paradigm.

Installing Subgraph OS

Downloading and verifying the Subgraph OS ISO

Installation methods

Installing from a CD or DVD

Installing from a USB key

Booting from a USB key (Live mode)

Subgraph OS also features a 'live' mode. Subgraph OS live mode runs in memory, directly from the USB stick. While running in live mode, nothing will be saved to your hard-drive. When the live session ends, any data created during your session will disappear, leaving no traces behind on the hard-disk.

People normally run in live mode for the following reasons:

1. They want to demo Subgraph OS
2. They want to test Subgraph OS with their particular hardware
3. They want to perform certain tasks with extra security and privacy but do not want a permanent installation of Subgraph OS

When the Subgraph OS ISO starts, you will be presented with different options. To start the live mode, select `Live (amd64)`.

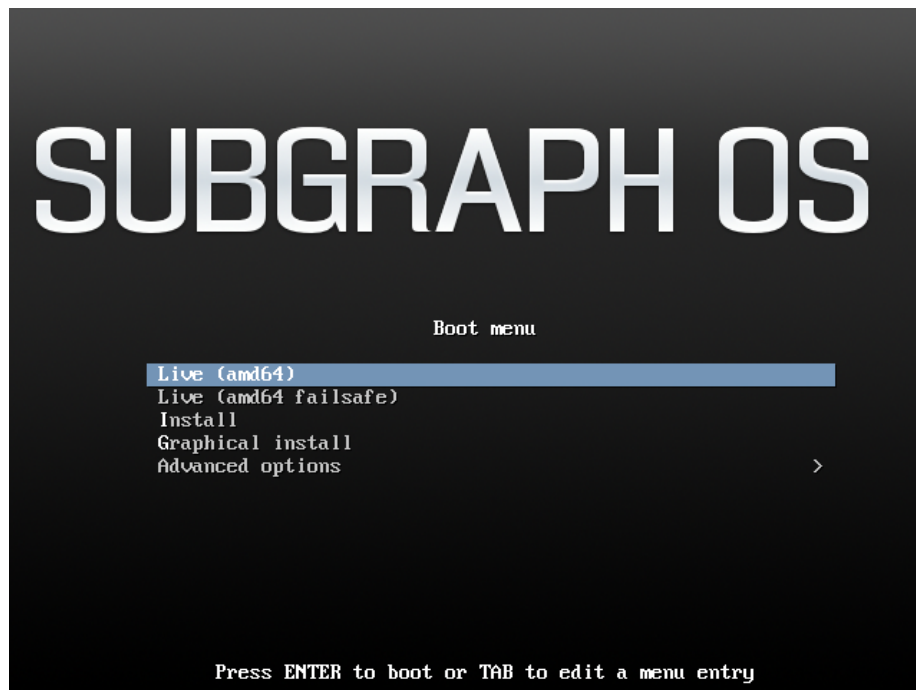


Figure 1: Subgraph OS boot screen

Please note that the user password on the live image is: `live`.

Installing, step-by-step

After the First Boot

Everyday usage

Subgraph OS comes with a number of applications that may already be familiar. We have also added newer alternatives that may be less familiar. This chapter shows you how to use these applications to perform everyday tasks.

Subgraph OS is also unique because the applications we have included are run inside of a security sandbox. We call this sandbox Oz. Oz helps protect the operating system and your personal files in case an application is compromised by a security vulnerability.

Each application described in this chapter runs inside an Oz sandbox. This means that they can only access the files and directories that they need to. Each of the applications is isolated from each other. They are also isolated from the system itself. Because the applications are isolated, they cannot access common directories such as `Pictures` or `Downloads` in the usual way. This chapter shows you how to manage your files in Oz, with some examples for each application.

Browsing the Web with Tor Browser

Tor Browser is the default web browser of Subgraph OS. It has a number of security and privacy advantages over other browsers.

The security and privacy features include:

- Anti-fingerprinting countermeasures to prevent websites from identifying individual users by their browser fingerprint
- A security slider that lets users disable browser features that may pose security and privacy risks

The Tor Browser runs inside a security sandbox, managed by Subgraph Oz. Web browsers represent some of the most complex software available. With complexity comes increased risk to security and privacy. This is what we call the `attack surface` of an application. Tor Browser is no different than other browsers in that it has a lot of attack surface. A successful compromise of Tor Browser could let an attacker gain access to things such as SSH keys, GPG encryption keys, personal files, email, etc. Our security sandbox technology helps to mitigate these risks.

Configuring the Tor Browser security slider

The Tor Browser includes a `security slider` that lets users choose the security and privacy features they want to enable. If they enable all of the security and privacy settings, some websites may be slower or may not work as expected. However, the security slider lets them instantly lower the settings if they need a particular website to work better.

We recommend setting the security slider to Medium-High or High. For websites you trust, you can lower the settings to make the website perform better.

We advise against lowering the security slider for any websites that are not accessed over HTTPS. HTTPS helps to make sure that the traffic between the Tor Browser and the website has not been tampered with. This is what we refer to as the 'integrity' security property. If you cannot verify the integrity of the traffic originating from a website by using HTTPS, it may be dangerous to visit the website using lowered security and privacy settings.

Downloading and saving files in the Tor Browser

The Tor Browser runs inside of Oz, our application sandbox. When files are downloaded by a sandboxed application such as the Tor Browser, they are saved within the sandbox. When you close the Tor Browser, Oz will cleanup the sandbox, causing files saved in the sandbox to be destroyed.

To allow the Tor Browser to download that can persist after the application is closed, Oz makes a special exception. This special exception is a `shared directory` where files can be saved and retrieved later, without being destroyed when Tor Browser is closed. `Shared directory`, in this case, means a directory that is shared inside and outside of the Oz sandbox. Oz sets up the the following shared directory for saving downloaded files:

```
~/Downloads/TorBrowser
```

The shared directory name may be localized depending on the language settings on your computer. In the case of French, the shared directory would be:

```
~/Téléchargements/TorBrowser
```

Files downloaded to the shared directory will persist after closing the Tor Browser.

Uploading files in the Tor Browser

When the Tor Browser starts, the Oz sandbox limits its access to files and directories on the computer. For example, a photo from the `Pictures` directory will not be visible in the sandbox by default. If you want to upload a photo from this directory, you must use the Oz menu to add it to the Tor Browser sandbox. The Oz menu is denoted by the little zebra icon at the top-right corner of the screen.



The following actions may be performed using the Oz menu:

- Add files to sandbox
- Open terminal in sandbox
- Shutdown sandbox

Click on the little zebra and then click Add file...

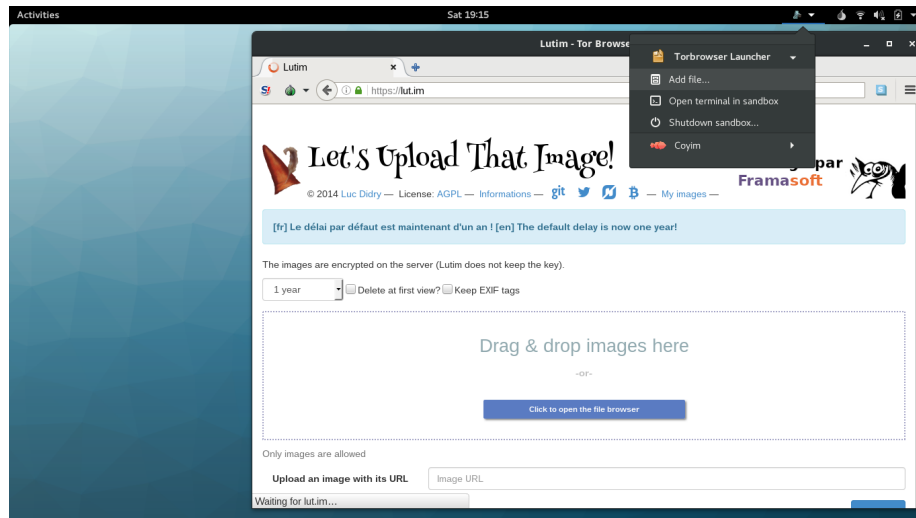


Figure 2: Oz menu - Add file

You may add more than one file at a time. You may also choose to make these files read-only, meaning that they can only be read and not written to while in the sandbox.

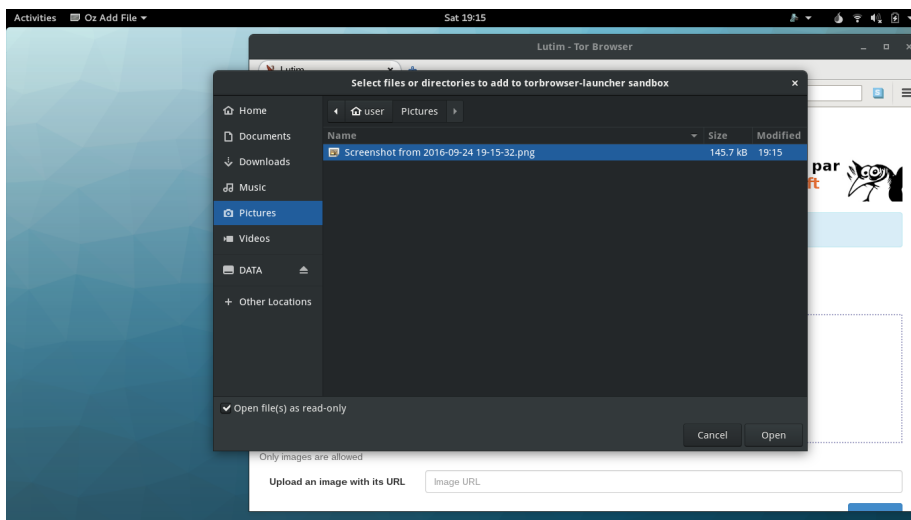


Figure 3: Oz menu - Select files or directories

Once the file(s) you want to upload are added to the Tor Browser sandbox, you may proceed to upload them normally.

Viewing PDFs

PDFs can present security and privacy risks to users. Subgraph OS sandboxes PDFs in a safe environment, minimizing those risks.

PDFs are affected by the following security and privacy risks:

1. Security vulnerabilities in the PDF reader software may allow adversaries to compromise a user who opens a malicious PDF
2. Privacy may be compromised if the PDF makes an outgoing connection to the Internet, such as when the user clicks on a link within the document or if the document automatically opens a link

Chatting with CoyIM

CoyIM is the default instant messaging application in Subgraph OS. CoyIM supports the XMPP instant messaging protocol. All chats are end-to-end encrypted using OTR (Off-the-Record) Messaging.

Adding an XMPP account to CoyIM

When CoyIM opens for the first time, it asks you if you want to encrypt your configuration file. We recommend that you encrypt your configuration.



Figure 4: CoyIM - Encrypt configuration file

If you have decided to encrypt your configuration file, you will be prompted to configure the master password that will be used to encrypt your configuration file. You will need to re-enter this password each time you use CoyIM, so choose something strong but memorable!

To begin using CoyIM, you must first add an existing account from an XMPP network.

Once you had added your account details, you can connect your account. If you have successfully connected to the chat network, a green dot will appear to the left of your username.

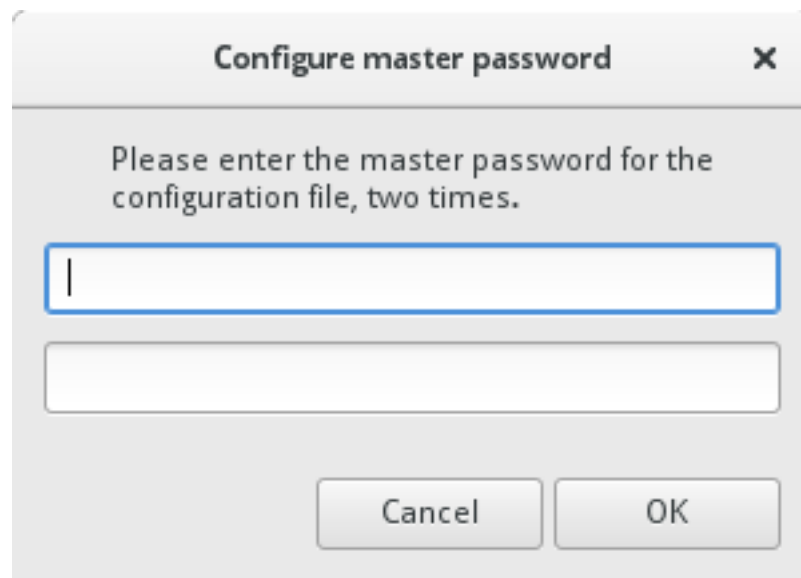
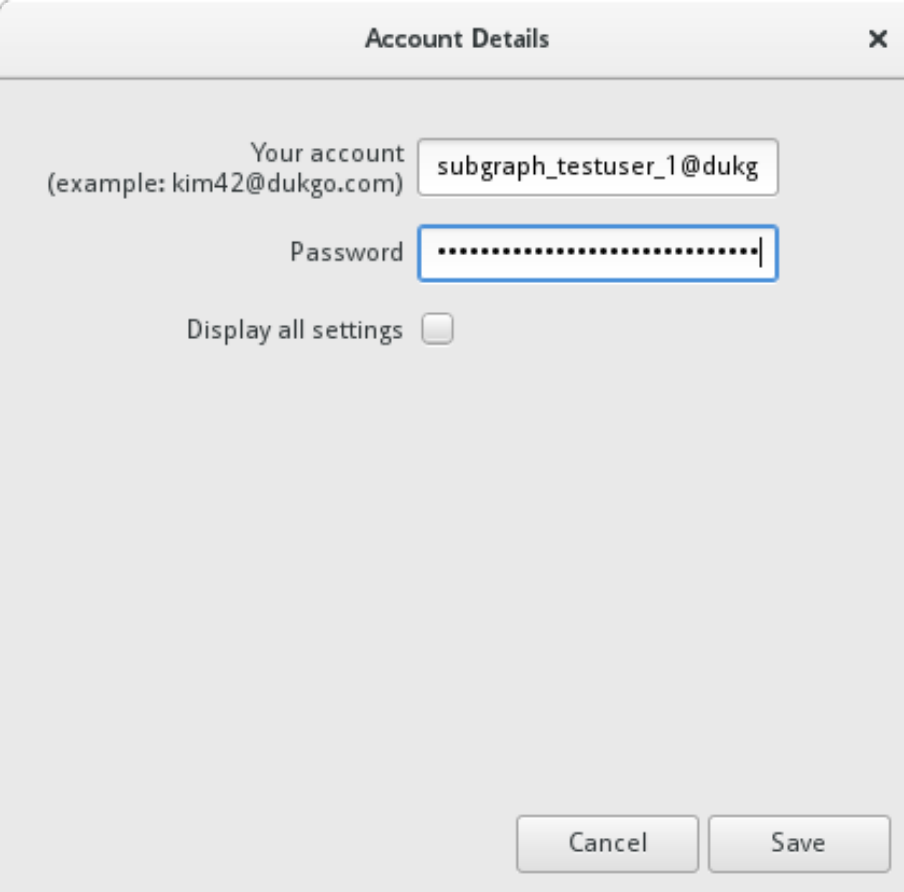


Figure 5: CoyIM - Configure master password

Chatting over Tor with Ricochet



The image shows a 'Account Details' dialog box with a close button (X) in the top right corner. It contains three input fields: 'Your account (example: kim42@dukgo.com)' with the value 'subgraph_testuser_1@dukg', 'Password' with a masked password of 20 dots, and 'Display all settings' with an unchecked checkbox. At the bottom are 'Cancel' and 'Save' buttons.

Account Details

Your account
(example: kim42@dukgo.com) subgraph_testuser_1@dukg

Password|

Display all settings ☐

Cancel Save

Figure 6: CoyIM - Account details: basic configuration

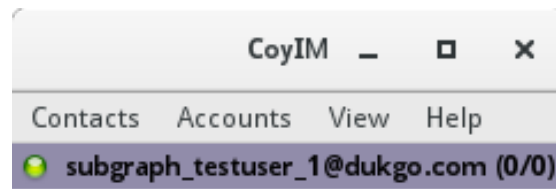


Figure 7: CoyIM - Successful connection

Sharing files with OnionShare

Monitoring outgoing connections with Subgraph Firewall

Features and advanced usage

Sandboxing applications with Subgraph Oz

Securing sandboxed applications with Oz seccomp

Anonymizing communications with Tor

Securing the Tor control port with roflcoptor

Routing applications through Tor with Subgraph Metaproxy

The Metaproxy is an important part of Subgraph OS. It is a service that runs in the background to help applications connect through the Tor network. This is done transparently, even with applications that are not configured or designed to work with Tor.

On a typical operating system, users must configure each application to connect to the Internet through Tor. This normally requires the user to configure the proxy settings of the application to use one of Tor's built-in proxies or to `torify` the application. Since Subgraph OS blocks outgoing connections that are not routed through Tor, this may pose a problem for applications that try to connect to the Internet but have not been manually `torified` or otherwise configured to work with Tor.

The Metaproxy addresses this problem by automatically relaying outgoing connections through Tor. When we say this is done transparently, we mean the following two things:

1. Users do not have to manually `torify` their applications or otherwise configure them to use Tor
2. Applications that are already configured to use Tor are ignored by the Metaproxy, therefore, it only helps those applications which need it

Hardening the operating system and applications with Grsecurity

Grsecurity is a third-party security enhancement to the Linux kernel. It is developed and maintained by the Grsecurity team. It is implemented as a patch to the upstream Linux kernel. Subgraph OS ships with a kernel that is patched with Grsecurity.

Configuring PaX flags with Paxrat

Anonymizing MAC addresses with Macouflage

MAC addresses are the unique identifiers for the network interface on the computer (such as Ethernet ports and WIFI cards). Due to their unique nature, they can also compromise the privacy of the user.

When connecting to a network, it is possible for other devices on the network to see the MAC address of the network interface that is connected. While this is not much of a concern on networks you trust such as your home network, it may compromise your privacy on those who do not trust. On untrustworthy or hostile networks, uniquely identifying characteristics such as the MAC address may allow others to track your computer.

Subgraph OS mitigates this privacy risk by always creating random MAC addresses for all of your network interfaces. Each time one of your interfaces connects to a network, it will use a different MAC address. This helps to anonymize you across different networks or when connecting to the same network over and over again.

Preventing unauthorized USB access with USB Lockout

Using virtual machines in Subgraph OS

Contrary to popular belief, there is nothing that stops the use of virtual machines in Subgraph OS. While there are, as of this writing, some known incompatibilities with VirtualBox, Qemu/KVM works as expected.

Qemu/KVM can be obtained by installing it the normal way: `sudo apt install qemu-system qemu-kvm qemu-utils`

Creating a virtual machine with Qemu.

The following are simple starter guides to using Qemu. For more detailed information regarding the operation of Qemu/KVM virtual machine see the official [Qemu manual](#).

There are also multiple user interfaces that allow interfacing with Qemu/KVM with various degrees of complexity and flexibility such as:

- [gnome-boxes](#)
- [virt-manager](#)
- [qemuctl](#)
- [virtualbricks](#)

Simple virtual machine creation

To create a virtual machine you will, if required, create a hard drive image for it:

```
qemu-img create -f qcow2 disk.qcow2 8G
```

Your virtual machine drive is now ready for use. You may launch a virtual machine using this drive like so:

```
qemu-system-x86_64 -enable-kvm -hda ./disk.qcow2 -m 4096
```

Where `-enable-kvm` enables KVM virtualisation instead of using emulation; `-hda ./disk.qcow2` attaches the disk image; and `-m 4096` allocates 4096MB of RAM to the virtual machine.

To attach a cdrom image, for example to install an operating system:

```
qemu-system-x86_64 -enable-kvm -hda ./disk.qcow2 -m 4096 -cdrom ./subgraph-os-alpha_2016-06-16_2.is
```


Advanced virtual machine creation

For more control and easier installation of Debian releases inside of a virtual machine, one may use debootstrap to create pre installed images without going through the installer process.

Let's start by creating an 8GB raw sparse image for our VM, then format and mount it:

```
truncate --size 8G ./disk.img
# Here you could decide to create a proper partition table if you wanted... or not...
/sbin/mkfs.ext4 ./disk.img
sudo mount -o loop ./disk.img /mnt
```

It's worth nothing that you should have enough free disk space for the image you create (and possible twice as much if you want to convert it later on).

However, the truncated image will only take as much as space as required:

```
du -sh disk.img
189M    disk.img
du --apparent-size -sh disk.img
8.0G    disk.img
```

Now that we have an image created and mounted, we can use debootstrap to expand a basic install into it:

```
sudo debootstrap --variant=mintbase --include=systemd-sysv stretch /mnt
```

```
# And set a root password
sudo chroot /mnt passwd
```

```
# Create a standard fstab
sudo tee /mnt/etc/fstab << EOL
/dev/sda    /    ext4    defaults,errors=remount-ro 0    1
EOL
```

```
# Let's download the subgraph grsec kernel and install it
cd /tmp
apt-get download linux-{image,headers}-grsec-amd64-subgraph linux-{image,headers}-${uname -r}
sudo cp ./linux-{image,headers}-${uname -r} /mnt/tmp
sudo chroot /mnt
$ dpkg -i /tmp/linux-{image,headers}-*
$ update-initramfs -u -k all
$ exit
```

```
# Now we grab a copy of the kernel and initramfs we just installed to boot the system
cp /mnt/boot/vmlinuz-<version>-amd64 /mnt/boot/initrd.img-<version>-amd64 /home/user/pa
```

```
# After, we will sync and umount
sync
sudo umount /mnt
```

Once done, we can use it as is with Qemu/KVM, or if you prefer it can be converted to a qcow2 image for convenience:

```
qemu-img convert -f raw -O qcow2 ./disk.img ./disk.qcow2
```

We can now launch our image:

```
qemu-system-x86_64 -enable-kvm -hda ./disk.qcow2 \
    -kernel ./vmlinuz-<version>-amd64 \
    -initrd ./initrd.img-<version>-amd64 \
    -append root=/dev/sda
```

If you want to install grub to keep the kernel and initrd images inside the virtual machine you'll have to create a full partition table, and potentially a separate /boot partition. But this is out of scope for this short tutorial.

Simple networking

By default, Qemu will transparently NAT your virtual machines to the host network. This can be disabled by using the `-net none` flag.

Alternatively, you can also open simple tunnels between the host and the virtual machine using the port redirection mechanism with the `-redir` flag:

```
-redir tcp:55700::55700
```

For more on networking in Qemu/KVM see:

- <http://wiki.qemu.org/Documentation/Networking>
- <https://en.wikibooks.org/wiki/QEMU/Networking>