Split Handbook

Split Linux is a general operating system optimized for safely navigating hostile environments like the Internet and physical check points.

Split Handbook

07 August 2023

Contents

1 About

Both, *Split* and *The Beast Desktop* are developed in the spare time of user **kevcrumb**. While I develop the former to support your privacy and sovereignty, the latter is meant to allow maximum speed and efficacy for your daily computer use.

1.1 Split

Split is as stable as its parent, Void Linux. It can function as your daily driver.

It extends the Void Linux Live disk with functionality to setup and manage LXC containers that are used to start graphical interfaces, simultanously, on separate TTYs.

Every container gets separate Tor circuits to the outside world, which prevents the TTYs from being correlated to one another. Thus, in a Split context, the containers are referred to as *"identities"*.

1.2 Beast

Beast builds on dwm, the window manager that is so stable, it rarely gets updates. It doesn't need them.

Beast aims to give you the best tool for each job and hands you scripts for optimized workflows. A handful of these scripts are already included. Hundreds more are ready and waiting to pass quality-control. They will be hand-selected and added as time permits.

1.3 A Note About Development

Development will typically happen in bursts, depending on my personal schedule. Expect periods of several months without new features being added to Split or Beast.

But don't worry: In case of Split you do not need the newest packages - it mostly provides the Kernel for your identities - and in Beast you will be using Void Linux packages directly anyway.

2 About This Handbook

Documenting Split and The Beast Desktop has just begun.

It is one of the most time-consuming parts. If you want to help, look at any yet-undocumented script and describe it's function for inclusion in this manual.

A local copy of this handbook, in several formats, can be installed via the `splitlinux-docs` package and accessed from the `/usr/share/doc/splitlinux/` directory structure.

The purpose of this document is to explain how to setup, configure, and use Split Linux systems.

To search for a particular term within the Handbook, select the 'magnifying glass' icon, or press 's'.

2.1 Example Commands

Examples in this guide may have snippets of commands to be run in your shell. When you see these, any line beginning with `$` is run as your normal user. Lines beginning with `#` are run as `root`. After either of these lines, there may be example output from the command.

2.1.1 Placeholders

Some examples include text with placeholders. Placeholders indicate where you should substitute the appropriate information. For example:

```
# ln -s /etc/sv/<service_name> /var/service/
```

This means you need to substitute the text `<service_name>` with the actual service name.

3 Split Linux

This section and its subsections provide information about Split Linux.

4 Installation

Split Linux is run as a Live OS which mounts an encrypted hard disk containing LXC containers.

Please refer to https://splitlinux.org for basic setup instructions.

5 The Recommended Setup

At the current stage Split Linux is still bare bones. This guides walk you from the initial hard disk preparation all the way through to starting your first container session.

At the beginning of every essential section there's a link to a screencast that may help spot potential errors in your process.

If you're running into any issues come and ask your questions over at our Reddit r/splitlinux.

5.1 Hard disk preparation

1. Create two primary partitions on your hard disk using your preferred partitioning tool (‘cfdisk /dev/<DEVICE>‘). The first is reserved for a decoy OS. Use the second for Split.
2. Command `format_for_splitlinux /dev/<DEVICE>` where "‘<DEVICE>‘" is the path to the disk you want to use.
3. Follow the instructions on the screen. Remember the password you set.
4. When finished follow the final suggestions on how to continue.

If you are interested in the manual steps involved, you may watch this ASCII Cast about the hard disk preparation.

5.2 Container setup

Boot your system from the Split Linux pendrive. Split will detect the partition you just created, ask for its password and mount it.

1. Switch to the second terminal (<kbd>Ctrl+Alt+F2</kbd>) and log in as **root**.
2. Command `create_voidlinux_container v 122` where "v" is the desired name of the container/identity and "122" an ID in the range 100-254.
3. Follow the instructions on the screen. Remember the password you set.
4. When finished run the suggested command to install the Beast desktop environment. Name any additional packages you want at the end of the `xbps-install` command.

When you now log out and log in using the container name as user name (eg **v**) you should end up *within* the container's graphical environment.

And this is where the journey begins.

Continue at Configuration: Internet for Applications to learn how to connect your applications through Tor.

6 Creating A Custom Container (Optional)

This explains manual container creation. You should only use it if the automated process doesn't satisfy your needs.

6.1 Container setup

Click here to watch the steps for "Container setup" as ASCII Cast

Boot your system from the Split Linux pendrive. Split will detect the partition you just created, ask for its password and mount it.

1. Switch to the second terminal (<kbd>Ctrl+Alt+F2</kbd>) and log in as **root**. *(Currently, the first terminal displays Tor debug output which renders it unfit for interactive use. This will be changed in a future release.)*
2. Ensure that **horde** is mounted (‘mount | grep split-horde‘) and that you're connected to the Internet (‘ping -c1 splitlinux.org‘).
3. Create a container. *You can replace name and amend the last row of parameters to install a different distro as offered at images.linuxcontainers.org.* <pre>$ lxc-create –config /usr/share/splitlinux/config/splitlinux-default.conf –name v –template download – –release current –dist voidlinux –arch amd64 –variant musl</pre>
4. Remove superfluous includes from newly created container configuration (‘sed 's#l̂xc.*lxc/config.*##g' /var/lib/lxc/v/config‘).
5. Set a unique IP address for the container (‘sed -i 's#100#122#g' /var/lib/lxc/v/config‘).
6. Start the container (‘lxc-start v‘) and step into it (‘lxc-attach v‘).
7. In the container create a user. Its name must match the container name! (‘useradd –create-home v‘).
8. Set a password for that user (‘passwd v‘).

*Our example names the container **v** as in Void Linux. Using single letters for container- and user names is generally a great way to save time typing.*

6.2 Container package installation

Click here to watch the steps for "Container package installation" as ASCII Cast

1. Make the container use Void's Tor mirror for packages (‘echo 'repository=http://lysator7eknrfl47rlyxvgeamrv7ucefgrrlhk7rouv3sna25asetwid.onion/pub/voidlinux/current/m > /etc/xbps.d/00-repository-main.conf‘).
2. Fetch the package index (‘SOCKS_PROXY="socks5://172.18.0.2:9050" xbps-install -Su‘).
3. Install a basic graphical environment (‘SOCKS_PROXY="socks5://172.18.0.2:9050" xbps-install -S dwm st xorg-minimal xorg-fonts monero curl torsocks‘).
4. Configure **dwm** to start as graphical environment (‘echo 'exec dwm' » /home/v/.xinitrc‘).

5. Exit the container (‘exit‘) and verify that you’re back in the host system (‘hostname‘).

6. Reboot (‘reboot‘).

6.2.1 Notes

* A window manager or a desktop environment (like **dwm**) is highly recommended if you want a graphical interface. * If **xorg-minimal** is not installed, dwm will fail to start complaining about `DISPLAY` not being set. * If **xorg-fonts** is not installed, dwm will fail to start complaining that `monospace:10` is missing. * **torsocks** is essential as it allows pretty much any application to route through Tor.

When you now log in using the container name as user name (eg **v**) you should end up *within* the container’s graphical environment.

Continue at Configuration: Internet for Applications to learn how to connect your applications through Tor.

7 Configuration

7.1 Overrides

Overrides enable users to customize the configuration of the live environment (i.e. the Split host system).

Thanks to overrides, one can boot vanilla Split Linux as intended (as Live OS) and still be able to mount additional devices (*fstab*), execute arbitrary commands on boot (*rc.local*), manage access to devices (*udev/rules.d/*) and more.

7.1.1 Usage

Any file placed into `/var/lib/lxc/_host/overrides/etc/` will replace the corresponding file in `/etc/` upon boot (subdirectories supported).

For example, to activate the amount of huge pages required for optimum Monero-mining performance set the corresponding sysctl value and reboot:

```
# cd /var/lib/lxc && mkdir -p _host/overrides/etc/
# echo 'vm.number_hugepages=1280' >> /var/lib/lxc/_host/
   overrides/etc/sysctl.conf
# reboot
```

8 Internet for Applications

If an application directly supports SOCKS proxies you can pass it the IP of the Tor router directly by whatever means the application supports.

Some, like the Void package manager XBPS support the `SOCKS_PROXY` environment variable. Others, like Monero, will take the proxy IP as command line argument.

8.1 torsocks (for any application)

If an application lacks direct SOCKS support, the `torsocks` command can usually be used to transparently route that application's traffic through Tor anyway.

Just prepend a command with "‘torsocks -a 172.18.0.2 -P 9050 –isolate‘".

8.2 Container connection test

If your host is correctly connected to the Internet you should be able to establish a connection via Tor. Try it:

1. Bring up a terminal (<kbd>Alt+Shift+Enter</kbd>) and check your Tor IP by telling `curl` to use the Tor router as proxy: <pre>$ curl –proxy socks5h://172.18.0.2:9050 https://check.torproject.org/api/ip</pre>
2. Check your Tor IP transparently by wrapping `curl` within the `torsocks` command: <pre>$ torsocks -a 172.18.0.2 -P 9050 –isolate curl https://check.torproject.org/api/ip</pre>

Note that your IP displays differently on every invocation as every connection establishes a new Tor circuit.

—

8.3 Further examples

8.3.1 Ansible

$ ANSIBLE_SSH_ARGS="-o ProxyCommand='openbsd-nc -X 5 -x 172.18.0.2:9050

See footnotes ansible[ânsible], cows[ĉows]

8.3.2 Chromium

$ chromium –proxy-server=socks://172.18.0.2:9050

8.3.3 Git

$ GIT_SSH_COMMAND="ssh -o ProxyCommand='nc –proxy-type socks5 –proxy 172.18.0.2:9050

See footnotes git[ĝit], ssh[ŝsh]

8.3.4 Monero

$ monero-wallet-cli –proxy 172.18.0.2:9050 –untrusted-daemon –daemon-host 4egylyolrzsk6rskorqvocipdo4tqqoyzxnplbjorns7issmgpoxvtyd.onion

See footnote monero[m̂onero]

8.3.5 Rsync over SSH

$ rsync –rsh 'ssh -o ProxyCommand="nc –proxy-type socks5 –proxy 172.18.0.2:9050
See footnote ssh[ŝsh]

8.3.6 Secure Shell

$ ssh -o ProxyCommand='nc –proxy-type socks5 –proxy 172.18.0.2:9050
See footnote ssh[ŝsh]

8.3.7 Tor Browser

!Connecting Tor Browser

8.3.8 youtube-dl

$ youtube-dl –proxy socks5://172.18.0.2:‘shuf -n1 -i 9050-9059‘ <VIDEO_ID>

8.3.9 XBPS

$ SOCKS_PROXY=socks5://172.18.0.2:9050 xbps-install -Su

—

* [ânsible]: For Ansible, nmap's netcat often fails with UNREACHABLE: **Failed to connect to the host via ssh: kex_exchange_identification: banner line contains invalid charactersńbanner exchange: Connection to UNKNOWN port 65535: invalid format**. `openbsd-nc` (package `openbsd-netcat`) doesn't seem to trigger such errors. * [ĉows]: For readability you will probably want to call `ansible-playbook` with the `ANSIBLE_NOCOWS=1` environment variable set. * [ĝit]: If this fails and a warning regarding LFS is displayed, try disabling it as instructed. * [ŝsh]: These examples use the `netcat` variant provided by the `nmap` package. * [m̂onero]: See monero.fail for a list of Tor remote nodes.

9 Virtual Private Networks

In order for a VPN to be compatible with Split's protected routing setup it must allow connection from Tor.

```
Like so:
you -> tor -> VPN -> internet

NOT so:
you -> VPN -> tor -> internet
```

It must be possbile to establish a connection after adding `socks-proxy 172.XX.0.2 9050` to the OpenVPN config file, where the "XX" octet depends on the type of container (isolated/leaky/exposed) you are using.

9.1 Why a VPN?

Split Linux torifies internet connections. Many parties, while not completely blocking connections via Tor, still make them very hard to access. You may end up solving countless loops of reCAPTCHA only to be denied access anyways or DDoS protection leaves you hanging at their check for validity.

The best way to deal with this is to avoid such sites altogether, but when there's no alternative provider of a service yet, or when you don't want to play into the "something to hide" narrative, it's helpful to appear as yet another citizen.

9.2 Commercial Providers

This is a list of commercial VPN providers that did not block our connection from Tor and let us surf the web normally. Please note the date and be aware that policies may change.

Provider	Comment	Cost (1y)	Ch
AirVPN (onion)	price when on offer	31.85 EUR	No
IVPN		60.00 USD	Fel
SnowHaze	only their "Firewall Bypass", Germany-location, confirmed working	50.00 CHF	No
Xeovo	support claims it's working - to be confirmed	35.88 EUR	No

If you know further VPN's that work, please let us know. A good starting point for finding VPN's is the list at https://monerica.com.

9.2.1 Incompatible

Most providers do not allow establishing a VPN through Tor. This list will be extended as more data is being gathered:

- ProtonVPN

9.3 Homemade

Tech-savvy users can turn any VPS hosting to use as VPN. As a major advantage this means that your traffic will look even more generic, since you do not surf using any of the known IP addresses associated with a VPN provider. If said VPS is on a dedicated IP, it might be a downside to have all traffic tied to one unique user.

While pretty much every "Big Tech" company offers a free-tier hosting, it's counter-productive to empower those by handing over even more statistics to analyze.

Know that you can find great offers for roughly one dollar a month, some of which payable with Monero (through BTC gateways).

10 The Beast Desktop

This section and its subsections provide information about The Beast Desktop Environment.

11 Beast dwm

This section has yet to be written.

12 Beast Scripts

This section has yet to be written.

13 Mine Monero

cmd: BDmine_monero kbd: Shift+Pause

Every time you want to donate hash rate to Split Linux, simply lock the screen using <kbd>Shift+Pause</kbd>.

To start donating from the command line, command `BDmine_monero` without any arguments.

If you want to mine to your own account run `BDmine_monero` providing a **handle** of your choosing and a wallet **address** as arguments. Example:

$ BDmine_monero mary 847vSaDnoiFehRQLwfnETRiEQwPWxzWWyeJtw6o3k1DUTvrfD2DCoHUKjpZ7Ui1PoQ5JcbdcEcpVcAiQYDA2Vyda8EsAbzX

If the handle is already associated with a different wallet address, the miner will exit with a message indicating this.

After the command ran successfully once, handle and address are linked and you can leave out the wallet address on future invocations.

Note: The miner will only be able to connect if a working default network route for the internet connection is active (check `ip route`).

13.1 Tweaking

13.1.1 Hugepages

Activating hugepages is crucial for efficient mining. Use Split Linux Overrides to persistently define `vm.number_hugepages=1280` in `/etc/sysctl.conf`.

13.1.2 CPU threads

The amount of CPU threads used defaults to the amount of processing units available. On a typical system, the amount of units available equals the amount of CPU cores.

Set the environment variable `BEAST_MINER_THREADS` to override the value, ideally in `~/.xinitrc.d/50-custom.sh` as `export BEAST_MINER_THREADS=<number>`.

14 Merge skel

```
cmd: BMmerge_skel
kbd: <none>
```

For an optimum desktop experience run `BMmerge_skel` whenever the **beast-skel** package gets updated. This pulls new optimizations into your identity's configuration. `sdiff(1)`'s interactive mode is used to process any changed files. Be sure to familiarize yourself with the tool beforehand.

15 Beast skel

The package **beast-skel** provides optimized defaults for Beast Desktop applications.

When an identity is first created, the contents of `/etc/skel` are copied into the main user's HOME directory.

15.1 Updates

Changes in **beast-skel** only affect the original files in `/etc/skel`.

Whenever the package is updated, it makes sense to merge the provided optimizations into the user's configuration.

The maintenance script BMmerge_skel simplifies the task.

16 Beast CommandLine

This section has yet to be written.

17 Beast Extras

This section has yet to be written.