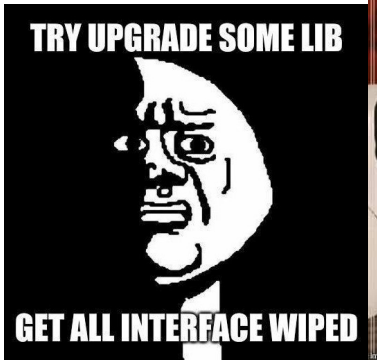
Secure Linux Networking v2

By Xe1phix

@Xe1phix | Xe1phix's GnuPG Key | Xe1phix @protonmail.ch | Telegram @Xe1phix

About Xe1phix

- https://gitlab.com/users/xe1phix/projects
- https://t.me/xe1phix
- https://twitter.com/xe1phix
- GnuPG Keys:
    - Xe1phix-Public-GnuPG-Key
    - Xe1phix-ProtonMail-Public-GnuPG-Key
- https://gitlab.com/xe1phix/xe1phix-linux-infosec-professional



Telegram @Xelphix | Slack @Xelphix | @Xelphix | Xelphix's GnuPG Key | Xelphix @protonmail.ch

InfoSec Community - Contact

- SecDSM Slack - https://secdsm.slack.com
- ParrotSec Telegram - https://t.me/parrotsecgroup
- Central Iowa LUG - https://cialug.slack.com/

InfoSec Community – IRC Channels

- Central Iowa Linux User Group
    - https://web.libera.chat/#cialug
- AskNoah WebIRC
    - https://web.libera.chat/#asknoahshow
- Parrotsec WebIRC
    - https://web.libera.chat/#parrotsec

Firejail Features:

- Seccomp-BPF (Restrict System Call)
- AppArmor Confinement
- User Namespaces (CLONE_NEWUSER)
- Mount namespaces (CLONE_NEWNS)
- Chroot Containers
- PID Namespaces (CLONE_NEWPID)
- OverlayFS
- Linux rlimits (Resource Allocation)
- Grsecurity Support
- CGroupV2 (Linux Control Groups)
- Berkeley Packet Filter (BPF) Support)
- Extended Berkeley Packet Filter (eBPF) Support
- NoGroup
- NoNewPrivs
- NoRoot (User Namespace Mounts)
- IPC Namespaces (CLONE_NEWIP) - isolate certain interprocess communication (IPC) resources
- Filesystem Containers

Firejail Security Sandbox

```
$ firejail --version
firejail version 0.9.27

Compile time support:
    - AppArmor support is enabled
    - AppImage support is enabled
    - chroot support is enabled
    - file and directory whitelisting support is enabled
    - file transfer support is enabled
    - firetunnel support is enabled
    - networking support is enabled
    - overlayfs support is enabled
    - private-home support is enabled
    - seccomp-bpf support is enabled
    - user namespace support is enabled
    - X11 sandboxing support is enabled
```

Xe1phix - Firejail Cheatsheet Template

https://gitlab.com/xe1phix/Xe1phix-Firejail#firejail-features

- Filesystem Containers
- Linux Capabilities (POSIX 1003.1e)
- Whitelist Linux Capabilities (POSIX 1003.1e
- Blacklist Linux Capabilities (POSIX 1003.1e)
- Audit Linux Capabilities (POSIX 1003.1e)
- Network Namespaces (CLONE_NEWNET)
- Protocol Filtering (unix, inet and inet6)
- UTS Namespaces (CLONE_NEWUTS)
- Overlayfs Filesystems
- Private Mounting
- Bind Mounts
- TmpFS Mounting (Temporary Filesystem)
- Read-Only | File(s) & Directories
- Read-Write | File(s) & Directories
- NoExec (No Execution)
- Blacklist | File(s) & Directories
- Whitelist | File(s) & Directories
- Blacklist External Devices
- Blacklist 3D
- Blacklist /dev/
- Blacklist /mnt/
- Blacklist /media/
- Read-Only Bind Mounts
- X11 Sandboxing
- Xpra Support
- Xephyr Server Support
- Network Interface Support: | macvlan, Bridged Interfaces, VLANs
- TUN Network Driver Support (Ethernet Virtual Network Interface)
- TAP Network Driver Support (Wireless Virtual Network Interface)
- Trustworthy DNS (Enforced)
- Netfilter (IPTables) Packet Filtering/Firewall
- Bridged Network Interfaces
- VLAN Network Interfaces
- NoNet (Isolate Network Interface In Its Own Namespace)
- EncFS and SSHFS
- Traffic Shaping
- Sandbox Auditing
- System Call Tracing
- DNS Auditing

Understanding namespace isolation

Namespaces are a kernel security feature that was introduced in Linux kernel version 2.4.19, all the way back in 2002. A namespace allows a process to have its own set of computer resources that other processes can't see. They're especially handy for times when you might have multiple customers sharing resources on the same server. The processes for each user will have their own namespaces. Currently, there are seven types of namespaces:

- **Mount (mnt)**: This is the original namespace, which was introduced in Linux kernel 2.4.19. At the time, this was the only namespace. This allows each process to have its own root filesystem that no other processes can see, unless you choose to share it. This is a good way of preventing information leakage.
- **UTS**: The UTS namespace allows each process to have its own unique hostname and domain name.
- **PID**: Every running process can have its own set of PID numbers. PID namespaces can be nested so that a parent namespace can see the PIDs of child namespaces. (Note that child namespaces can't see into the parent namespaces.)
- **Network (net)**: This allows you to create a whole virtual network for each process. Each virtual network can have its own subnets, virtual network interfaces, routing tables, and firewalls.
- **Interprocess Communication (ipc)**: This also prevents data leakage by preventing two processes from sharing the same memory space. Each running process can access its own memory space, but other processes will be blocked.
- **Control group (cgroup)**: This namespace hides the identity of the cgroup that a process is a member of.
- **User**: The User namespace allows a user to have different levels of privilege on different processes. For example, a user could have root-level privileges on one process, but only normal-user privileges on another process.

Mount namespaces (`CLONE_NEWNS`, Linux 2.4.19) isolate the set of filesystem mount points seen by a group of processes. Thus, processes in different mount namespaces can have different views of the filesystem hierarchy. With the addition of mount namespaces, the `mount()` and `umount()` system calls ceased operating on a global set of mount points visible to all processes on the system and instead performed operations that affected just the mount namespace associated with the calling process.

Network namespaces (`CLONE_NEWNET`, started in Linux ~~2.4.19~~ 2.6 and largely completed by about Linux 2.6.29) provide isolation of the system resources associated with networking. Thus, each network namespace has its own network devices, IP addresses, routing tables, `/proc/net` directory, port numbers, and so on.

PID namespaces (`CLONE_NEWPID`, Linux 2.6.24) isolate the process ID number space. In other words, processes in different PID namespaces can have the same PID. One of the main benefits of PID namespaces is that containers can be migrated between hosts while keeping the same process IDs for the processes inside the container. PID namespaces also allow each container to have its own `init` (PID 1), the "ancestor of all processes" that manages various system initialization tasks and reaps orphaned child processes when they terminate.

User namespaces (`CLONE_NEWUSER`, started in Linux 2.6.23 and completed in Linux 3.8) isolate the user and group ID number spaces. In other words, a process's user and group IDs can be different inside and outside a user namespace. The most interesting case here is that a process can have a normal unprivileged user ID outside a user namespace while at the same time having a user ID of 0 inside the namespace. This means that the process has full root privileges for operations inside the user namespace, but is unprivileged for operations outside the namespace.

UTS namespaces (`CLONE_NEWUTS`, Linux 2.6.19) isolate two system identifiers—`nodename` and `domainname`—returned by the `uname()` system call; the names are set using the `sethostname()` and `setdomainname()` system calls. In the context of containers, the UTS namespaces feature allows each container to have its own hostname and NIS domain name. This can be useful for initialization and configuration scripts that tailor their actions based on these names. The term "UTS" derives from the name of the structure passed to the `uname()` system call: `struct utsname`. The name of that structure in turn derives from "UNIX Time-sharing System".

IPC namespaces (`CLONE_NEWIPC`, Linux 2.6.19) isolate certain interprocess communication (IPC) resources, namely, System V IPC objects and (since Linux 2.6.30) POSIX message queues. The common characteristic of these IPC mechanisms is that IPC objects are identified by mechanisms other than filesystem pathnames. Each IPC namespace has its own set of System V IPC identifiers and its own POSIX message queue filesystem.

CAP_SETPCAP	Without VFS support for capabilities:

Filtering Linux Capabilities

Capability Name	Meaning
CAP_ALL	CAP_ALL is not a real capability, but was coded into `gradm` to represent all capabilities. Therefore to denote dropping of all capabilities, but CAP_SETUID, -CAP_ALL and +CAP_SETUID would be used.
CAP_CHOWN	In a system with the [_POSIX_CHOWN_RESTRICTED] option defined, this overrides the restriction of changing file ownership and group ownership.
CAP_DAC_OVERRIDE	Override all DAC access, including ACL execute access if [_POSIX_ACL] is defined. Excluding DAC access covered by CAP_LINUX_IMMUTABLE.
CAP_DAC_READ_SEARCH	Overrides all DAC restrictions, regarding read and search on files and directories, including ACL restrictions, if [_POSIX_ACL] is defined. Excluding DAC access covered by CAP_LINUX_IMMUTABLE.
CAP_FOWNER	Overrides all restrictions about allowed operations on files, where file owner ID must be equal to the user ID, except where CAP_FSETID is applicable. It doesn't override MAC and DAC restrictions.
CAP_FSETID	Overrides the following restrictions, that the effective user ID shall match the file owner ID, when setting the S_ISUID and S_ISGID bits on that file; that the effective group ID (or one of the supplementary group IDs) shall match the file owner ID when setting the S_ISGID bit on that file; that the S_ISUID and S_ISGID bits are cleared on successful return from `chown(2)` (not implemented).
CAP_KILL	Overrides the restriction, that the real or effective user ID of a process, sending a signal, must match the real or effective user ID of the process receiving the signal.
CAP_SETGID	▪ Allows `setgid(2)` manipulation. ▪ Allows `setgroups(2)`. ▪ Allows forged gids on socket credentials passing.
CAP_SETUID	▪ Allows `set*uid(2)` manipulation (including fsuid). ▪ Allows forged pids on socket credentials passing.
	▪ Transfer any capability in your permitted set to any pid, remove any capability in your permitted set from any pid. With VFS support for capabilities (neither of above, but) ▪ Add any capability from current's capability bounding set to the current process' inheritable set ▪ Allow taking bits out of capability bounding set. ▪ Allow modification of the securebits for a process.
CAP_LINUX_IMMUTABLE	Allow modification of S_IMMUTABLE and S_APPEND file attributes.
CAP_NET_BIND_SERVICE	▪ Allows binding to TCP/UDP sockets below 1024. ▪ Allows binding to ATM VCIs below 32.
CAP_NET_BROADCAST	Allow broadcasting, listen to multicast.
CAP_NET_ADMIN	▪ Allow interface configuration. ▪ Allow administration of IP firewall, masquerading and accounting. ▪ Allow setting debug option on sockets. ▪ Allow modification of routing tables. ▪ Allow setting arbitrary process / process group ownership on sockets. ▪ Allow binding to any address for transparent proxying. ▪ Allow setting TOS (type of service). ▪ Allow setting promiscuous mode. ▪ Allow clearing driver statistics. ▪ Allow multicasting. ▪ Allow read/write of device–specific registers. ▪ Allow activation of ATM control sockets.
CAP_NET_RAW	▪ Allow use of RAW sockets. ▪ Allow use of PACKET sockets.
CAP_IPC_LOCK	▪ Allow locking of shared memory segments. ▪ Allow `mlock` and `mlockall` (which doesn't really have

CAP_IPC_OWNER	Override IPC ownership checks.
CAP_SYS_MODULE	Insert and remove kernel modules – modify kernel without limit.
CAP_SYS_RAWIO	▪ Allow ioperm/iopl access ▪ Allow sending USB messages to any device via */proc/bus/usb'*
CAP_SYS_CHROOT	Allow use of `chroot()`.
CAP_SYS_PTRACE	Allow `ptrace()` of any process.
CAP_SYS_PACCT	Allow configuration of process accounting.
CAP_SYS_ADMIN	▪ Allow configuration of the secure attention key. ▪ Allow administration of the random device. ▪ Allow enabling/disabling tagged queuing on SCSI controllers and sending arbitrary SCSI commands. ▪ Allow setting encryption key on loopback filesystem. ▪ Allow setting zone reclaim policy.
CAP_SYS_BOOT	▪ Allow use of `reboot()` ▪ Allow use of `kexec()` syscall
CAP_SYS_NICE	▪ Allow raising priority and setting priority on other (different UID) processes. ▪ Allow use of FIFO and round–robin (realtime) scheduling on own processes and setting the scheduling algorithm used by another process.
CAP_SYS_RESOURCE	▪ Override resource limits. Set resource limits. ▪ Override quota limits ▪ Override reserved space on ext2 filesystem ▪ Modify data journaling mode on ext3 filesystem (uses journaling resources). NOTE: ext2 honors fsuid when checking for resource overrides, so you can override using fsuid too. ▪ Override size restrictions on IPC message queues. ▪ Allow more than 64Hz interrupts from the real–time clock. ▪ Override max number of consoles on console allocation. ▪ Override max number of keymaps.
CAP_SYS_TIME	▪ Allow manipulation of system clock. ▪ Allow `irix_stime` on mips. ▪ Allow setting the real–time clock.
CAP_SYS_TTY_CONFIG	▪ Allow configuration of tty devices. ▪ Allow `vhangup()` of tty.
CAP_MKNOD	Allow the privileged aspects of `mknod()`.
CAP_LEASE	Allow taking of leases on files.
CAP_AUDIT_WRITE	Allow emitting auditing messages.
CAP_AUDIT_CONTROL	Allow administration of the kernel's auditing system.
CAP_SETFCAP	Allow the setting of file capabilities.
CAP_MAC_OVERRIDE	Override MAC access. The base kernel enforces no MAC policy. An LSM may enforce a MAC policy and if it does and it chooses to implement capability based overrides of that policy, this is the capability it should use to do so.
CAP_MAC_ADMIN	Allow MAC configuration or state changes. The base kernel requires no MAC configuration. An LSM may enforce a MAC policy, and if it does and it chooses to implement capability based checks on modifications to that policy or the data required to maintain it, this is the capability it should use to do so.
CAP_SYSLOG	Allow configuring the kernel's syslog (printk behaviour).

```
--caps.drop=capability,capability,capability
       Define a custom blacklist Linux capabilities filter.

       Example:
       $ firejail --caps.drop=net_broadcast,net_admin,net_raw

--net=none
       Enable a new, unconnected network namespace. The only interface
       available in the new namespace is a new loopback interface
       (lo). Use this option to deny network access to programs that
       don't really need network access.

--net=tap_interface
       Enable a new network namespace and connect it to this ethernet
       tap interface using the standard Linux macvlan driver. If the
       tap interface is not configured, the sandbox will not try to
       configure the interface inside the sandbox. Please use --ip,
       --netmask and --defaultgw to specify the configuration.

       Example:
       $ firejail --net=tap0 --ip=10.10.20.80 --netmask=255.255.255.0
       --defaultgw=10.10.20.1 firefox
```

firetools

a Firejail graphical user interface

Name firefox

Description Mozilla Firefox Browser [Seccomp-bpf Sandbox]

Command firejail --profile=/etc/firejail/firefox.profile --protocol=unix,inet,packet --dns=193.138.218.74

The launcher will us
installed on your sy
You can supply your

Name vlc

Description VLC Media Player {Firejail NoNet Profile}

Command private-dev --private-tmp --whitelist=${Downloads}/ --net=none /usr/bin/vlc

Help

Name qbittorrent

Description qBittorrent {Firejail Sandbox}

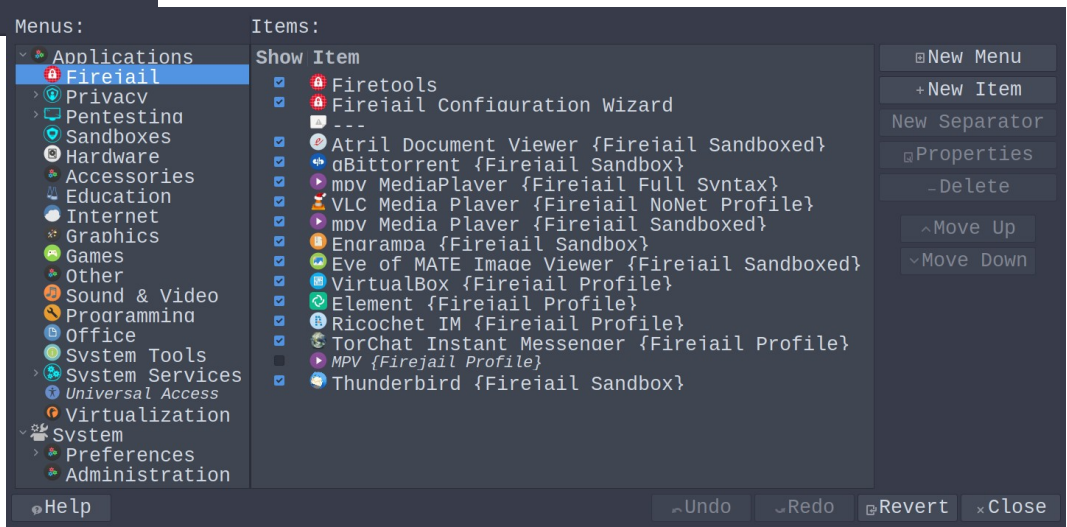
Command firejail --profile=/etc/firejail/qbittorrent.profile --protocol=unix,inet,netlink --dns=193.138.218.74 --dns=198.98.49.91 /usr/bin/qbittorrent

The launcher will use the name above to find an icon already
installed on your system (name.png files in /usr directory).
You can supply your own icon by placing it in ~/.config/firetools.

Help

× Cancel ✓ OK

Firejail – Custom Menu



```
└─$sudo firecfg --debug --clean
[sudo] password for parrotseckiosk:
Removing all firejail symlinks:
   VirtualBox removed
   atril removed
   atril-previewer removed
   atril-thumbnailer removed
   audacious removed
   bleachbit removed
   brasero removed
   claws-mail removed
   cola removed
   coyim removed
   cvlc removed
   dconf-editor removed
   dig removed
   display removed
   dnsmasq removed
   element-desktop removed
   enchant-2 removed
   enchant-lsmod-2 removed
   engrampa removed
   eom removed
   exiftool removed
   ffplay removed
   ffprobe removed
   firefox removed
```

```
└─$sudo firecfg --debug
parrotseckiosk 0 0 0 0
Removing all firejail symlinks:

Configuring symlinks in /usr/local/bin based on firecfg.config
found VirtualBox in directory /bin
   VirtualBox created
found atril in directory /bin
   atril created
Adding user parrotseckiosk to Firejail access database in /usr/local/etc/firejail/firejail.users
User parrotseckiosk already in the database

Loading AppArmor profile

Configuring symlinks in /usr/local/bin based on local firejail config directory
parrotseckiosk 1000 1001 1000 1001

Fixing desktop files in /home/parrotseckiosk/.local/share/applications
checking profile for anon-change-identity.desktop
checking profile for anon-check-ip.desktop
checking profile for anon-gui.desktop
checking profile for anon-surf-start.desktop
checking profile for anon-surf-stop.desktop
checking profile for atril.desktop
found /usr/local/etc/firejail/atril.profile
checking profile for bleachbit-root.desktop
checking profile for brasero-nautilus.desktop
checking profile for brasero.desktop
found /usr/local/etc/firejail/brasero.profile
checking profile for ca.desrt.dconf-editor.desktop
found /usr/local/etc/firejail/dconf-editor.profile
   ca.desrt.dconf-editor.desktop skipped: file exists
checking profile for cact.desktop
checking profile for caja-autorun-software.desktop
checking profile for caja-browser.desktop
checking profile for caja-computer.desktop
```

Abusive ISPs

These are Internet Service Providers that have been found to tamper with your DNS (or OpenNIC related) traffic, do note that this list is only for previously mentioned abuse, nothing else.

Is my ISP intercepting DNS traffic?

Some abusive ISPs will intercept DNS traffic on port 53 and return results from their own servers instead. This makes access to alternative TLDs difficult, and is a privacy concern as it allows the ISPs to carry out more detailed logging of the domains you resolve.

Some OpenNIC DNS servers also listen on an alternative port (generally 5353) which is less likely to be tampered with by ISPs.

To test if an ISP is tampering with DNS traffic, you can use the dig command from the `dnsutils` package. Select a server from the Tier 2 page which supports an alternative port. In my example I have used `106.186.17.181`. First, try querying for the root zone (`.`) on the default port:

```
dig SOA . @106.186.17.181
...
.       58346   IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2015080300 1800 900 604800 86400
```

You can see from the returned SOA above that the DNS request has been hijacked by the ISP as `a.root-servers.net` is not an OpenNIC DNS server. If the SOA you get looks more like the one below, then your ISP is probably not hijacking your DNS requests.

Now try again on the alternative port:

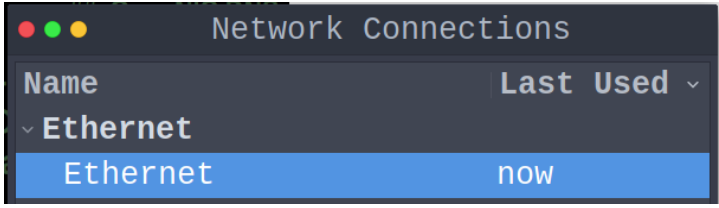
```
dig SOA . @106.186.17.181 -p 5353
...
.       86319   IN      SOA     ns0.opennic.glue. hostmaster.opennic.glue. 2015080301 1800 900 604800 3600
```

You can see that the SOA returned is OpenNIC's, meaning no hijacking has taken place on the alternative port. If this result differs from the previous result or the first result times out with `connection timed out; no servers could be reached`, then your ISP is likely to be hijacking DNS.

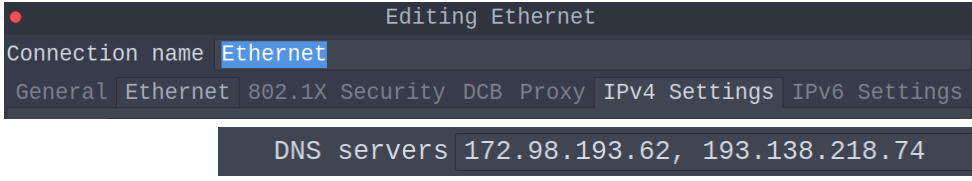
OpenNIC Abusive ISPs Wiki

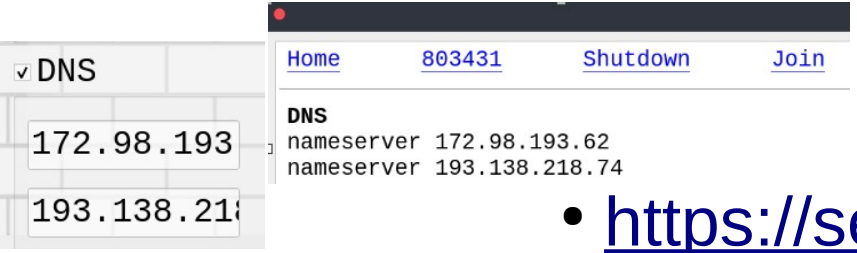
Mullvad|OpenNIC DNS Servers

```
##  ------------------------------------------------- ##
##     [+] Primary DNS - OpenNIC
##     [+] Secondary DNS - Mullvad DNS
##  ------------------------------------------------- ##
nmcli> set ipv4.dns 172.98.193.62,193.138.218.74
nmcli> set ipv4.dns 172.98.193.62          ## OpenNIC DNS
nmcli> set ipv4.dns 193.138.218.74         ## Mullvad DNS
└──  $nmcli -g ip4.dns connection show Ethernet
172.98.193.62 | 193.138.218.74
```




```
firejail --dns=172.98.193.62 --dns=193.138.218.74

└──  $firejail --dns.print=706943
Switching to pid 706946, the first
nameserver 172.98.193.62
nameserver 193.138.218.74
```





- https://servers.opennic.org/

Firetools - Using A Trustworthy DNS

Home 803431 Shutdown Join

DNS
nameserver 172.98.193.62
nameserver 193.138.218.74

Firejail reduces the risk of security breaches by restricting the running environment of untrusted applications using the latest Linux kernel sandboxing technologies.

Step 3: Configure the sandbox

File System
- [x] Restrict /home directory
- [x] Restrict /dev directory
- [x] Restrict /tmp directory
- [x] Restrict /mnt and /media

Networking
- (•) System network
- ( ) Namespace
- ( ) Disable networking

- [x] DNS

.98.193.62

138.218.74

Home Directory
- [ ] .config
- [ ] Desktop
- [ ] Templates
- [x] Downloads
- [ ] Public
- [ ] Documents
- [ ] Music
- [ ] Pictures
- [ ] Videos

- [x] Network Protocol
  - [x] unix
  - [x] inet
  - [ ] inet6
  - [ ] netlink
  - [ ] packet

Help Go Back Continue Cancel

Firejail reduces the risk of security breaches by restricting the running environment of untrusted applications using the latest Linux kernel sandboxing technologies.

Step 3: Configure the sandbox... continued...

Multimedia
- [ ] Disable sound
- [x] Disable video camera devices
- [x] Disable CD-ROM/DVD devices
- [ ] Disable TV/DVB devices
- [ ] Disable 3D acceleration
- [ ] Disable X11 support

Kernel
- [x] Enable seccomp-bpf
- [x] Disable all Linux capabilities
- [x] Restricted user namespace (noroot)

Help Go Back Continue Cancel

DNS Whitelisting API

Due to repeated amplification attacks, Tier 2 servers now have the option to only allow access from registered IP addresses. Please ch
2 wiki page to determine which servers are making use of this feature.

Privacy is important, so we have taken steps to ensure that your identity is protected. When you register your IP address, your userna
of the servers. The servers will only receive a list of what addresses have been registered, nothing else.

Advantages

- Because these servers do not respond to attackers, you will receive faster and more reliable responses to your DNS queries.

Disadvantages

- When your IP address changes, there may be a 1-2 minute delay before you are able to make queries again. IP address changes m
intervals depending on your ISP.
- To circumvent any problems, you should always have a non-whitelisted DNS server as the last entry in your list. This will allow co
lookups even while your new IP address is being registered.

You may register multiple IP addresses under your account. After 4 weeks your old addresses will expire and be removed from the sys

Registering Static/Dynamic IP

Linux

First you need an OpenNIC member account. Visit http://www.opennicproject.org/members/ and create a new account, or log in to yo

After signing in, towards the bottom of the page you will see a box marked “If you wish to register your IP for whitelisting”. The comma your username and a unique hash to authenticate you.

Run the following command to install a script and cron job to update the whitelist with your IP every 5 minutes. If you are in a text-only see this script for install instructions.

```
bash <(curl -s https://raw.githubusercontent.com/CalumMc/OpenNIC-Whitelist-Updater/master/install.sh)
```

Optionally, you can visit this URL to verify the content of the script. You will need to enter your username and the unique hash, both of the command on http://www.opennicproject.org/members/. For more details please see OpenNIC Whitelist Updater on GitHub.

Windows

An easy-to-use Windows IP update script will be available in the near future.

https://opennicproject.org/members

Xe1phix-[Dig]-Cheatsheet

```
## ---------------------------------------------------------------------------------------- ##
   dig +short +identify $Server                  ## see what name server
                                                 ## or whose cache is providing answers
## ---------------------------------------------------------------------------------------- ##
   dig $Domain.com | grep status                 ## receive the NXDOMAIN status.
## ---------------------------------------------------------------------------------------- ##
   dig $Domain.com | grep Query                  ## query time only
## ---------------------------------------------------------------------------------------- ##
   dig $Domain.com $Type                         ## a, mx, ns, soa, srv, txt, any
## ---------------------------------------------------------------------------------------- ##
   dig -x $TargetIP                              ## Pointer records
## ---------------------------------------------------------------------------------------- ##
   dig @NameServerIP $Domain.com axfr            ## Zone transfer
## ---------------------------------------------------------------------------------------- ##
   dig @NameServerIP $Domain.com afro            ## Forward zone transfer
## ---------------------------------------------------------------------------------------- ##
   dig @$IPAddr $Domain +norecurse               ## Non recursive query (cache lookup)
## ---------------------------------------------------------------------------------------- ##
   dig MX +short $Domain                         ## Perform MX Record Lookup
## ---------------------------------------------------------------------------------------- ##
   dig ns $Domain                                ## List the Name Servers for google.com
## ---------------------------------------------------------------------------------------- ##
   dig a $Domain.com @nameserver                 ## Perform DNS IP Lookup
## ---------------------------------------------------------------------------------------- ##
```

Xe1phix-[NSLookup]-Cheatsheet

```
## ------------------------------------------------------------------------------------ ##
   nslookup $Domain.com                          ## Query A and PTR Records
## ------------------------------------------------------------------------------------ ##
   nslookup $Domain.com x.x.x.x                  ## Query A and PTR record
                                                 ## using a different name server
## ------------------------------------------------------------------------------------ ##
   nslookup -debug google.com                    ## Debug Mode for nslookup
## ------------------------------------------------------------------------------------ ##
   nslookup -query=ns $Domain.com                ## Query Nameserver records
## ------------------------------------------------------------------------------------ ##
   nslookup -querytype=mx $Domain.com            ## Query MX record
## ------------------------------------------------------------------------------------ ##
   nslookup set type=mx $Domain.com              ## Interactive option
## ------------------------------------------------------------------------------------ ##
   nslookup -norecursive $Domain.com             ## Non Recursive lookup
## ------------------------------------------------------------------------------------ ##
   nslookup recursive $Domain.com                ## Recursive lookup
## ------------------------------------------------------------------------------------ ##
   nslookup ns4.google.com                       ## Resolve the IP Address for ns4.google.com
## ------------------------------------------------------------------------------------ ##


## ------------------------------------------------------------------------------------ ##
   nslookup server $Server set type=any ls -d $Target      ## DNS zone transfer
## ------------------------------------------------------------------------------------ ##
```

Xe1phix-[Host]-Cheatsheet

```
## ------------------------------------------------------------------------ ##
      host -t ns $Domain                        ## Show name servers
      host -t mx $Domain                        ## Show mail servers
      host -t CNAME $Domain                     ## CNAME Records
      host -t SOA $Domain                       ## SOA Records
      host -t TXT $Domain                       ## TXT Records
      host -t DNSKEY $Domain                    ## DNSKEY Records
      host -t AXFR $Domain                      ## AXFR Records
## ------------------------------------------------------------------------ ##
      host -l $Domain $NameServer               ## Zone transfer
## ------------------------------------------------------------------------ ##
      host -C $Domain                           ## SOA Records
## ------------------------------------------------------------------------ ##
      host -a $Domain                           ## All Query Types
      host -v -t ANY $Domain                    ## enables verbose output
## ------------------------------------------------------------------------ ##
```

Xe1phix-[DNSRecon]-Cheatsheet

```
## ------------------------------------------------------------------------------------------ ##
    dnsrecon -t rvs -i 192.1.1.1,192.1.1.20          ## Reverse Lookup For IP Range:
## ------------------------------------------------------------------------------------------ ##
    dnsrecon -t std -d $Domain.com                   ## Retrieve Standard DNS Records:
## ------------------------------------------------------------------------------------------ ##
    dnsrecon -t brt -d $Domain.com -w hosts.txt      ## Enumerate SubDornains:
## ------------------------------------------------------------------------------------------ ##
    dnsrecon -d $Domain.com -t axfr                  ## DNS Zone Transfer:
## ------------------------------------------------------------------------------------------ ##
    dnsrecon --type snoop -n $Server -D $Dict        ## Cache Snooping
## ------------------------------------------------------------------------------------------ ##
    dnsrecon -d $Host -t zonewalk                    ## Zone Walking
## ------------------------------------------------------------------------------------------ ##
```

NetworkManager

```
#nmcli general status
STATE      CONNECTIVITY  WIFI-HW  WIFI     WWAN-HW  WWAN
connected  full          enabled  enabled  enabled  enabled
```

```
[parrotsec-kiosk@parrot]-[~]
$nmcli con show
NAME             UUID                                  TYPE      DEVICE
Ethernet         4c335b2b-2cd4-4078-a7c1-e2c6f018f07f  ethernet  eth0
mullvad_se_sto   37c91a90-e8fa-444e-9fc9-9ca680c87997  vpn       --
```

```
##=============================##
##   [+] Bring The connection Down:
##-============================-##
nmcli con down "Wired connection 1"
```

```
#rfkill list
#rfkill block all
#rfkill unblock all
```

```
[parrotsec-kiosk@parrot]-[~]
$nmcli radio all off
```

```
iw reg set US
iwconfig wlan0 txpower 25
```

```
iw dev wlan0 station dump
iw dev wlan0 scan | less
iwlist wlan0 scanning | egrep "ESSID|Channel"
iwconfig wlan0 mode monitor channel 3
iw scan
```

```
nmcli> set ipv6.method ignore
nmcli> set 802-3-ethernet.wake-on-lan ignore
nmcli> set ipv6.never-default yes
nmcli> set connection.autoconnect no
nmcli> set ipv6.ignore-auto-dns yes
nmcli> set ipv6.dhcp-send-hostname no
nmcli> set ipv4.dns 198.98.49.91,193.138.218.74
nmcli> set ipv6.ignore-auto-routes yes
```

```
capability net_admin,
/etc/NetworkManager/NetworkManager.conf r,
/etc/NetworkManager/VPN/ r,
/etc/NetworkManager/conf.d/ r,
/etc/NetworkManager/dispatcher.d/ r,
/etc/NetworkManager/dispatcher.d/pre-down.d/ r,
/etc/NetworkManager/dispatcher.d/pre-up.d/ r,
/etc/NetworkManager/system-connections/ r,
/etc/NetworkManager/system-connections/* rw,
/etc/dhcp/dhclient.conf r,
/etc/hostname r,
/etc/network/interfaces r,
/etc/network/interfaces.d/ r,
/etc/udev/udev.conf r,
/sbin/dhclient Px,
/sys/bus/ r,
/sys/class/ r,
/sys/class/net/ r,
```

NetworkManager AppArmor

AppArmor - Overview

AppArmor - kernel enhancement to confine programs to a limited set of resources.

AppArmor confinement is provided via **profiles** loaded into the kernel via **apparmor_parser**(8),

AppArmor can operate in two modes: **enforcement**, and **complain or learning**:

- **enforcement** - Profiles loaded in enforcement mode will result in enforcement of the policy defined in the profile as well as reporting policy violation attempts to syslogd.

- **complain** - Profiles loaded in "complain" mode will not enforce policy. Instead, it will report policy violation attempts. This mode is convenient for developing profiles. To manage complain mode for individual profiles the utilities **aa-complain**(8) and **aa-enforce**(8) can be used. These utilities take a

ERRORS

When a confined process tries to access a file it does not have permission to access, the kernel will report a message through audit, similar to:

```
audit(1386511672.612:238): apparmor="DENIED" operation="exec"
  parent=7589 profile="/tmp/sh" name="/bin/uname" pid=7605
  comm="sh" requested_mask="x" denied_mask="x" fsuid=0 ouid=0

audit(1386511672.613:239): apparmor="DENIED" operation="open"
  parent=7589 profile="/tmp/sh" name="/bin/uname" pid=7605
  comm="sh" requested_mask="r" denied_mask="r" fsuid=0 ouid=0

audit(1386511772.804:246): apparmor="DENIED" operation="capable"
  parent=7246 profile="/tmp/sh" pid=7589 comm="sh" pid=7589
  comm="sh" capability=2  capname="dac_override"
```

```
┌─[root@parrotseckiosk-optiplex990]─[/home/parrotseckiosk/
└──╼ #aa-status --verbose
apparmor module is loaded.
183 profiles are loaded.
145 profiles are in enforce mode.
   /etc/cron.daily/slocate.cron
   /etc/cron.daily/tmpwatch
   /usr/NX/bin/nxclient
   /usr/X11R6/bin/acroread
   /usr/bin/apropos
   /usr/bin/evolution-2.10
   /usr/bin/fam
   /usr/bin/gaim
   /usr/bin/i2prouter
   /usr/bin/i2prouter//sanitized_helper
   /usr/bin/lxc-start
   /usr/sbin/sshd
   /usr/sbin/sshd//passwd
   /usr/sbin/unbound
   /usr/sbin/useradd
   /usr/sbin/useradd//pam_tally2
   /usr/sbin/userdel
   /usr/sbin/vsftpd
   /usr/sbin/xinetd
   apt-cacher-ng
   avahi-daemon
   dhclient
   dhclient-script
   dhcpcd
   firejail-default
   identd
   klogd
   system_i2p
   system_i2p//sanitized_helper
   system_tor
   tcpdump
   thunderbird
   thunderbird//browser_java
   thunderbird//browser_openjdk
   thunderbird//gpg
   thunderbird//sanitized_helper
   torbrowser_firefox
   torbrowser_tor
   virt-aa-helper
```

Apparmor - Status

Apparmor – Parser + Enforce

```
┌─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #cp -v /usr/share/apparmor/extra-profiles/usr.sbin.dhcpd /etc/apparmor.d/
'/usr/share/apparmor/extra-profiles/usr.sbin.dhcpd' -> '/etc/apparmor.d/usr.sbin.dhcpd'
┌─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #apparmor_parser --verbose -r /etc/apparmor.d/usr.sbin.dhcpd
Replacement succeeded for "/usr/sbin/dhcpd".
```

```
┌─[✗]─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #aa-enforce /etc/apparmor.d/usr.sbin.dhcpd
Setting /etc/apparmor.d/usr.sbin.dhcpd to enforce mode.
```

```
┌─[root@ParrotSec-Kiosk]─[/home/xe1phix]
└──╼ #/etc/init.d/apparmor reload
Reloading apparmor configuration (via systemctl): apparmor.service.
```

```
└──╼ $firejail --list
216203:parrotseckiosk::firejail --dns=172.98.193.62 --dns=193.138.218.74 --apparmor --protocol=unix,inet,netlink --seccomp --nonewprivs
 --caps.drop=all --shell=none --whitelist=~/.mozilla --private-tmp --private-dev --netfilter --nogroups --nodvd /usr/bin/firefox
```

```
└──╼ $firemon --apparmor 216203
  AppArmor: firejail-default enforce
```

```
└──╼ $firejail --apparmor.print=1243548
1243548:parrotseckiosk::/usr/local/bin/firejail /usr/bin/firefox
  AppArmor: firejail-default enforce
```

Firejail – Print Apparmor

```
┌─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #firejail --list
193119:parrotsec-kiosk::firejail --profile=/etc/firejail/firefox.profile --protocol=unix,in
et,netlink,packet --dns=193.138.218.74 --dns=10.8.0.1 --dns=139.99.96.146 /usr/bin/firefox
┌─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #firejail --apparmor.print=193119
193119:parrotsec-kiosk::firejail --profile=/etc/firejail/firefox.profile --protocol=unix,in
et,netlink,packet --dns=193.138.218.74 --dns=10.8.0.1 --dns=139.99.96.146 /usr/bin/firefox
  AppArmor: firejail-default enforce
```

Xe1phix - Service Killer Script

```
for SERVICE in                                  \
    greenbone-security-assistant.service        \
    greenbone-security-assistant                \
    selinux-autorelabel                         \
    selinux-autorelabel-mark.service            \
    winbind                     \
    wine;
do
    if [ -e /etc/init.d/$SERVICE ]; then
        /etc/init.d/$SERVICE stop
        $SERVICEBIN $SERVICE stop
        ## /sbin/chkconfig --level 0123456 $SERVICE off
        $UPDATE $SERVICE disable
    else
        echo "SERVICE doesn't exist in /etc/init.d/ ($SERVICE)."
        echo "Trying to disable it through systemctl..."
        systemctl stop $SERVICE
        systemctl disable $SERVICE
        ## /sbin/chkconfig --level 12345 $SERVICE off
        ## systemctl mask $SERVICE                          ## Uncomment this line to mask services
        ## chkconfig --del
        update-rc.d $SERVICE disable
        ## update-rc.d $SERVICE remove
    fi
done
```

[Xe1phix-Disable-Services-ParrotSec-Kiosk-[.v9.6.8.].sh]

Secure Apt

A secure hash function (a type of checksum) is a method of taking a file and boiling it down to a reasonably short number that will uniquely identify the content of the file, even if people are deliberately trying to create a pair of different files with the same checksum or create a new file that matches a previous checksum. APT was originally designed around MD5 but people have since managed to construct collisions and so support for newer hash functions has been added.

Public key cryptography is based on pairs of keys, a public key and a private key. The public key is given out to the world; the private key must be kept a secret. Anyone possessing the public key can encrypt a message so that it can only be read by someone possessing the private key. It's also possible to use a private key to sign a file, not encrypt it. If a private key is used to sign a file, then anyone who has the public key can check that the file was signed by that key. Anyone who doesn't have the private key can't forge such a signature.
gpg is the tool used in secure apt to sign files and check their signatures.

A Debian archive contains a Release file, which is updated each time any of the packages in the archive change. Among other things, the Release file contains some checksums of other files in the archive. An excerpt of an example Release file:
secure apt adds a gpg signature for the Release file. This is put in a file named Release.gpg that's shipped alongside the Release file.

Secure apt always downloads Release.gpg files when it's downloading Release files, and if it cannot download the Release.gpg, or if the signature is bad, it will complain, and will make note that the Packages files that the Release file points to, and all the packages listed therein, are from an untrusted source. Here's how it looks during an apt-get update:

```
└─ #apt-get update
W: GPG error: http://ftp.us.debian.org testing Release: The following signatures
        couldn't be verified because the public key is not available: NO_PUBKEY 010908312D230C5F
```

So the security of the whole system depends on there being a Release.gpg file, which signs a Release file, and of apt checking that signature using gpg. To check the signature, it has to know the public key of the person who signed the file. These keys are kept in apt's own keyring (/etc/apt/trusted.gpg), and managing the keys is where secure apt comes in.

Basic Apt Concepts

```
##-=========================================================================-##
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~##
##-=========================================================================-##
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian Debian 11 (Bullseye): Bullseye Stable Release Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/release-11.asc
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 11 (Bullseye): Archive Automatic Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/archive-key-11.asc
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 11 (Bullseye): Security Archive Automatic Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/archive-key-11-security.asc
##
##-=========================================================================-##
```

```
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~##
##-=========================================================================-##
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 10 (Buster): Stable Release Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/release-10.asc
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 10 (Buster): Archive Automatic Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/archive-key-10.asc
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 10 (Buster): Security Archive Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/archive-key-10-security.asc
##
##-=========================================================================-##
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~##
##-=========================================================================-##
```

```
##-=========================================================================-##
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~##
##-=========================================================================-##
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 9 (Stretch): Stable Release Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/release-9.asc
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 9 (Stretch): Archive Automatic Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/archive-key-9.asc
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 9 (Stretch): Security Archive Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/archive-key-9-security.asc
##
##
```

```
##-=========================================================================-##
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~##
##-=========================================================================-##
##
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 8 (Jessie): Stable Release Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/release-8.asc
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 8 (Jessie): Archive Automatic Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/archive-key-8.asc
##
## ------------------------------------------------------------------------ ##
echo "Downloading The  [+] Debian 8 (Jessie): Security Archive Signing Key..."
##\________________________________________________________________________/##
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://ftp-master.debian.org/keys/archive-key-8-security.asc
##
##
##-=========================================================================-##
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~##
##-=========================================================================-##
```

```
echo "##-==========================================-##"
echo "      [+] Fetch The ParrotSec Repository Signing Key:      "
echo "##-==========================================-##"
└─── $wget -q -O - https://deb.parrotsec.org/parrot/misc/parrotsec.gpg | gpg --import
gpg: key 363A96A5CEA9EA27: public key "Parrot Project (2020-2022) <team@parrotsec.org>" imported
gpg: Total number processed: 1
gpg:               imported: 1
## ------------------------------------------------------------------------------------------ ##
echo "Signing The  [+] Parrot Security:  Archive Automatic Signing Key"
## ____________________________________________________________ ##
gpg --lsign 0x0A912CDE87F9972236AF8B50363A96A5CEA9EA27
```

ParrotSec Apt Repository Signing Key

```
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] I2P Project - ZZZ On I2P Signing Key..."
##\____________________________________________________/##
gpg --verbose --default-key 0x3BC00B17180C200A --lsign 0x2D3D2D03910C6504C1210C65EE60C0C8EE7256A8

## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] I2P (KillYourTV) Debian Package Repository..."
##\____________________________________________________/##
gpg --verbose --default-key 0x3BC00B17180C200A --lsign 0x7840E7610F28B904753549D767ECE5605BCF1346
## ------------------------------------------------------------------------------------------ ##
```

GnuPG-Signing-I2P-Keys.sh

```
echo "##-==================================-##"
echo "      [+] Import Mullvads GPG Signing Key:      "
echo "##-==================================-##"
gpg --keyid-format 0xlong --import mullvad-support-mail.asc
gpg --keyid-format 0xlong --import mullvad-code-signing.asc
echo "##-============================-##"
echo "      [+] Sign Mullvads GPG Key:      "
echo "##-============================-##"
gpg --lsign A1198702FC3E0A09A9AE5B75D5A1D4F266DE8DDF
```

Mullvad GnuPG Code Signing Key

```
##-==========================================================================================-##
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] Debian 9 (Stretch): Archive Signing Key..."
##\____________________________________________________/##
gpg --lsign 0xE1CF20DDFFE4B89E802658F1E0B11894F66AEC98
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] Debian 9 (Stretch): Security Archive Signing Key..."
##\____________________________________________________/##
gpg --lsign 0x6ED6F5CB5FA6FB2F460AE88EEDA0D2388AE22BA9
## ------------------------------------------------------------------------------------------ ##
##-==========================================================================================-##
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] Debian 10 (Buster): Archive Automatic Signing Key..."
##\____________________________________________________/##
gpg --lsign 0x80D15823B7FD1561F9F7BCDDDC30D7C23CBBABEE
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] Debian 10 (Buster): Security Archive Signing Key..."
##\____________________________________________________/##
gpg --lsign 0x5E61B217265DA9807A23C5FF4DFAB270CAA96DFA
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] Debian 10 (Buster): Stable Release Key..."
##\____________________________________________________/##
gpg --lsign 0x6D33866EDD8FFA41C0143AEDDCC9EFBF77E11517
## ------------------------------------------------------------------------------------------ ##
##-==========================================================================================-##
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] Debian 11 (Bullseye): Archive Automatic Signing Key..."
##\____________________________________________________/##
gpg --lsign 0x1F89983E0081FDE018F3CC9673A4F27B8DD47936
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] Debian 11 (Bullseye): Security Archive Automatic Signing Key..."
##\____________________________________________________/##
gpg --lsign 0xAC530D520F2F3269F5E98313A48449044AAD5C5D
## ------------------------------------------------------------------------------------------ ##
echo "Signing  [+] Debian 11 (Bullseye): Bullseye Stable Release Key..."
##\____________________________________________________/##
gpg --lsign 0xA4285295FC7B1A81600062A9605C66F00D6C9793
## ------------------------------------------------------------------------------------------ ##
##-==========================================================================================-##
```

Debian Archive Signing Keys

```
└─# apt-get update
Hit:1 https://mirror.clarkson.edu/parrot rolling InRelease
Hit:2 https://mirror.clarkson.edu/parrot rolling-security InRelease
Ign:3 http://deb.debian.org/debian stretch InRelease
Get:4 http://deb.debian.org/debian stretch-updates InRelease [93.6 kB]
Get:5 http://deb.debian.org/debian-security stretch/updates InRelease [53.0 kB]
Get:6 http://deb.debian.org/debian buster InRelease [122 kB]
Get:7 http://deb.debian.org/debian buster-updates InRelease [51.9 kB]
Get:8 http://deb.debian.org/debian-security buster/updates InRelease [65.4 kB]
Get:9 http://deb.debian.org/debian stretch Release [118 kB]
Get:10 http://deb.debian.org/debian-security stretch/updates/main amd64 Packages [709 kB]
Get:11 http://deb.debian.org/debian-security stretch/updates/main Translation-en [326 kB]
Get:12 http://deb.debian.org/debian stretch Release.gpg [3,177 B]
Get:13 http://deb.debian.org/debian-security buster/updates/main amd64 Packages [302 kB]
Get:14 http://deb.debian.org/debian-security buster/updates/main Translation-en [158 kB]
Get:15 https://mirrors.mit.edu/parrot rolling InRelease [14.4 kB]
Get:16 https://mirrors.mit.edu/parrot rolling-security InRelease [8,599 B]
Get:17 https://mirrors.mit.edu/parrot rolling/main amd64 Packages [18.1 MB]
Get:18 https://mirrors.ocf.berkeley.edu/parrot rolling InRelease [14.4 kB]
Get:19 https://mirrors.ocf.berkeley.edu/parrot rolling-security InRelease [8,599 B]
Get:20 https://mirrors.ocf.berkeley.edu/parrot rolling/main amd64 Packages [18.1 MB]
Get:21 https://deb.parrot.sh/parrot rolling InRelease [14.4 kB]
Get:22 https://deb.parrot.sh/parrot rolling-security InRelease [8,599 B]
Get:23 https://deb.parrot.sh/parrot rolling/non-free amd64 Packages [260 kB]
Get:24 https://mirrors.mit.edu/parrot rolling/contrib amd64 Packages [152 kB]
Get:25 https://mirrors.mit.edu/parrot rolling/non-free amd64 Packages [260 kB]
```

GnuPG – Asymmetric Encryption

```
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
##    [+] Create A GPG Key:
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
gpg --enable-large-rsa --full-gen-key
```

```
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
##    [?] The Owner Exports His GPG Public Key For The Recipient
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
gpg --export --armor Xe1phix > Xe1phix.asc
```

```
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
##    [+] Import A GPG Public Key:
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
gpg --keyid-format 0xlong --import Xe1phix.asc
```

```
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
##     [+] Encrypt & Sign A File (Using Xe1phix As The Recipient):
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
gpg --encrypt --sign --armor --recipient Xe1phix@mail.i2p $File
gpg -se -r Xe1phix@mail.i2p $File
```

```
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
##     [+] Verify The Recipients Signature File Against The Base File:
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
gpg --verify --keyid-format 0xlong $file.txt.gpg $file.txt
```

Xe1phix-GnuPG-Signing-Files-With-Friends-Cheatsheet.sh

GnuPG – Symmetric Encryption

```
echo "##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##"
echo "    [+] Create A GPG Key:"
echo "##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##"
gpg --enable-large-rsa --full-gen-key
```

```
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
##    [+] Encrypts $File With A Symmetric Cipher Encryption (Using A Passphrase):
##-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-##
## ------------------------------------------------------------------------------------------------------------ ##
##    [?] Uses The AES-256 Cipher Algorithm To Encrypt The Passphrase
##    [?] Uses The SHA-512 Digest Algorithm To Mangle The Passphrase
##    [?] Mangles The Passphrase For 65536 Iterations
## ------------------------------------------------------------------------------------------------------------ ##
```

```
└─■ $gpg --verbose --symmetric --cipher-algo aes256 --digest-algo sha512 --cert-digest-algo sha512 --s2k-mode 3 --s2k-count 65011712 --s2k-cipher-algo AES256 --s2k-digest-algo SHA512 mullvad-n.pem
gpg: pinentry launched (952861 gnome3 1.1.0 /dev/pts/1 xterm-256color :0)
gpg: pinentry launched (952881 gnome3 1.1.0 /dev/pts/1 xterm-256color :0)
gpg: using cipher AES256
gpg: writing to 'mullvad-n.pem.gpg'
```

Xe1phix-GnuPG-Cheatsheet.sh

The best and most reliable VPN Services for your Privacy

We have compared 185 different VPN providers, but our strict criteria left only the two best providers. Our recommended providers are operating outside the USA or other Five Eyes countries, use a strong encryption, accept Crypto currencies or cash payments, support OpenVPN, have a no logging policy and have a long history of operating.

Mullvad: 60 Euro Yearly

Win | Android | iOS | Mac | Linux | Bitcoin | Cash

Based in Sweden. Operating since 2009. Accepts Bitcoin, BCH and Cash. Native desktop and mobile clients are available for Android and iOS and are easy to use. Money back guarantee for 30 days.

Amount of servers in Oct 2021: 763 VPN servers, in 38 different countries. Source

ProtonVPN: Limited free version available, otherwise 48 EUR Yearly

Freemium | Win | Android | iOS | Mac | Linux | Bitcoin

Based in Switzerland. Operating since 2016. Accepts Bitcoin, but you need an existing account or contact their support team in advance. Easy to use native desktop and mobile clients are available for Android and iOS.

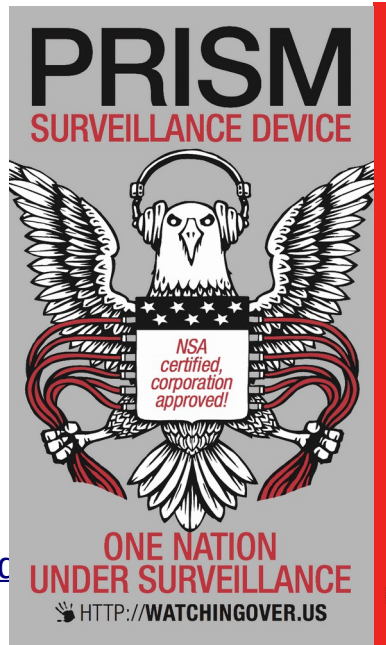
Amount of servers in Oct 2021: 1200+ VPN servers available in 55 different countries. Source

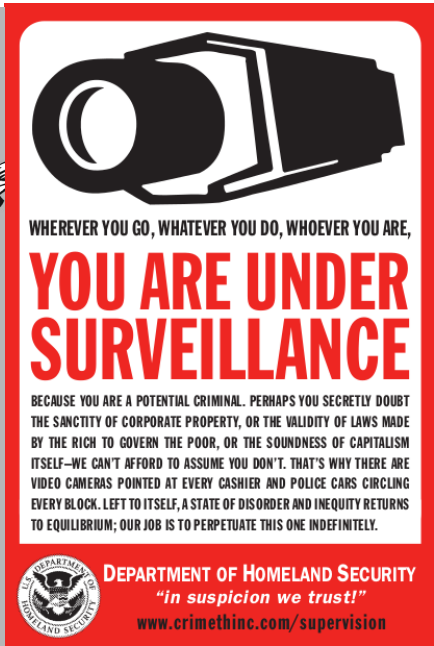
Mullvad Activism

Mullvad (Amagicom AB) offers a VPN service with a focus on the right to privacy and freedom of expression without censorship, both upheld in the UN's Universal Declaration of Human Rights (articles 12 and 19) and the European Convention on Human Rights (articles 8 and 10).

- European Court of Human Rights passes legislation to make mass surveillance official against human rights:
- **[European Court of Human Rights - mass surveillance announcement]**

- Data Collection Techniques | Whonix Wiki
- Surveillance Capabilities | Whonix Wiki
- Abusive ISPs [OpenNIC Wiki]
- https://www.privacytools.io/providers/vpn/#mullvad
- https://prism-break.org/en/all/#vpn
- Privacy is A Universal Right | Mullvad Blog
- https://mullvad.net/en/blog/2020/6/25/results-available-audit-mullvad-app/
- Debian-Privacy-Server-Guide
- Rebel-Alliance-Tech-Manual
- Tradecraft DOs and DON'Ts
- A Declaration of The Independence of Cyberspace | Electronic Frontier Found





Mullvad Features

- No logging policy
- Open source clients
- WireGuard support
- Mullvad Android client available on F-Droid.
- Built-in killswitch to block internet connections outside of the VPN.

https://www.privacytools.io/providers/vpn/#mullvad

Fetching Mullvad GPG Keys Using GPG And SKS Keyservers

```
└─ $gpg --keyserver pool.sks-keyservers.net --recv-keys A1198702FC3E0A09A9AE5B75D5A1D4F266DE8DDF
gpg: key D5A1D4F266DE8DDF: "Mullvad (code signing) <admin@mullvad.net>" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
└─ $gpg --keyid-format 0xlong --fingerprint 0xA1198702FC3E0A09A9AE5B75D5A1D4F266DE8DDF
pub   rsa4096/0xD5A1D4F266DE8DDF 2016-10-27 [SC]
      Key fingerprint = A119 8702 FC3E 0A09 A9AE  5B75 D5A1 D4F2 66DE 8DDF
uid                   [  full  ] Mullvad (code signing) <admin@mullvad.net>
sub   rsa4096/0xC187D22C089EF64E 2016-10-27 [E]
sub   rsa4096/0xA26581F219C8314C 2016-10-27 [S]
```

Xe1phix-GnuPG-[Mullvad]-Trust-Verified-Signatures.sh

https://mullvad.net/en/help/verifying-signatures/

Securely Fetching Mullvads GPG Signing Key Using Curl (TLS)

```
└─ #curl --verbose --progress-bar --ssl-reqd --url https://www.mullvad.net/static/mullvad
-code-signing.asc --output mullvad-code-signing.asc
*   Trying 45.83.220.101:443...
* TCP_NODELAY set
* Connected to www.mullvad.net (45.83.220.101) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*   CAfile: /etc/ssl/certs/ca-certificates.crt
  CApath: /etc/ssl/certs
* SSL connection using TLSv1.2 / ECDHE-RSA-CHACHA20-POLY1305
* ALPN, server accepted to use h2
* Server certificate:
*  subject: CN=mullvad.net
*  start date: Sep 14 06:32:35 2020 GMT
*  expire date: Dec 13 06:32:35 2020 GMT
*  subjectAltName: host "www.mullvad.net" matched cert's "www.mullvad.net"
*  issuer: C=US; O=Let's Encrypt; CN=Let's Encrypt Authority X3
*  SSL certificate verify ok.
```

Verifying The Mullvad Binary

```
└──╼ $gpg --keyid-format 0xlong -v --verify MullvadVPN-2020.5_amd64.deb.asc Mul
lvadVPN-2020.5_amd64.deb
gpg: Signature made Thu 25 Jun 2020 03:42:23 AM CDT
gpg:                using RSA key CA83A46153BC58D69518ED49A26581F219C8314C
gpg: using subkey 0xA26581F219C8314C instead of primary key 0xD5A1D4F266DE8DDF
gpg: using subkey 0xA26581F219C8314C instead of primary key 0xD5A1D4F266DE8DDF
gpg: using pgp trust model
gpg: Good signature from "Mullvad (code signing) <admin@mullvad.net>" [full]
gpg: using subkey 0xA26581F219C8314C instead of primary key 0xD5A1D4F266DE8DDF
gpg: binary signature, digest algorithm SHA256, key algorithm rsa4096
```

Xe1phix-GnuPG-[Mullvad]-Trust-Verified-Signatures.sh

https://mullvad.net/en/help/verifying-signatures/

Mullvad – OpenVPN Details

Mullvad's OpenVPN servers

Our OpenVPN servers have the following characteristics:

- 4096 bit RSA certificates (with SHA512) are used for server authentication
- 4096 bit Diffie-Hellman parameters are used for key exchange
- DHE is utilized for perfect forward secrecy
- all available data channel ciphers on all ports are offered, including AES-256-GCM (default), AES-256-CBC, and BF-CBC
- re-keying is performed every 60 minutes.

https://mullvad.net/en/what-is-vpn/

OpenVPN In-depth Analysis

CVE-2019-14899

Published 2019-12-11T15:15:00

A vulnerability was discovered in Linux, FreeBSD, OpenBSD, MacOS, iOS, and Android that allows a malicious access point, or an adjacent user, to determine if a connected user is using a VPN, make positive inferences about the websites they are visiting, and determine the correct sequence and acknowledgement numbers in use, allowing the bad actor to inject data into the TCP stream. This provides everything that is needed for an attacker to hijack active connections inside the VPN tunnel.

The victim device connects to the access point, which for most of our testing was a laptop running create_ap. The victim device then establishes a connection with their VPN provider.

The access point can then determine the virtual IP of the victim by sending SYN-ACK packets to the victim device across the entire virtual IP space (the default for OpenVPN is 10.8.0.0/24). When a SYN-ACK is sent to the correct virtual IP on the victim device, the device responds with a RST; when the SYN-ACK is sent to the incorrect virtual IP, nothing is received by the attacker.

To quickly demonstrate this difference, we use the nping commands on the AP device running create_ap. The source IP is the gateway of our AP, the destination IP is the virtual IP assigned to the tun interface by the VPN client, ap0 is the interface create_ap created on the attacker device, and the destination MAC is the victim's wireless MAC

To mitigate CVE-2019-14899, Linux clients have two possible solutions:

- Enable strict reverse path filtering:

```
sysctl net.ipv4.conf.all.rp_filter=1
```

- Employ IPTables:

```
iptables -t raw \! -i tun0 -d 10.0.0.0/8 -j DROP
```

[CVE-2019-14899]

> OpenVPNs uses TLS control channels.
  Which are signed, and every packet on the
  control channel is authenticated
  using SHA256 HMAC signatures.
  and a unique ID for replay protection.

> The Server and Client certificates
  are hashed using SHA256 fingerprints.

> OpenVPN Servers Require the clients certificate
  is signed using a key based on RFC3280 TLS rules.

> After OpenVPN negotiates a TLS session, a new set
  of keys for protecting the tunnel data channel
  is generated and exchanged over the TLS session.

> TUN/TAP virtual network interface is created
> Which facilitates virtual point-to-point IP connection.
  Once either side of the tunnel hits the 20s TTL
  without any pings, the connection is terminated.

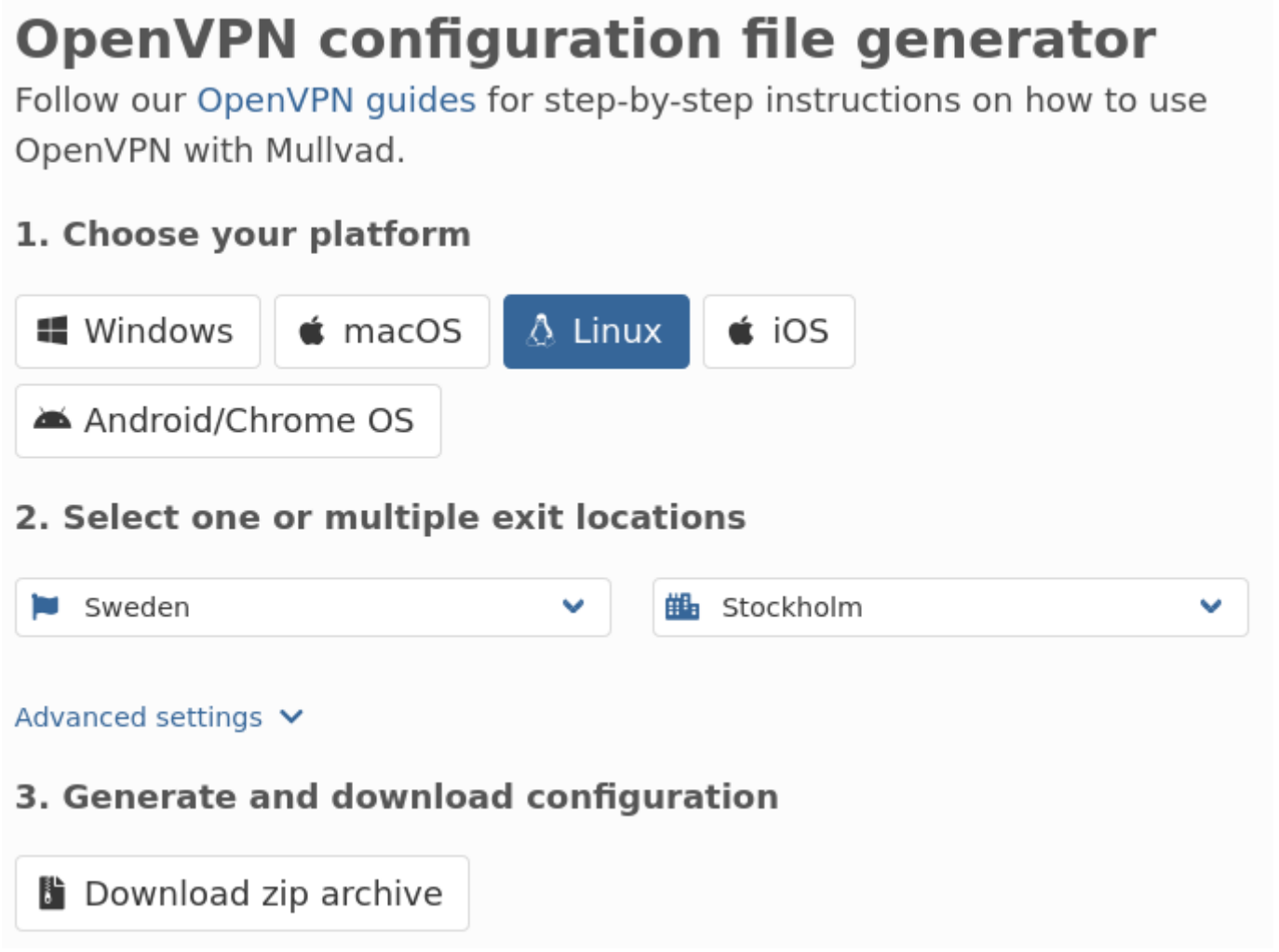
> A TUN/TAP virtual network interface is created
> Which facilitates virtual point-to-point IP connection.
> Once either side of the tunnel hits the 20s TTL
  without any pings, the connection is terminated

> subjectAltName and issuerAltName X.509 extensions are supported.

OpenVPN – Connection Types

- **Point to Point :** the most commonly used VPN. PPTP VPNs are used by remote users to connect them to the VPN network using their existing internet connection. This is a useful VPN for both business users and home users.
- **Site to Site :** is mostly used in corporate based operations. The fact that many companies have offices located both nationally and internationally, a Site-to-Site VPN is used to connect the network of the main office location to multiple offices. This is also known as an Intranet based VPN.

Note: It is generally a bad idea to use TCP for VPN connections, unless your connection to the server is very stable. High reliability sounds great in theory but any disruption (packet drop, lag spikes, etc...) to the connection will potentially snowball into a $TCPMeltdown.



- https://mullvad.net/en/account/#/openvpn-config/

Mullvad – DNS Server + OpenVPN And WireGuard SOCKS Proxy Servers

- Mullvad's DNS server IP: 193.138.218.74

When using WireGuard protocol:

SOCKS Proxy Server:
10.64.0.1 : 1080

When using OpenVPN protocol:

SOCKS Proxy Server:
10.8.0.1 : 1080

- https://mullvad.net/en/help/socks5-proxy/

Mullvad – Copying Certificates

Copy the following files to **/etc/openvpn/** (use `sudo`):

- mullvad_ca.crt
- mullvad_xx.conf
- mullvad_userpass.txt

```
#cp -v mullvad_ca.crt /etc/openvpn/ && cp -v mullvad_se_sto.conf /etc/openvpn/ &&
 cp -v mullvad_userpass.txt /etc/openvpn/ && cp -v update-resolv-conf /etc/openvpn/
'mullvad_ca.crt' -> '/etc/openvpn/mullvad_ca.crt'
'mullvad_se_sto.conf' -> '/etc/openvpn/mullvad_se_sto.conf'
'mullvad_userpass.txt' -> '/etc/openvpn/mullvad_userpass.txt'
'update-resolv-conf' -> '/etc/openvpn/update-resolv-conf'
```

- https://mullvad.net/en/help/linux-openvpn-installation/
- https://mullvad.net/en/account/#/openvpn-config/

Mullvad – Immutable Bits:

```
echo "##-==========================================-##"
echo "##    [+] Turn on The Immutable Bit For The VPN Keys & Certs:          "
echo "##-==========================================-##"
chattr +i /etc/openvpn/mullvad_ca.crt
chmod -v 0644 mullvad_ca.crt
chown -v root mullvad_ca.crt
chattr +i /etc/openvpn/mullvad_crl.pem
chmod -v 0644 mullvad_crl.pem
chmod ug+r mullvad_userpass.txt
chown -v root mullvad_userpass.txt
```

```
┌─[root@parrot]─[/]
└──╼ #lsattr -l /etc/openvpn/mullvad_ca.crt
/etc/openvpn/mullvad_ca.crt  Immutable
┌─[root@parrot]─[/]
└──╼ #chattr +i /etc/openvpn/mullvad_se_sto.conf
┌─[root@parrot]─[/]
└──╼ #lsattr -l /etc/openvpn/mullvad_se_sto.conf
/etc/openvpn/mullvad_se_sto.conf Immutable
┌─[root@parrot]─[/]
└──╼ #chattr +i /etc/openvpn/mullvad_userpass.txt
┌─[root@parrot]─[/]
└──╼ #lsattr -l /etc/openvpn/mullvad_userpass.txt
/etc/openvpn/mullvad_userpass.txt Immutable
```

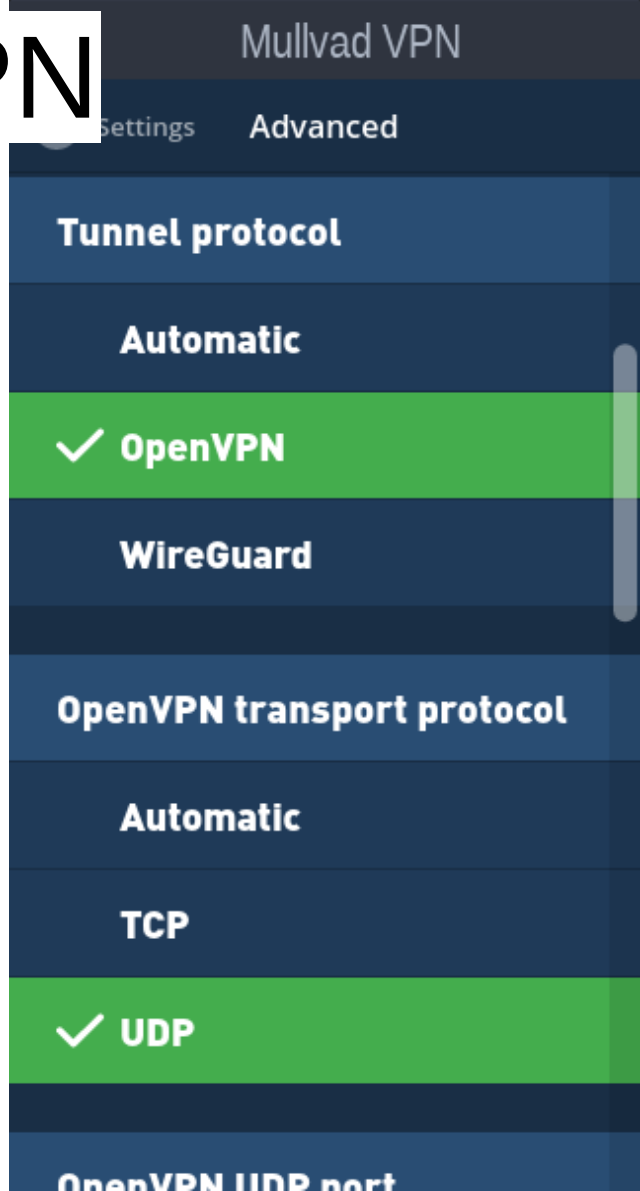
Loading The Virtual TUN/TAP Interface

```
┌─[✗]─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #modprobe --verbose tun
insmod /lib/modules/5.8.0-2parrot1-amd64/kernel/drivers/net/tun.ko
┌─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #modinfo tun
filename:       /lib/modules/5.8.0-2parrot1-amd64/kernel/drivers/net/tun
description:    Universal TUN/TAP device driver
┌─[✗]─[root@parrot]─[/etc/openvpn]
└──╼ #sudo /usr/sbin/openvpn --mktun --dev tun0
2021-07-07 18:56:06 TUN/TAP device tun0 opened
2021-07-07 18:56:06 Persist state set to: ON
```

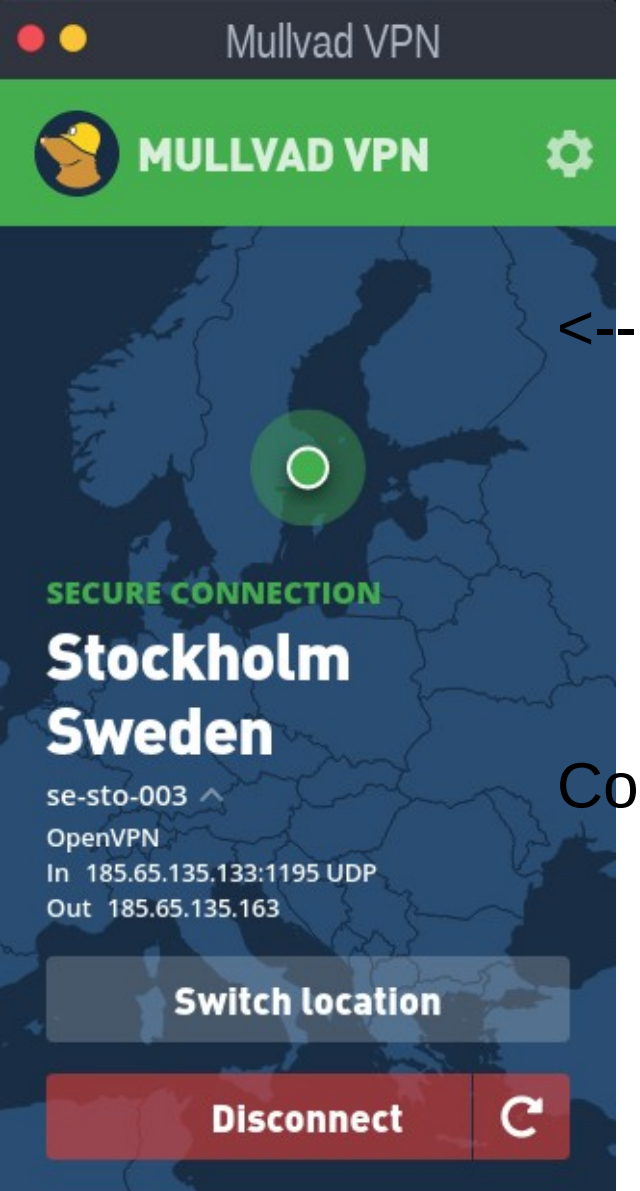
Loading The OpenVPN Config

```
└──╼ $sudo openvpn --config /etc/openvpn/mullvad_se_sto.conf
Wed Sep 30 10:17:01 2020 VERIFY OK: depth=2, C=SE, ST=Gotaland, L=Gothenburg, O=Amagicom AB
, OU=Mullvad, CN=Mullvad Root CA v2, emailAddress=security@mullvad.net
Wed Sep 30 10:17:01 2020 VERIFY OK: depth=1, C=SE, ST=Gotaland, O=Amagicom AB, OU=Mullvad,
CN=Mullvad Intermediate CA v3, emailAddress=security@mullvad.net
Wed Sep 30 10:17:01 2020 VERIFY KU OK
Wed Sep 30 10:17:01 2020 Validating certificate extended key usage
Wed Sep 30 10:17:01 2020 ++ Certificate has EKU (str) TLS Web Server Authentication, expect
s TLS Web Server Authentication
Wed Sep 30 10:17:01 2020 VERIFY EKU OK
Wed Sep 30 10:17:01 2020 VERIFY OK: depth=0, C=SE, ST=Gotaland, O=Amagicom AB, OU=Mullvad,
CN=se-sto-007.mullvad.net, emailAddress=security@mullvad.net
Wed Sep 30 10:17:01 2020 WARNING: 'link-mtu' is used inconsistently, local='link-mtu 1557',
 remote='link-mtu 1558'
Wed Sep 30 10:17:01 2020 WARNING: 'comp-lzo' is present in remote config but missing in loc
al config, remote='comp-lzo'
Wed Sep 30 10:17:01 2020 Control Channel: TLSv1.3, cipher TLSv1.3 TLS_CHACHA20_POLY1305_SHA
256, 4096 bit RSA
Wed Sep 30 10:17:01 2020 [se-sto-007.mullvad.net] Peer Connection Initiated with [AF_INET]1
85.65.135.137:1194
Wed Sep 30 10:17:02 2020 SENT CONTROL [se-sto-007.mullvad.net]: 'PUSH_REQUEST' (status=1)
```

Mullvad - OpenVPN

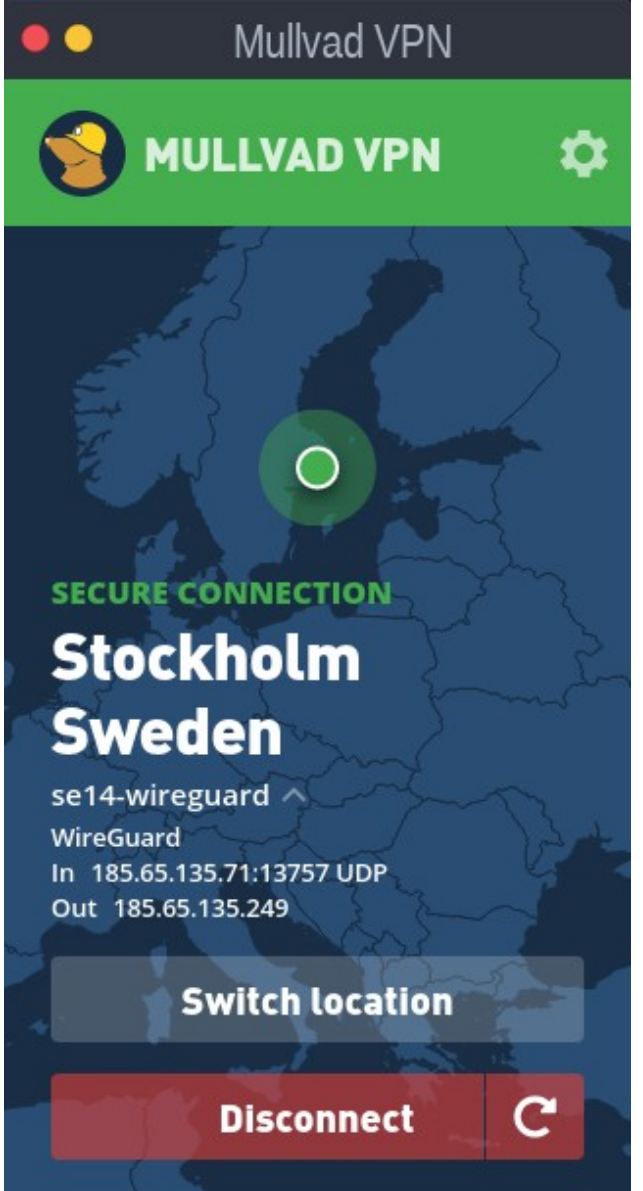


Linux: /var/log/mullvad-vpn/



<--Connected Via OpenVPN

Connected Via Wireguard-->



Mullvad – NetworkManager Setup

Click on the Network icon.

Click on **VPN-Connections** > **Configure VPN**.

Click on **Add**.

Select **Import a saved vpn configuration**.

Navigate to where you saved the downloaded file, select it and then click **open**.

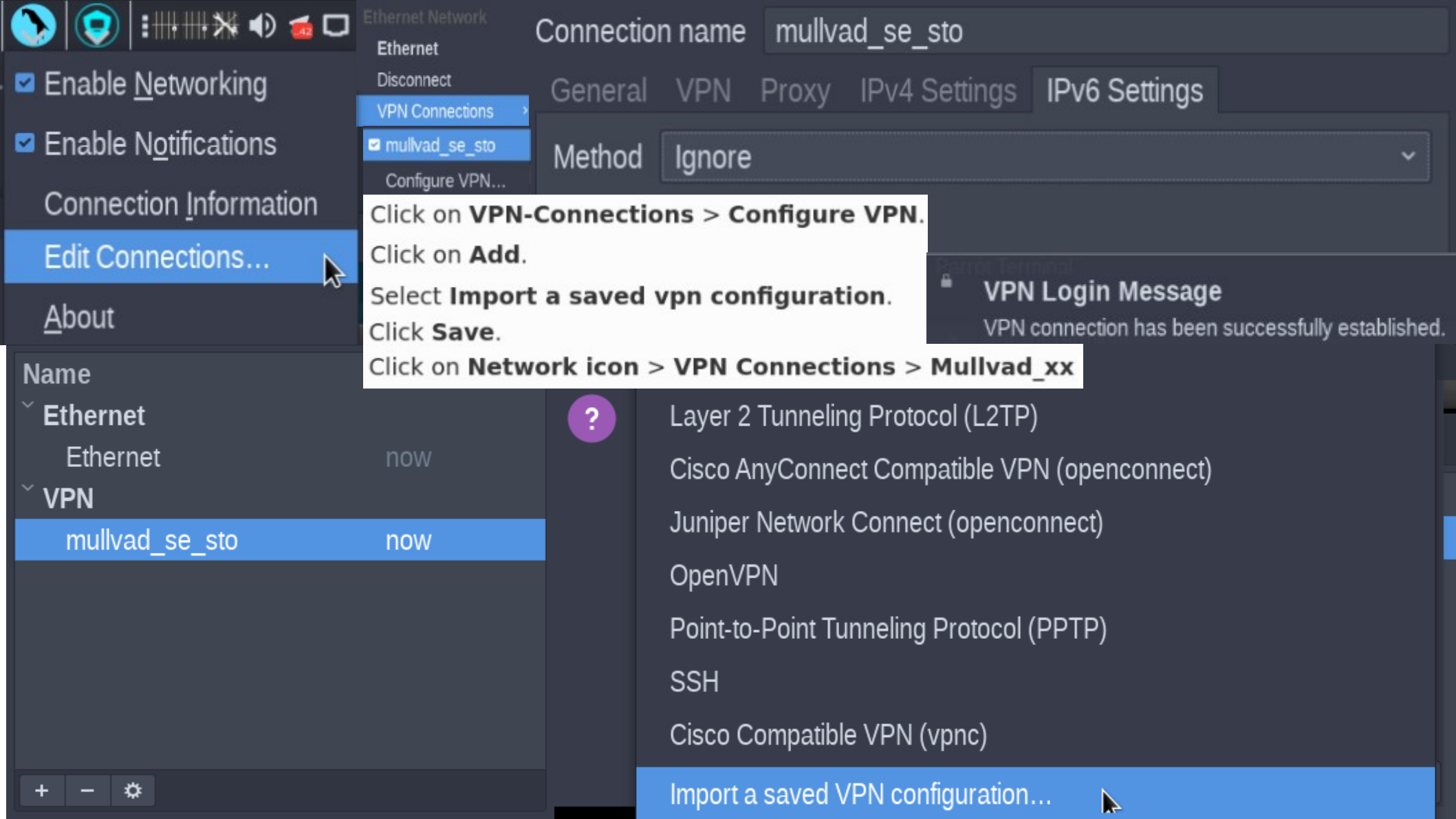
In the user name field, enter your Mullvad account number.

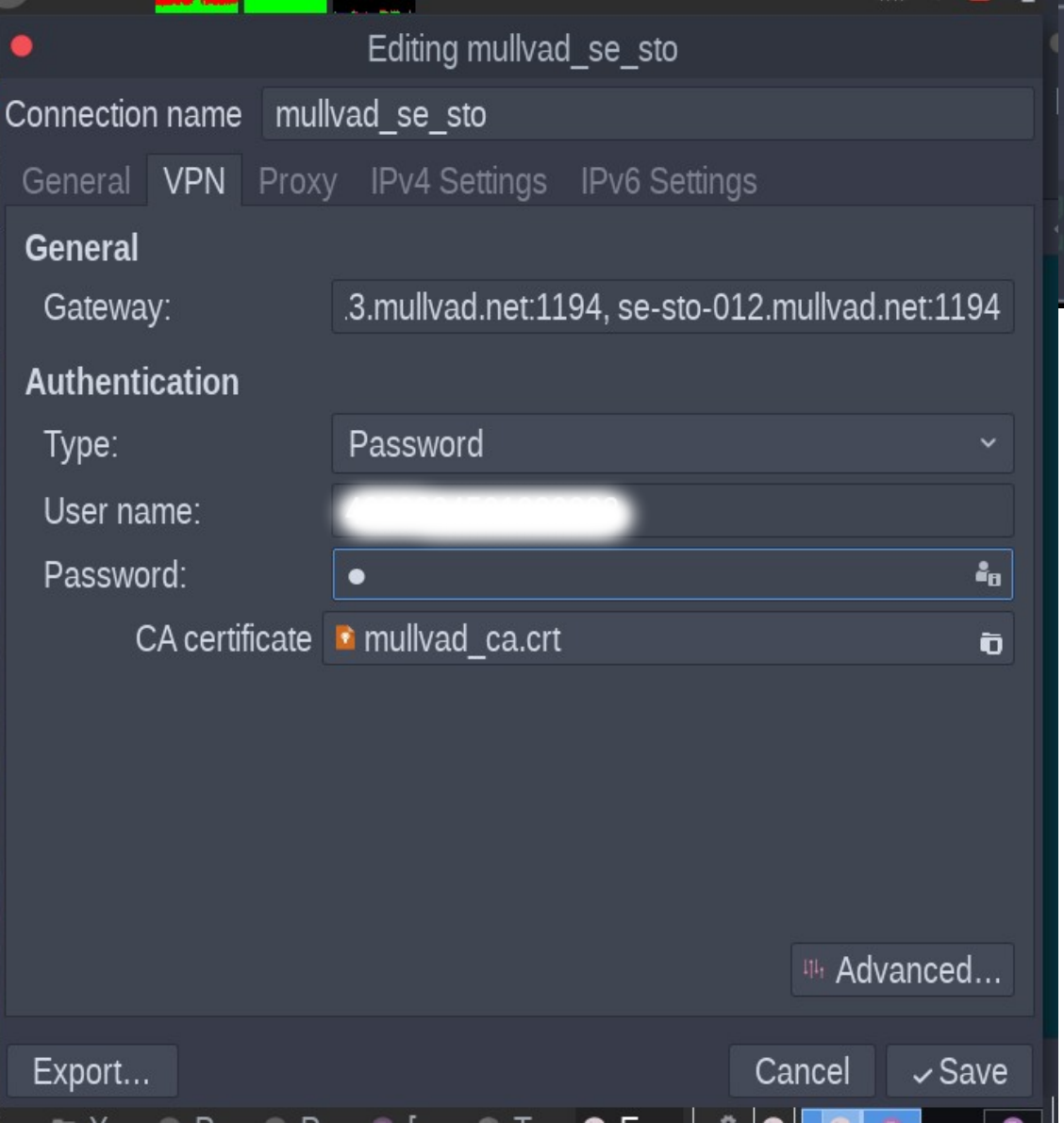
In the password field, enter "**m**".

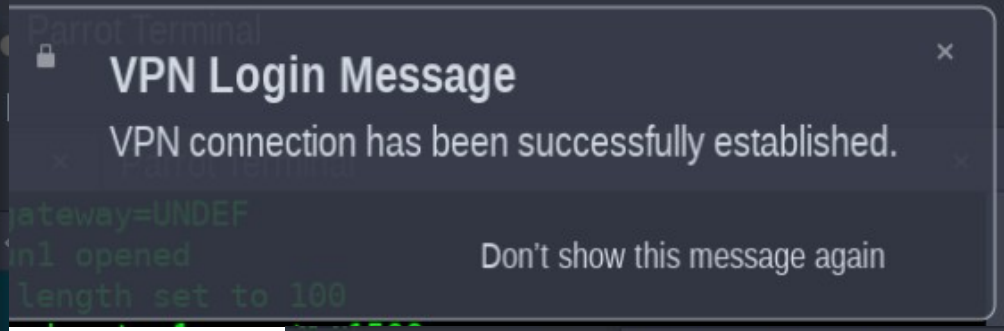
Click **Save**.

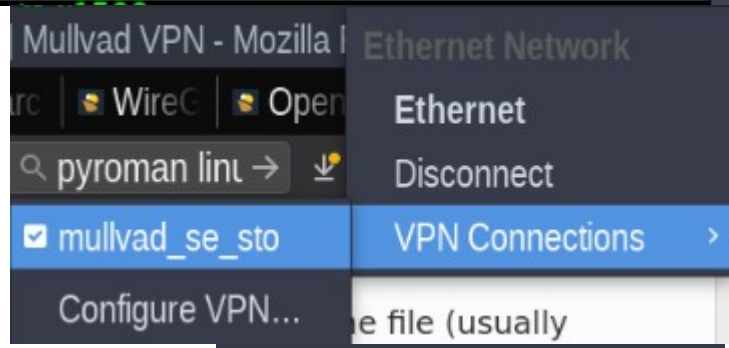
Click on **Network icon** > **VPN Connections** > **Mullvad_xx**

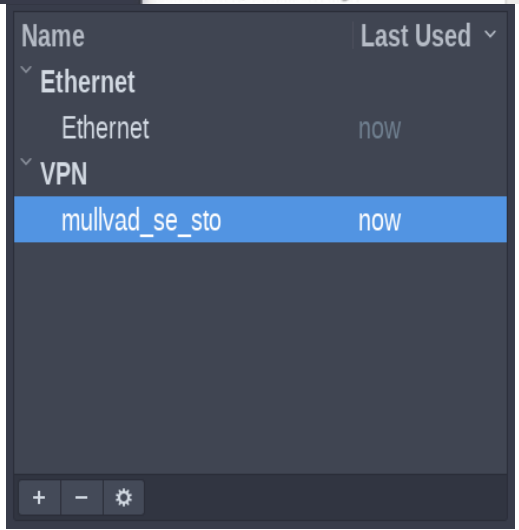
https://mullvad.net/en/help/linux-openvpn-installation/

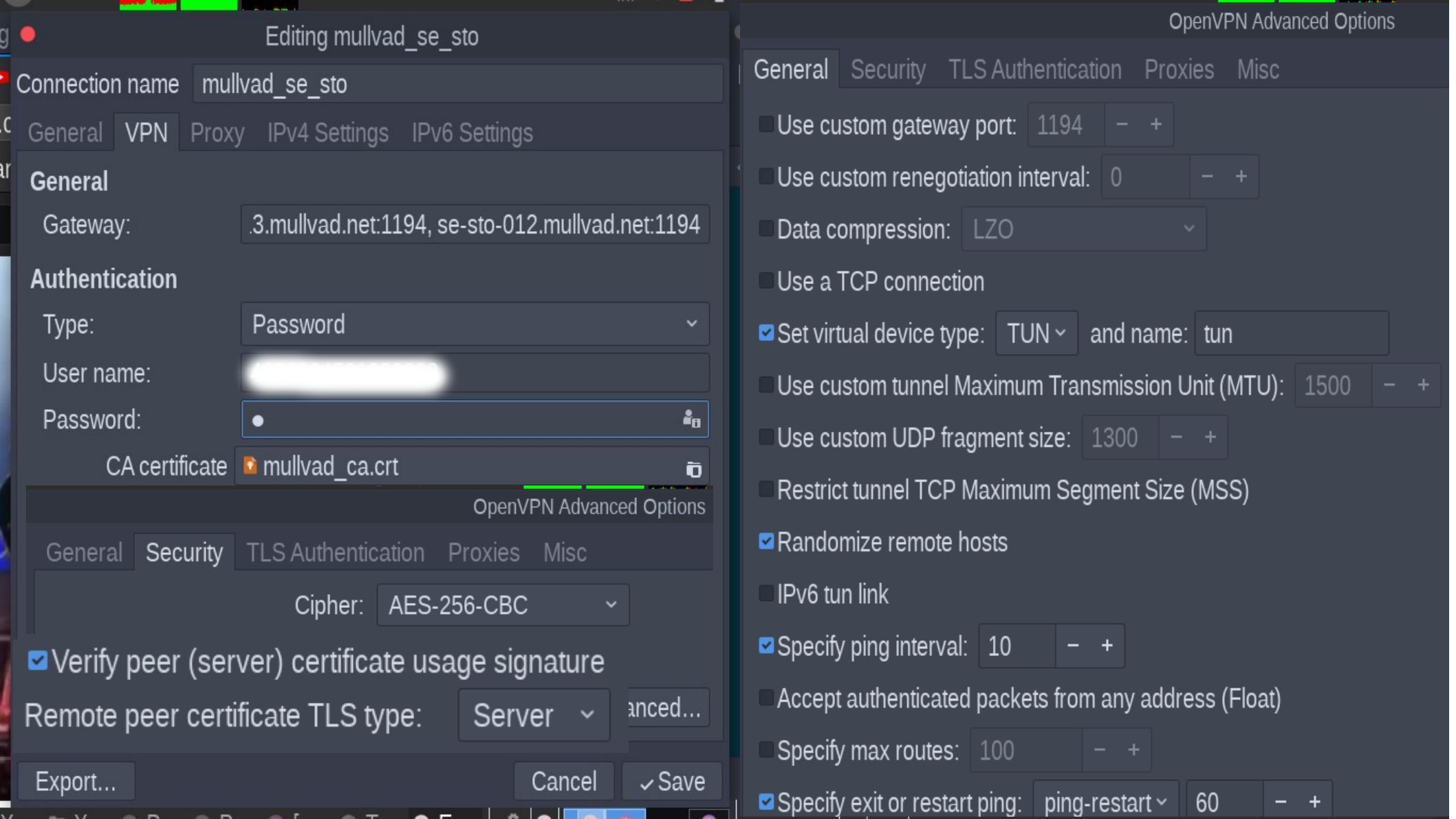

Enable Networking
Enable Notifications
Connection Information
Edit Connections...
About
Ethernet Network
Ethernet
Disconnect
VPN Connections
mullvad_se_sto
Configure VPN...
Connection name
mullvad_se_sto
General
VPN
Proxy
IPv4 Settings
IPv6 Settings
Method
Ignore
Click on VPN-Connections > Configure VPN.
Click on Add.
Select Import a saved vpn configuration.
Click Save.
Click on Network icon > VPN Connections > Mullvad_xx
VPN Login Message
VPN connection has been successfully established.
Name
Ethernet
Ethernet
now
VPN
mullvad_se_sto
now
Layer 2 Tunneling Protocol (L2TP)
Cisco AnyConnect Compatible VPN (openconnect)
Juniper Network Connect (openconnect)
OpenVPN
Point-to-Point Tunneling Protocol (PPTP)
SSH
Cisco Compatible VPN (vpnc)
Import a saved VPN configuration...

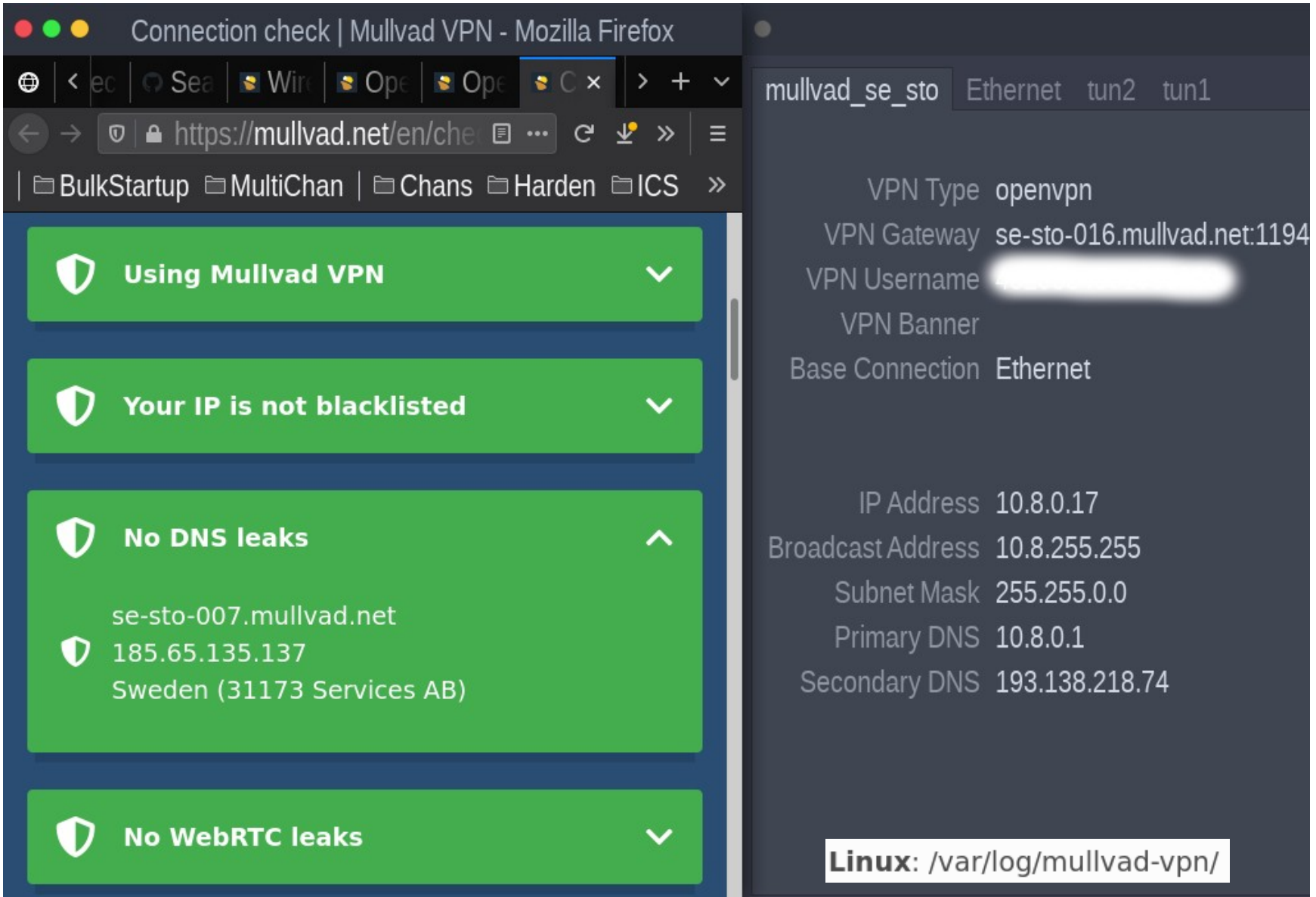

Editing mullvad_se_sto
Connection name
mullvad_se_sto
General
VPN
Proxy
IPv4 Settings
IPv6 Settings
General
Gateway:
.3.mullvad.net:1194, se-sto-012.mullvad.net:1194
Authentication
Type:
Password
User name:
Password:
CA certificate
mullvad_ca.crt
Advanced...
Export...
Cancel
Save


Parrot Terminal
VPN Login Message
VPN connection has been successfully established.
Don't show this message again


Mullvad VPN - Mozilla
WireG
Open
pyroman linu
mullvad_se_sto
Configure VPN...
Ethernet Network
Ethernet
Disconnect
VPN Connections
e file (usually


Name
Last Used
Ethernet
Ethernet
now
VPN
mullvad_se_sto
now


Editing mullvad_se_sto
Connection name
mullvad_se_sto
General
VPN
Proxy
IPv4 Settings
IPv6 Settings
General
Gateway:
.3.mullvad.net:1194, se-sto-012.mullvad.net:1194
Authentication
Type:
Password
User name:
Password:
CA certificate
mullvad_ca.crt
OpenVPN Advanced Options
General
Security
TLS Authentication
Proxies
Misc
Cipher:
AES-256-CBC
Verify peer (server) certificate usage signature
Remote peer certificate TLS type:
Server
anced...
Export...
Cancel
Save
OpenVPN Advanced Options
General
Security
TLS Authentication
Proxies
Misc
Use custom gateway port:
1194
Use custom renegotiation interval:
0
Data compression:
LZO
Use a TCP connection
Set virtual device type:
TUN
and name:
tun
Use custom tunnel Maximum Transmission Unit (MTU):
1500
Use custom UDP fragment size:
1300
Restrict tunnel TCP Maximum Segment Size (MSS)
Randomize remote hosts
IPv6 tun link
Specify ping interval:
10
Accept authenticated packets from any address (Float)
Specify max routes:
100
Specify exit or restart ping:
ping-restart
60


Connection check | Mullvad VPN - Mozilla Firefox
https://mullvad.net/en/che
BulkStartup
MultiChan
Chans
Harden
ICS
Using Mullvad VPN
Your IP is not blacklisted
No DNS leaks
se-sto-007.mullvad.net
185.65.135.137
Sweden (31173 Services AB)
No WebRTC leaks
mullvad_se_sto
Ethernet
tun2
tun1
VPN Type openvpn
VPN Gateway se-sto-016.mullvad.net:1194
VPN Username
VPN Banner
Base Connection Ethernet
IP Address 10.8.0.17
Broadcast Address 10.8.255.255
Subnet Mask 255.255.0.0
Primary DNS 10.8.0.1
Secondary DNS 193.138.218.74
Linux: /var/log/mullvad-vpn/

Mullvad - Configure Firefox

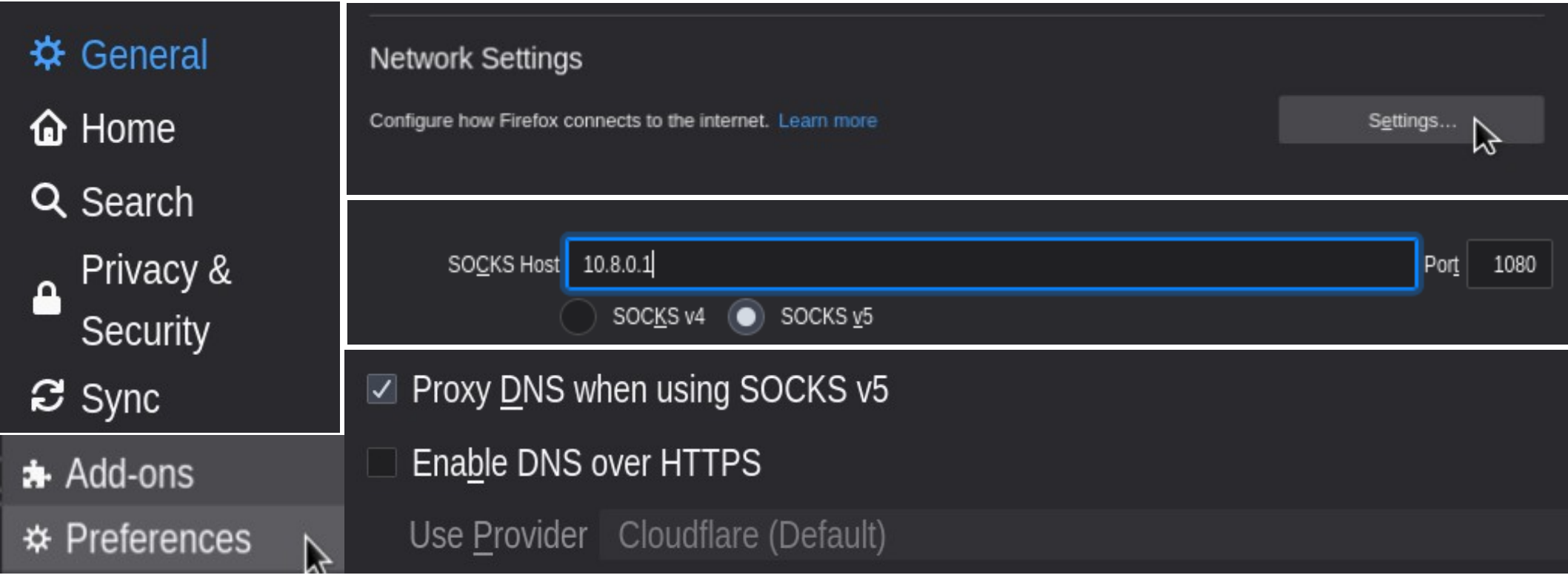
1. In the **Firefox menu**, click on **Edit** (on some operating systems, click Tools).
2. Click on **Preferences** (on some operating systems, click Options).
3. Scroll down to **Network Proxy**.
4. Click on **Settings**.
5. Select **Manual proxy configuration**.
6. Make sure **HTTP/SSL** and **FTP proxy** fields are **empty**.
7. In the **SOCKS Host:** field, enter **10.8.0.1** with port **1080**.
8. Click on **SOCKS v5** and enable **Remote DNS** or tick **Proxy DNS when using SOCKS v5**.
9. Click on **OK**.

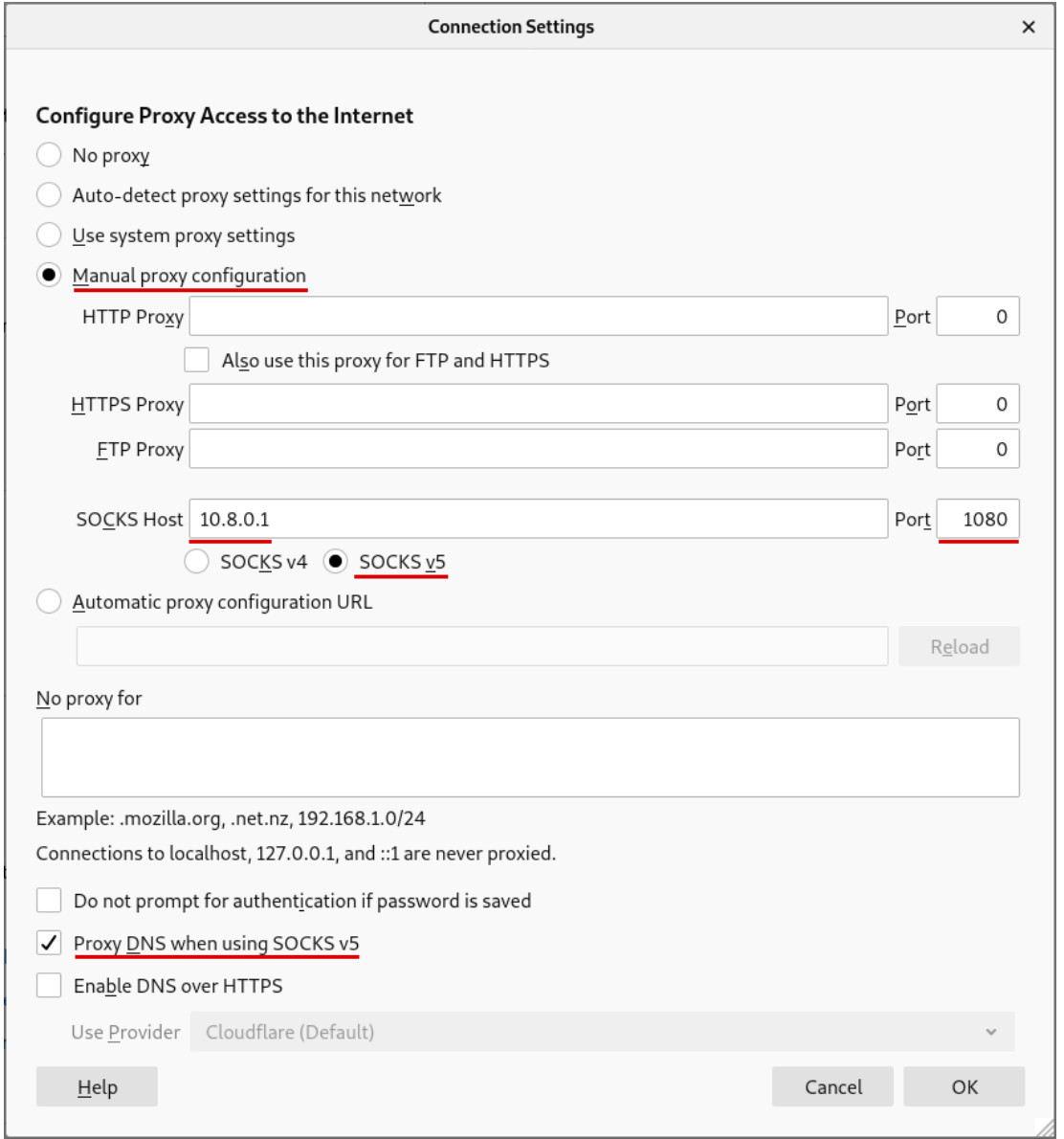
https://mullvad.net/en/help/socks5-proxy/#get-started

Firefox users

You are at risk of leaking DNS requests to Cloudflare, no matter which Mullvad setup you have. To prevent this, open Firefox Options > General > Network settings > Settings, then **deselect** “Enable DNS over HTTPS.”

You can then visit am.i.mullvad.net to easily check whether or not you’re leaking information.




Connection Settings
Configure Proxy Access to the Internet
No proxy
Auto-detect proxy settings for this network
Use system proxy settings
Manual proxy configuration
HTTP Proxy
Port
0
Also use this proxy for FTP and HTTPS
HTTPS Proxy
Port
0
FTP Proxy
Port
0
SOCKS Host
10.8.0.1
Port
1080
SOCKS v4
SOCKS v5
Automatic proxy configuration URL
Reload
No proxy for
Example: .mozilla.org, .net.nz, 192.168.1.0/24
Connections to localhost, 127.0.0.1, and ::1 are never proxied.
Do not prompt for authentication if password is saved
Proxy DNS when using SOCKS v5
Enable DNS over HTTPS
Use Provider
Cloudflare (Default)
Help
Cancel
OK

nmcli – Show Mullvad Connection

```
└──╼ #nmcli connection show mullvad_se_sto
connection.id:                          mullvad_se_sto
connection.uuid:                        37c91a90-e8fa-444e-9fc9-9ca680c87997
connection.stable-id:                   --
connection.type:                        vpn
connection.interface-name:              --
connection.autoconnect:                 no
connection.autoconnect-priority:        0
connection.autoconnect-retries:         -1 (default)
connection.multi-connect:               0 (default)
connection.auth-retries:                -1
connection.timestamp:                   1601480718
connection.read-only:                   no
connection.permissions:                 user:parrotsec-kiosk
```

nmcli – Examine VPN Connection

```
vpn.service-type:                       org.freedesktop.NetworkManager.openvpn
vpn.user-name:                          --
vpn.data:                               ca = /etc/openvpn/mullvad_ca.crt, cipher = AES-256-
vpn.secrets:                            <hidden>
vpn.persistent:                         no
vpn.timeout:                            0
proxy.method:                           none
proxy.browser-only:                     no
proxy.pac-url:                          --
proxy.pac-script:                       --
GENERAL.NAME:                           mullvad_se_sto
GENERAL.UUID:                           37c91a90-e8fa-444e-9fc9-9ca680c87997
GENERAL.DEVICES:                        eth0
GENERAL.IP-IFACE:                       eth0
```

Route configuration fails with systemd-networkd

When using **systemd-networkd** to manage network connections and attempting to tunnel all outgoing traffic through the VPN, OpenVPN may fail to add routes. This is a result of systemd-networkd attempting to manage the tun interface before OpenVPN finishes configuring the routes. When this happens, the following message will appear in the OpenVPN log.

```
openvpn[458]: RTNETLINK answers: Network is unreachable
openvpn[458]: ERROR: Linux route add command failed: external program exited with error status: 2
```

With systemd-233 (currently in **testing**), systemd-networkd can be configured to ignore the tun connections and allow OpenVPN to manage them. To do this, create the following file:

/etc/systemd/network/90-tun-ignore.network

```
[Match]
Name=tun*

[Link]
Unmanaged=true
```

90-tun-ignore.network

OpenVPN – Restart After Suspend

/etc/systemd/system/openvpn-reconnect.service

```
[Unit]
Description=Restart OpenVPN after suspend

[Service]
ExecStart=/usr/bin/pkill --signal SIGHUP --exact openvpn

[Install]
WantedBy=sleep.target
```

OpenVPN-Reconnect.service

OpenVPN – Restart After Suspend

```
##-==========================================-##
##   [+] /etc/systemd/system/openvpn-reconnect.service
##-==========================================-##
## ------------------------------------------------------------------------ ##
##   [?] kill and restart OpenVPN after suspend
## ------------------------------------------------------------------------ ##
[Unit]
Description=Restart OpenVPN after suspend
[Service]
ExecStart=/usr/bin/pkill --signal SIGHUP --exact openvpn
[Install]
WantedBy=sleep.target
```

OpenVPN-Reconnect.service

Generating OpenVPN Keys On The Server (Site-to-Site VPN)

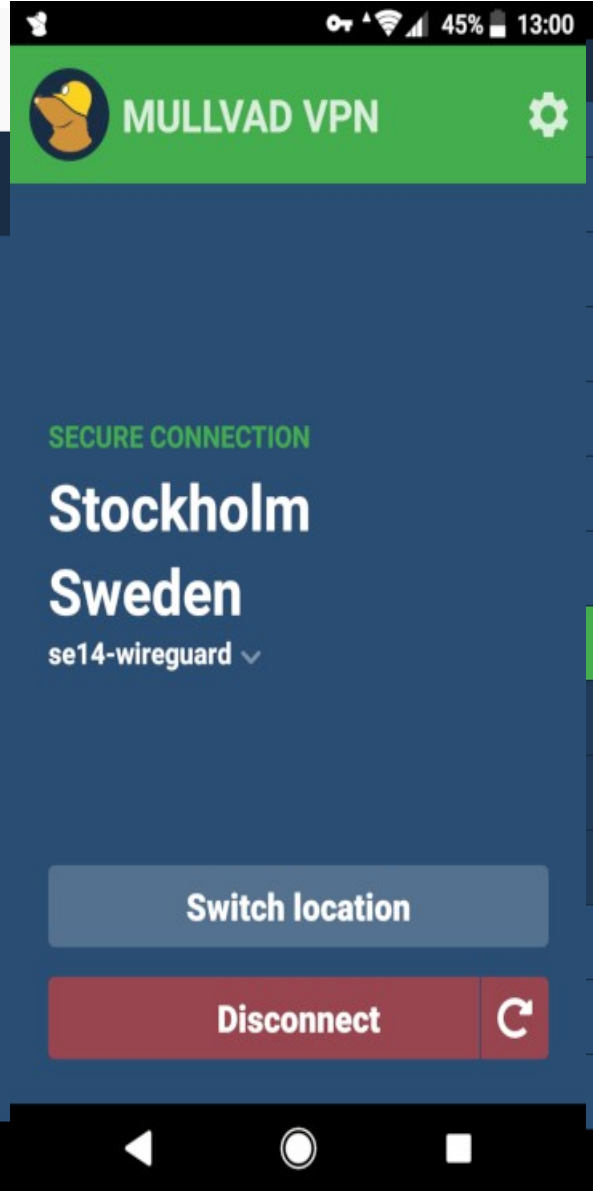
```
echo "## ================================================ ##"
echo "##       [+] Generate key for tls-auth              "
echo "## ================================================ ##"
openvpn --genkey --secret /etc/openvpn/ta.key

## ------------------------------------------------- ##
##    [?]  In the server configuration, add:
## ------------------------------------------------- ##
tls-auth ta.key 0

## ------------------------------------------------- ##
##  [?]  In the client configuration, add:
## ------------------------------------------------- ##
tls-auth ta.key 1
```

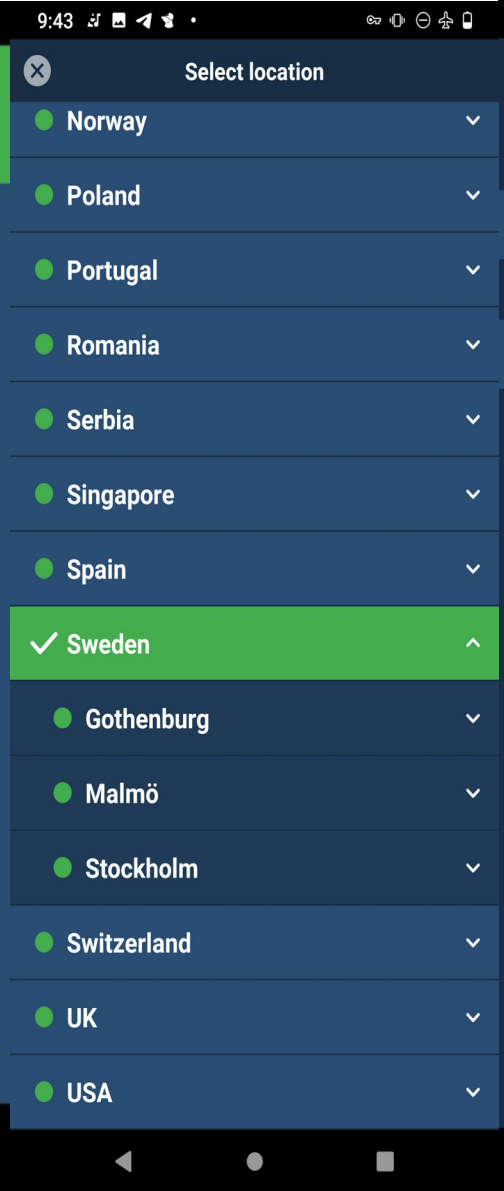
OpenVPN – TLS Auth Config

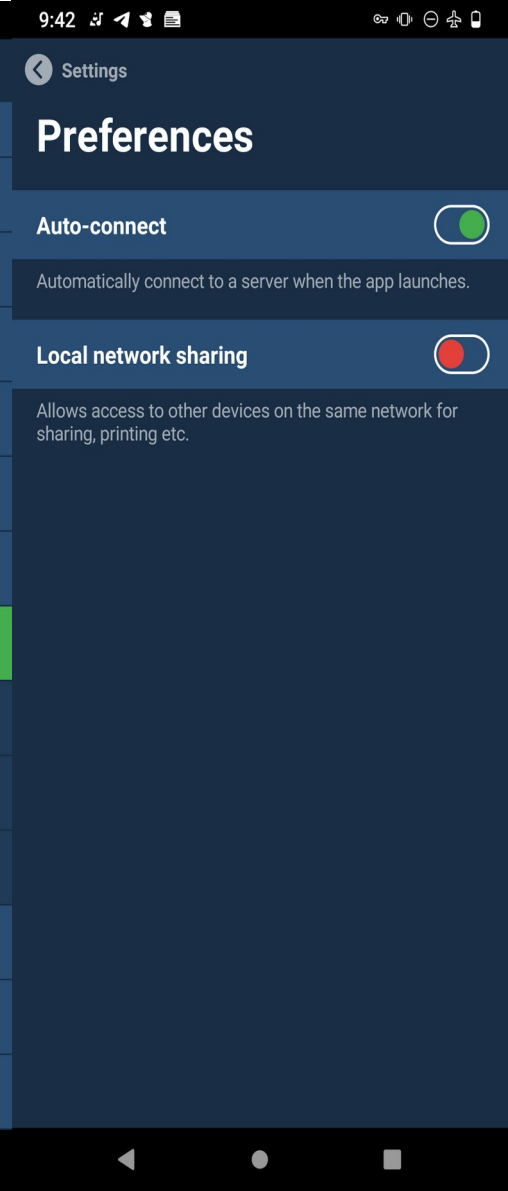
- Enable the TLS HMAC handshake protection (`--tls-crypt` or `--tls-auth`).

```
/etc/openvpn/client/client.conf
-------------------------------------------------------------------------------
remote elmer.acmecorp.org 1194
.
user nobody
group nobody
ca ca.crt
cert client.crt
key client.key
.
tls-crypt ta.key
```

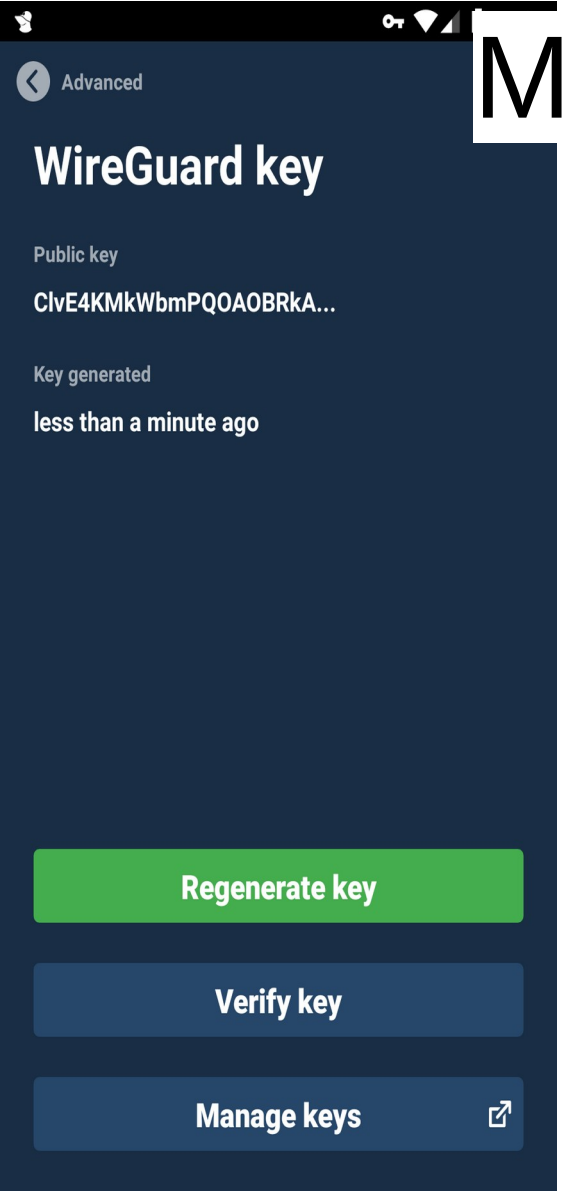

10:26
https://mullvad.net/en/dow
Using Mullvad (se6-wireguard)
Stockholm, Sweden
Check for leaks
MULLVAD VPN
Mullvad VPN for Android
Latest version: 2021.1 (see changes)
GET IT ON Google Play
GET IT ON F-Droid
Works on Android 7.0+
Download .apk
GPG signature
What is this?
Installation instructions

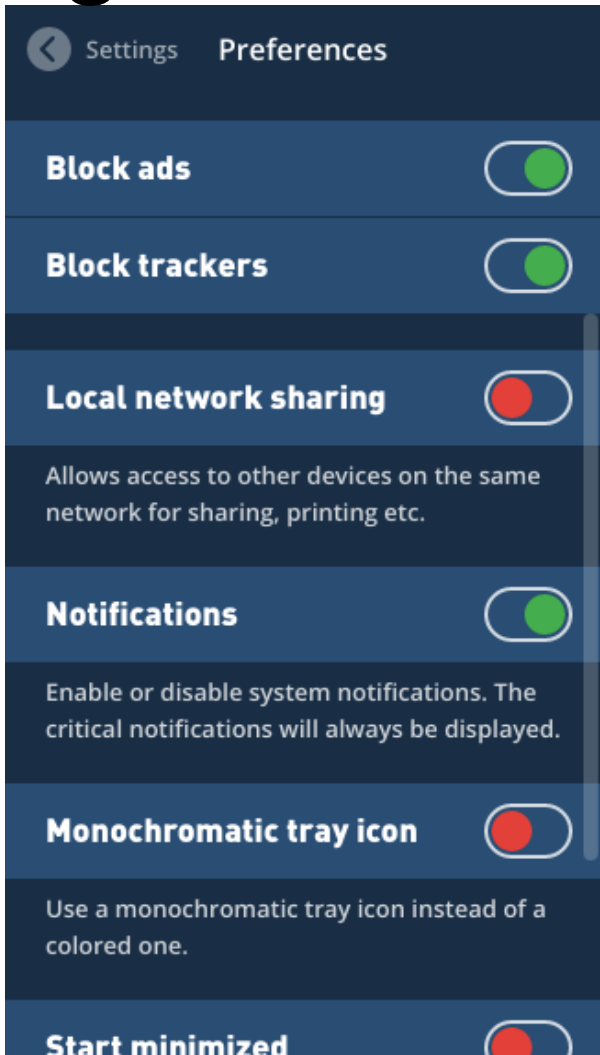

45% 13:00
MULLVAD VPN
SECURE CONNECTION
Stockholm
Sweden
se14-wireguard
Switch location
Disconnect

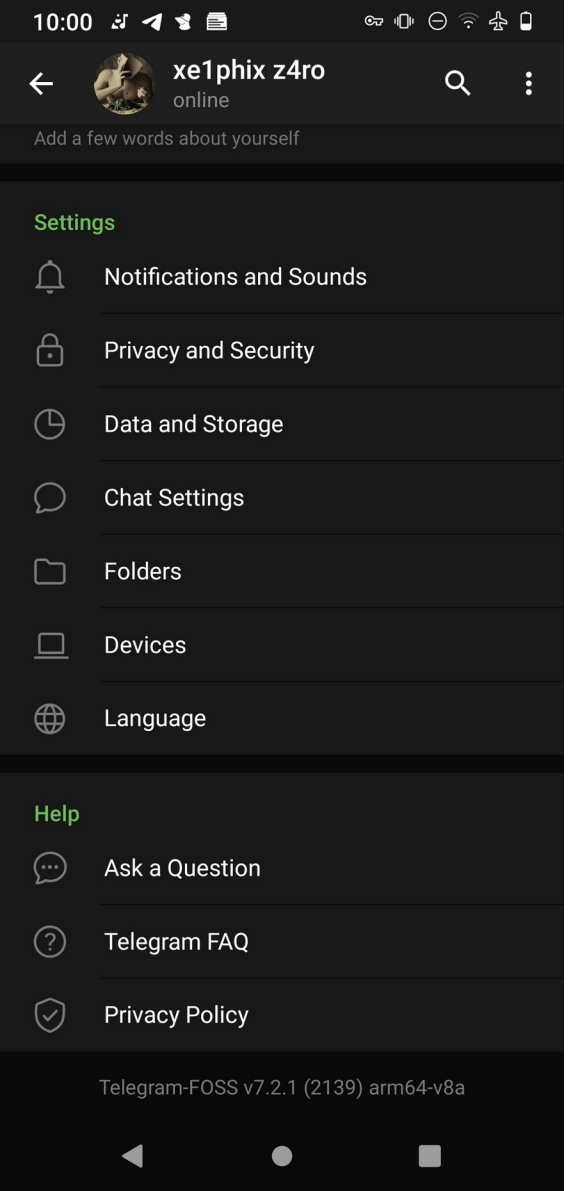

9:43
Select location
Norway
Poland
Portugal
Romania
Serbia
Singapore
Spain
Sweden
Gothenburg
Malmö
Stockholm
Switzerland
UK
USA

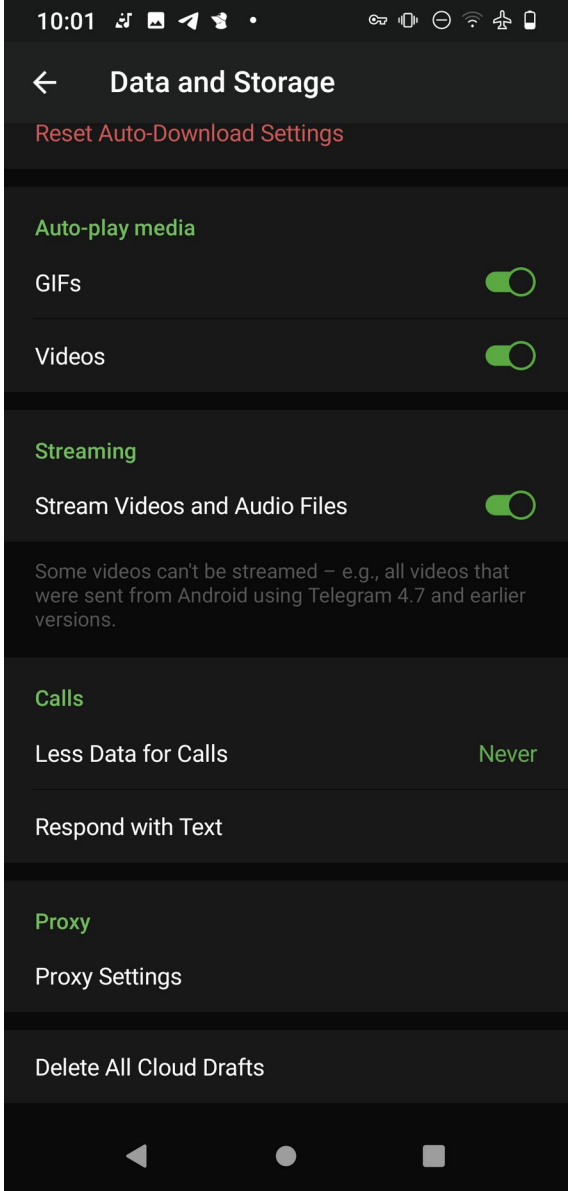

9:42
Settings
Preferences
Auto-connect
Automatically connect to a server when the app launches.
Local network sharing
Allows access to other devices on the same network for sharing, printing etc.

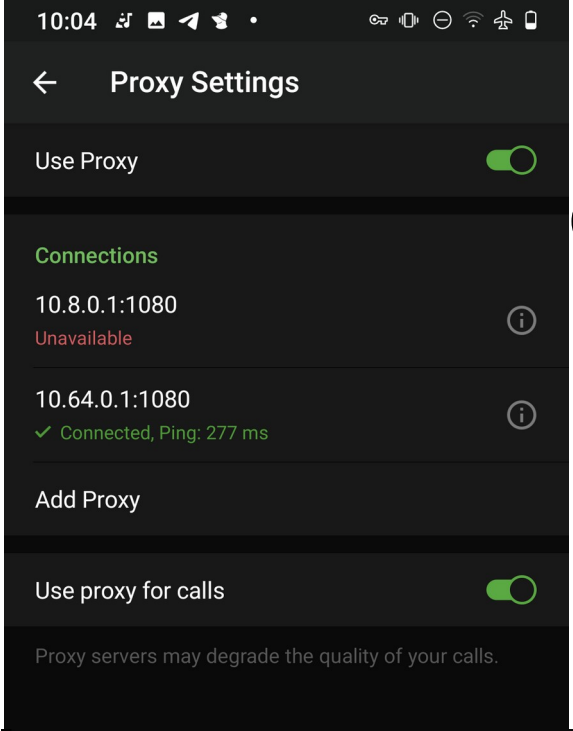
Mullvad – AD Blocking



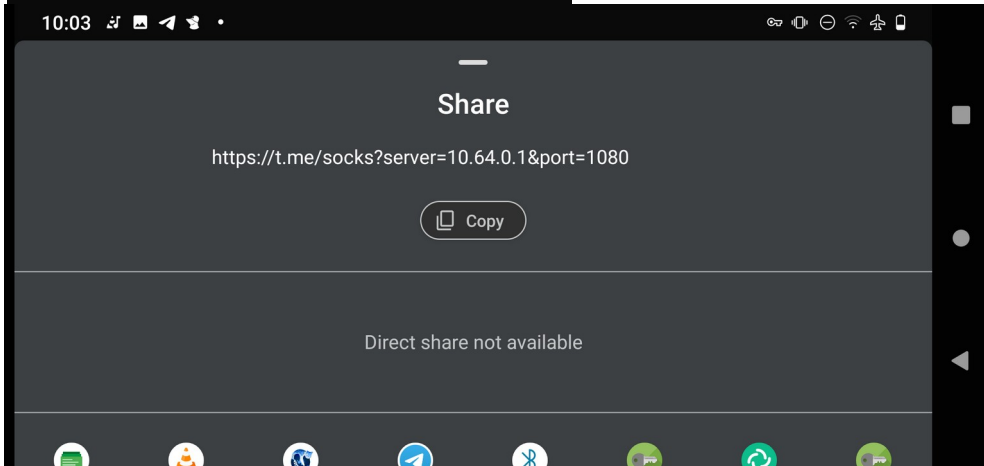


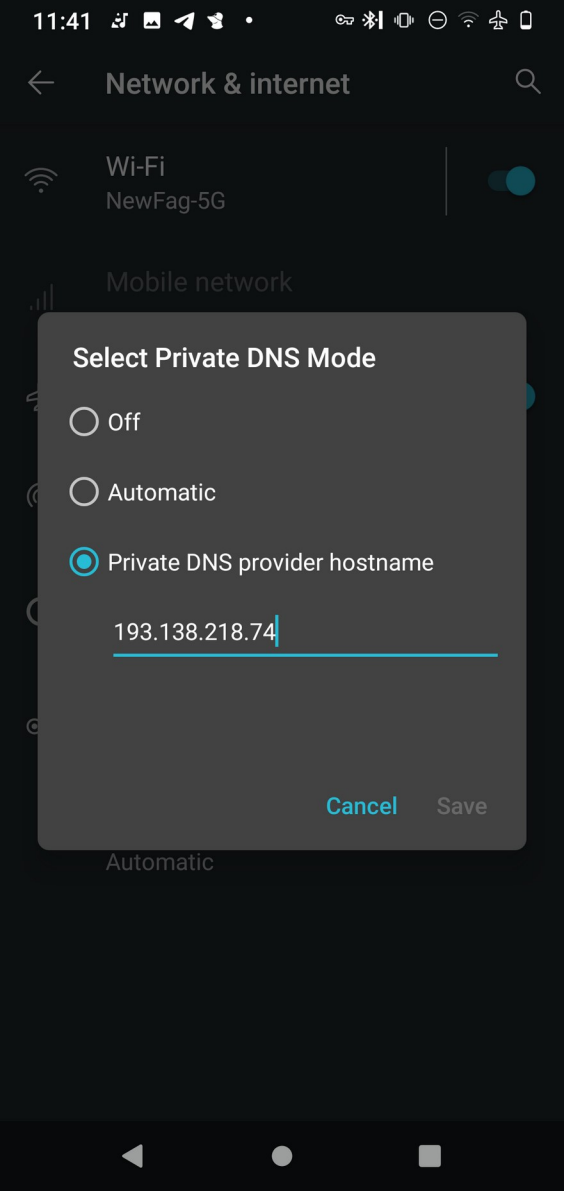


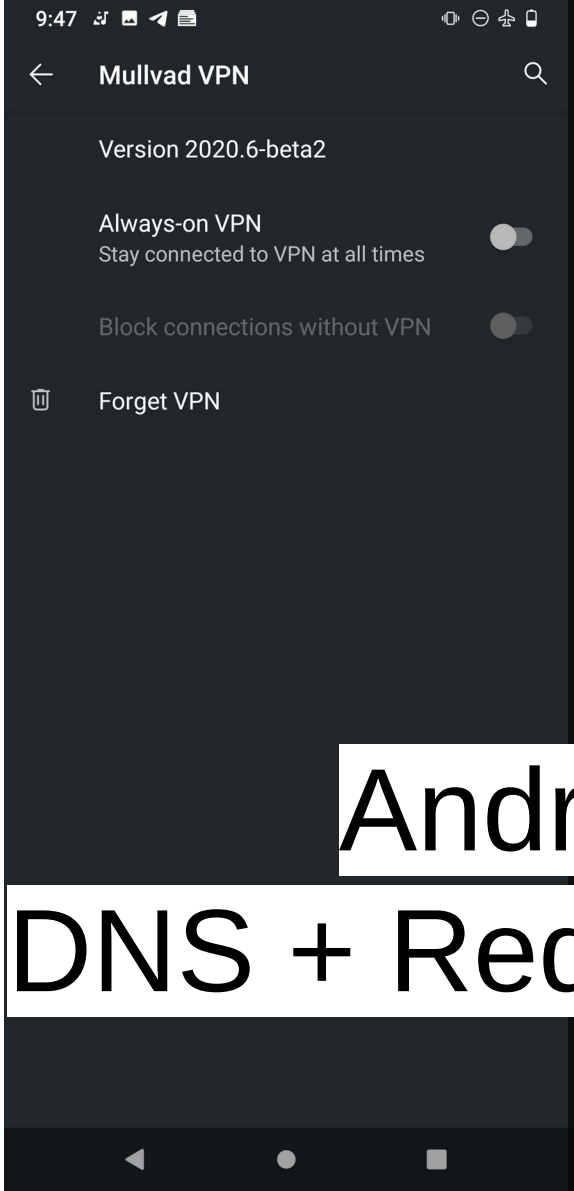


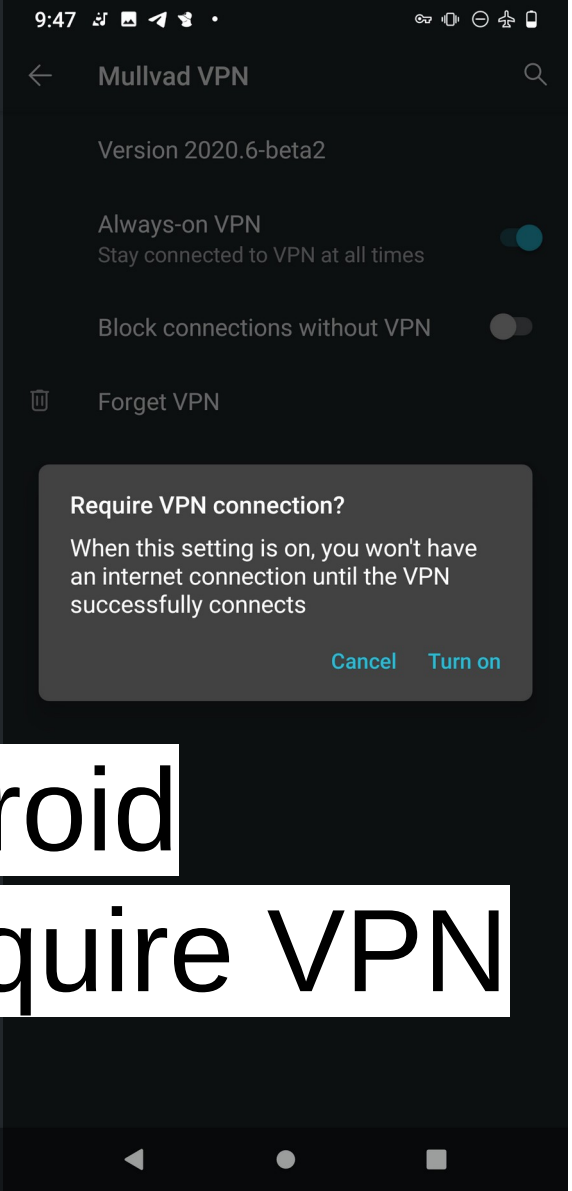


Wireguard Telegram Configuration









Android DNS + Require VPN

Connection check

Using Mullvad VPN

WireGuard
se28-wireguard.mullvad.net
185.195.233.168
Stockholm, Sweden (31173 Services AB)

WireGuard
se28-wireguard.mullvad.net
2a03:1b20:4:f011::a28e
Stockholm, Sweden (31173 Services AB)

Your IP is not blacklisted

Your IP was not found on any blacklists.

No DNS leaks

se28-wireguard.mullvad.net
185.195.233.68
Sweden (31173 Services AB)

No WebRTC leaks

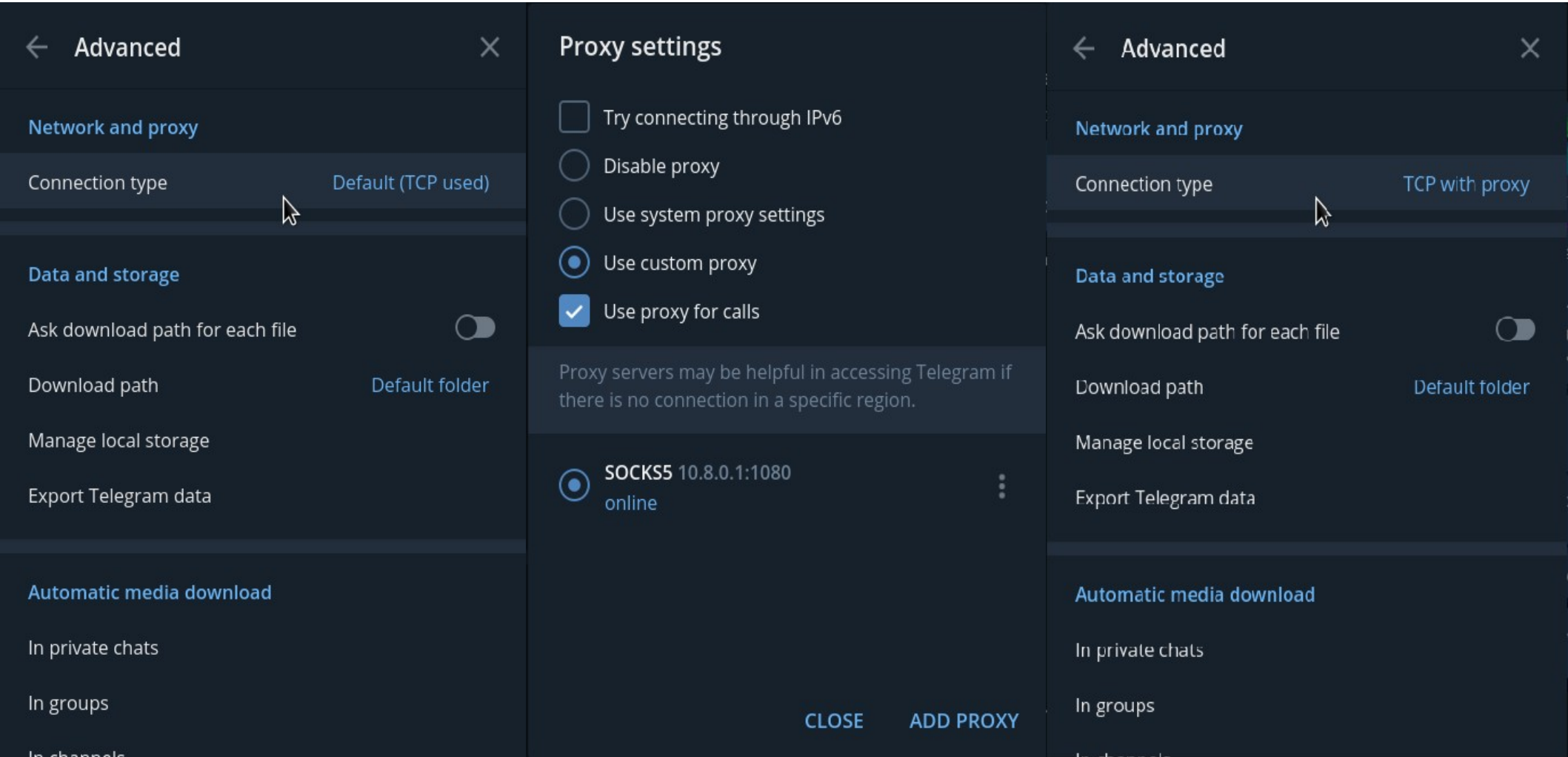
185.195.233.168

OpenVPN – Packet Capture

```
└─ #tshark -i any -f 'udp port 1194'
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'
    1 0.000000000 173.22.75.86 → 193.138.218.136 OpenVPN 58 MessageType: P_CONTROL_HARD_RESET_CLIENT_V2
    2 0.128643529 193.138.218.136 → 173.22.75.86 OpenVPN 70 MessageType: P_CONTROL_HARD_RESET_SERVER_V2
    3 0.128865762 173.22.75.86 → 193.138.218.136 OpenVPN 66 MessageType: P_ACK_V1
    4 0.128944171 173.22.75.86 → 193.138.218.136 TLSv1 267 Client Hello
    5 0.272074537 193.138.218.136 → 173.22.75.86 TLSv1.3 1244 Server Hello, Change Cipher Spec, Application Data, Application Data
    6 0.272276797 193.138.218.136 → 173.22.75.86 TLSv1.3 1232 Continuation Data
    7 0.272276872 193.138.218.136 → 173.22.75.86 TLSv1.3 1232 Continuation Data
    8 0.272329335 193.138.218.136 → 173.22.75.86 TLSv1.3 851 Continuation Data
    9 0.272372003 173.22.75.86 → 193.138.218.136 OpenVPN 66 MessageType: P_ACK_V1
   10 0.272395651 173.22.75.86 → 193.138.218.136 OpenVPN 66 MessageType: P_ACK_V1
   11 0.272421831 173.22.75.86 → 193.138.218.136 OpenVPN 66 MessageType: P_ACK_V1
   12 0.273627595 173.22.75.86 → 193.138.218.136 TLSv1.3 619 Change Cipher Spec, Application Data, Application Data, Application Data
   13 0.404691384 193.138.218.136 → 173.22.75.86 TLSv1.3 228 Application Data, Application Data
```

```
└─ #tcpdump -vn -i any 'port 1194'
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
23:24:11.394485 eth0  Out IP (tos 0x0, ttl 64, id 31803, offset 0, flags [DF], proto UDP (17), length 42)
    173.22.75.86.60526 > 193.138.218.133.1194: UDP, length 14
23:24:11.515617 eth0  In  IP (tos 0x0, ttl 47, id 43676, offset 0, flags [DF], proto UDP (17), length 54)
    193.138.218.133.1194 > 173.22.75.86.60526: UDP, length 26
23:24:11.515794 eth0  Out IP (tos 0x0, ttl 64, id 31818, offset 0, flags [DF], proto UDP (17), length 50)
    173.22.75.86.60526 > 193.138.218.133.1194: UDP, length 22
23:24:11.515841 eth0  Out IP (tos 0x0, ttl 64, id 31819, offset 0, flags [DF], proto UDP (17), length 251)
    173.22.75.86.60526 > 193.138.218.133.1194: UDP, length 223
23:24:11.645779 eth0  In  IP (tos 0x0, ttl 47, id 43696, offset 0, flags [DF], proto UDP (17), length 1228)
    193.138.218.133.1194 > 173.22.75.86.60526: UDP, length 1200
```

Mullvad – Telegram SOCKS5 Proxy



##-==-##
[+] Connect To Telegram Using SOCKS5 Proxy Connections
##-==-##

##-==================================-##
[+] Connect To Telegram Using Wireguard
##-==================================-##
https://t.me/socks?server=10.64.0.1&port=1080

##-==================================-##
[+] Connect To Telegram Using OpenVPN
##-==================================-##
https://t.me/socks?server=10.8.0.1&port=1080

```
owner @{HOME}/.local/share r,
owner @{HOME}/.local/share/TelegramDesktop/ rwk,
owner @{HOME}/.local/share/TelegramDesktop/** rwkl,
owner @{HOME}/.icons/** r,
owner @{HOME}/.cache/** rwk,

# home files
owner @{HOME}/ r,
owner @{HOME}/* rwk,

# shared libraries
/usr/lib/** rm,

# /proc
owner @{PROC}/*/cmdline r,
```

Telegram – AppArmor Profile

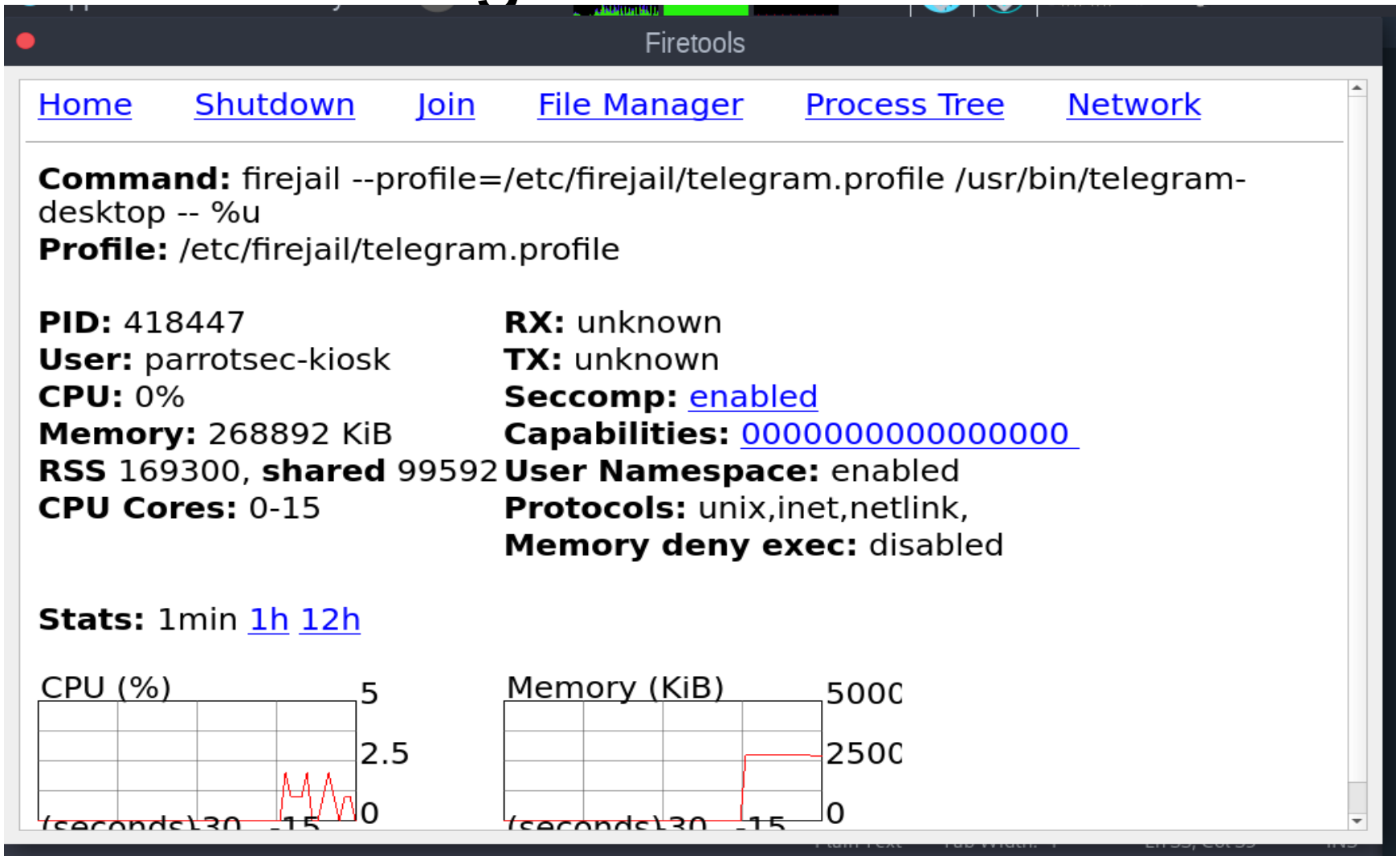
```
noblacklist ${HOME}/.TelegramDesktop
noblacklist ${HOME}/.local/share/TelegramDesktop
mkdir ${HOME}/.TelegramDesktop
mkdir ${HOME}/.local/share/TelegramDesktop
whitelist ${DOWNLOADS}
whitelist ${HOME}/.TelegramDesktop
whitelist ${HOME}/.local/share/TelegramDesktop
include whitelist-common.inc
include whitelist-runuser-common.inc
include whitelist-usr-share-common.inc
include whitelist-var-common.inc

include disable-common.inc
include disable-devel.inc
include disable-exec.inc
include disable-interpreters.inc
include disable-programs.inc
include disable-shell.inc
include disable-xdg.inc
```

Telegram - Firejail

Telegram – Firejail 2

```
seccomp
apparmor
caps.drop all
netfilter
nodvd
nonewprivs
noroot
notv
protocol unix,inet,netlink
shell none

disable-mnt
private-cache
private-etc ca-certificates,crypto-policies,fonts,ld.so.cache,l
ocaltime,machine-id,pki,pulse,resolv.conf,ssl
private-tmp
seccomp.block-secondary
```

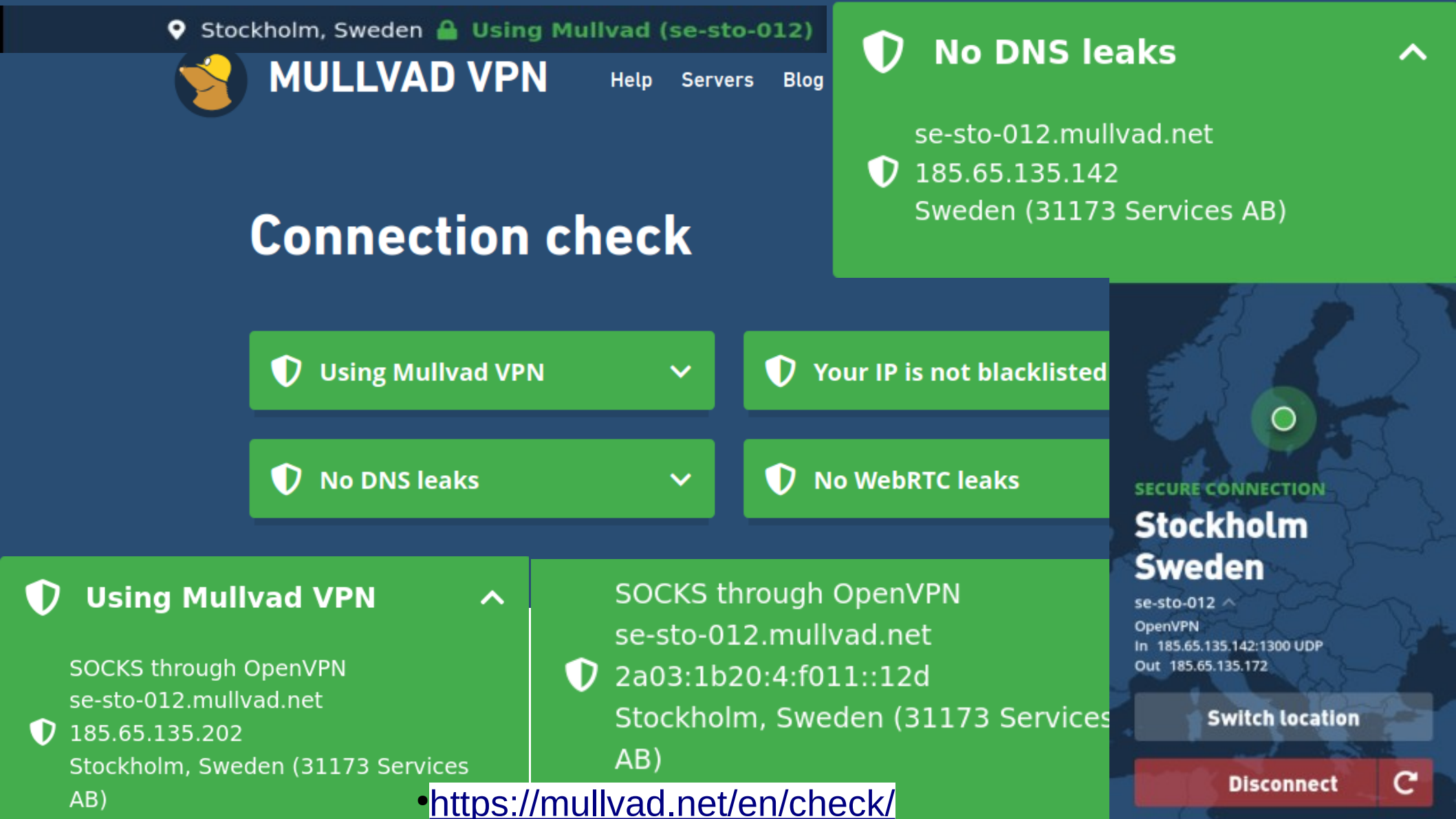
Telegram - Firetools



What are DNS leaks?

A DNS server is the first point of contact that your browser makes when you try to access information over the Internet. This is the case for every URL you visit, every file you download, and every image that loads on a website, including ads.

The DNS server therefore knows which pages you are visiting and which resources you are looking at, and as a result, you are constantly leaking information to your DNS server provider about your activity. Our DNS leaks guide has information on how to prevent this.



- https://mullvad.net/en/check/

```
└─ $curl https://am.i.mullvad.net/connected
You are connected to Mullvad (server se-mma-024). Your IP address is 141.98.255.154
```

FAQ | Port check | Torrent check | API

```
$ curl https://am.i.mullvad.net/connected
You are connected to Mullvad (server se-sto-012). Your IP address is 185.65.135.202
```

```
$ curl https://am.i.mullvad.net/ip
185.65.135.202
```

```
$ curl https://am.i.mullvad.net/city
Stockholm
```

```
$ curl https://am.i.mullvad.net/country
Sweden
```

- https://mullvad.net/en/check/

```
$ curl https://am.i.mullvad.net/json
{
  "ip": "185.65.135.202",
  "country": "Sweden",
  "city": "Stockholm",
  "longitude": 18.0717,
  "latitude": 59.3287,
  "mullvad_exit_ip": true,
  "mullvad_exit_ip_hostname": "se-sto-012",
  "mullvad_server_type": "SOCKS through OpenVPN",
  "blacklisted": {
    "blacklisted": false,
    "results": [
      {
        "name": "Project Honeypot",
        "link": "https://www.projecthoneypot.org/about_us.php",
        "blacklisted": false
      },
      {
        "name": "Spamhaus",
        "link": "https://www.spamhaus.org/organization/",
```

DNS Leak Test

With insufficient configuration, it is possible that the browser's DNS requests will be sent to the ISP DNS server directly, and not sent through the VPN or Proxy. Thus, a malicious website will be able to find out the name of your real ISP, and the ISP will know your endpoint IP and which sites you visit.

DNS Leak Test shows which DNS servers your browser uses to resolve domain names. This test attempts to resolve 100 randomly generated domain names asynchronously, 50 with A record (IPv4-only) and 50 with both A and AAAA records (IPv4+IPv6).

Your IP Address :			
IP Address	185.65.135.191		
ISP	31173 Services AB		
Location	Sweden, Stockholm		
DNS Leak Test :			
Test Results	Found 2 Servers, 1 ISP, 1 Location		
Your DNS Servers	IP Address :	ISP :	Location :
	185.65.135.131	31173 Services AB	Sweden, Stockholm
	2a03:1b20:4:f011::1d	31173 Services AB	Sweden, Stockholm

My IP Address :

IP address	185.65.135.191
Hostname	n/a

IP Address Location :

Country	Sweden (SE)
State/Region	Stockholm County (AB)
City	Stockholm
ISP	31173 Services AB
ASN	39351
Timezone	Europe/Stockholm
Local Time	Wed, 30 Sep 2020 09:09:20 +0200
Latitude/Longitude	59.3287,18.0717

IPv6 Leak Test :

IPv6 Address	2a03:1b20:4:f011::1d more

WebRTC Leak Test :

Local IP address	n/a
Public IP address	n/a

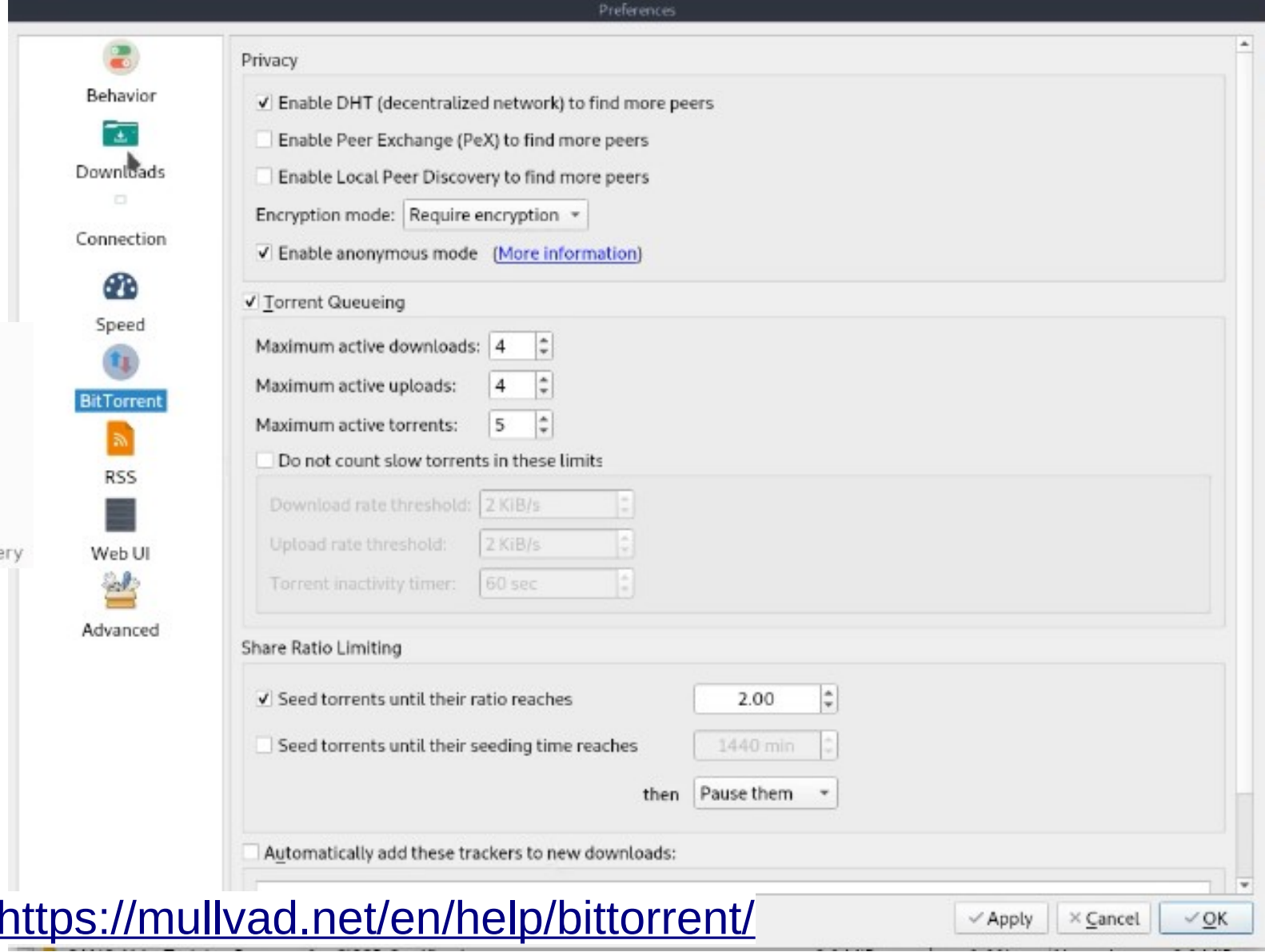
Qbittorrent

1. Click on **Tools** followed by **Options** (Alt-O).
2. Click on **Connection**.
3. Click on BitTorrent
4. Enable (Check) Enable anonymous mode
5. Disable (Uncheck) Enable DHT
6. Disable (Uncheck) Enable PeX
7. Disable (Uncheck) Enable Local peer discovery
8. Click on **Connection**
9. For Enabled protocol: Use the drop-down bar and select **TCP**
10. Under **Proxy Server** change Type to **SOCKS5**.
11. Change **Host:** to **10.8.0.1**.
12. Change **Port** to **1080**.
13. Checkmark the box next to **use proxy for peer connections**.
14. Checkmark the box next to **Disable connections not supported by proxies**.
15. Disable (Uncheck) Use PNP / NAT - PMP
16. Click on **Connection**

To

- https://mullvad.net/en/help/bittorrent/

1. Click on **Tools** followed by **Options** (Alt-O).
2. Click on **Connection**.
3. Click on BitTorrent
4. Enable (Check) Enable anonymous mode
5. Disable (Uncheck) Enable DHT
6. Disable (Uncheck) Enable PeX
7. Disable (Uncheck) Enable Local peer discovery

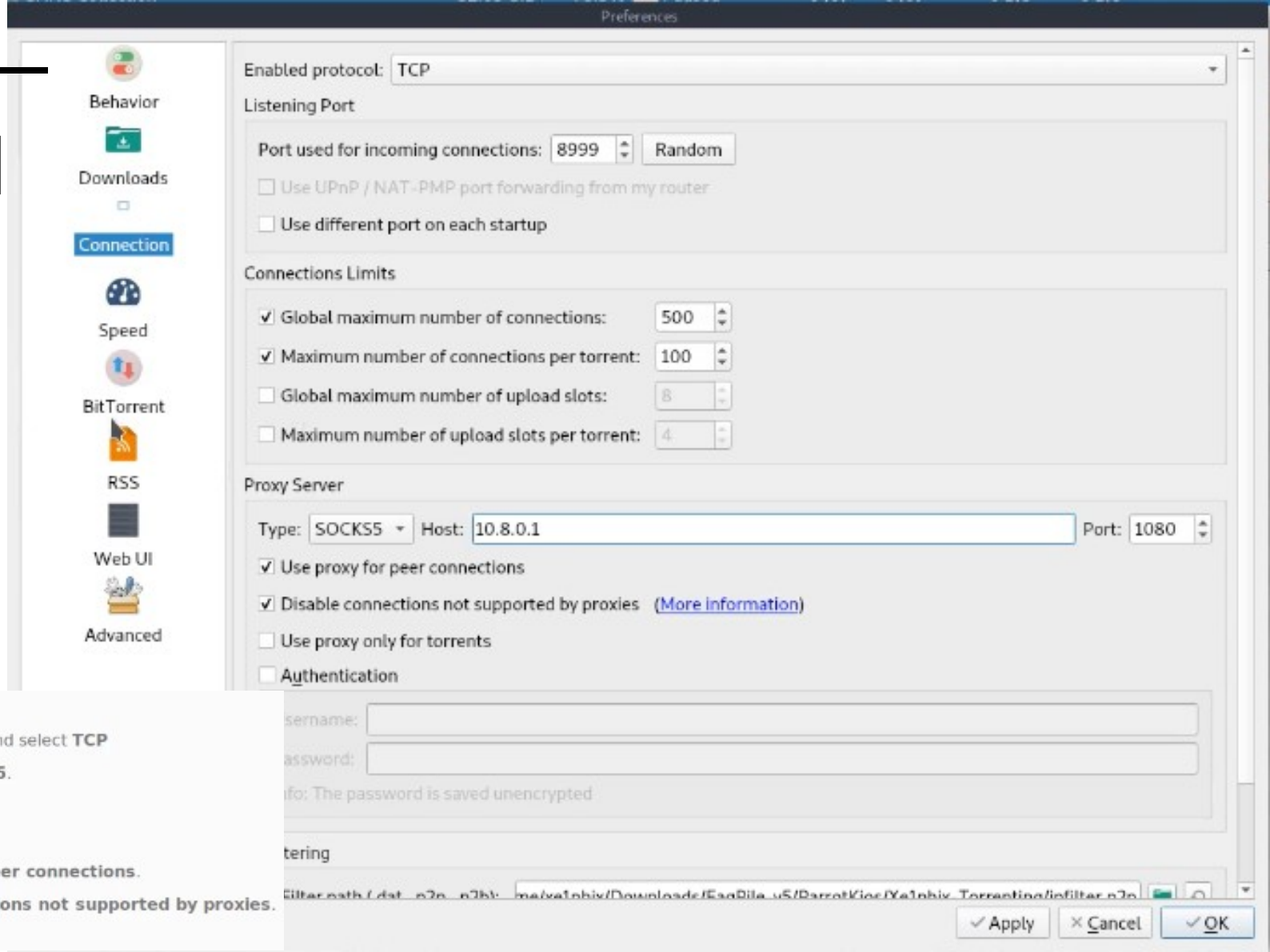


- https://mullvad.net/en/help/bittorrent/

qBittorrent – Anonymous Mode v2

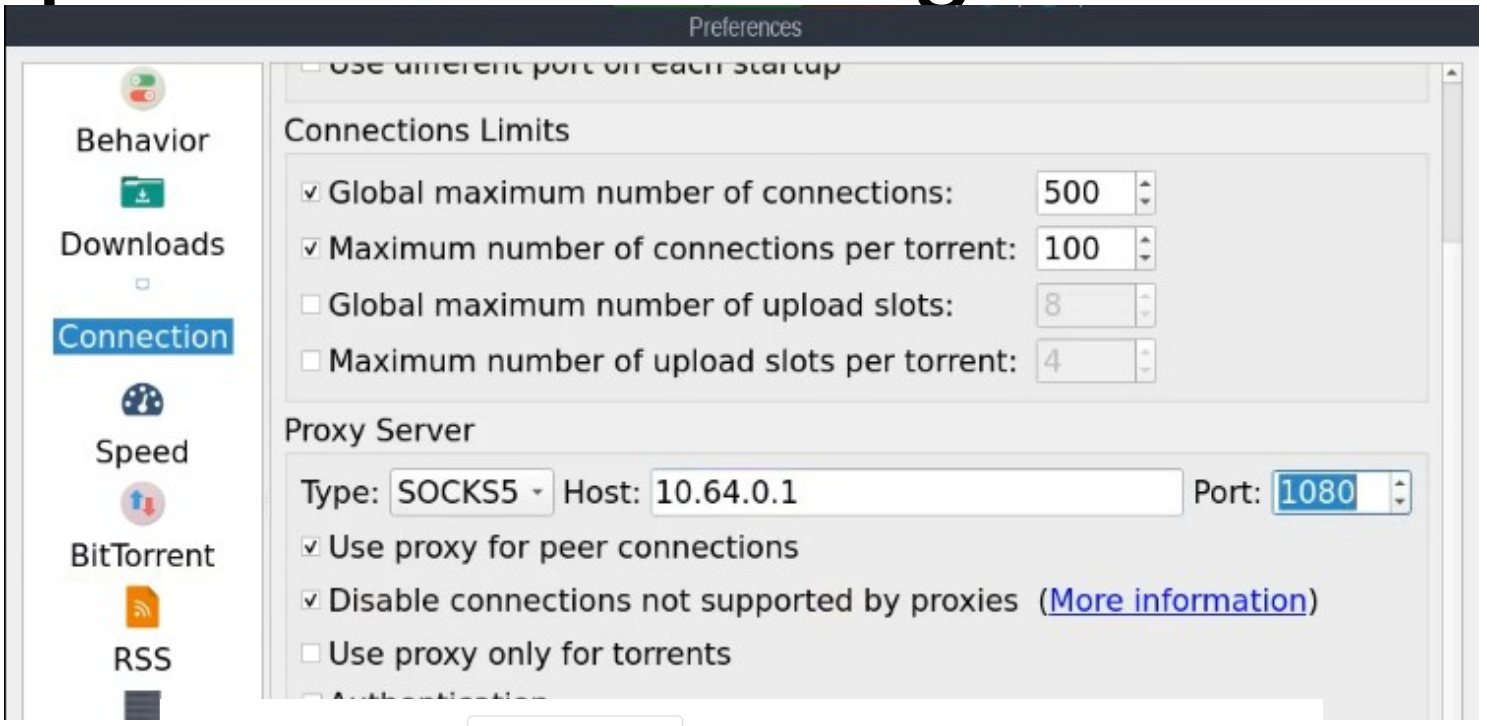
- Disables Local Peer Discovery
- Disables DHT
- Disables UPnP & NAT-PMP
- Only talks to http(s) trackers via (any) proxy
- Only talks to udp trackers via SOCKS5/I2P proxy
- The peer-ID will no longer include the client's fingerprint
- The user-agent will be reset to an empty string
- Other identifying information will not be exposed to the public directly, such as IP, listening port, etc.
- The announced port used to be `0` but changed to `1` for versions using libtorrent 1.2.5 or later, since some trackers would fail the announce for `0`.

qBittorrent – OpenVPN SOCKS5 Proxy



8. Click on **Connection**
9. For Enabled protocol: Use the drop-down bar and select **TCP**
10. Under **Proxy Server** change Type to **SOCKS5**.
11. Change **Host:** to **10.8.0.1**.
12. Change **Port** to **1080**.
13. Checkmark the box next to **use proxy for peer connections**.
14. Checkmark the box next to **Disable connections not supported by proxies**.
15. Disable (Uncheck) Use PNP / NAT - PMP

qBittorrent – Wireguard VPN



FAQ | Port check | Torrent check | API

Use this to check if your torrent client is leaking your actual IP address.

No leaks detected

Detected IP addresses

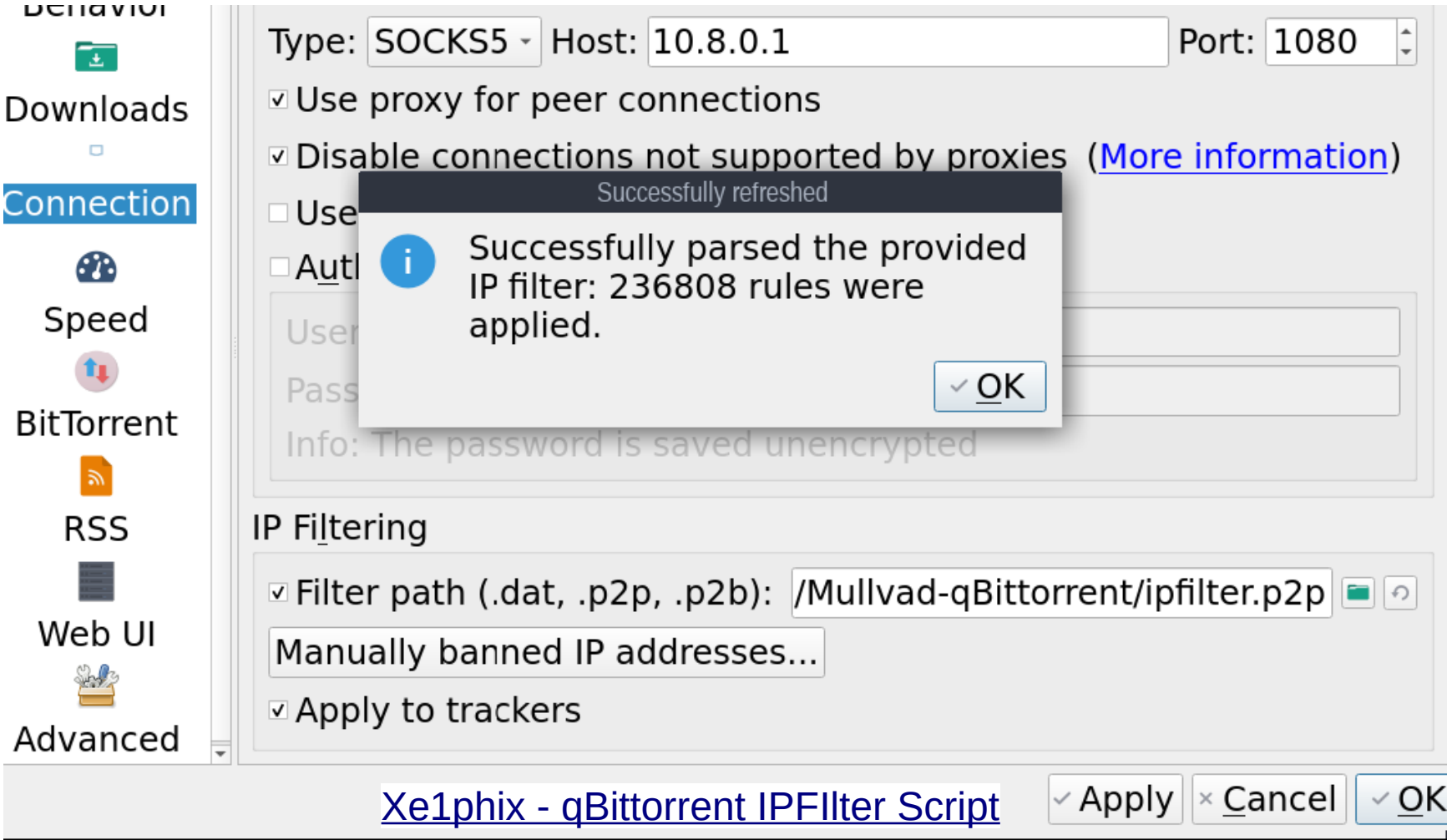
- 185.65.135.168 - 31173 Services AB (Sweden) **Mullvad IP: se-sto-008**

Fetch IP Filter For qBittorrent

```
└──╼ $wget -O - http://list.iblocklist.com/\?list\=ydxerpxkpcfqjaybcssw\&fileformat\=p2p\&a
rchiveformat\=gz | gunzip > ~/ipfilter.p2p
--2020-09-29 05:03:08--  http://list.iblocklist.com/?list=ydxerpxkpcfqjaybcssw&fileformat=p
2p&archiveformat=gz
Resolving list.iblocklist.com (list.iblocklist.com)... 54.175.167.220
Connecting to list.iblocklist.com (list.iblocklist.com)|54.175.167.220|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://cdn1.iblocklist.com/files/7rcwpdsg7s6t9l5774gk/ydxerpxkpcfqjaybcssw.gz [fo
llowing]
--2020-09-29 05:03:09--  http://cdn1.iblocklist.com/files/7rcwpdsg7s6t9l5774gk/ydxerpxkpcfq
jaybcssw.gz
Resolving cdn1.iblocklist.com (cdn1.iblocklist.com)... 165.227.124.62
Connecting to cdn1.iblocklist.com (cdn1.iblocklist.com)|165.227.124.62|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3664582 (3.5M) [application/x-gzip]
Saving to: ‘STDOUT’

-                   100%[=========================>]   3.49M  9.40MB/s    in 0.4s

2020-09-29 05:03:09 (9.40 MB/s) - written to stdout [3664582/3664582]
```

[Xe1phix - qBittorrent IPFIlter Script]

qBittorrent - Applying IPFilters



qBittorrent – Firejail Profile

```
apparmor
caps.drop all
machine-id
netfilter
nodvd
nogroups
nonewprivs
noroot
nosound
notv
nou2f
novideo
protocol unix,inet,netlink
seccomp
shell none

private-bin python*,qbittorrent
private-dev
```

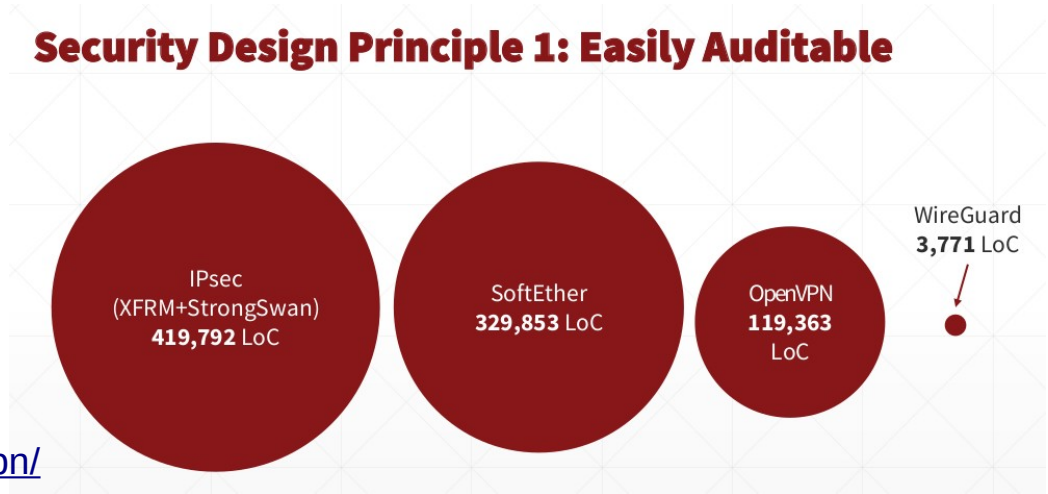
```
include allow-python2.inc
include allow-python3.inc

include disable-common.inc
include disable-devel.inc
include disable-exec.inc
include disable-interpreters.inc
include disable-passwdmgr.inc
include disable-programs.inc
include disable-shell.inc

mkdir ${HOME}/.cache/qBittorrent
mkdir ${HOME}/.config/qBittorrent
mkfile ${HOME}/.config/qBittorrentrc
mkdir ${HOME}/.local/share/data/qBittorrent
whitelist ${DOWNLOADS}
whitelist ${HOME}/.cache/qBittorrent
whitelist ${HOME}/.config/qBittorrent
whitelist ${HOME}/.config/qBittorrentrc
whitelist ${HOME}/.local/share/data/qBittorrent
include whitelist-common.inc
```

qBittorrent – Firejail Profile

Mullvad's WireGuard servers

Our WireGuard servers utilize the following protocols and primitives:

- ChaCha20 for symmetric encryption, authenticated with Poly1305, using RFC7539's AEAD construction
- Curve25519 for ECDH
- BLAKE2s for hashing and keyed hashing, as described in RFC7693
- SipHash24 for hashtable keys
- HKDF for key derivation, as described in RFC5869
- Noise_IK handshake from Noise, building on the work of CurveCP, NaCL, KEA+, SIGMA, FHMQV, and HOMQV.



https://mullvad.net/en/what-is-vpn/

WIREGUARD®
FAST, MODERN, SECURE VPN TUNNEL

WireGuard configuration file generator

Follow our WireGuard guides for step-by-step instructions on how to use WireGuard with Mullvad.

1. Choose your platform

Windows | macOS | Linux | iOS

Android/Chrome OS

2. Generate a WireGuard key

Generate key | No key generated

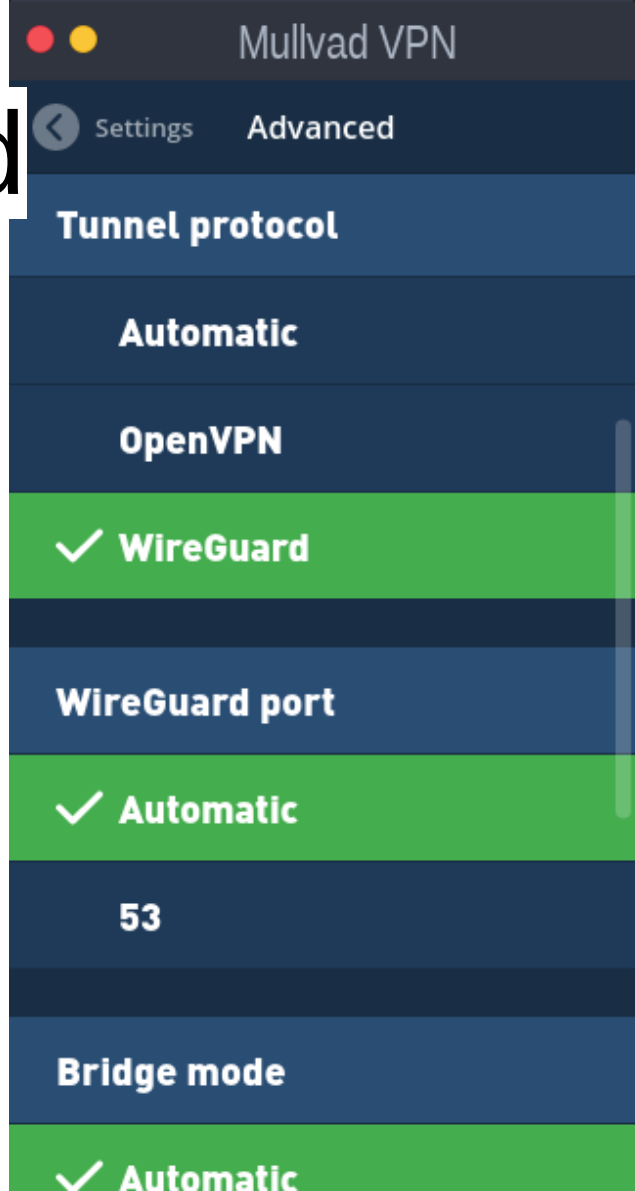
Manage keys

https://mullvad.net/en/account/#/wireguard-config/

Using WireGuard

1. Click on the **gear icon.**
2. Click on **Advanced**.
3. Under **Tunnel protocol**, select **WireGuard**.

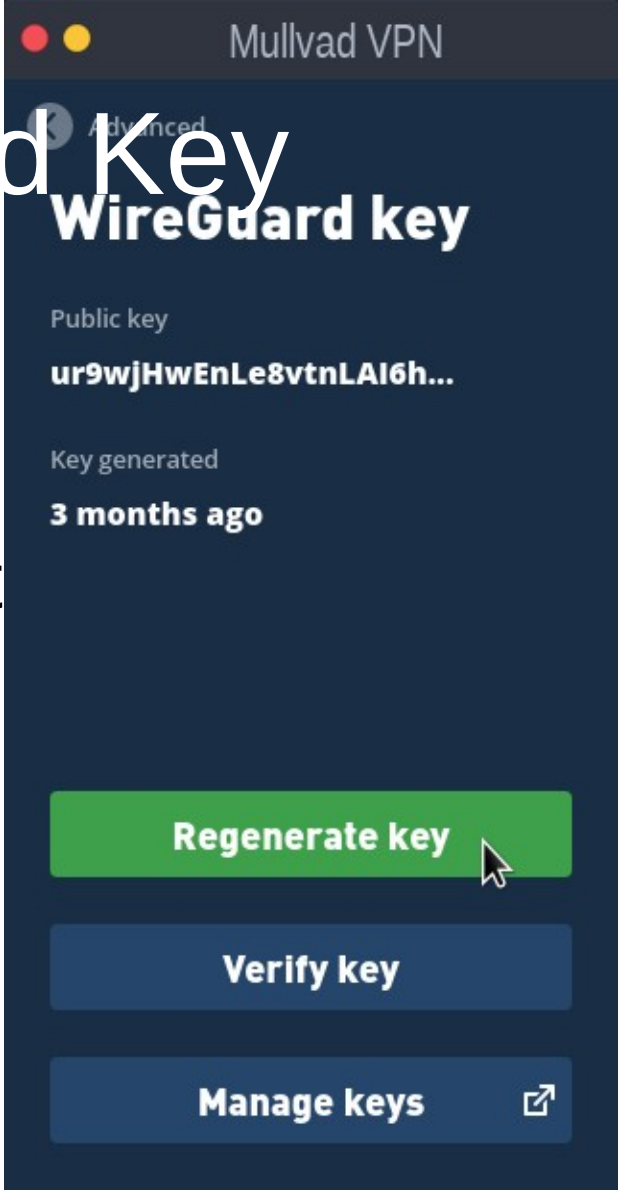
WireGuard is the default protocol.



https://mullvad.net/en/help/how-turn-wireguard-mullvad-app/

Regenerate WireGuard Key

- **Regenerate key**: Replace your current key. This will also replace your internal static IP address.
- **Verify key**: This will verify your current key.
- **Manage keys**: Open your account page on our website so you can get an overview of all your keys.



Mullvad - WireGuard Configuration Script:

```
##-=============================================-##
##    [+] Run The Mullvad - WireGuard Configuration Script:
##-=============================================-##
curl -LO https://mullvad.net/media/files/mullvad-wg.sh
chmod +x ./mullvad-wg.sh
./mullvad-wg.sh
```

Executing Mullvads WireGuard Configuration Script:

```
└──╼ #chmod +x mullvad-wg.sh
┌─[root@parrot]─[/home/parrotsec-kiosk/Downloads/Scripts/ParrotLinux-Publ
pdated/[05-11-20]/Xe1phix-[Mullvad]/Xe1phix-Mullvad-[Wireguard]-(Configs)
└──╼ #./mullvad-wg.sh
[?] Please enter your Mullvad account number: [redacted]
[+] Contacting Mullvad API for server locations.
[+] Using existing private key.
[+] Contacting Mullvad API.
[+] Writing WriteGuard configuration files.
[+] Success. The following commands may be run for connecting to Mullvad:
- Melbourne, Australia:
  $ wg-quick up mullvad-au3
- Melbourne, Australia:
  $ wg-quick up mullvad-au4
- Sydney, Australia:
  $ wg-quick up mullvad-au10
- Sydney, Australia:
  $ wg-quick up mullvad-au11
- Sydney, Australia:
  $ wg-quick up mullvad-au12
- Sydney, Australia:
  $ wg-quick up mullvad-au14
- Sydney, Australia:
  $ wg-quick up mullvad-au1
- Sydney, Australia:
  $ wg-quick up mullvad-au2
- Sydney, Australia:
```

Setting Wireguard Directory Permissions

```
##-==========================================-##
##    [+] Set Strict Permissions For The Wireguard Directory:
##-==========================================-##
## ------------------------------------------ ##
##    [?] So Only Root Can Read Them
## ------------------------------------------ ##
chown root:root -R /etc/wireguard
chmod 600 -R /etc/wireguard
```

```
##-==========================================-##
##    [+] Start WireGuard automatically on boot:
##-==========================================-##
systemctl enable wg-quick@mullvad-se4

##-==========================-##
##    [+] Turn on WireGuard
##-==========================-##
wg-quick up mullvad-se4

##-==========================-##
##    [+] Turn off WireGuard
##-==========================-##
wg-quick down mullvad-se4
```

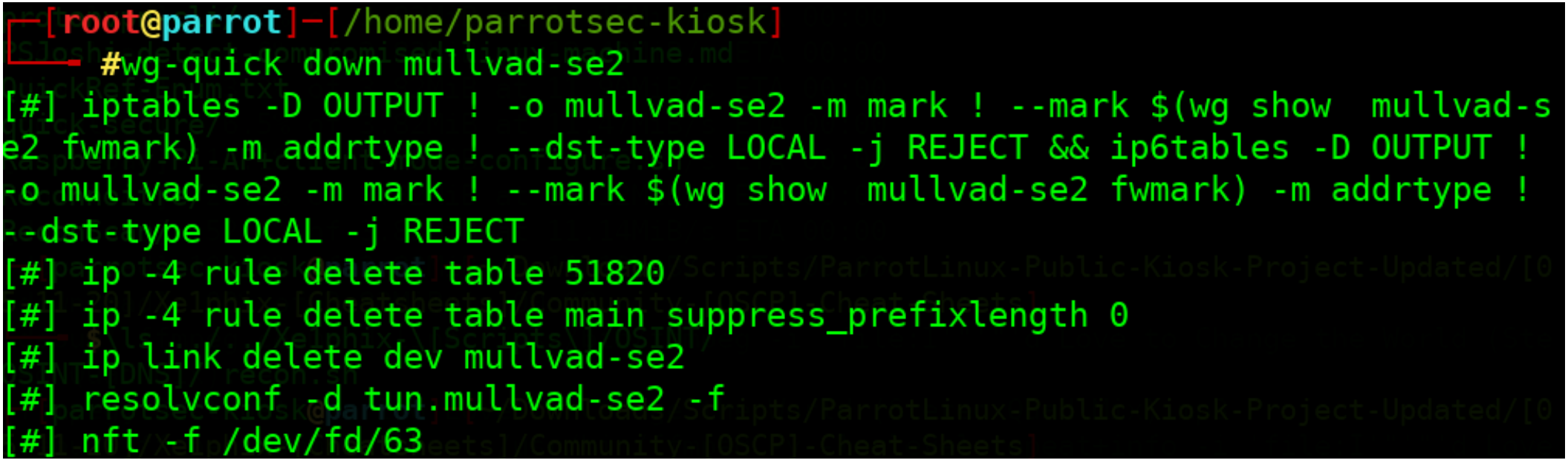
```
┌─[parrotsec-kiosk@parrot]─[~]
└──╼ $ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.101  netmask 255.255.255.0  broadcast 192.168.1.255
        ether [redacted]  txqueuelen 1000  (Ethernet)
        RX packets 110525696  bytes 154223654529 (143.6 GiB)
        RX errors 0  dropped 2971  overruns 0  frame 0
        TX packets 60166078  bytes 15588710581 (14.5 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xe1500000-e1520000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 76017  bytes 6290667 (5.9 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 76017  bytes 6290667 (5.9 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

mullvad-se2: flags=209<UP,POINTOPOINT,RUNNING,NOARP>  mtu 1420
        inet [redacted]  netmask 255.255.255.255  destination [redacted]
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 1000  (UNSPEC)
        RX packets 4572595  bytes 6355760108 (5.9 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3382508  bytes 357405840 (340.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #wg show all
interface: mullvad-se2
  public key: [redacted]
  private key: (hidden)
  listening port: 39401
  fwmark: 0xca6c

peer: [redacted]
  endpoint: [redacted]:51820
  allowed ips: 0.0.0.0/0
  latest handshake: 1 minute, 51 seconds ago
  transfer: 2.73 GiB received, 126.99 MiB sent
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #wg show
show    showconf
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #wg showconf mullvad-se2
[Interface]
ListenPort = 39401
FwMark = 0xca6c
PrivateKey = [redacted]

[Peer]
PublicKey = [redacted]
AllowedIPs = 0.0.0.0/0
Endpoint = [redacted]:51820
```

Setting The Wireguard Interface Up

```
└─# wg-quick up mullvad-se2
[#] ip link add mullvad-se2 type wireguard
[#] wg setconf mullvad-se2 /dev/fd/63
[#] ip -4 address add [redacted]/32 dev mullvad-se2
[#] ip link set mtu 1420 up dev mullvad-se2
[#] resolvconf -a tun.mullvad-se2 -m 0 -x
[#] wg set mullvad-se2 fwmark 51820
[#] ip -4 route add 0.0.0.0/0 dev mullvad-se2 table 51820
[#] ip -4 rule add not fwmark 51820 table 51820
[#] ip -4 rule add table main suppress_prefixlength 0
[#] sysctl -q net.ipv4.conf.all.src_valid_mark=1
[#] nft -f /dev/fd/63
[#] systemd-resolve -i mullvad-se2 --set-dns=193.138.218.74 --set-domain=~.
[#] iptables -I OUTPUT ! -o mullvad-se2 -m mark ! --mark $(wg show mullvad-se2 fwmark) -m a
ddrtype ! --dst-type LOCAL -j REJECT && ip6tables -I OUTPUT ! -o mullvad-se2 -m mark ! --ma
rk $(wg show mullvad-se2 fwmark) -m addrtype ! --dst-type LOCAL -j REJECT
```

Setting The Wireguard Interface Down



WireGuard configuration file generator

Follow our WireGuard guides for step-by-step instructions on how to use WireGuard with Mullvad.

1. Choose your platform

Windows | macOS | Linux | iOS | Android/Chrome OS

2. Generate a WireGuard key

Generate key | No key generated

Manage keys

3. Select one or multiple exit locations

USA | Denver, CO | us10-wireguard

Advanced settings

Multihop

enable

Select an entry server

Sweden | Stockholm | se2-wireguarc

Server connection protocol

IPv4 IPv6

Tunnel traffic

both Only IPv4 Only IPv6

Custom port

51820

Enable kill switch (Linux only)

Multihop - Another layer of security

Even though a standard, single-hop VPN configuration will be adequate for the majority of users, incoming/outgoing traffic correlation may still be possible. Multihop adds another level of security for those concerned where the correlating of in and outgoing traffic over several locations (with different ISP and hosting providers) and preferably nations, becomes even more difficult.

How

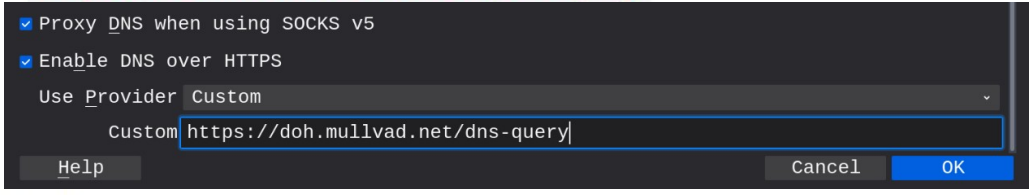
For instance, if you are connected to se1-wireguard.mullvad.net and then want to exit via us3-wireguard.mullvad.net, you would configure your browser/program to use us3-wg.socks5.mullvad.net on port 1080 as your exit node.

DoH [DNS-over-HTTPS] + DoT [DNS-over-TLS]

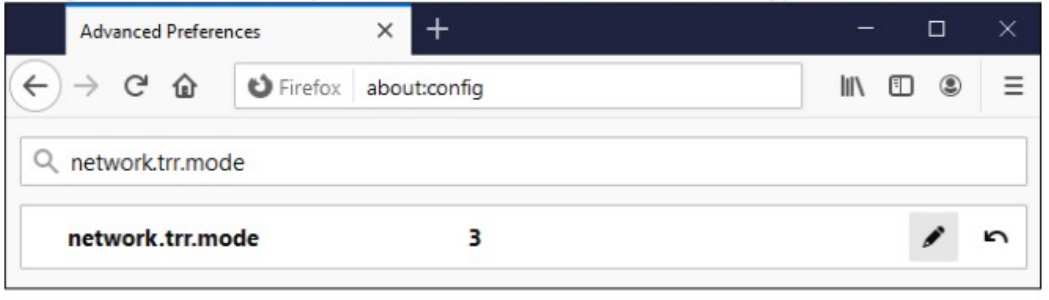
How to use our DNS service

Firefox

1. In a Firefox browser window, click the **menu button** and choose **Options** or **Preferences**.
2. In the search box, type "**network**", then click on the **Settings** button in the results.
3. At the bottom, check the box next to **Enable DNS over HTTPS**.
4. Next to **Use Provider**, choose **Custom**.
5. In the text box that appears, enter `https://doh.mullvad.net/dns-query` or `https://adblock.doh.mullvad.net/dns-query`



6. Click OK.
7. In the address bar of the browser, type in `about:config` and hit Enter.
8. If a warning pops up, click "Accept the Risk and Continue".
9. In the search box, type `network.trr.mode`
10. Change the value to **3** and press **Enter**. (this will disable the unencrypted fallback).



DoT only uses port **853**, while DoH uses port **443**.

Without ad blocking

doh.mullvad.net has address 194.242.2.2
doh.mullvad.net has address 193.19.108.2
doh.mullvad.net has IPv6 address 2a07:e340::2

With ad blocking

adblock.doh.mullvad.net has address 194.242.2.3
adblock.doh.mullvad.net has address 193.19.108.3
adblock.doh.mullvad.net has IPv6 address 2a07:e340::3

Wireguard – Packet Capturing

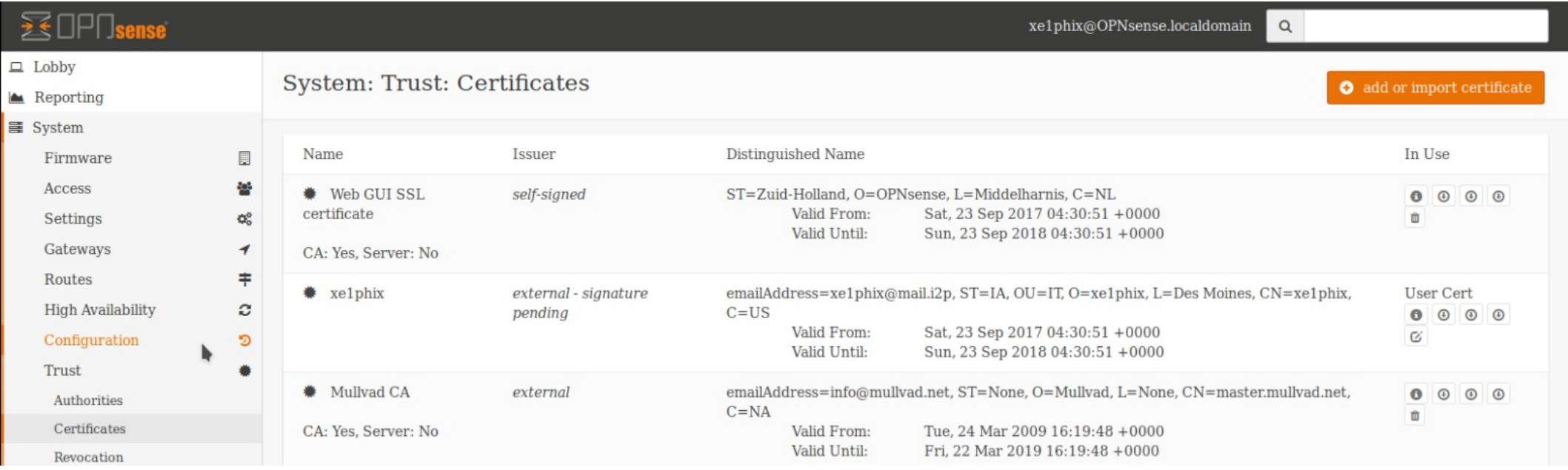
```
┌─[✗]─[root@parrotseckiosk-optiplex990]─[/home/parrotseckiosk/Downloads]
└──╼ #tcpdump -vn -i 2 -w Wireguard2.pcap
tcpdump: listening on wg-mullvad, link-type RAW (Raw IP), snapshot length 262144 bytes
^C1522 packets captured
1522 packets received by filter
0 packets dropped by kernel
```

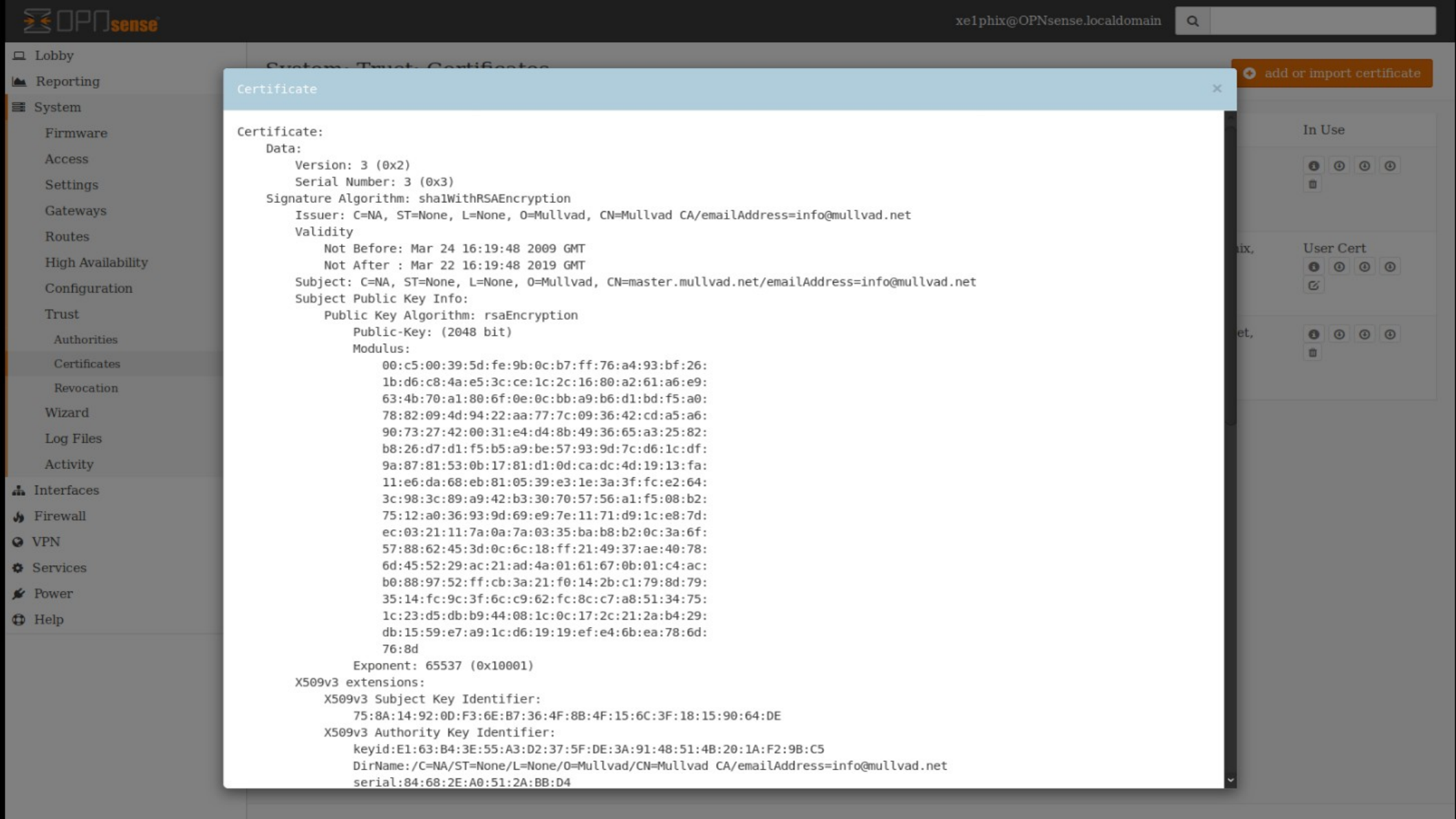
```
└──╼ #tcpdump -vn -i any 'port 51820'
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
23:18:49.592702 eth0  Out IP (tos 0x88, ttl 64, id 49418, offset 0, flags [none], proto UDP (17), length 176)
    173.22.75.86.35865 > 193.138.218.80.51820: UDP, length 148
23:18:49.725905 eth0  In  IP (tos 0x0, ttl 47, id 3184, offset 0, flags [none], proto UDP (17), length 120)
    193.138.218.80.51820 > 173.22.75.86.35865: UDP, length 92
23:18:49.726272 eth0  Out IP (tos 0x0, ttl 64, id 49439, offset 0, flags [none], proto UDP (17), length 124)
    173.22.75.86.35865 > 193.138.218.80.51820: UDP, length 96
23:18:49.726291 eth0  Out IP (tos 0x0, ttl 64, id 49440, offset 0, flags [none], proto UDP (17), length 124)
    173.22.75.86.35865 > 193.138.218.80.51820: UDP, length 96
└──╼ #tshark -i any -f 'port 51820'
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'
    1 0.000000000 193.138.218.82 → 173.22.75.86 WireGuard 140 Transport Data, receiver=0x3AB8D0E6, counter=19, datalen=64
    2 0.000137455 173.22.75.86 → 193.138.218.82 WireGuard 140 Transport Data, receiver=0xF118CCB1, counter=64, datalen=64
    3 0.000321491 173.22.75.86 → 193.138.218.82 WireGuard 428 Transport Data, receiver=0xF118CCB1, counter=65, datalen=352
    4 0.010524773 173.22.75.86 → 193.138.218.82 WireGuard 172 Transport Data, receiver=0xF118CCB1, counter=66, datalen=96
    5 0.010535063 173.22.75.86 → 193.138.218.82 WireGuard 172 Transport Data, receiver=0xF118CCB1, counter=67, datalen=96
    6 0.010540163 173.22.75.86 → 193.138.218.82 WireGuard 156 Transport Data, receiver=0xF118CCB1, counter=68, datalen=80
    7 0.060060201 193.138.218.82 → 173.22.75.86 WireGuard 140 Transport Data, receiver=0x3AB8D0E6, counter=20, datalen=64
    8 0.060190618 173.22.75.86 → 193.138.218.82 WireGuard 140 Transport Data, receiver=0xF118CCB1, counter=69, datalen=64
```

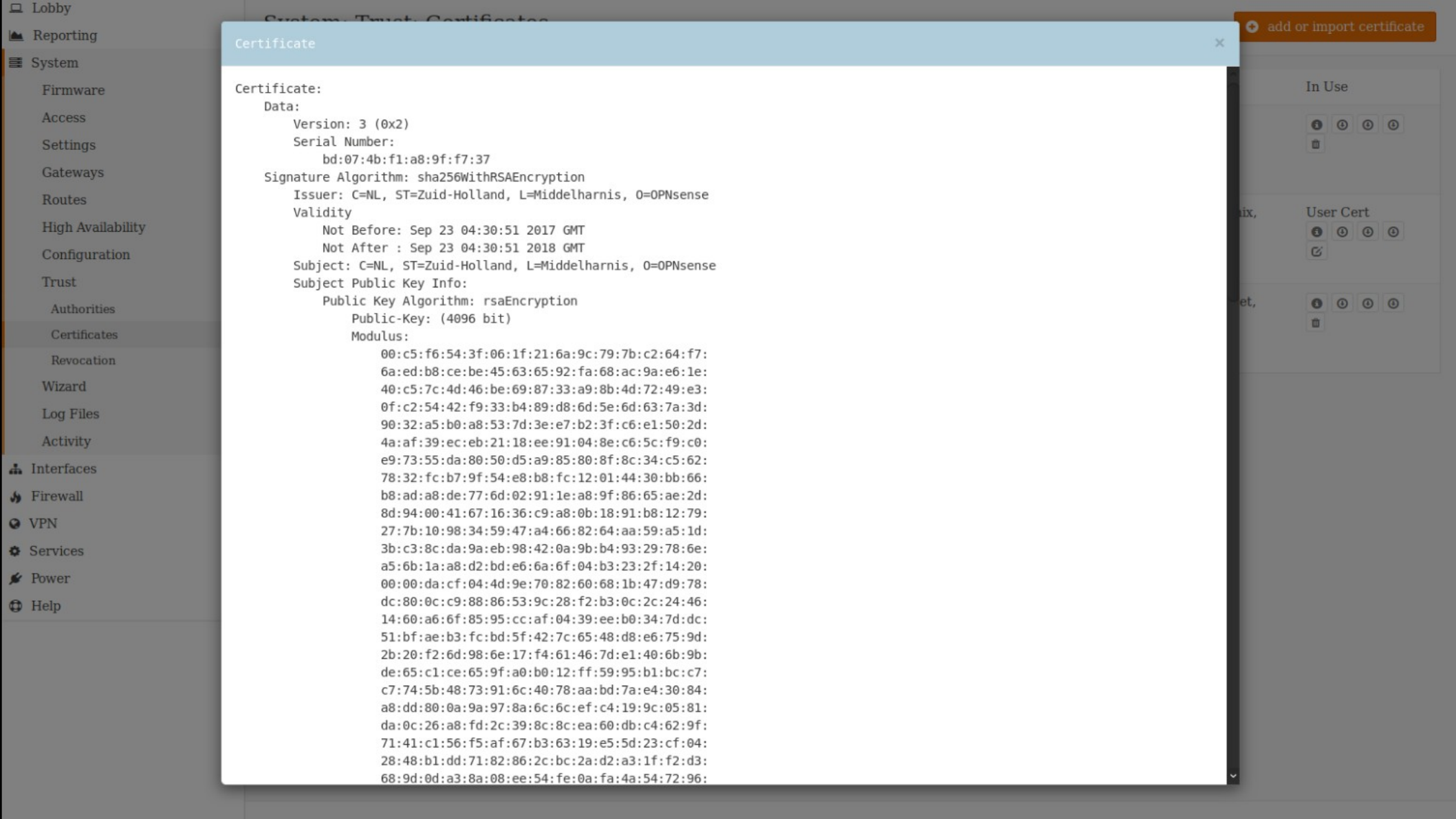
```
$ curl https://am.i.mullvad.net/json
{
  "ip": "[redacted]",
  "country": "Sweden",
  "city": null,
  "longitude": [redacted],
  "latitude": [redacted],
  "mullvad_exit_ip": true,
  "mullvad_exit_ip_hostname": "se2-wireguard",
  "mullvad_server_type": "SOCKS through WireGuard",
  "blacklisted": {
    "blacklisted": false,
    "results": [
      {
        "name": "Project Honeypot",
        "link": "https://www.projecthoneypot.org/about_us.php",
        "blacklisted": false
      },
      {
        "name": "Spamhaus",
        "link": "https://www.spamhaus.org/organization/",
        "blacklisted": false
```

OPNSense

OPNSense – Import Mullvad Certificate




xe1phix@OPNsense.localdomain
Lobby
Reporting
System
Firmware
Access
Settings
Gateways
Routes
High Availability
Configuration
Trust
Authorities
Certificates
Revocation
Wizard
Log Files
Activity
Interfaces
Firewall
VPN
Services
Power
Help
add or import certificate
In Use
User Cert
Certificate
Certificate:
Data:
Version: 3 (0x2)
Serial Number: 3 (0x3)
Signature Algorithm: sha1WithRSAEncryption
Issuer: C=NA, ST=None, L=None, O=Mullvad, CN=Mullvad CA/emailAddress=info@mullvad.net
Validity
Not Before: Mar 24 16:19:48 2009 GMT
Not After : Mar 22 16:19:48 2019 GMT
Subject: C=NA, ST=None, L=None, O=Mullvad, CN=master.mullvad.net/emailAddress=info@mullvad.net
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
Public-Key: (2048 bit)
Modulus:
00:c5:00:39:5d:fe:9b:0c:b7:ff:76:a4:93:bf:26:
1b:d6:c8:4a:e5:3c:ce:1c:2c:16:80:a2:61:a6:e9:
63:4b:70:a1:80:6f:0e:0c:bb:a9:b6:d1:bd:f5:a0:
78:82:09:4d:94:22:aa:77:7c:09:36:42:cd:a5:a6:
90:73:27:42:00:31:e4:d4:8b:49:36:65:a3:25:82:
b8:26:d7:d1:f5:b5:a9:be:57:93:9d:7c:d6:1c:df:
9a:87:81:53:0b:17:81:d1:0d:ca:dc:4d:19:13:fa:
11:e6:da:68:eb:81:05:39:e3:1e:3a:3f:fc:e2:64:
3c:98:3c:89:a9:42:b3:30:70:57:56:a1:f5:08:b2:
75:12:a0:36:93:9d:69:e9:7e:11:71:d9:1c:e8:7d:
ec:03:21:11:7a:0a:7a:03:35:ba:b8:b2:0c:3a:6f:
57:88:62:45:3d:0c:6c:18:ff:21:49:37:ae:40:78:
6d:45:52:29:ac:21:ad:4a:01:61:67:0b:01:c4:ac:
b0:88:97:52:ff:cb:3a:21:f0:14:2b:c1:79:8d:79:
35:14:fc:9c:3f:6c:c9:62:fc:8c:c7:a8:51:34:75:
1c:23:d5:db:b9:44:08:1c:0c:17:2c:21:2a:b4:29:
db:15:59:e7:a9:1c:d6:19:19:ef:e4:6b:ea:78:6d:
76:8d
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Subject Key Identifier:
75:8A:14:92:0D:F3:6E:B7:36:4F:8B:4F:15:6C:3F:18:15:90:64:DE
X509v3 Authority Key Identifier:
keyid:E1:63:B4:3E:55:A3:D2:37:5F:DE:3A:91:48:51:4B:20:1A:F2:9B:C5
DirName:/C=NA/ST=None/L=None/O=Mullvad/CN=Mullvad CA/emailAddress=info@mullvad.net
serial:84:68:2E:A0:51:2A:BB:D4


Lobby
Reporting
System
Firmware
Access
Settings
Gateways
Routes
High Availability
Configuration
Trust
Authorities
Certificates
Revocation
Wizard
Log Files
Activity
Interfaces
Firewall
VPN
Services
Power
Help
add or import certificate
In Use
User Cert
Certificate
Certificate:
Data:
Version: 3 (0x2)
Serial Number:
bd:07:4b:f1:a8:9f:f7:37
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=NL, ST=Zuid-Holland, L=Middelharnis, O=OPNsense
Validity
Not Before: Sep 23 04:30:51 2017 GMT
Not After : Sep 23 04:30:51 2018 GMT
Subject: C=NL, ST=Zuid-Holland, L=Middelharnis, O=OPNsense
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
Public-Key: (4096 bit)
Modulus:
00:c5:f6:54:3f:06:1f:21:6a:9c:79:7b:c2:64:f7:
6a:ed:b8:ce:be:45:63:65:92:fa:68:ac:9a:e6:1e:
40:c5:7c:4d:46:be:69:87:33:a9:8b:4d:72:49:e3:
0f:c2:54:42:f9:33:b4:89:d8:6d:5e:6d:63:7a:3d:
90:32:a5:b0:a8:53:7d:3e:e7:b2:3f:c6:e1:50:2d:
4a:af:39:ec:eb:21:18:ee:91:04:8e:c6:5c:f9:c0:
e9:73:55:da:80:50:d5:a9:85:80:8f:8c:34:c5:62:
78:32:fc:b7:9f:54:e8:b8:fc:12:01:44:30:bb:66:
b8:ad:a8:de:77:6d:02:91:1e:a8:9f:86:65:ae:2d:
8d:94:00:41:67:16:36:c9:a8:0b:18:91:b8:12:79:
27:7b:10:98:34:59:47:a4:66:82:64:aa:59:a5:1d:
3b:c3:8c:da:9a:eb:98:42:0a:9b:b4:93:29:78:6e:
a5:6b:1a:a8:d2:bd:e6:6a:6f:04:b3:23:2f:14:20:
00:00:da:cf:04:4d:9e:70:82:60:68:1b:47:d9:78:
dc:80:0c:c9:88:86:53:9c:28:f2:b3:0c:2c:24:46:
14:60:a6:6f:85:95:cc:af:04:39:ee:b0:34:7d:dc:
51:bf:ae:b3:fc:bd:5f:42:7c:65:48:d8:e6:75:9d:
2b:20:f2:6d:98:6e:17:f4:61:46:7d:e1:40:6b:9b:
de:65:c1:ce:65:9f:a0:b0:12:ff:59:95:b1:bc:c7:
c7:74:5b:48:73:91:6c:40:78:aa:bd:7a:e4:30:84:
a8:dd:80:0a:9a:97:8a:6c:6c:ef:c4:19:9c:05:81:
da:0c:26:a8:fd:2c:39:8c:8c:ea:60:db:c4:62:9f:
71:41:c1:56:f5:af:67:b3:63:19:e5:5d:23:cf:04:
28:48:b1:dd:71:82:86:2c:bc:2a:d2:a3:1f:f2:d3:
68:9d:0d:a3:8a:08:ee:54:fe:0a:fa:4a:54:72:96:

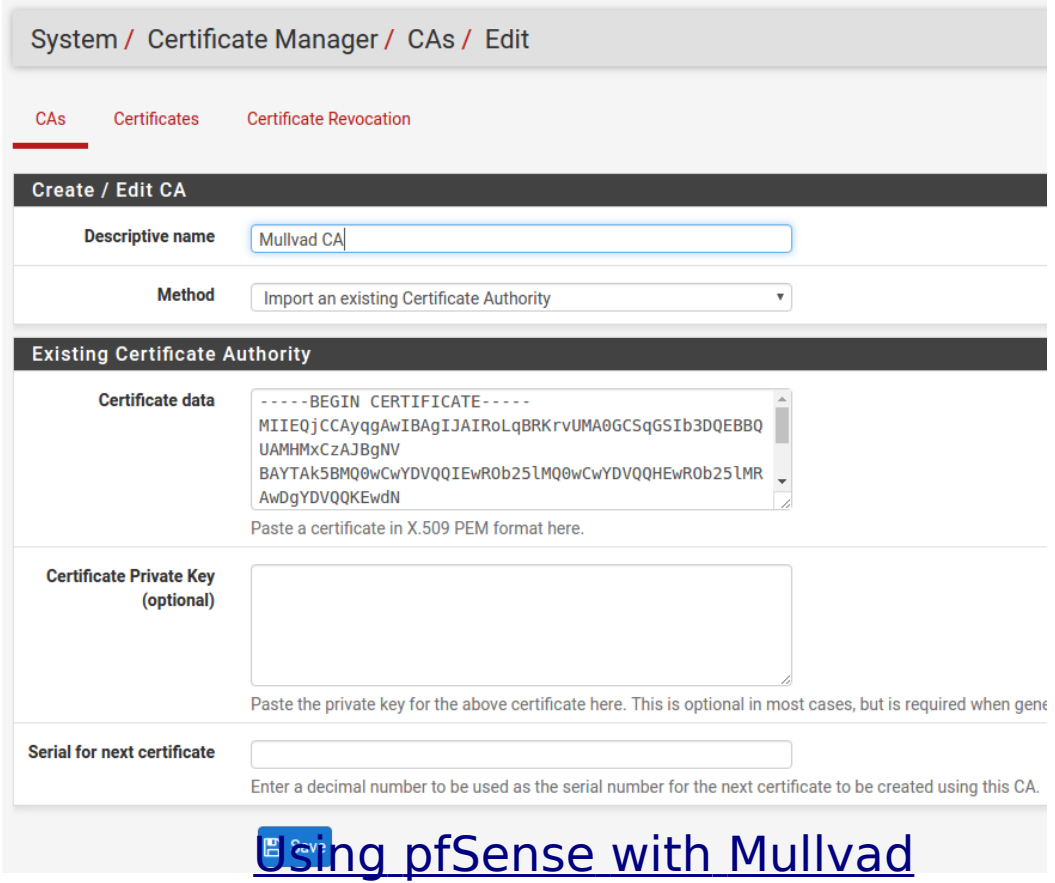
PFSense – OpenVPN OTP setup

OpenVPN One-Time-Password Authentication Using

OpenVPN One-Time-Password Authentication Using

Add the Ca.crt to the Certificate Manager

1. Log in to your pfSense device click on "System" -> "Cert. manager" -> "CAs" and then click on "+Add"
2. Edit the descriptive name and name it Mullvad CA .
3. Set the Method to **Import an existing Certificate Authority**
4. Paste the certificates found in mullvad_ca.crt that was extracted earlier into the "Certificate data" field.
5. Click on Save.

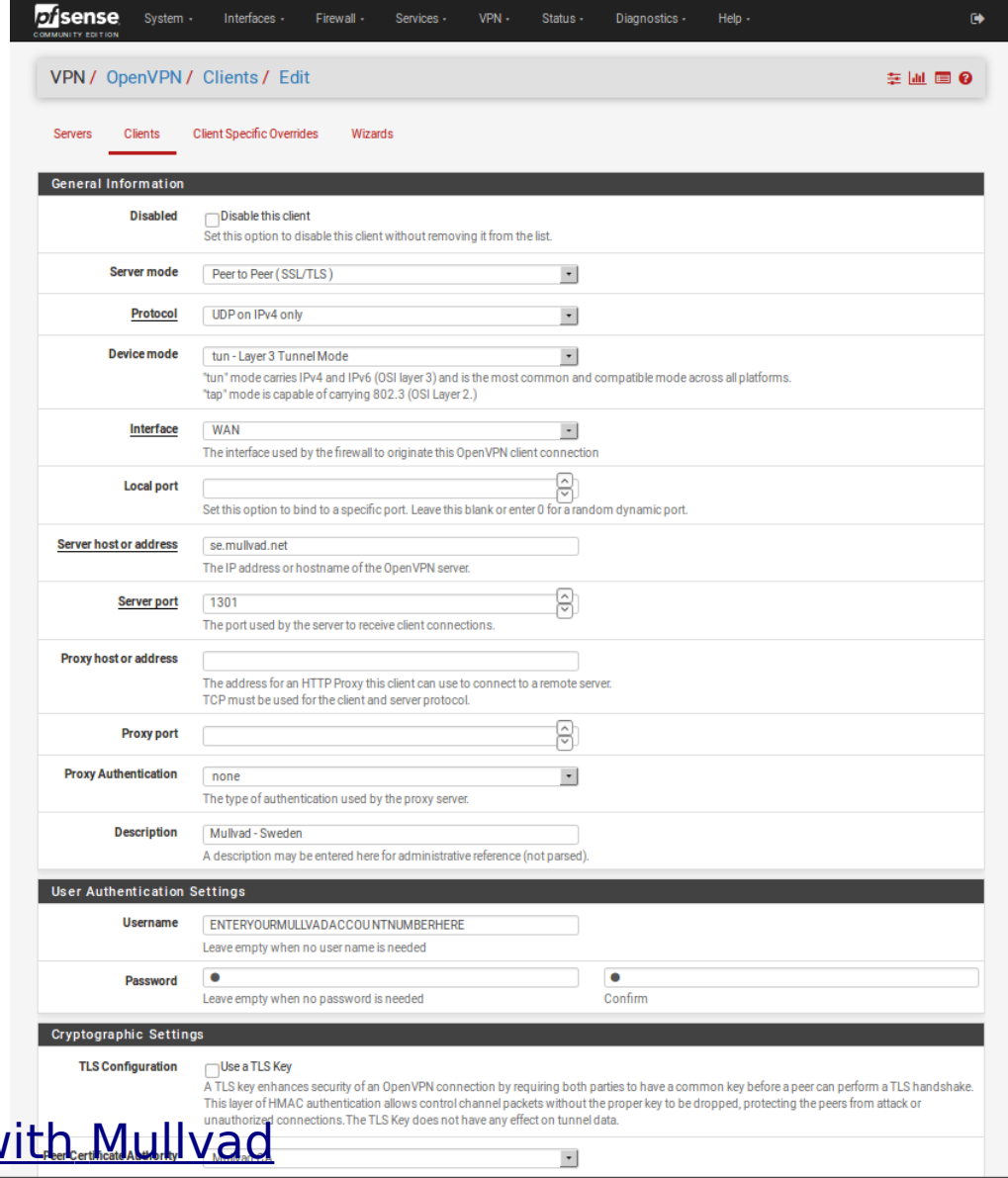


Add a VPN connection

This example will make use of **se**.mullvad.net. You can of course replace this server with any other country, region, or specific server that you wish to use. See our list of available servers.

Click on VPN -> OpenVPN -> Clients and then click on +Add

1. Set Server Mode to: Peer to Peer (SSL/TLS)
2. Set **Protocol** to: **UDP on IPV4 only**
3. Set **Device mode** to: **tun Layer 3 Tunnel Mode**
4. Set **Interface** to: **WAN**
5. Set **Server host** to: **se.mullvad.net**
6. Set **Server port** to: **1301**
7. Set Description to: Mullvad Sweden
8. Set your **mullvad account number as Username** under User Authentication Settings **(make sure it does not contain any spaces)**
9. set M as Password under User Authentication Settings
10. Set **TLS Configuration** to: **Unchecked**
11. Set Peer: **Certificate Authority** to: **Mullvad CA**
12. Set Client Certificate to: None (Username or Password required)
13. Set Encryption Algorithm to: **AES-256-GCM**
14. Set Enable Negotiate Cryptographic Parameters to: Checked
15. Add AES-256-GCM to the Allowed NCP Encryption Algorithms field.
16. Set **Auth digest Algorithm** to: **SHA384**
17. Set **Compression** to: **No LZO Compression** [Legacy style, comp-lzo no]
18. In the **Custom options** field, paste: **remote-cert-tls server**
19. Set UDP Fast I/O to : checked
20. Set Send/Recieve buffer to 1.00 MiB
21. Click Save.



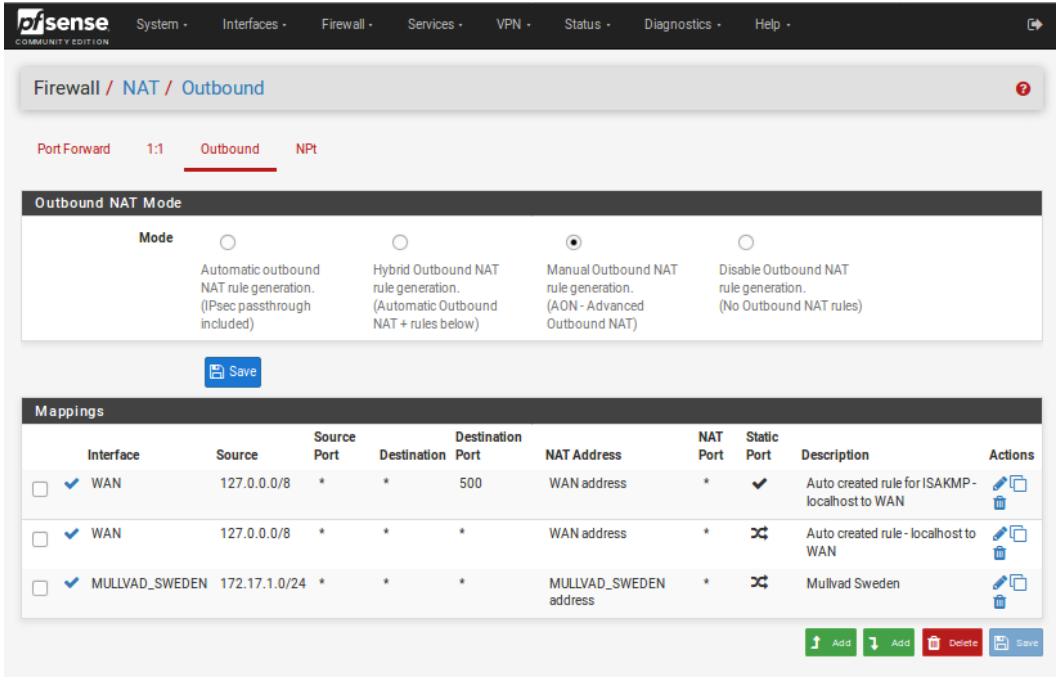
Add an Interface and NAT Rules

Add an Interface

1. Click on **Interfaces** -> **Assignments**
2. Use the Drop-down menu for the Available network ports: and select ovpnc* and then click on **+Add**
3. Click on the New interface name, it is usually named OPT1 or OPT2.
4. Set **Enable: Enable Interface** to be **checked**
5. Click on Save.

Add NAT rules

1. Click on Firewall -> NAT -> Outbound and then select Mode: "Manual Outbound NAT rule Generation (AON) and then click on Save.
2. Copy the entry that contains your local IP address (The one that does not contain port 500 nor 127.0.0.0 , In this example 172.17.1.0/24 is used, for you this will most like differ and will probably be 192.168.1.0/24) by clicking on the Copy icon found under Actions to the right of the NAT entry (Add a new mapping based on this one)
3. Click on the Pen icon (Edit mapping) and change so that interface is the mullvad one and write a description.
4. Make sure that both **Disabled** and **do not NAT** are **unchecked**
5. Delete the other rules that contain your local IP that exists via WAN , (keep the 127.0.0.0) This will ensure that you can not reach the internet if the VPN tunnel is down from your clients behind the pfSense router.
6. Click on Save.



[Using pfSense with Mullvad]

OpenSSH – Config Files

`/etc/ssh/sshd_config`	SSH server daemon configuration file
`/etc/ssh/ssh_config`	SSH client global configuration file
`/etc/ssh/ssh_host_key`	Host's private key (should be mode 0600)
`/etc/ssh/ssh_host_key.pub`	Host's public key
`/etc/ssh/shosts.equiv`	Names of trusted hosts for host-based authentication
`/etc/ssh/ssh_known_hosts`	Database of host public keys that were previously accepted as legitimate
`~/.ssh/`	User's SSH directory (must be mode 0700)
`~/.ssh/config`	SSH client user configuration file
`~/.ssh/id_rsa` `~/.ssh/id_dsa`	User's RSA or DSA private key, as generated by `ssh-keygen`
`~/.ssh/id_rsa.pub` `~/.ssh/id_dsa.pub`	User's RSA or DSA public key, as generated by `ssh-keygen`
`~/.ssh/known_hosts`	Host public keys that were previously accepted as legitimate by the user
`~/.ssh/authorized_keys` `~/.ssh/authorized_keys2` (obsolete)	Trusted public keys; the corresponding private keys allow the user to authenticate on this host

OpenSSH - SFTP

```
## Fetch File using SFTP:
curl sftp://$URL.com/$File.zip -u $User

## Require TLS security for your FTP transfer:
curl --ssl-reqd ftp://ftp.$URL.com/$File.txt

## Fetch File using SCP:
curl scp://$URL.com/$File.zip -u $User
```

Mullvad - Connect-Via-Bridges

- SSH tunneling is one method of using bridges to get around a restrictive firewall.
- Configure SSH SOCKS5
- This involves logging in to our bridge servers and then running a local SOCKS proxy that you can connect OpenVPN to

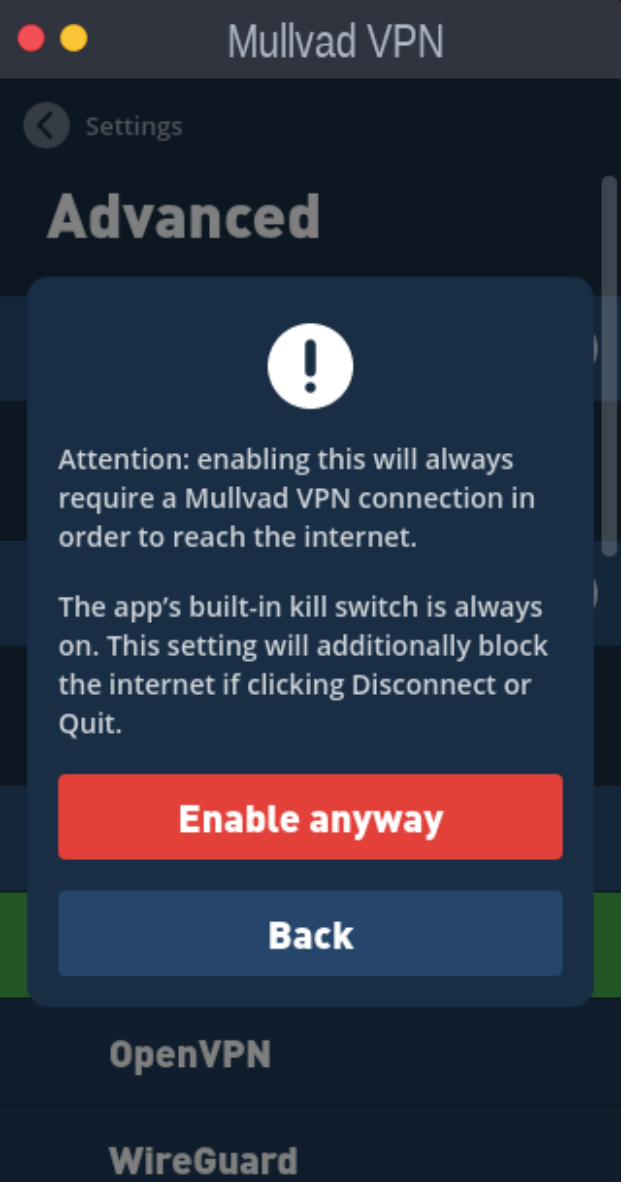
  - https://mullvad.net/en/help/ssh-and-mullvad-vpn/

Mullvad – SSH Tunneling

```
echo "##-==========================================-##"
echo "       [+] SSH tunneling to connect to Mullvads VPN servers      "
echo "##-==========================================-##"
ssh -f -N -D 1234 mullvad@193.138.219.43
```

- https://mullvad.net/en/help/ssh-and-mullvad-vpn/

Enable Mullvad KillSwitch



Enabling Kill Switch Via IPTables

```
 sudo iptables -P OUTPUT DROP
sudo iptables -A OUTPUT -o tun+ -j ACCEPT
sudo iptables -A INPUT -i lo -j ACCEPT
sudo iptables -A OUTPUT -o lo -j ACCEPT
sudo iptables -A OUTPUT -d 255.255.255.255 -j ACCEPT
sudo iptables -A INPUT -s 255.255.255.255 -j ACCEPT
sudo iptables -A OUTPUT -o eth+ -p udp -m multiport --dports 53,1300:1302,1194:1197 -d
141.98.255.0/24,193.138.218.0/24,45.83.220.0/24,185.213.152.0/24,185.213.154.0/24,185.65.135.0
/24,185.65.134.0/24 -j ACCEPT
sudo iptables -A OUTPUT -o eth+ -p tcp -m multiport --dports 53,443 -d
141.98.255.0/24,193.138.218.0/24,45.83.220.0/24,185.213.152.0/24,185.213.154.0/24,185.65.135.0
/24,185.65.134.0/24 -j ACCEPT
sudo iptables -A OUTPUT -o eth+ ! -d 193.138.218.74 -p tcp --dport 53 -j DROP
sudo ip6tables -P OUTPUT DROP
sudo ip6tables -A OUTPUT -o tun+ -j ACCEPT
```

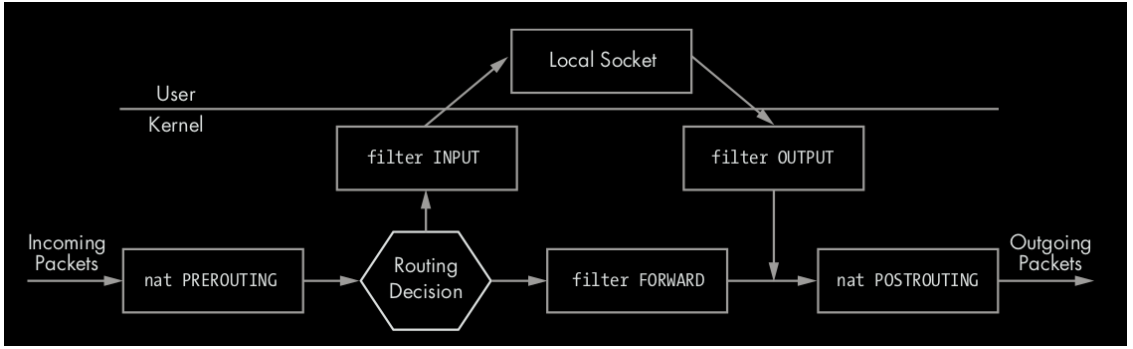
https://mullvad.net/en/help/linux-openvpn-installation/

IPTables – Load Modules

```
┌─[✗]─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #modprobe --verbose tun
insmod /lib/modules/5.8.0-2parrot1-amd64/kernel/drivers/net/tun.ko
┌─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #modinfo tun
filename:       /lib/modules/5.8.0-2parrot1-amd64/kernel/drivers/net/tun
description:    Universal TUN/TAP device driver
```

```
┌─[✗]─[root@parrot]─[/etc/openvpn]
└──╼ #sudo /usr/sbin/openvpn --mktun --dev tun0
2021-07-07 18:56:06 TUN/TAP device tun0 opened
2021-07-07 18:56:06 Persist state set to: ON
```

```
┌─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #sudo /usr/sbin/openvpn --rmtun --dev tun0
2021-07-07 18:52:11 TUN/TAP device tun0 opened
2021-07-07 18:52:11 Persist state set to: OFF
```



IPTables

```
##-=============================================-##
##    [+] Drop All Input Packets That Are in An Invalid ctstate:
##-=============================================-##
/sbin/iptables -A INPUT -m state --state INVALID -j LOG --log-prefix "DROP INVALID " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -m state --state INVALID -j DROP
/sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -A INPUT -m conntrack --ctstate NEW,RELATED,ESTABLISHED -j ACCEPT
##-=============================================-##
##    [+] Drop All Output Packets That Are in An Invalid ctstate:
##-=============================================-##
/sbin/iptables -A OUTPUT -m conntrack --ctstate INVALID -j LOG --log-prefix "Invalid ctstate: Blocked: " --log-uid
/sbin/iptables -A OUTPUT -m conntrack --ctstate INVALID -j DROP
/sbin/iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Xe1phix-IPTables-Custom-Script.sh

IPTables – Drop WebRTC

```
/sbin/iptables -A INPUT -p udp --dport 3478 -j LOG --log-prefix "DROP WebRTC"
/sbin/iptables -A INPUT -p udp --dport 3478 -j DROP
/sbin/iptables -A INPUT -p udp --dport 3479 -j LOG --log-prefix "DROP WebRTC"
/sbin/iptables -A INPUT -p udp --dport 3479 -j DROP
/sbin/iptables -A INPUT -p tcp --dport 3478 -j LOG --log-prefix "DROP WebRTC" .
/sbin/iptables -A INPUT -p tcp --dport 3478 -j DROP
/sbin/iptables -A INPUT -p tcp --dport 3479 -j LOG --log-prefix "DROP WebRTC" .
/sbin/iptables -A INPUT -p tcp --dport 3479 -j DROP
```

TCP FIN, XMAS, and NULL Scans

The FIN, XMAS, and NULL scans operate on the principle that any TCP stack (that adheres to the RFC) should respond in a particular way if a surprise TCP packet that does not set the SYN, ACK, or RST control bits is received on a port. If the port is closed, then TCP responds with a RST/ACK, but if the port is open, TCP does not respond with *any* packet at all.

NULL Scan

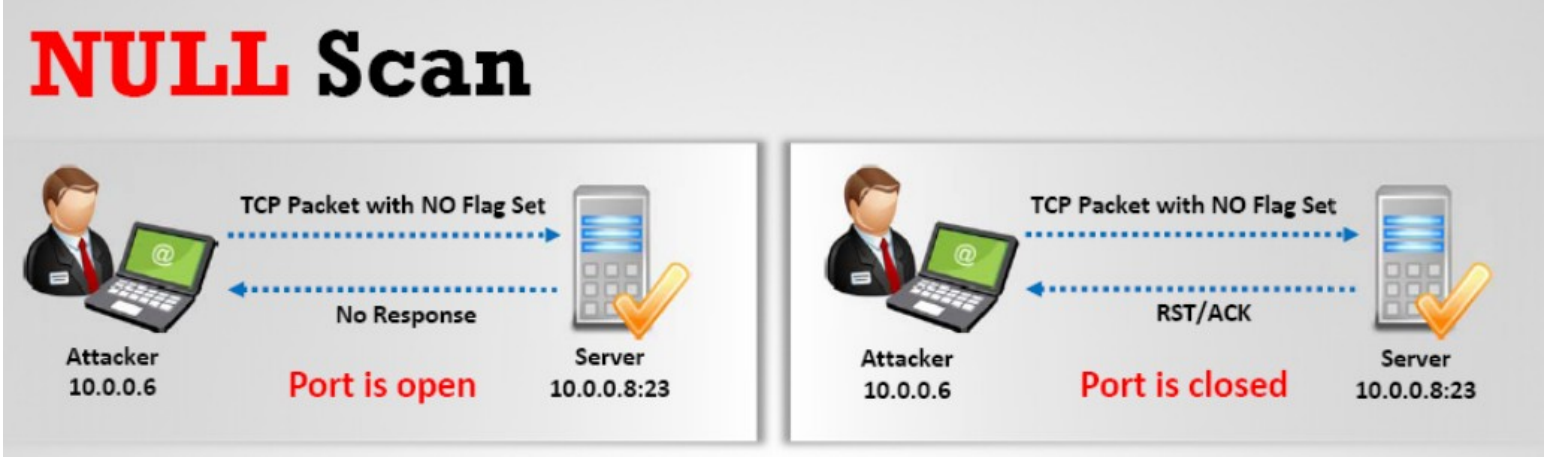
- NULL scan, attackers send a TCP frame to a remote host with NO Flags

FIN Scan

- FIN scan sends a TCP frame to a remote device with FIN flag set

Xmas Scan

- Xmas scan sends a TCP frame to a remote device with URG, ACK, RST, SYN, and FIN flags set



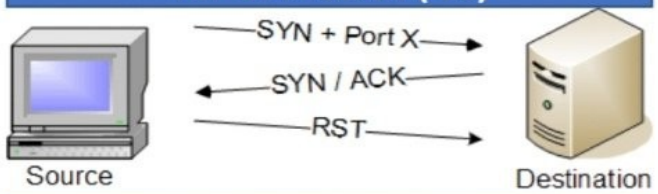

```
/sbin/iptables -A INPUT -p tcp --tcp-flags ALL NONE -j LOG --log-prefix "IPT: Null Flag " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
/sbin/iptables -A INPUT -s 255.0.0.0/8 -j LOG --log-prefix "Spoofed source IP!" --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -s 255.0.0.0/8 -j DROP
```

- Xmas scan sends a TCP frame to a remote device with URG, ACK, RST, SYN, and FIN flags set

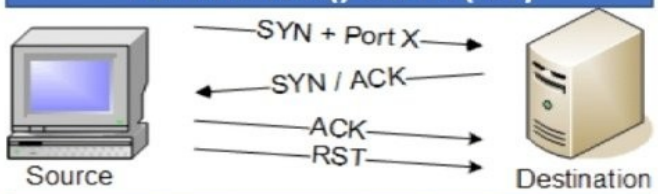
```
/sbin/iptables -N check-flags
/sbin/iptables -A check-flags -p tcp --tcp-flags ALL FIN,URG,PSH -m limit --limit 5/minute -j LOG --log-level alert --log-prefix "NMAP-XMAS:"
/sbin/iptables -A check-flags -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP
/sbin/iptables -A check-flags -p tcp --tcp-flags ALL ALL -m limit --limit 5/minute -j LOG --log-level 1 --log-prefix "XMAS:"
/sbin/iptables -A check-flags -p tcp --tcp-flags ALL ALL -j DROP
/sbin/iptables -A check-flags -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -m limit --limit 5/minute -j LOG --log-level 1 --log-prefix "XMAS-PSH:"
/sbin/iptables -A check-flags -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP
```

Identifying Open Ports with Nmap

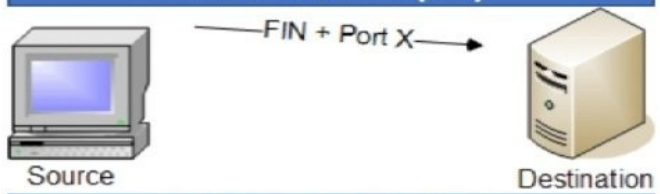
TCP SYN SCAN (-sS)



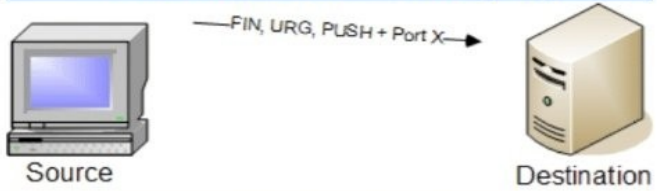
TCP connect() SCAN (-sT)



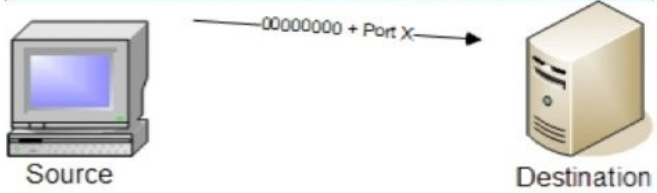
TCP FIN SCAN (-sF)



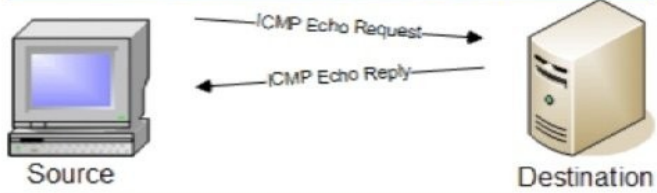
TCP XMAS TREE SCAN (-sX)



TCP NULL SCAN (-sN)

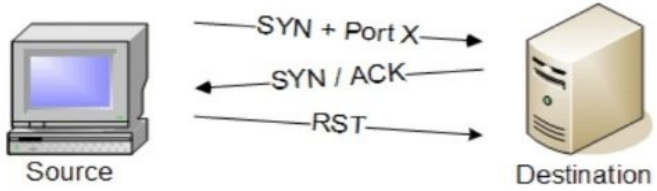


TCP PING SCAN (-sP)

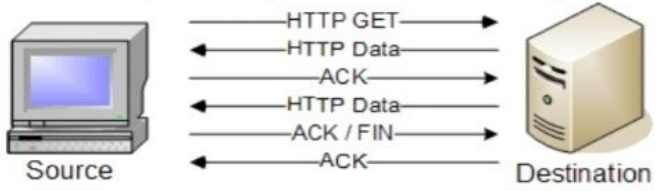


VERSION DETECTION SCAN (-sV)

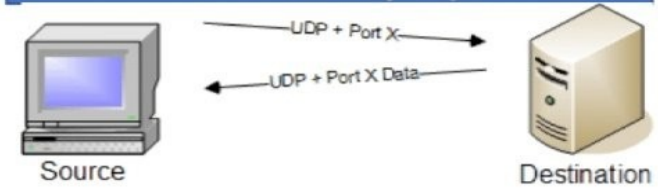
Version scan identifies open pots with a TCP SYN scan...



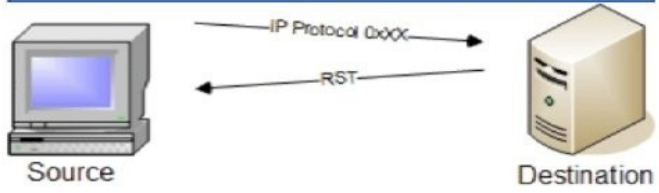
...and then queries the port with a customized signature.



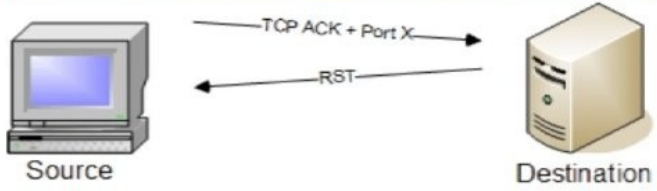
UDP SCAN (-sU)



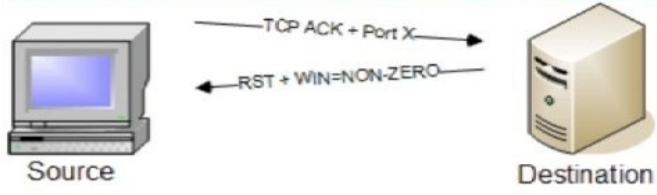
IP PROTOCOL SCAN (-sO)



TCP ACK SCAN (-sA)



TCP WINDOW SCAN (-sW)



IPTables – Blocking Port Scans

```
/sbin/iptables -A INPUT -p tcp --tcp-flags ALL ALL -j LOG --log-prefix "IPT: All Flags " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP
/sbin/iptables -A INPUT -p tcp --tcp-flags ALL ACK,RST,SYN,FIN -j LOG --log-prefix "IPTables: Bad SF Flag " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags ALL ACK,RST,SYN,FIN -j DROP
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j LOG --log-prefix "IPTables: Bad SF Flag " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j LOG --log-prefix "IPT: Bad SR Flag " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,FIN,PSH SYN,FIN,PSH -j LOG --log-prefix "IPT: Bad SFP Flag " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,FIN,PSH SYN,FIN,PSH -j DROP
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,FIN,RST SYN,FIN,RST -j LOG --log-prefix "IPT: Bad SFR Flag " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,FIN,RST SYN,FIN,RST -j DROP
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,FIN,RST,PSH SYN,FIN,RST,PSH -j LOG --log-prefix "IPT: Bad SFRP Flag " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags SYN,FIN,RST,PSH SYN,FIN,RST,PSH -j DROP
/sbin/iptables -A INPUT -p tcp --tcp-flags FIN FIN -j LOG --log-prefix "IPT: Bad F Flag " --log-ip-options --log-tcp-options
/sbin/iptables -A INPUT -p tcp --tcp-flags FIN FIN -j DROP
```

Xe1phix-IPTables-Custom-Script.sh

IP6Tables

```
##-==============================-##
##    [+] Drop + Reject all IPv6 Traffic:
##-==============================-##
/sbin/ip6tables -P INPUT DROP
/sbin/ip6tables -P OUTPUT DROP
/sbin/ip6tables -A OUTPUT -j REJECT
/sbin/ip6tables -P FORWARD DROP
/sbin/ip6tables -A FORWARD -j REJECT
##-==========================================-##
##    [+] Block All IPv6 Packets VIA IPTables As Well:
##-==========================================-##
/sbin/iptables -A INPUT -p ipv6 -j DROP
/sbin/iptables -A OUTPUT -p ipv6 -j DROP
/sbin/iptables -A FORWARD -p ipv6 -j DROP
/sbin/ip6tables -A OUTPUT -d fd00::/8 -j BLOCK
```

IPTables - OpenVPN

```
##-=======================================================-##
##     [+] Allow Packets Through The Tun/Tap Virtual VPN Interface Only:
##-=======================================================-##
/sbin/iptables -I INPUT -o tun+ -j ACCEPT
/sbin/iptables -I OUTPUT -o tun+ -j ACCEPT
/sbin/iptables -A INPUT ! -i tun+ -j DROP
/sbin/iptables -A OUTPUT ! -o tun+ -j DROP
```

```
/sbin/iptables -A INPUT -i tun+ -s 10.64.0.0/24 --dport 1080 -j ACCEPT
/sbin/iptables -A OUTPUT -i tun+ -d 10.64.0.0/24 --dport 1080 -j ACCEPT
/sbin/iptables -A INPUT -i tun+ -s 10.8.0.0/24 --dport 1080 -j ACCEPT
/sbin/iptables -A INPUT -i tun+ -s 10.8.0.0/24 --dport 1194 -j ACCEPT
/sbin/iptables -A OUTPUT  -i tun+ -d 10.8.0.0/24 --dport 1080 -j ACCEPT
/sbin/iptables -A OUTPUT  -i tun+ -d 10.8.0.0/24 --dport 1194 -j ACCEPT
/sbin/iptables -A FORWARD -i tun+ -s 10.8.0.0/24 --dport 1080 -j ACCEPT
/sbin/iptables -A FORWARD -i tun+ -s 10.8.0.0/24 --dport 1194 -j ACCEPT
```

IPTables – Force DNS to use OpenNIC or Mullvad DNS

```
iptables -A OUTPUT -o eth+ ! -d 193.138.218.74 -p tcp --dport 53 -j DROP
```

IPTables – Save + Restore

```
#iptables-save > /etc/iptables/rules.v4
#iptables-restore < /etc/iptables/rules.v4
#ip6tables-save > /etc/iptables/rules.v6
#ip6tables-restore < /etc/iptables/rules.v6
#ls /etc/iptables/rules.v*
/etc/iptables/rules.v4  /etc/iptables/rules.v6
```

Xe1phix-IPTables-Custom-Script.sh

```
#nft list ruleset
table ip filter {
        chain INPUT {
                type filter hook input priority filter; policy accept;
                meta l4proto ipv6 counter packets 0 bytes 0 drop
                ct state invalid counter packets 21352 bytes 866850 log prefix "DROP INVALID " flags tcp options flags ip options
                ct state invalid counter packets 21352 bytes 866850 drop
                ct state related,established counter packets 13709383 bytes 40277516547 accept
                meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|syn|rst|psh|ack|urg counter packets 0 bytes 0 log prefix "IPT: All Flags " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|syn|rst|psh|ack|urg counter packets 0 bytes 0 drop
                meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|syn|rst|ack counter packets 0 bytes 0 log prefix "IPTables: Bad SF Flag " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|syn|rst|ack counter packets 0 bytes 0 drop
                meta l4proto tcp tcp flags & (fin|syn) == fin|syn counter packets 0 bytes 0 log prefix "IPTables: Bad SF Flag " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (fin|syn) == fin|syn counter packets 0 bytes 0 drop
                meta l4proto tcp tcp flags & (syn|rst) == syn|rst counter packets 0 bytes 0 log prefix "IPT: Bad SR Flag " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (syn|rst) == syn|rst counter packets 0 bytes 0 drop
                meta l4proto tcp tcp flags & (fin|syn|psh) == fin|syn|psh counter packets 0 bytes 0 log prefix "IPT: Bad SFP Flag " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (fin|syn|psh) == fin|syn|psh counter packets 0 bytes 0 drop
                meta l4proto tcp tcp flags & (fin|syn|rst) == fin|syn|rst counter packets 0 bytes 0 log prefix "IPT: Bad SFR Flag " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (fin|syn|rst) == fin|syn|rst counter packets 0 bytes 0 drop
                meta l4proto tcp tcp flags & (fin|syn|rst|psh) == fin|syn|rst|psh counter packets 0 bytes 0 log prefix "IPT: Bad SFRP Flag " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (fin|syn|rst|psh) == fin|syn|rst|psh counter packets 0 bytes 0 drop
                meta l4proto tcp tcp flags & (fin) == fin counter packets 0 bytes 0 log prefix "IPT: Bad F Flag " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (fin) == fin counter packets 0 bytes 0 drop
                meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == 0x0 counter packets 0 bytes 0 log prefix "IPT: Null Flag " flags tcp options flags ip options
                meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == 0x0 counter packets 0 bytes 0 drop
                ip saddr 255.0.0.0/8 counter packets 0 bytes 0 log prefix "Spoofed source IP!" flags tcp options flags ip options
                meta l4proto udp udp dport 3478 counter packets 6 bytes 296 log prefix "DROP WebRTC" flags tcp options flags ip options flags skuid
                meta l4proto udp udp dport 3478 counter packets 6 bytes 296 drop
                meta l4proto udp udp dport 3479 counter packets 0 bytes 0 log prefix "DROP WebRTC" flags tcp options flags ip options flags skuid
                meta l4proto udp udp dport 3479 counter packets 0 bytes 0 drop
                meta l4proto tcp tcp dport 3478 counter packets 1 bytes 40 log prefix "DROP WebRTC" flags tcp options flags ip options flags skuid
                meta l4proto tcp tcp dport 3478 counter packets 1 bytes 40 drop
                meta l4proto tcp tcp dport 3479 counter packets 2 bytes 84 log prefix "DROP WebRTC" flags tcp options flags ip options flags skuid
                meta l4proto tcp tcp dport 3479 counter packets 2 bytes 84 drop
                iifname "wlan0" counter packets 0 bytes 0 accept
                ct state related,established counter packets 0 bytes 0 accept
                iifname "wlan0" meta l4proto tcp ip saddr 139.99.96.146 tcp dport 53 counter packets 0 bytes 0 accept
                iifname "wlan0" meta l4proto tcp ip saddr 37.59.40.15 tcp dport 53 counter packets 0 bytes 0 accept
                iifname "wlan0" meta l4proto tcp ip saddr 185.121.177.177 tcp dport 53 counter packets 0 bytes 0 accept
                iifname "wlan0" meta l4proto udp udp sport 68 udp dport 67 counter packets 0 bytes 0 accept
                meta l4proto tcp tcp dport { 135,137,138,139,445,1433,1434} counter packets 285 bytes 12740 log prefix "Blocked: Faggot Microsoft Ser" flags tcp options flags ip options
                meta l4proto tcp tcp dport { 135,137,138,139,445,1433,1434} counter packets 285 bytes 12740 drop
                meta l4proto udp udp dport { 135,137,138,139,445,1433,1434} counter packets 79 bytes 4951 log prefix "Blocked: Faggot Microsoft Ser" flags tcp options flags ip options
                meta l4proto udp udp dport { 135,137,138,139,445,1433,1434} counter packets 79 bytes 4951 drop
                meta l4proto icmp icmp type echo-request meta length 86-65535 counter packets 0 bytes 0 log prefix "Ping Packet Size Larger Than " level debug
                meta l4proto icmp icmp type echo-request meta length 86-65535 counter packets 0 bytes 0 drop
```

```
chain check-flags {
        meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|psh|urg limit rate 5/minute counter packets 0 bytes 0 log prefix "NMAP-XMAS:" level alert
        meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|psh|urg counter packets 0 bytes 0 drop
        meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|syn|rst|psh|ack|urg limit rate 5/minute counter packets 0 bytes 0 log prefix "XMAS:" level alert
        meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|syn|rst|psh|ack|urg counter packets 0 bytes 0 drop
        meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|syn|rst|ack|urg limit rate 5/minute counter packets 0 bytes 0 log prefix "XMAS-PSH:" level alert
        meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == fin|syn|rst|ack|urg counter packets 0 bytes 0 drop
        meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == 0x0 limit rate 5/minute counter packets 0 bytes 0 log prefix "NULL_SCAN:" level alert
        meta l4proto tcp tcp flags & (fin|syn|rst|psh|ack|urg) == 0x0 counter packets 0 bytes 0 drop
        meta l4proto tcp tcp flags & (syn|rst) == syn|rst limit rate 5/minute counter packets 0 bytes 0 log prefix "SYN/RST:" level notice
        meta l4proto tcp tcp flags & (syn|rst) == syn|rst counter packets 0 bytes 0 drop
        meta l4proto tcp tcp flags & (fin|syn) == fin|syn limit rate 5/minute counter packets 0 bytes 0 log prefix "SYN/FIN:" level notice
        meta l4proto tcp tcp flags & (fin|syn) == fin|syn counter packets 0 bytes 0 drop
}
chain OUTPUT {
        type filter hook output priority filter; policy accept;
        meta l4proto ipv6 counter packets 0 bytes 0 drop
        ct state invalid counter packets 329 bytes 22461 log prefix "Invalid ctstate: Blocked: " flags skuid
        ct state invalid counter packets 329 bytes 22461 drop
        ct state related,established counter packets 5170763 bytes 576794978 accept
        meta l4proto tcp tcp dport 80 tcp flags & (fin|syn|rst|ack) == syn ct state new counter packets 3491 bytes 209460 accept
        meta l4proto udp udp dport 53 ct state new counter packets 19990 bytes 1582710 accept
        meta l4proto tcp tcp dport 53 ct state new counter packets 224 bytes 14092 accept
        meta l4proto icmp icmp type echo-request counter packets 0 bytes 0 accept
        meta l4proto tcp tcp dport 443 tcp flags & (fin|syn|rst|ack) == syn ct state new counter packets 9765 bytes 585900 accept
        oifname "wlan0" counter packets 0 bytes 0 accept
        meta l4proto tcp ip daddr 139.99.96.146 tcp dport 53 counter packets 0 bytes 0 accept
        meta l4proto tcp ip daddr 37.59.40.15 tcp dport 53 counter packets 0 bytes 0 accept
        meta l4proto tcp ip daddr 185.121.177.177 tcp dport 53 counter packets 0 bytes 0 accept
        meta l4proto udp udp dport 53 ct state new counter packets 0 bytes 0 accept
        meta l4proto tcp tcp dport 53 ct state new counter packets 0 bytes 0 accept
}
```

IPSet – Git Clone

```
└──╼ $git clone https://github.com/firehol/blocklist-ipsets
Cloning into 'blocklist-ipsets'...
remote: Enumerating objects: 7536, done.
remote: Counting objects: 100% (1559/1559), done.
remote: Compressing objects: 100% (662/662), done.
remote: Total 7536 (delta 1044), reused 1222 (delta 897), pack-reused 5977
Receiving objects: 100% (7536/7536), 28.06 MiB | 6.54 MiB/s, done.
Resolving deltas: 100% (3919/3919), done.
```

https://github.com/firehol/blocklist-ipsets

IPSet – Enabling Lists

```
└──╼ #update-ipsets enable dshield dshield_top_1000 et_compromised et_spamhaus spamhaus_dro
p spamhaus_edrop malwaredomainlist snort_ipfilter talosintel_ipfilter firehol_level4 fireho
l_level3 firehol_level2 firehol_level1
dshield: Enabling dshield...
dshield_top_1000: Enabling dshield_top_1000...
et_compromised: Enabling et_compromised...
et_spamhaus: Enabling et_spamhaus...
spamhaus_drop: Enabling spamhaus_drop...
spamhaus_edrop: Enabling spamhaus_edrop...
malwaredomainlist: Enabling malwaredomainlist...
snort_ipfilter: Enabling snort_ipfilter...
talosintel_ipfilter: Enabling talosintel_ipfilter...
firehol_level4: Enabling firehol_level4...
firehol_level3: Enabling firehol_level3...
firehol_level2: Enabling firehol_level2...
firehol_level1: Enabling firehol_level1...
```

```
┌──[root@parrot]─[/home/parrotsec-kiosk/Downloads]
└─── #update-ipsets --verbose

Mon May 17 22:20:27 CDT 2021: /usr/sbin/update-ipsets

Getting list of active ipsets...
Found these ipsets active:
                                    |
                           dshield| parsing attributes:
                                  | 11243360/10 mins passed, downloading...
                                  | fetch: 'http://feeds.dshield.org/block.txt'
                                  | running downloader 'geturl'
                                  | curl  'http://feeds.dshield.org/block.txt'
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1125  100  1125    0     0   7450      0 --:--:-- --:--:-- --:--:--  7450
                                  | HTTP/200 OK
                                  | downloaded successfully
                                  | saving downloaded file
                                  | forced reprocessing (ignoring download status)
                                  | converting with 'dshield_parser'
                                  | parsing attributes:
                                  |  SAVED  no need to load ipset in kernel
                                  | version 1, 17 subnets, 5120 unique IPs
```

IPSet – Custom Bash For Loop

```
for IP in $(cat /etc/ipset/snort_ipfilter.restore)
    do
        ipset -A Snort $IP
    done
```

https://github.com/firehol/blocklist-ipsets

IPSet – List Blacklist IPs

```
#ipset list
Name: Snort
Type: hash:ip
Revision: 4
Header: family inet hashsize 1024 maxelem 65536
Size in memory: 28280
References: 1
Number of entries: 836
Members:
23.129.64.209
27.69.166.251
2.50.24.70
112.238.151.252
92.222.92.152
117.194.161.66
```

IPSet – Saving Rules
+ Listing IPTables Rule

```
└──  #ipset save
create Snort hash:ip family inet hashsize 1024 maxelem 65536
add Snort 23.129.64.209
add Snort 27.69.166.251
add Snort 2.50.24.70
add Snort 112.238.151.252
```

```
└──  #iptables -L
# Warning: iptables-legacy tables present, use iptables-legacy to see them
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  anywhere             anywhere             match-set Snort src
```

https://github.com/firehol/blocklist-ipsets

FWSnort – Fetch Rules

```
└─ #fwsnort --update-rules
[+] Downloading latest rules into /etc/fwsnort/snort_rules/--2021-07-22 10:42:41--  http://rules.emergingthreats.net/open/snort-2.9.0/emerging-all.rules
Resolving rules.emergingthreats.net (rules.emergingthreats.net)... 52.5.252.216, 23.21.164.163
Connecting to rules.emergingthreats.net (rules.emergingthreats.net)|52.5.252.216|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 18266286 (17M) [text/plain]
Saving to: ‘emerging-all.rules’

emerging-all.rules        100%[=====================================>]  17.42M  14.6MB/s    in 1.2s

2021-07-22 10:42:42 (14.6 MB/s) - ‘emerging-all.rules’ saved [18266286/18266286]

--2021-07-22 10:42:42--  https://rules.emergingthreats.net/fwrules/emerging-IPTABLES-ALL.rules
Resolving rules.emergingthreats.net (rules.emergingthreats.net)... 23.21.164.163, 52.5.252.216
Connecting to rules.emergingthreats.net (rules.emergingthreats.net)|23.21.164.163|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 88177 (86K) [text/plain]
Saving to: ‘emerging-IPTABLES-ALL.rules’

emerging-IPTABLES-ALL.rul 100%[=====================================>]  86.11K  --.-KB/s    in 0.04s

2021-07-22 10:42:42 (2.04 MB/s) - ‘emerging-IPTABLES-ALL.rules’ saved [88177/88177]

[+] Finished.
```

FWSnort – IPTables Script

```
┌─[root@parrotseckiosk-optiplex990]─[/var/lib/fwsnort]
└──╼ #fwsnort --ipt-list
[+] Listing FWSNORT_INPUT chain...
Chain FWSNORT_INPUT (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 RETURN     all  --  *      *       96.43.137.99         0.0.0.0/0
    0     0 RETURN     all  --  *      *       204.12.217.19        0.0.0.0/0
 1599 7390K FWSNORT_INPUT_ESTAB  tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            ctstate ESTABLISHED
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp STRING match  "|535353535343534353ffd06668|" ALGO name bm TO 65535 STRING match  "|665389e19568a41a|" ALGO name bm F
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|0b|" ALGO name bm F
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|1f00|" ALGO name bm
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|1f00|" ALGO name bm
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|1f00|" ALGO name bm
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|1f00|" ALGO name bm
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|2000|" ALGO name bm
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|2000|" ALGO name bm
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|2000|" ALGO name bm
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:139 STRING match  "|c84f324b7016d30112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|2000|" ALGO name bm
    0     0 LOG        udp  --  *      *      !192.168.1.0/24       192.168.1.0/24       udp spts:1024:65535 dpts:1024:65535 length 28:48 STRING match  "|e30d|" ALGO name bm TO 44 /* sid:2003311; msg:ET P2P Ed
    0     0 LOG        udp  --  *      *       192.168.1.0/24      !192.168.1.0/24       udp spts:1024:65535 dpts:1024:65535 length 53 STRING match  "|e30a|" ALGO name bm TO 44 /* sid:2003312; msg:ET P2P Edonk
    0     0 LOG        udp  --  *      *      !192.168.1.0/24       192.168.1.0/24       udp spt:41170 length 28:76 STRING match  "|3d4ad9|" ALGO name bm TO 45 /* sid:2009097; msg:ET P2P Manolito Connection (1
    0     0 LOG        udp  --  *      *       192.168.1.0/24      !192.168.1.0/24       udp dpt:3544 STRING match  "|fe8000000000000008000544552454 44f|" ALGO name bm FROM 63 TO 79 /* sid:2003155; msg:ET POLICY
    0     0 LOG        udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp STRING match  "UK00760S7G10" ALGO name bm TO 65535 /* sid:2001597; msg:ET POLICY Netop Remote Control Usage; classty
    0     0 LOG        udp  --  *      *       192.168.1.0/24      !192.168.1.0/24       udp dpt:69 STRING match  "|0003|" ALGO name bm TO 44 /* sid:2008117; msg:ET TFTP Outbound TFTP Data Transfer; classtype:
    0     0 LOG        udp  --  *      *       192.168.1.0/24      !192.168.1.0/24       udp dpt:69 STRING match  "|0004|" ALGO name bm TO 44 /* sid:2008118; msg:ET TFTP Outbound TFTP ACK; classtype:policy-vio
    0     0 LOG        udp  --  *      *       192.168.1.0/24      !192.168.1.0/24       udp dpt:69 STRING match  "|0005|" ALGO name bm TO 44 /* sid:2008119; msg:ET TFTP Outbound TFTP Error Message; classtype:
```

```
┌─[root@parrotseckiosk-optiplex990]─[/var/lib/fwsnort]
└──╼ #iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
FWSNORT_INPUT  all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
FWSNORT_FORWARD  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
FWSNORT_OUTPUT  all  --  anywhere             anywhere

Chain FWSNORT_FORWARD (1 references)
target     prot opt source               destination
RETURN     all  --  buygetpromo.com      anywhere
RETURN     all  --  anywhere             buygetpromo.com
RETURN     all  --  204.12.217.19        anywhere
RETURN     all  --  anywhere             204.12.217.19
FWSNORT_FORWARD_ESTAB  tcp  --  anywhere             anywhere            ctstate ESTABLISHED
LOG        udp  --  anywhere             anywhere            udp STRING match  "|535353535343534353ffd0
6668|" ALGO name bm TO 65535 STRING match  "|665389e19568a41a|" ALGO name bm FROM 55 TO 65535 /* sid:200
9285; msg:ET SHELLCODE Bindshell2 Decoder Shellcode (UDP); classtype:shellcode-detect; reference:url,doc
.emergingthreats.net/2009285; rev:2; FWS:1.6.8; */ LOG level warning ip-options prefix "[1] SID2009285 "
LOG        udp  --  anywhere             192.168.1.0/24      udp dpt:139 STRING match  "|c84f324b7016d3
0112785a47bf6ee188|" ALGO name bm TO 65535 STRING match  "|0b|" ALGO name bm FROM 44 TO 45 /* sid:200869
```

FWSnort – Save + Restore

```
    Rules added: 25798
[+] Finished.
```

```
└── #iptables-save > /etc/iptables/FWSnort.rules
```

```
┌─[root@parrotseckiosk-optiplex990]─[/var/lib/fwsnort]
└── #iptables-restore < /etc/iptables/FWSnort.rules
```

Auditd Cheatsheet

```
## ---------------------------------------------------------- ##
## See All Syscalls Made By A Specific Program:
## ---------------------------------------------------------- ##
auditctl -a always,exit -S all -F pid=$PID

## See Files Opened By A Specific User:
## ---------------------------------------------------------- ##
auditctl -a always,exit -S openat -F auid=$UID

## See Unsuccessful Openat Calls:
## ---------------------------------------------------------- ##
auditctl -a always,exit -S openat -F success=0
```

```
## ---------------------------------------------------------- ##
## Watch A File For Changes
##
## ---------------------------------------------------------- ##
auditctl -w /etc/shadow -p wa
auditctl -a always,exit -F path=/etc/shadow -F perm=wa
```

```
## ---------------------------------------------------------- ##
## Recursively Watch
## A Directory For Changes
## ---------------------------------------------------------- ##
auditctl -w /etc/ -p wa
auditctl -a always,exit -F dir=/etc/ -F perm=wa
```

Auditd – Ausearch Cheatsheet

```
## ------------------------------------------------------------------------ ##
##  [?] If  your  investigation showed a lot of failed accesses to a particular file
## ------------------------------------------------------------------------ ##
##-=============================================================-##
##   [+] Run the following report to see who is doing it:
##-=============================================================-##
ausearch --start this-week -k access -f /path-to/file --raw | aureport --user -i


## ------------------------------------------------------------------------ ##
##  [?] list showing which files are being accessed with the EPERM failure.
## ------------------------------------------------------------------------ ##
ausearch --start this-week -k access --raw | aureport --user --summary


##-=============================================================-##
##   [+] see the files that unauthorized access has been attempted
##-=============================================================-##
ausearch --start this-week -k access --raw | aureport --file --summary


##-=============================================================-##
##   [+] listing of the keys associated with rules that have been triggering
##-=============================================================-##
auditctl -a always,exit -F arch=b64 -S open -S openat -F exit=-EPERM -k access
```

Pluggable Authentication Modules Cheatsheet

/etc/pam.d/*service*

```
auth        requisite   pam_securetty.so
auth        required    pam_nologin.so
auth        required    pam_env.so
auth        required    pam_unix.so nullok
account     required    pam_unix.so
session     required    pam_unix.so
session     optional    pam_lastlog.so
password    required    pam_unix.so nullok obscure min=4 max=8
```

type	`auth`	Authentication module to verify user identity and group membership
	`account`	Authorization module to determine user's right to access a resource (other than his identity)
	`password`	Module to update an user's authentication credentials
	`session`	Module (run at end and beginning of an user session) to set up the user environment
control	`optional`	Module is not critical to the success or failure of *service*
	`sufficient`	If this module successes, and no previous module has failed, module stack processing ends successfully. If this module fails, it is non-fatal and processing of the stack continues
	`required`	If this module fails, processing of the stack continues until the end, and *service* fails
	`requisite`	If this module fails, *service* fails and control returns to the application that invoked *service*
	`include`	Include modules from another PAM service file
module	PAM module and its options, e.g.:	
	`pam_unix.so`	Standard UNIX authentication module via `/etc/passwd` and `/etc/shadow`
	`pam_nis.so`	Module for authentication via NIS
	`pam_ldap.so`	Module for authentication via LDAP
	`pam_fshadow.so`	Module for authentication against an alternative shadow passwords file
	`pam_cracklib.so`	Module for password strength policies (e.g. length, case, max n of retries)
	`pam_limits.so`	Module for system policies and system resource usage limits
	`pam_listfile.so`	Module to deny or allow the service based on an arbitrary text file

Sysctl - Kernel

```
┌─[✗]─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #sysctl -w kernel.randomize_va_space=2
kernel.randomize_va_space = 2
└──╼ #sysctl -w net.core.bpf_jit_harden=2
net.core.bpf_jit_harden = 2
┌─[root@parrotseckiosk-optiplex990]─[/home/parrotseck
└──╼ #sysctl -w net.ipv4.conf.all.log_martians=1
net.ipv4.conf.all.log_martians = 1
┌─[root@parrotseckiosk-optiplex990]─[/home/parrotseck
└──╼ #sysctl -w net.ipv4.conf.all.forwarding=0
net.ipv4.conf.all.forwarding = 0
┌─[root@parrotseckiosk-optiplex990]─[/home/parrotseck
└──╼ #sysctl -w net.ipv4.conf.all.rp_filter=1
net.ipv4.conf.all.rp_filter = 1
┌─[root@parrotseckiosk-optiplex990]─[/home/parrotseck
└──╼ #sysctl -w net.ipv4.conf.default.log_martians=1
net.ipv4.conf.default.log_martians = 1
```

Sysctl

```
##-========================================-##
##    [+] Logging Packets With Impossible Addresses
##-========================================-##
for i in /proc/sys/net/ipv4/conf/*/log_martians; do echo 1 > $i; done
┌─[root@parrot]─[/home/parrotsec-kiosk]
└──╼ #cat /proc/sys/net/ipv4/conf/all/log_martians
1
##-====================================-##
##    [+] Protect against SYN flood attacks
##-====================================-##
sysctl -w net.ipv4.tcp_syncookies=1
└──╼ #sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
##-====================================-##
##    [+] Enabling Bad Error Message Protection
##-====================================-##
sysctl -w net.ipv4.icmp_ignore_bogus_error_responses=1
└──╼ #sysctl -w net.ipv4.icmp_ignore_bogus_error_responses=1
net.ipv4.icmp_ignore_bogus_error_responses = 1
```

Xe1phix – Hardened Sysctl.conf

Xe1phix-Sysctl-[v17.8.94].conf

Brief List of IPv6 Weaknesses

- Man in the middle with spoofed ICMPv6 neighbor advertisement.
- Man in the middle with spoofed ICMPv6 router advertisement.
- Man in the middle using ICMPv6 redirect or ICMPv6 too big to implant route.
- Man in the middle to attack mobile IPv6 but requires ipsec to be disabled.
- Man in the middle with rogue DHCPv6 server.
- Traffic flooding with ICMPv6 router advertisement, neighbor advertisement, neighbor solicitation, multicast listener discovery, or smurf attack.
- Denial of Service which prevents new IPv6 attack on the network.
- Denial of Service which is related to fragmentation.
- Traffic flooding with ICMPv6 neighbor solicitation and a lot of crypto stuff to make CPU target busy.

IPv6 Attack Cheatsheet

A Complete Guide on IPv6 Attack and Defense

```
The THC IPV6 ATTACK TOOLKIT comes already with lots of effective attacking
tools:
 - parasite6: ICMPv6 neighbor solitication/advertisement spoofer, puts you
   as man-in-the-middle, same as ARP mitm (and parasite)
 - alive6: an effective alive scanng, which will detect all systems
   listening to this address
 - dnsdict6: parallized DNS IPv6 dictionary bruteforcer
 - fake_router6: announce yourself as a router on the network, with the
   highest priority
 - redir6: redirect traffic to you intelligently (man-in-the-middle) with
   a clever ICMPv6 redirect spoofer
 - toobig6: mtu decreaser with the same intelligence as redir6
 - detect-new-ip6: detect new IPv6 devices which join the network, you can
   run a script to automatically scan these systems etc.
 - dos-new-ip6: detect new IPv6 devices and tell them that their chosen IP
   collides on the network (DOS).
 - trace6: very fast traceroute6 with supports ICMP6 echo request and TCP-SYN
 - flood_router6: flood a target with random router advertisements
 - flood_advertise6: flood a target with random neighbor advertisements
 - fuzz_ip6: fuzzer for IPv6
 - implementation6: performs various implementation checks on IPv6
 - implementation6d: listen daemon for implementation6 to check behind a FW
 - fake_mld6: announce yourself in a multicast group of your choice on the n
 - fake_mld26: same but for MLDv2
 - fake_mldrouter6: fake MLD router messages
 - fake_mipv6: steal a mobile IP to yours if IPSEC is not needed for
authentication
 - fake_advertiser6: announce yourself on the network
 - smurf6: local smurfer
 - rsmurf6: remote smurfer, known to work only against linux at the moment
 - exploit6: known IPv6 vulnerabilities to test against a target
 - denial6: a collection of denial-of-service tests againsts a target
 - thcping6: sends a hand crafted ping6 packet
 - sendpees6: a tool by willdamn@gmail.com, which generates a neighbor
   solicitation requests with a lot of CGAs (crypto stuff ;-) to keep the
   CPU busy. nice.
and about 25 more tools for you to discover :-)
```

Smurf attack

Flood the target with network traffic amplification. Send ICMPv6 echo requests to 'FF02::1' with the spoofed source from the attack target.

```
$ sudo ./smurf6 eth0 TARGETIPv6ADDR
```

Router Advertisement MITM 👍

Announce yourself as a router and become the default router.

```
$ sudo ./fake_router26 eth0
```
'fake_router26 -h' have many interesting options

Neighbor Advertisement

Flood the local network with neighbor advertisements. The performance on IPv6 host neighbor tables will degrade and cause a DoS.

```
$ sudo ./flood_advertise6 eth0 TARGETIPv6ADDR
$ sudo ./na6 -i eth0 --target TARGETIPv6ADDR --dst-address ff02::1
--override -E 1:2:3:4:5:6 --loop --verbose
```

Neighbor Solicitation 👍

Flood the network with neighbor solicitations. If no target is supplied, query address will be 'ff02::1'.

```
$ sudo ./flood_solicitate6 eth0 TARGETIPv6ADDR
```

Firewall audit & Filter bypass tests

Performs various access control & bypass attempts to check implementations.

```
$ sudo ./firewall6 -H eth0 TARGETIPv6ADDR DSTPORT
```
Option '-u' for UDP

Disable an Existing Router

Impersonate the local router and send a Router Advertisement with a "Router Lifetime" small value. The victim host will remove the router from the 'default routers list'.

```
$ sudo ./ra6 -i eth0 --src-address ROUTERADDR --dst-address TARGETIPv6ADDR
--lifetime 0 --loop 1 --verbose
```

IPv6 Attack Toolkit Github

IPv6 Scans

```
└──╼ $sudo tcpdump -i eth0 -evv ip6 or proto ipv6
[sudo] password for parrotseckiosk:
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
15:35:50.240905 [redacted] (oui Unknown) > [redacted] (oui Unknown), ethertype IPv6 (0x86dd), length 118: (hlim 255, next-header ICMPv6 (58) payload length: 64)
15:35:50.243074 [redacted] (oui Unknown) > [redacted] (oui Unknown), ethertype IPv6 (0x86dd), length 86: (hlim 255, next-header ICMPv6 (58) payload length: 32)
└──╼ #tshark -i any -f 'proto ipv6'
```

Sysctl – IPv6

```
└─# sysctl -w net.ipv6.conf.default.autoconf=0
net.ipv6.conf.default.autoconf = 0
└─# sysctl -w net.ipv6.conf.all.autoconf=0
net.ipv6.conf.all.autoconf = 0
```

```
┌─[✗]─[root@parrot]─[/home/parrotsec]
└─# sysctl -w net.ipv6.conf.all.disable_ipv6=1
net.ipv6.conf.all.disable_ipv6 = 1
```

```
└─# sysctl -w net.ipv6.conf.default.disable_ipv6=1
net.ipv6.conf.default.disable_ipv6 = 1
```

IPv6 Module Parameters

Sysctl
40-ipv6.conf
sysctl.conf

```
## touch /etc/sysctl.d/40-ipv6.conf

net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.eth0.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1

net.ipv6.conf.all.forwarding = 0
net.ipv6.conf.default.forwarding = 0
net.ipv6.conf.eth0.forwarding = 0
net.ipv6.conf.lo.forwarding = 0

net.ipv6.conf.all.accept_ra = 0
net.ipv6.conf.default.accept_ra = 0
net.ipv6.conf.eth0.accept_ra = 0
net.ipv6.conf.lo.accept_ra = 0

net.ipv6.conf.all.router_solicitations = 0
net.ipv6.conf.default.router_solicitations = 0
net.ipv6.conf.eth0.router_solicitations = 0
net.ipv6.conf.lo.router_solicitations = 0

net.ipv6.conf.all.accept_dad = 0
net.ipv6.conf.default.accept_dad = 0
net.ipv6.conf.eth0.accept_dad = 0
net.ipv6.conf.lo.accept_dad = 0

net.ipv6.conf.all.dad_transmits = 0
net.ipv6.conf.default.dad_transmits = 0
net.ipv6.conf.eth0.dad_transmits = 0
net.ipv6.conf.lo.dad_transmits = 0

net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0
net.ipv6.conf.eth0.accept_redirects = 0
net.ipv6.conf.lo.accept_redirects = 0

net.ipv6.conf.all.autoconf = 0
net.ipv6.conf.default.autoconf = 0
net.ipv6.conf.eth0.autoconf = 0
net.ipv6.conf.lo.autoconf = 0
```

40-ipv6.conf

Modprobe - Blacklisting Unsafe Network Drivers

```
nonet               ## Do not probe for network devices
nonfs               ## Disables portmap/nfsmount

noipv6              ## Do not enable IPv6 networking
ipv6.disable=1      ## Disable IPv6
disable_ipv6=1      ## Disable IPv6
ipv6.autoconf=0     ## Disable auto configuring IPv6
```

IPv6 Module Parameters

```
bluetooth           ##  Bluetooth Core ver 2.22
rfcomm              ##  Bluetooth RFCOMM ver 1.11
btusb               ##  Generic Bluetooth USB driver ver 0.8
btsdio              ##  Generic Bluetooth SDIO driver ver 0.1
btintel             ##  Bluetooth support for Intel devices ver 0.1
btrtl               ##  Bluetooth support for Realtek devices ver 0.1
bt3c_cs             ##  Bluetooth driver for the 3Com Bluetooth PCMCIA card
btmrvl              ##  Marvell Bluetooth driver ver 1.0
btmrvl_sdio         ##  Marvell BT-over-SDIO driver ver 1.0
btqca               ##  Bluetooth support for Qualcomm Atheros family ver 0.1
btbcm               ##  Bluetooth support for Broadcom devices ver 0.1
bluetooth_6lowpan   ##  Bluetooth 6LoWPAN
bluecard_cs         ##  Bluetooth driver for the Anycom BlueCard (LSE039/LSE041)
hci_uart            ##  Bluetooth HCI UART driver ver 2.3
hci_vhci            ##  Bluetooth virtual HCI driver ver 1.5
bnep                ##  Bluetooth BNEP ver 1.3
cmtp                ##  Bluetooth CMTP ver 1.0
hidp                ##  Bluetooth HIDP ver 1.2
```

```
##  /lib/modprobe.d/bluetooth-blacklist.conf
blacklist bluetooth
blacklist rfcomm
blacklist btusb
blacklist btsdio
blacklist btrtl
blacklist btmrvl_sdio
blacklist btqca
blacklist btintel
blacklist btmrvl
blacklist btcoexist
blacklist btbcm
blacklist bt3c_cs
blacklist bluecard_cs
blacklist bluetooth_6lowpan
blacklist 6lowpan
blacklist bnep
blacklist cmtp
blacklist hidp
```

Modprobe - Blacklisting Apple Drivers

```
hfs             ## Apple HFS Filesystem
hfsplus         ## Extended Macintosh Filesystem

hid-apple       ## Apple Bullshit
hid_microsoft   ## Microsoft Bullshit
appletouch      ## Apple MacBook USB touchpad driver

appletalk       ## AppleTalk 0.20
apple_gmux      ## Apple Gmux Driver
applesmc        ## Apple SMC
ipheth          ## Apple iPhone USB Ethernet driver
thunderbolt_net ## Thunderbolt network driver
```

```
## /lib/modprobe.d/apple-blacklist.conf
blacklist hfs
blacklist hfsplus
blacklist hid-apple
blacklist hid_microsoft
blacklist appletouch
blacklist appletalk
blacklist apple_gmux
blacklist applesmc
blacklist ipheth
blacklist thunderbolt_net
```

Modprobe – Checking Config

```
┌─[root@parrotseckiosk-optiplex990]─[/home/parrotseckiosk/Downloads]
└──╼ #modprobe --showconfig | grep blacklist
blacklist drm
blacklist bluetooth
blacklist hfs
blacklist hfsplus
blacklist appletalk
blacklist nfs
blacklist nfsd
blacklist nfsv4
blacklist nfsv3
blacklist nfsv2
blacklist btsdio
blacklist btrtl
blacklist btmrvl_sdio
blacklist btqca
blacklist btintel
blacklist btmrvl
blacklist btcoexist
blacklist btbcm
blacklist bt3c_cs
blacklist bluecard_cs
blacklist bluetooth_6lowpan
blacklist 6lowpan
blacklist bnep
```

Kernel Self Protection Project

```
## ------------------------------------------------------------------ ##
##   [+] Enable slub/slab allocator free poisoning
## ------------------------------------------------------------------ ##
slub_debug=P

## ------------------------------------------------------------------------ ##
##   [+] Enable buddy allocator free poisoning
##       (Wipe higher-level memory allocations when freed)
## ------------------------------------------------------------------------ ##
page_poison=1

## ------------------------------------------------------------------ ##
##   [+] Wipe slab and page allocations
##       (supersedes slub_debug and page_poison)
## ------------------------------------------------------------------ ##
init_on_alloc=1
init_on_free=1
## ------------------------------------------------------------------ ##
##   [+] Disable slab merging
##       (heap overflow attacks more difficult)
## ------------------------------------------------------------------ ##
slab_nomerge

## ------------------------------------------------------------ ##
##   [+] Enable Kernel Page Table Isolation
## ------------------------------------------------------------ ##
pti=on

## ------------------------------------- ##
##   [+] Prevent against L1TF
## ------------------------------------- ##
nosmt
```

```
## ------------------------------------------- ##
##    [+] Block non-uid-0 profiling
## ------------------------------------------- ##
kernel.perf_event_paranoid = 3

## ------------------------------------------------------------ ##
##    [+] Turn off kexec, even if it's built in.
## ------------------------------------------------------------ ##
kernel.kexec_load_disabled = 1

## ------------------------------------------------------------------------ ##
##    [+] Avoid non-ancestor ptrace access
##        from running processes and their credentials.
## ------------------------------------------------------------------------ ##
kernel.yama.ptrace_scope = 1

## ------------------------------------------------------------ ##
##    [+] Turn off unprivileged eBPF access
## ------------------------------------------------------------ ##
kernel.unprivileged_bpf_disabled = 1

## ------------------------------------------- ##
##    [+] Turn on BPF JIT hardening
## ------------------------------------------- ##
net.core.bpf_jit_harden = 2
```

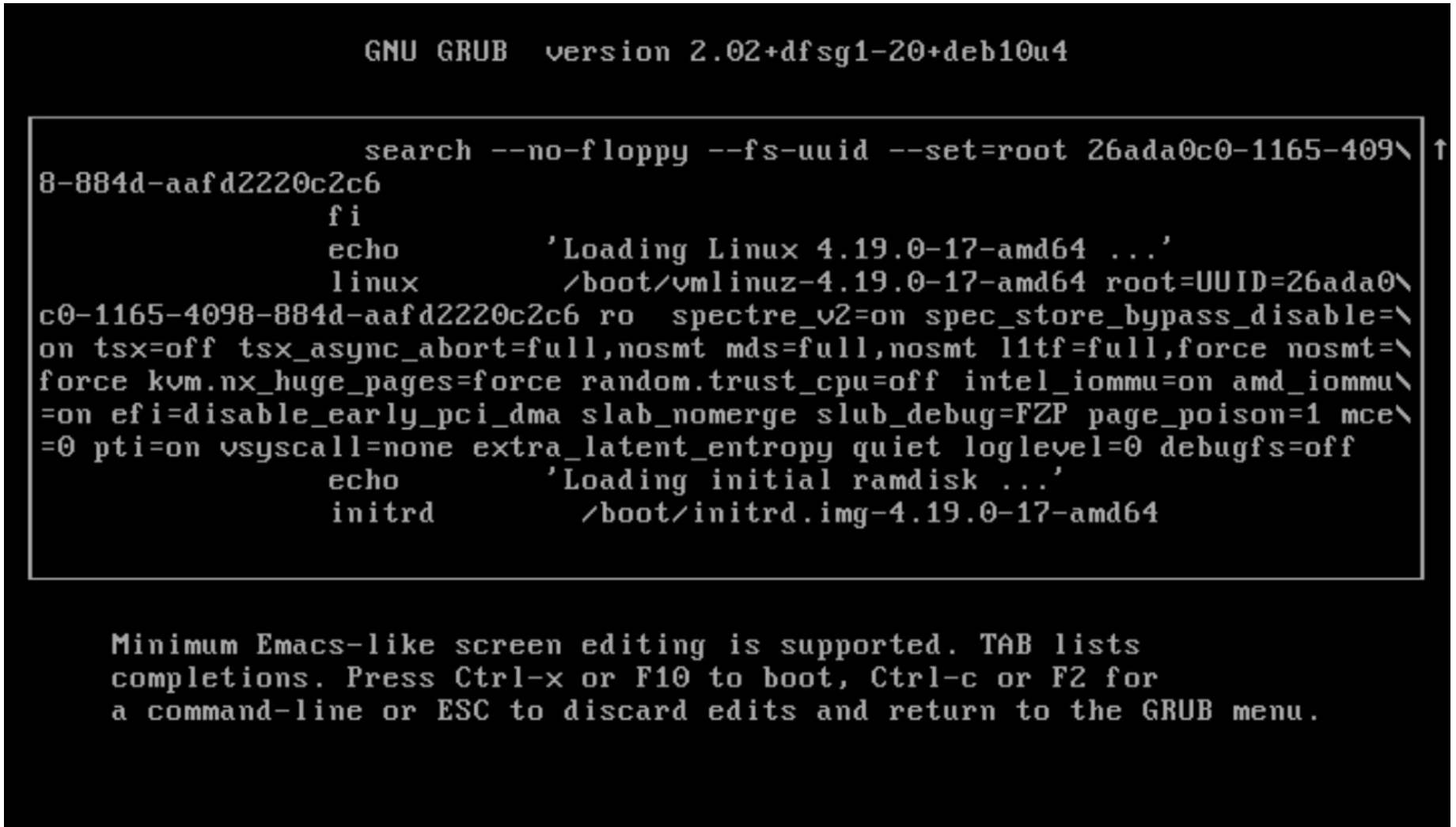
[Kernel Self Protection Project - Grub Boot Commands]

[Kernel Self Protection Project - Sysctl Settings]

Xe1phix-Kernel-Boot-Arguements

```
GRUB_DEFAULT=0
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX="debug security=apparmor apparmor=1 randomize_kstack_offset=on rodata=on kalsr pti=on
slab_nomerge init_on_alloc=1 init_on_free=1 nosmt page_poison=1 noipv6 disable_ipv6=1 ipv6.disable=1 ipv6.autoconf=0 acl
user_xattr noconfig=sudo noefi efi=noruntime rfkill.default_state=0 rfkill.master_switch_mode=2 noautologin lp=0 lang=US
xkeyboard=US keyboard-layout=en uid=1000 nosuid noautomount edd=off nonfs vga=normal xforcevesa
modprobe.blacklist=drm,bluetooth,btusb,btrtl,btsdio,btqca,btbcm,bluetooth_6lowpan,btrsi,rfcomm,cmtp,bluecard_cs,bfusb,
bt3c_cs,bnep,hci_uart,hci_vhci,nfc,nfc_digital,cifs,scsi_transport_iscsi,scsi_transport_sas,libiscsi,libiscsi_tcp,
iscsi_target_mod,nfs,nfsv2,nfsv3,nfsv4,nfsd,nfs_layout_flexfiles,nfs_layout_nfsv41_files,blocklayoutdriver,
usbip_core,usbip_host,usbip_vudc,usbnet,usb_wwan,vhci_hcd,hfs,hfsplus,appletalk,appledisplay,apple_bl,apple-
gmux,applesmc,efivars,efivarfs,efi_pstore,hid_microsoft,hid-apple
systemd.mask=ModemManager,mysql,postgresql,apache2,mysql,lighttpd,postfix,iscsi,iscsid,rwhod,sshd,ssh,beef-
xss,mysqld,printer,rpcbind,smbd,snmpd,nmbd,sendmail,samba-ad-
dc,mariadb,nginx,geoclue,ipsec,strongswan,cups,exim4,httpd,nfs-server,redis-server,pppd-dns,mountnfs,freeradius,cups-
browsed,openvpn-server"
```

Whonix – Kernel Arguments



Matrix

Element

Federated P2P VoIP

Element is the reference client for the Matrix protocol, an open standard for secure decentralized real-time communication.

Briar

P2P Anonymous Routing

Briar is an encrypted instant messenger that connects to other clients using the Tor Network. Briar can also connect via Wi-Fi or Bluetooth when in local proximity. Briar's local mesh mode can be useful when internet availability is a problem.

How it works

Briar is a messaging app designed for activists, journalists, and anyone else who needs a safe, easy and robust way to communicate. Unlike traditional messaging apps, Briar doesn't rely on a central server - messages are synchronized directly between the users' devices. If the internet's down, Briar can sync via Bluetooth or Wi-Fi, keeping the information flowing in a crisis. If the internet's up, Briar can sync via the Tor network, protecting users and their relationships from surveillance.

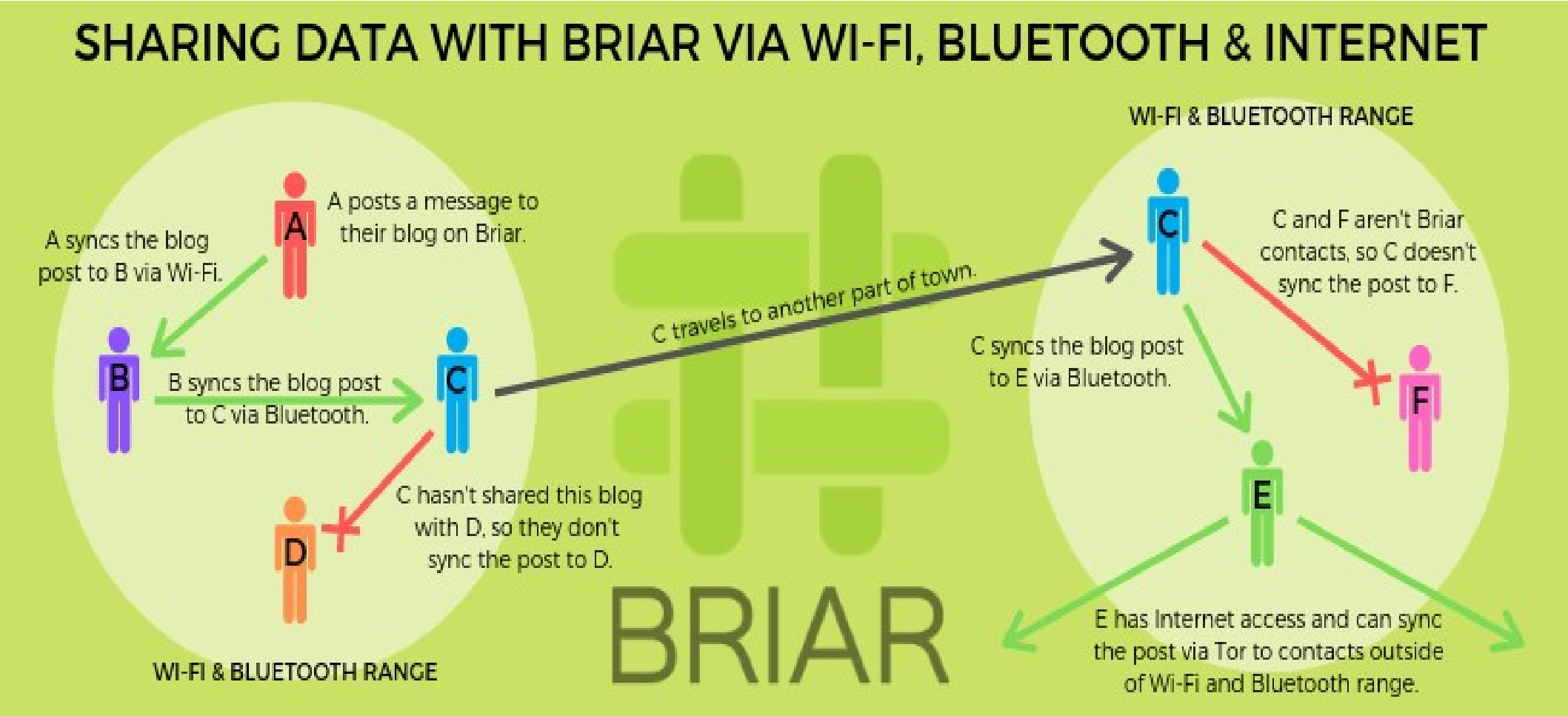
https://briarproject.org/how-it-works/

Briar Messenger – Threat Model

- **Metadata surveillance.** Briar uses the Tor network to prevent eavesdroppers from learning which users are talking to each other. Each user's contact list is encrypted and stored on her own device.
- **Content surveillance.** All communication between devices is encrypted end-to-end, protecting the content from eavesdropping or tampering.
- **Content filtering.** Briar's end-to-end encryption prevents keyword filtering, and because of its decentralized design there are no servers to block.
- **Takedown orders.** Every user who subscribes to a forum keeps a copy of its content, so there's no single point where a post can be deleted.
- **Denial of service attacks.** Briar's forums have no central server to attack, and every subscriber has access to the content even if they're offline.
- **Internet blackouts.** Briar can operate over Bluetooth and Wi-Fi to keep information flowing during blackouts.

https://briarproject.org/how-it-works/

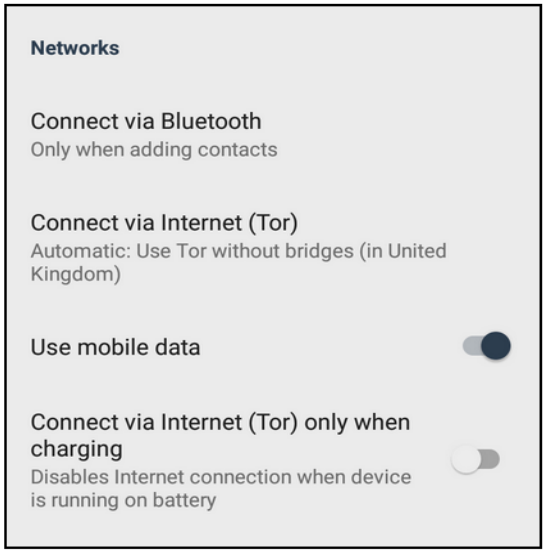
Briar – Bluetooth & WiFi connections



https://briarproject.org/how-it-works/

Connect via Internet (Tor)

> ***Tip:*** *Briar uses Tor to connect to the internet. Tor is a network of computers run by volunteers around the world to help people access the internet privately and without censorship. “Bridges” are computers that can help you connect to Tor if your government or internet provider is blocking it.*



You can control how Briar connects to the internet:

- **Automatic based on location:** Briar will choose how to connect based on your current location.
- **Use Tor without bridges:** Briar will connect to Tor without using bridges.
- **Use Tor with bridges:** Briar will use bridges to connect to Tor.
- **Don’t connect:** Briar won’t connect to the internet at all.

Hardened Thunderbird User.js

```
## ------------------------------------------------------------------ ##
##    [+] Backup the original Thunderbird settings:
## ------------------------------------------------------------------ ##
└─── #cp -v /etc/thunderbird/pref/thunderbird.js /etc/thunderbird/pref/thunderbird.js.bak
'/etc/thunderbird/pref/thunderbird.js' -> '/etc/thunderbird/pref/thunderbird.js.bak'
```

```
## ------------------------------------------------------------------ ##
##    [+] Copy Hardened Thunderbird settings:
## ------------------------------------------------------------------ ##
└─── $cp -v Thunderbird-user.js /home/parrotseckiosk/.thunderbird/zuruwlb5.default-defau
'Thunderbird-user.js' -> '/home/parrotseckiosk/.thunderbird/zuruwlb5.default-default/use
```

```
## ------------------------------------------------------------------ ##
##    [+] Backup the original Thunderbird settings:
## ------------------------------------------------------------------ ##
└─── #cp -v /etc/thunderbird/pref/thunderbird.js /etc/thunderbird/pref/thunderbird.js.bak
'/etc/thunderbird/pref/thunderbird.js' -> '/etc/thunderbird/pref/thunderbird.js.bak'
```

```
## ------------------------------------------------------------------ ##
##    [+] Copy Hardened Thunderbird settings:
## ------------------------------------------------------------------ ##
└─── #cp -v Thunderbird-user.js /etc/thunderbird/pref/thunderbird.js
'Thunderbird-user.js' -> '/etc/thunderbird/pref/thunderbird.js'
```

Hardened Thunderbird Settings

```
user_pref("privacy.firstparty.isolate.restrict_opener_access", true);
user_pref("privacy.resistFingerprinting", true);
user_pref("privacy.sanitize.sanitizeOnShutdown", true);
user_pref("privacy.trackingprotection.enabled", true);
user_pref("privacy.trackingprotection.pbmode.enabled", true);
user_pref("privacy.trackingprotection.ui.enabled", true);
user_pref("security.OCSP.GET.enabled", true);
user_pref("security.OCSP.enabled", 1);
user_pref("security.OCSP.require", true);
user_pref("security.ssl.require_safe_negotiation", true);
user_pref("security.ssl.treat_unsafe_negotiation_as_broken", true);
user_pref("security.tls.version.fallback-limit", 3);
user_pref("security.tls.version.min", 3);
user_pref("services.sync.enabled", false);
```

Thunderbird – Tor SOCKS5 Proxy

```
// Use a manual proxy configuration.
pref("network.proxy.type", 1);

// Configure Thunderbird to use the SOCKS5 proxy.
pref("network.proxy.socks", "127.0.0.1");
pref("network.proxy.socks_port", 9061);
pref("network.proxy.socks_version", 5);

// Set DNS proxying through SOCKS5.
pref("network.proxy.socks_remote_dns", true);

// Disable DNS prefetching.
pref("network.dns.disablePrefetch", true);
```

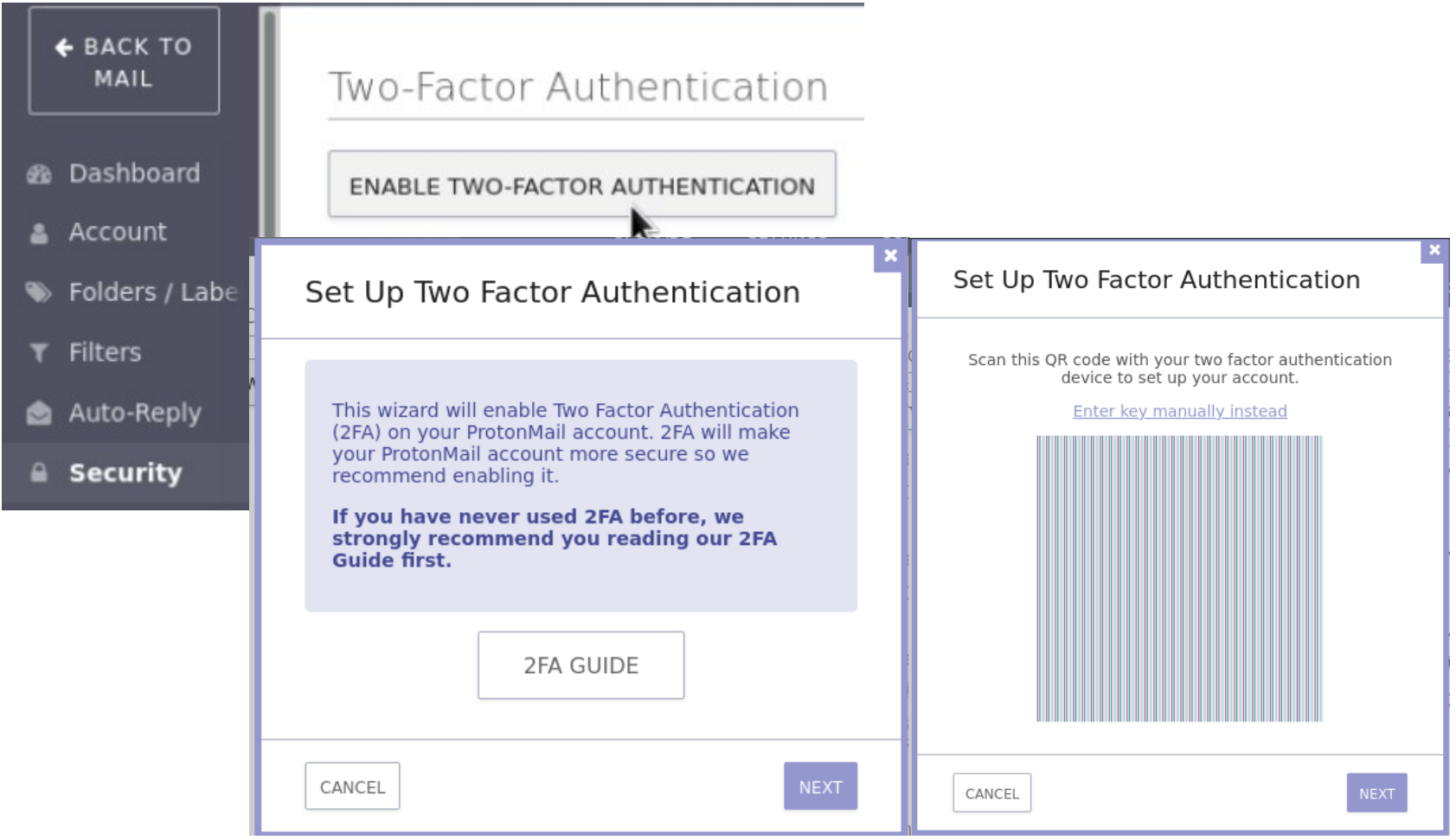
```
@{HOME}/.thunderbird/ rw,
@{HOME}/.thunderbird/** rwk,
@{HOME}/.cache/thunderbird/ rw,
@{HOME}/.cache/thunderbird/** rwk,
@{HOME}/.config/dconf/user r,
@{HOME}/.local/share/fonts/ r,
@{HOME}/.local/share/fonts/** r,

/etc/thunderbird/** r,
/etc/xul-ext/** r,
/etc/mailcap r,
/etc/mime.types r,
/usr/share/mime/ r,
/usr/share/mime/** r,
/usr/lib/thunderbird/** r,
/usr/lib/thunderbird-addons/** rk,
/usr/bin/gpg rix,
/usr/lib/gnupg/gpgkeys_hkp rix,
@{HOME}/.gnupg/ rw,
@{HOME}/.gnupg/** rwl,
@{HOME}/.gnupg/random_seed k,
owner /{,var/}run/user/*/keyring-** rw,
```

Thunderbird AppArmor Profile

Thunderbird – Firejail Profile

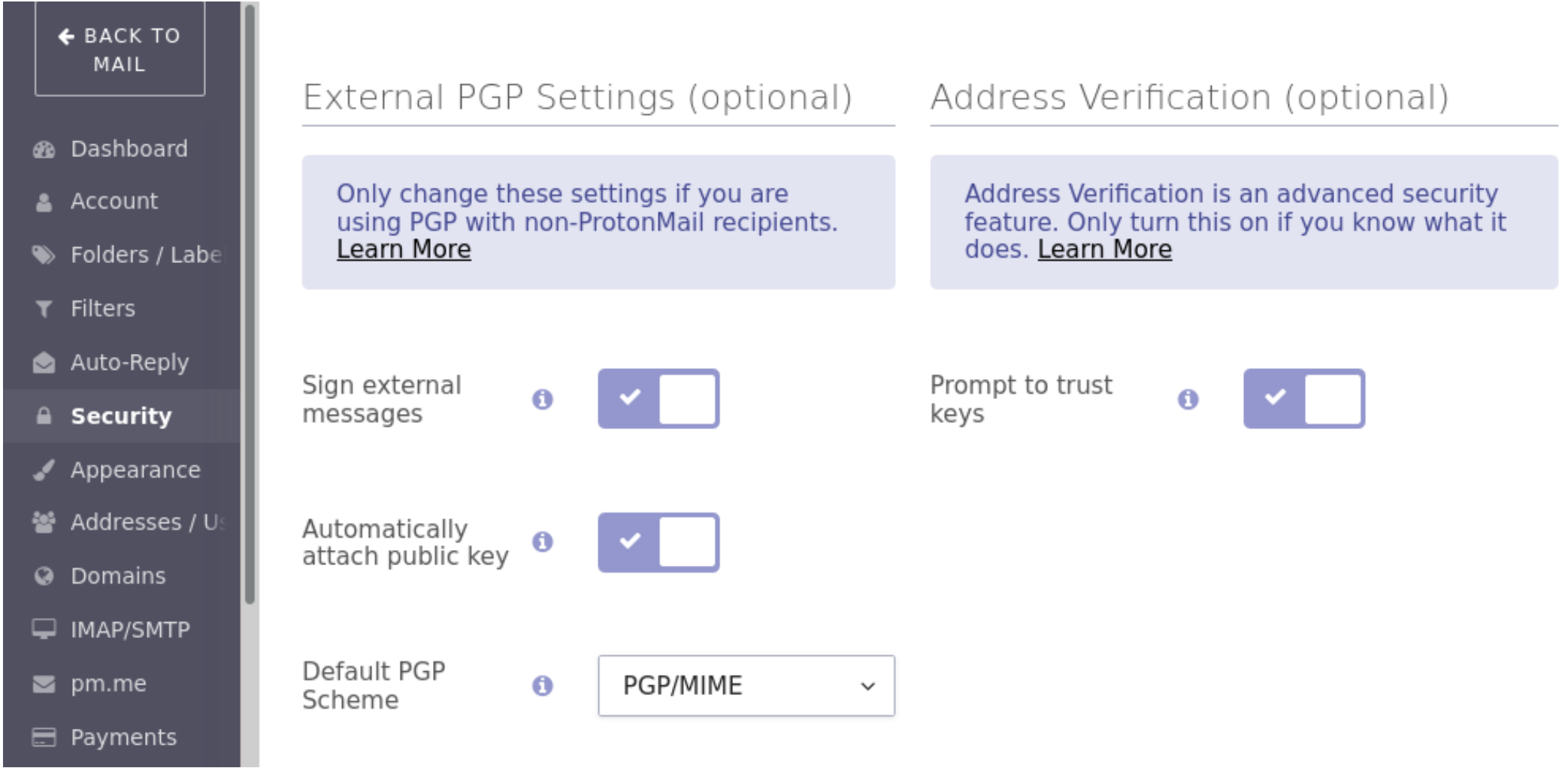
```
mkdir ${HOME}/.cache/thunderbird
mkdir ${HOME}/.gnupg
# mkdir ${HOME}/.icedove
mkdir ${HOME}/.thunderbird
whitelist ${HOME}/.cache/thunderbird
whitelist ${HOME}/.gnupg
# whitelist ${HOME}/.icedove
whitelist ${HOME}/.thunderbird

whitelist /usr/share/gnupg
whitelist /usr/share/mozilla
whitelist /usr/share/thunderbird
whitelist /usr/share/webext
include whitelist-usr-share-common.inc

# Redirect
include firefox-common.profile
```

ProtonMail - Two-Factor Authentication



- https://protonmail.com/support/knowledge-base/two-factor-authentication/

ProtonMail - PGP Settings + Address Verification

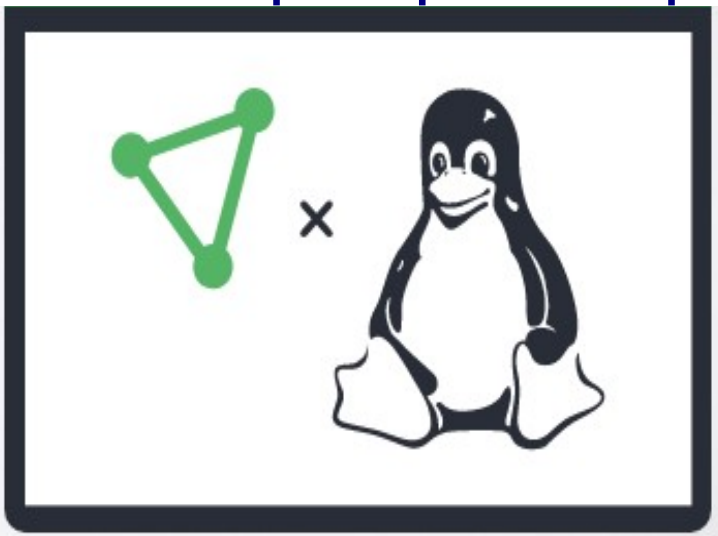


ProtonVPN

- ProtonVPN
  - https://www.privacytools.io/providers/vpn/#protonvpn
  - https://protonvpn.com

ProtonVPN's Linux

Graphical User Interface



  - https://github.com/ProtonVPN
    - https://github.com/ProtonVPN/protonvpn-cli
    - https://github.com/ProtonVPN/linux-cli-community

ProtonVPN

Strong Encryption

We use only **the highest strength encryption** to protect your Internet connection. This means all your network traffic is encrypted with AES-256, key exchange is done with 4096-bit RSA, and HMAC with SHA384 is used for message authentication.

Strong Protocols

We use only **VPN protocols which are known to be secure** - IKEv2/IPSec and OpenVPN. ProtonVPN does not have any servers that support PPTP and L2TP/IPSec, even though they are less costly to operate. By using ProtonVPN, you can be confident that your VPN tunnel is protected by the most reliable protocol.

No Logs Policy

Under Swiss law, we are **not obligated to save any user connection logs**, nor can we be forced to perform targeted logging on specific users. This allows us to ensure that your private browsing history does, in fact, stay private and cannot be turned over to a third-party under any circumstances. Our no logs policy applies to all our users, including anyone using our free VPN.

Features

DNS Management

DNS Leak Protection

ProtonVPN-CLI features a DNS Leak Protection feature, which makes sure that your online traffic uses ProtonVPN's DNS Servers. This prevents third parties (like your ISP) from being able to see your DNS queries (and, therefore, your browsing history).

ProtonVPN-CLI accomplishes this by updating the `/etc/resolv.conf` file when you connect to a VPN server, and makes sure that only ProtonVPN's DNS Server is written in this file. It will also backup the previous state of `/etc/resolv.conf` to revert all changes upon disconnection.

Please note that if you change your network (e.g., if you connect to a different WiFi hotspot) without first disconnecting, `/etc/resolv.conf` will likely be updated, which would remove ProtonVPN's DNS Servers. This could cause DNS leaks, so to keep your data safe, use `protonvpn reconnect` after changing your network.

Custom DNS

You can also make a custom DNS server your default for all your ProtonVPN connections. ProtonVPN-CLI lets you add up to 3 custom DNS Servers.

Enabling Custom DNS

To configure custom DNS Servers, use the `protonvpn configure` command, then press `4` to choose DNS Management. Then press `2` to choose that you want to configure a custom DNS Server. Now enter the IP addresses of up to 3 DNS Servers you want to use and confirm with Enter.

Disabling DNS Management

If you don't want ProtonVPN-CLI to do any changes to your DNS, you can do this as well. This will cause ProtonVPN-CLI to not touch `/etc/resolv.conf` and your device will always use the DNS servers configured by you or through your network.

Disabling any DNS management

To enable DNS Leak Protection use the `protonvpn configure` command, then pre choose DNS Management. Then press `3` to disable any DNS management.

IPv6 Leak Protection

ProtonVPN-CLI features an IPv6 Leak Protection feature. It makes sure that your IPv6 address is not leaked when you connect to a ProtonVPN server.

This feature is enabled by default, and for security reasons, it can't be disabled.

It works by detecting the IPv6 address, backing it up, and removing it from the default interface. When disconnecting, it adds the IPv6 address back to the default interface and deletes the backup.

Kill Switch and Always-on VPN

ProtonVPN applications offer a **built-in Kill Switch feature** or the **Always-on VPN feature**. In the event that you lose connection with the VPN server, Kill Switch blocks all network traffic, while Always-on automatically re-establishes a connection to a VPN server. These features prevent a VPN server disconnect from inadvertently compromising your privacy by revealing your true IP address.

Open Source

Founded by MIT and CERN scientists, ProtonVPN believes in transparency and peer review. Our apps are 100% open source, so anyone can examine our code. This transparency means that you can have confidence that our apps are doing what they are supposed to be doing, and only what they are supposed to be doing. You can see the code for all our apps on GitHub. Learn More

DNS Leak Prevention

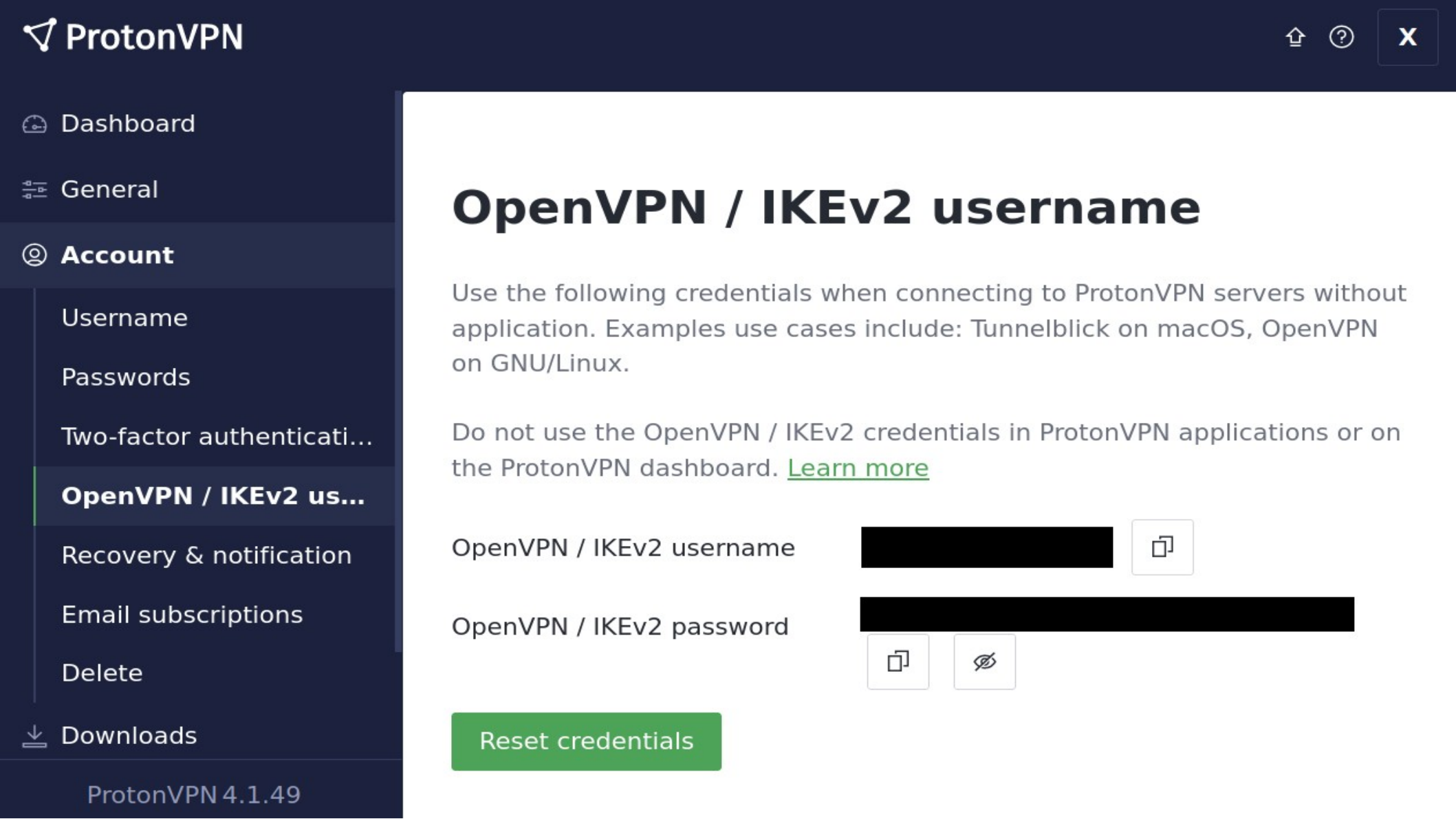
ProtonVPN doesn't just protect your browsing traffic, **we also protect your DNS queries**. By routing your DNS queries through the encrypted tunnel and not relying on third-party DNS providers, we ensure that your browsing activity cannot be exposed by leaks from DNS queries.

```
echo "## ------------------------------------------------------------------------------ ##"
echo "##     [+]  Downloading The ProtonVPN Code Signing Key:"
echo "##\_____________________________________________________________________________/##"
curl --verbose --progress-bar --tlsv1.2 --ssl-reqd --url https://repo.protonvpn.com/debian/public_key.asc --output ~/ProtonVPN-Public-Key.asc
echo "## ------------------------------------------------------------------------------ ##"
echo "##     [+]  Importing The ProtonVPN Code Signing Key:"
echo "##\_____________________________________________________________________________/##"
gpg --keyid-format 0xlong --import public_key.asc
echo "## ------------------------------------------------------------------------------ ##"
echo "##     [+]  Fingerprinting The ProtonVPN Code Signing Key..."
echo "##\_____________________________________________________________________________/##"
gpg --keyid-format 0xlong --fingerprint 0x0x71EB474019940E11
gpg --keyid-format 0xlong --fingerprint 0xA88441BD4864F95BEE08E63A71EB474019940E11
echo "## ------------------------------------------------------------------------------ ##"
echo "##                  [+]  ProtonVPN GPG Fingerprints (Verified):                   "
echo "##\_____________________________________________________________________________/##"
echo
echo "## ------------------------------------------------------------------------------ ##"
echo "      Key fingerprint = A884 41BD 4864 F95B EE08  E63A 71EB 4740 1994 0E11       "
echo "## ------------------------------------------------------------------------------ ##"
echo "            A884 41BD 4864 F95B EE08  E63A 71EB 4740 1994 0E11                  "
echo "            A884 41BD 4864 F95B EE08  E63A 71EB 4740 1994 0E11                  "
echo "## ------------------------------------------------------------------------------ ##"
echo "       [?]  See https://protonvpn.com/support/official-linux-client-arch/"
echo "## ------------------------------------------------------------------------------ ##"
echo "## ------------------------------------------------------------------------------ ##"
echo "##     [+]  Signing The ProtonVPN Code Signing Key..."
echo "##\_____________________________________________________________________________/##"
gpg --lsign 0xA88441BD4864F95BEE08E63A71EB474019940E11
```

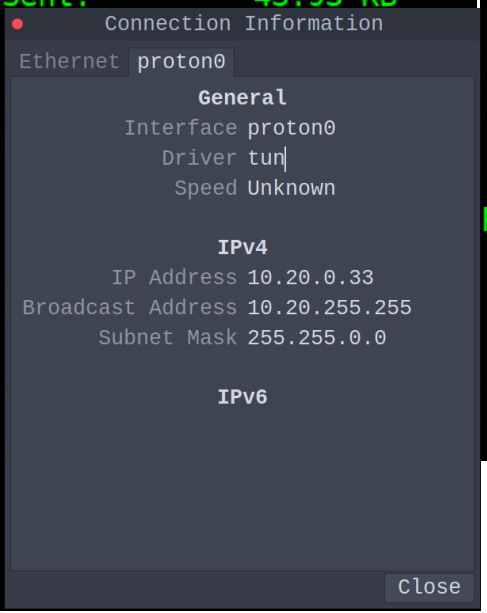
```
└── $curl --verbose --progress-bar --tlsv1.3 --ssl-reqd --url https://repo.protonvpn.com/debian/public_key.asc
*   Trying 104.26.9.21:443...
* Connected to repo.protonvpn.com (104.26.9.21) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*  CAfile: /etc/ssl/certs/ca-certificates.crt
*  CApath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server accepted to use h2
* Server certificate:
*  subject: C=US; ST=California; L=San Francisco; O=Cloudflare, Inc.; CN=sni.cloudflaressl.com
*  start date: Jul  8 00:00:00 2021 GMT
*  expire date: Jul  7 23:59:59 2022 GMT
*  subjectAltName: host "repo.protonvpn.com" matched cert's "*.protonvpn.com"
*  issuer: C=US; O=Cloudflare, Inc.; CN=Cloudflare Inc ECC CA-3
*  SSL certificate verify ok.
* Using HTTP2, server supports multi-use
* Connection state changed (HTTP/2 confirmed)
* Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
* Using Stream ID: 1 (easy handle 0x5634538a96d0)
> GET /debian/public_key.asc HTTP/2
> Host: repo.protonvpn.com
```

```
┌─[root@parrotseckiosk-optiplex990]─[/home/parrotseckiosk/Downloads]
└──╼ #protonvpn init
```

```
                                    [ -- PROTONVPN-CLI INIT -- ]

ProtonVPN uses two different sets of credentials, one for the website and official apps where the username is most
likely your e-mail, and one for connecting to the VPN servers.

You can find the OpenVPN credentials at https://account.protonvpn.com/account.

--- Please make sure to use the OpenVPN credentials ---

Enter your ProtonVPN OpenVPN username:
Enter your ProtonVPN OpenVPN password:
Confirm your ProtonVPN OpenVPN password:

Please choose your ProtonVPN Plan
1) Free
2) Basic
3) Plus
4) Visionary
Your plan: 1

Choose the default OpenVPN protocol.
OpenVPN can act on two different protocols: UDP and TCP.
UDP is preferred for speed but might be blocked in some networks.
TCP is not as fast but a lot harder to block.
Input your preferred protocol. (Default: UDP)

1) UDP
2) TCP

Your choice: 1

You entered the following information:
Username:
Password:
Tier: Free
Default protocol: UDP

Is this information correct? [Y/n]: Y
Writing configuration to disk...

Done! Your account has been successfully initialized.
```


ProtonVPN
X
Dashboard
General
Account
Username
Passwords
Two-factor authenticati...
OpenVPN / IKEv2 us...
Recovery & notification
Email subscriptions
Delete
Downloads
ProtonVPN 4.1.49
OpenVPN / IKEv2 username
Use the following credentials when connecting to ProtonVPN servers without application. Examples use cases include: Tunnelblick on macOS, OpenVPN on GNU/Linux.
Do not use the OpenVPN / IKEv2 credentials in ProtonVPN applications or on the ProtonVPN dashboard. Learn more
OpenVPN / IKEv2 username
OpenVPN / IKEv2 password
Reset credentials

```
#protonvpn status
Status:         Connected
Time:           0:00:11
IP:             217.23.3.92
Server:         NL-FREE#1
Features:       Normal
Protocol:       UDP
Kill Switch:    Disabled
Country:        Netherlands
City:           None
Load:           94%
Received:       53.42 KB
Sent:           43.93 KB
```




```
#ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet [redacted]  netmask 255.255.248.0  broadcast [redacted]
        ether [redacted]  txqueuelen 1000  (Ethernet)
        RX packets 26601869  bytes 30559580029 (28.4 GiB)
        RX errors 0  dropped 302  overruns 0  frame 0
        TX packets 4574340  bytes 786584361 (750.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xe1500000-e1520000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4817  bytes 648231 (633.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4817  bytes 648231 (633.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

proton0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1500
        inet 10.20.0.33  netmask 255.255.0.0  destination 10.20.0.33
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 500  (UNSPEC)
        RX packets 1821  bytes 592142 (578.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3203  bytes 348758 (340.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

https://github.com/ProtonVPN/linux-cli-community/blob/master/USAGE.md

ProtonVPN – DNS Leak Protection

```
└──╼ #protonvpn configure
What do you want to change?

1) Username and Password
2) ProtonVPN Plan
3) Default Protocol
4) DNS Management
5) Kill Switch
6) Split Tunneling
7) Purge Configuration

Please enter your choice or leave empty to quit: 4

DNS Leak Protection makes sure that you always use ProtonVPN's DNS servers.
For security reasons this option is recommended.

1) Enable DNS Leak Protection (recommended)
2) Configure Custom DNS Servers
3) Disable DNS Management

Please enter your choice or leave empty to quit: 1
```

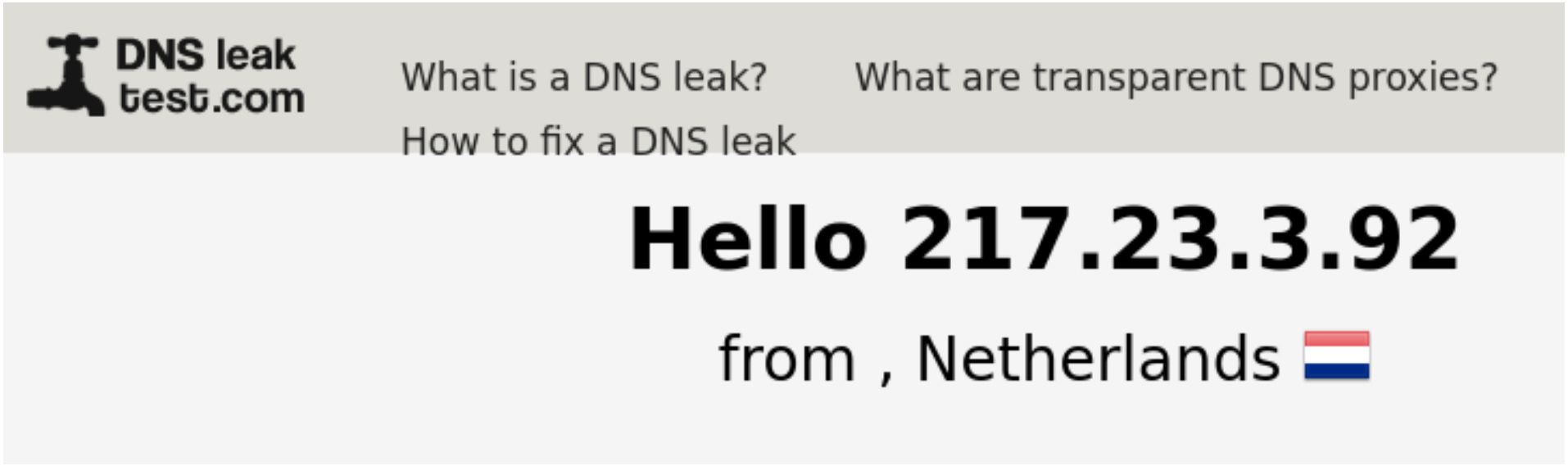
https://github.com/ProtonVPN/linux-cli-community/blob/master/USAGE.md

ProtonVPN – Enable Kill Switch

```
#protonvpn configure
What do you want to change?

1) Username and Password
2) ProtonVPN Plan
3) Default Protocol
4) DNS Management
5) Kill Switch
6) Split Tunneling
7) Purge Configuration

Please enter your choice or leave empty to quit: 5

The Kill Switch will block all network traffic
if the VPN connection drops unexpectedly.

Please note that the Kill Switch assumes only one network interface being active.

1) Enable Kill Switch (Block access to/from LAN)
2) Enable Kill Switch (Allow access to/from LAN)
3) Disable Kill Switch

Please enter your choice or leave empty to quit: 1

Kill Switch configuration updated.
```

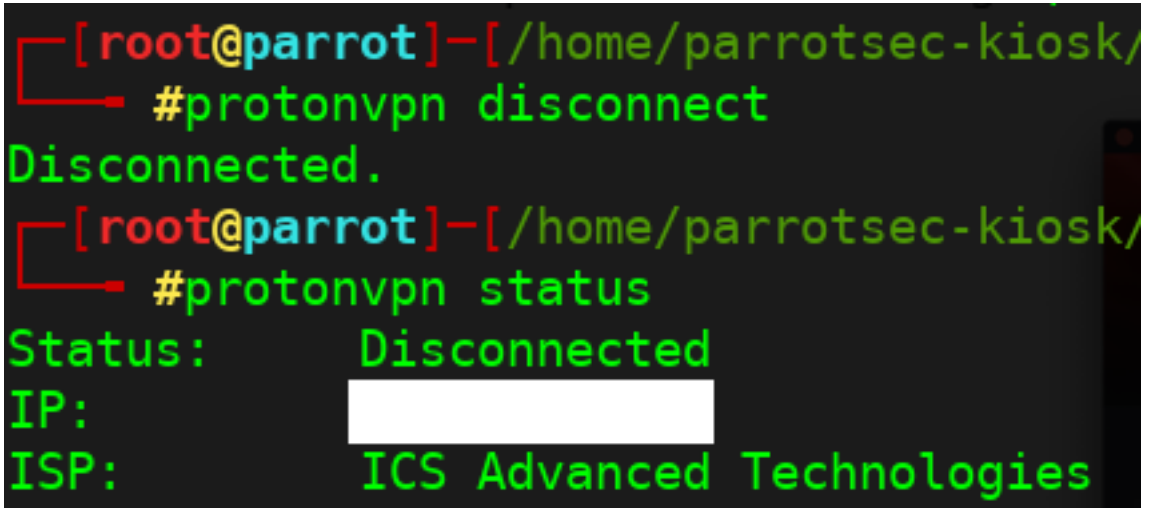
https://github.com/ProtonVPN/linux-cli-community/blob/master/USAGE.md

ProtonVPN – DNSLeakTest



https://github.com/ProtonVPN/linux-cli-community/blob/master/USAGE.md

ProtonVPN - Disconnect



https://github.com/ProtonVPN/linux-cli-community/blob/master/USAGE.md

What is I2P?

2.2 Tunnels

Every tunnel is unidirectional, and is formed by the gateway (entry point), a set of participants (intermediate nodes) and an endpoint (exit point). Two types of tunnels exist. *Inbound* tunnels allow a user to receive data, and *outbound* tunnels to send data. A fully bidirectional communication between two users will involve four tunnels, one inbound and one outbound for each user.

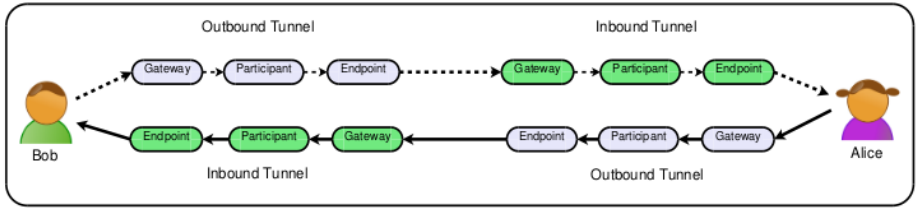
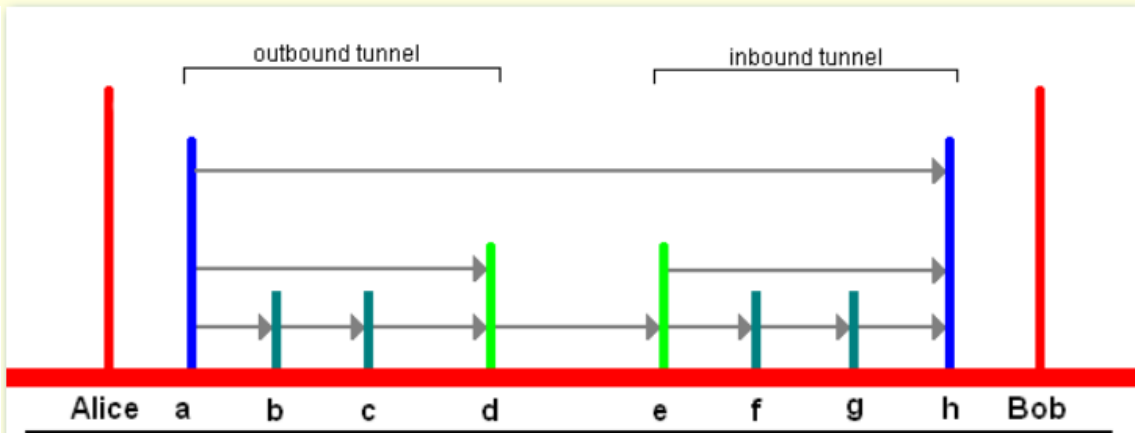


Fig. 1. Simple tunnel-oriented communication in I2P



		tunnel encryption (private key AES)	a to d, e to h
garlic encryption (ElGamal/SessionTag+AES)	a to h	transport encryption (DH/STS+AES, etc)	a to b, b to c, c to d, d to e, e to f, f to g, g to h

##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=-=-=-=-=-=~=-=-=-=-=-=~=-=-=-=-=-=-=-##
[+] I2P Technical Details:
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=-=-=-=-=-=~=-=-=-=-=-=~=-=-=-=-=-=-=-##

[+] RouterInfo - A data structure to provide routers the information necessary for contacting a specific router (their public keys, transport addresses, etc).

[+] LeaseSet - gives routers the information necessary for contacting a particular destination.

[+] ElGamal/AES+SessionTags - Are used for end-to-end encryption in several parts of I2P:

[?] To encrypt router-to-router tunnel building messages.
[?] For encryption of some netDb stores
--> Queries sent to Floodfill Routers (destination-to-router or router-to-router).
[?] ElGamal is used to encrypt IV's and Session keys in a single block.

[+] AES is used for Symmetric Encryption, in several cases:

[?] To Transport Encryption (see section "Transports") after DH Key Exchange
[?] AES Encrypted Payload using that key and IV.
[?] Encryption of some netDb stores
[?] Queries sent to Floodfill Routers (destination-to-router or router-to-router).

Darknet - I2P - Systemctl

```
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #systemctl enable i2p.service
```

```
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #systemctl status i2p.service
● i2p.service - load-balanced unspoofable packet switching network
     Loaded: loaded (/lib/systemd/system/i2p.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2021-06-18 09:01:06 CDT; 6s ago
```

I2P – Ports List

```
##-==============================================================================-##
##  I2P Router Home                    [[ xdg-open http://127.0.0.1:7657/home          ]]
##  Configure I2P Router HomePage      [[ xdg-open http://127.0.0.1:7657/confighome    ]]
##-==============================================================================-##
##  I2P Router Addressbook             [[ xdg-open http://127.0.0.1:7657/dns                   ]]
##  I2P Router DNS Subscriptions       [[ xdg-open http://127.0.0.1:7657/susidns/subscriptions ]]
##-==============================================================================-##
##  I2P Pluggins                       [[ xdg-open http://i2pwiki.i2p/index.php?title=Plugins  ]]
##  I2P Configure Plugins              [[ xdg-open http://127.0.0.1:7657/configplugins         ]]
##-==============================================================================-##
##  I2P HTTPS Outproxy                 [[ xdg-open http://127.0.0.1:4445                       ]]
##                                     [[ xdg-open https://outproxy-tor.meeh.i2p               ]]
##-==============================================================================-##
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~##
## ------------------------------------------------------------------------------ ##
##  Susimail Darknet Mail                  [[ xdg-open http://127.0.0.1:7657/susimail/ ]]
## ------------------------------------------------------------------------------ ##
##  Postman's SMTP Server                  [[ xdg-open http://127.0.0.1:7659           ]]
##                                         [[ xdg-open http://smtp.postman.i2p:25      ]]
## ------------------------------------------------------------------------------ ##
##  Postman's POP3 Server                  [[ xdg-open http://127.0.0.1:7660           ]]
##                                         [[ xdg-open http://pop.postman.i2p:110      ]]
## ------------------------------------------------------------------------------ ##
##~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~##
```

```
##
##-========================================================================-##
##  I2P HTTP Proxy                    [[ xdg-open http://127.0.0.1:4444   ]]
##  I2P HTTPS Proxy                   [[ xdg-open http://127.0.0.1:4444   ]]
##-========================================================================-##
##  I2NP Protocol                     [[ xdg-open http://127.0.0.1:26989  ]]
##  I2CP                              [[ xdg-open http://127.0.0.1:7654   ]]
##  I2CP SigType:                                  >> ECDSA_SHA256_P256 <<
##  SAM Bridge (SAMBridge)            [[ xdg-open http://127.0.0.1:7656   ]]
##-========================================================================-##
##  Irc2P IRC Tunnel                  [[ xdg-open http://127.0.0.1:6668   ]]
##  Postman Irc2p IRC Server          [[   hexchat irc.postman.i2p:6667   ]]
##  Echelons Irc2p IRC Server         [[   hexchat irc.echelon.i2p:6667   ]]
##-========================================================================-##
##  I2PSnark Torrent Client           [[ xdg-open http://127.0.0.1:7657/i2psnark/            ]]
##  Configure I2PSnark                [[ xdg-open http://127.0.0.1:7657/i2psnark/configure   ]]
##  I2PSnark Torrent Data Path        [[ caja /var/lib/i2p/i2p-config/i2psnark               ]]
##-========================================================================-##
##
##-========================================================================-##
##  I2P Webserver                       [[ xdg-open http://127.0.0.1:7658   ]]
##  I2P Monotone Server                 [[ xdg-open http://127.0.0.1:8998   ]]
##-========================================================================-##
##  I2P Router Tunnel Manager           [[ xdg-open http://127.0.0.1:7657/i2ptunnelmgr  ]]
##  I2P Router Configure Tunnels        [[ xdg-open http://127.0.0.1:7657/configtunnels ]]
##  I2P Router Configure Clients        [[ xdg-open http://127.0.0.1:7657/configclients ]]
##  I2P Router Main Config              [[ pluma /var/lib/i2p/i2p-config/router.config  ]]
##-========================================================================-##
```

```
echo "_____ I2P _____Localhost:Port___[owner|UID]__
echo "      [+] I2P HTTP Proxy              127.0.0.1:4444      i2pbrowser
echo "      [+] I2P HTTPS Proxy             127.0.0.1:4444
echo "      [+] I2P Bootstrapping           127.0.0.1:5353       i2psvc
echo "          (using Tors DNSPort)
echo "      [+] I2P HTTP Proxy              127.0.0.1:7657      i2pbrowser
echo "      [+] I2PWebserverPort            127.0.0.1:7658      i2pbrowser
echo "      [+] I2pSAMBridge                127.0.0.1:7656
echo "      [+] I2pUdpSAMBridge             127.0.0.1:7655
echo "      [+] I2pBobBridge                 127.0.0.1:2827
echo "      [+] I2pClientProtocolPort       127.0.0.1:7654
echo "      [+] I2pSSDPMulticastListener    127.0.0.1:1900
echo "      [+] TCPEventListener            127.0.0.1:7652
echo "      [+] I2pMonotone                  127.0.0.1:8998
```

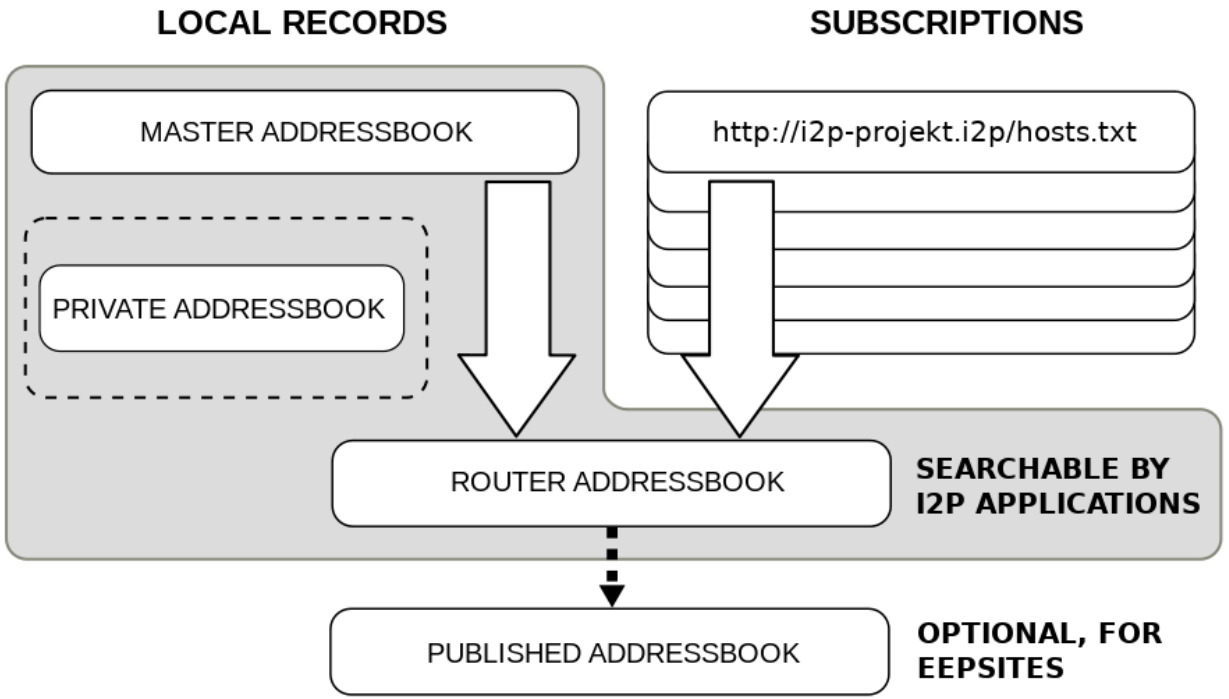
Addressbook Subscription technical details:

I2P supports Base32 hostnames similar to Tors .onion addresses.
Base32 addresses are much shorter and easier to handle
than the full 516-character Base64 Destinations or addresshelpers.
Example: ukeu3k5oycgaauneqgtnvselmt4yemvoilkln7jpvamvfx7dnkdq.b32.i2p

In Tor, the address is 16 characters (80 bits), or half of the SHA-1 hash.
I2P uses 52 characters (256 bits) to represent the full SHA-256 hash.
The form is {52 chars}.b32.i2p.

Base32 is implemented in the naming service,
which queries the router over I2CP to lookup the LeaseSet to get the full Destination.
Base32 lookups will only be successful when the Destination is up and publishing a LeaseSet.

I2P - Addressbook




```
http://i2p-projekt.i2p/hosts.txt
http://i2host.i2p/cgi-bin/i2hostetag
http://stats.i2p/cgi-bin/newhosts.txt
http://no.i2p/export/alive-hosts.txt
```

```
http://inr.i2p/browse/        ## I2P Name Registry is a domain name registration service for I2P
```

I2P

Configuration Help
Addressbook

Information: New Host Name with Address Helper

The address helper link you followed is for a new host name that is not in your address book. You may save this host name to your local address book. If you save it to your address book, you will not see this message again. If you do not save it, the host name will be forgotten after the next router restart. If you do not wish to visit this host, click the "back" button on your browser.

Host	*bote.i2p*
Base 32	bhjhc3lsdqzoyhxwzyrd63kvyg4br6n2337d74blyintae66mr2a.b32.i2p
Destination	6lZTYacjlXjSAxu-uXEO5oGsj-f4tfePHEvGjs5pu-AMXMwD7-xFdi8kdobDMJp9yRAl96U7yLI~0t9zI

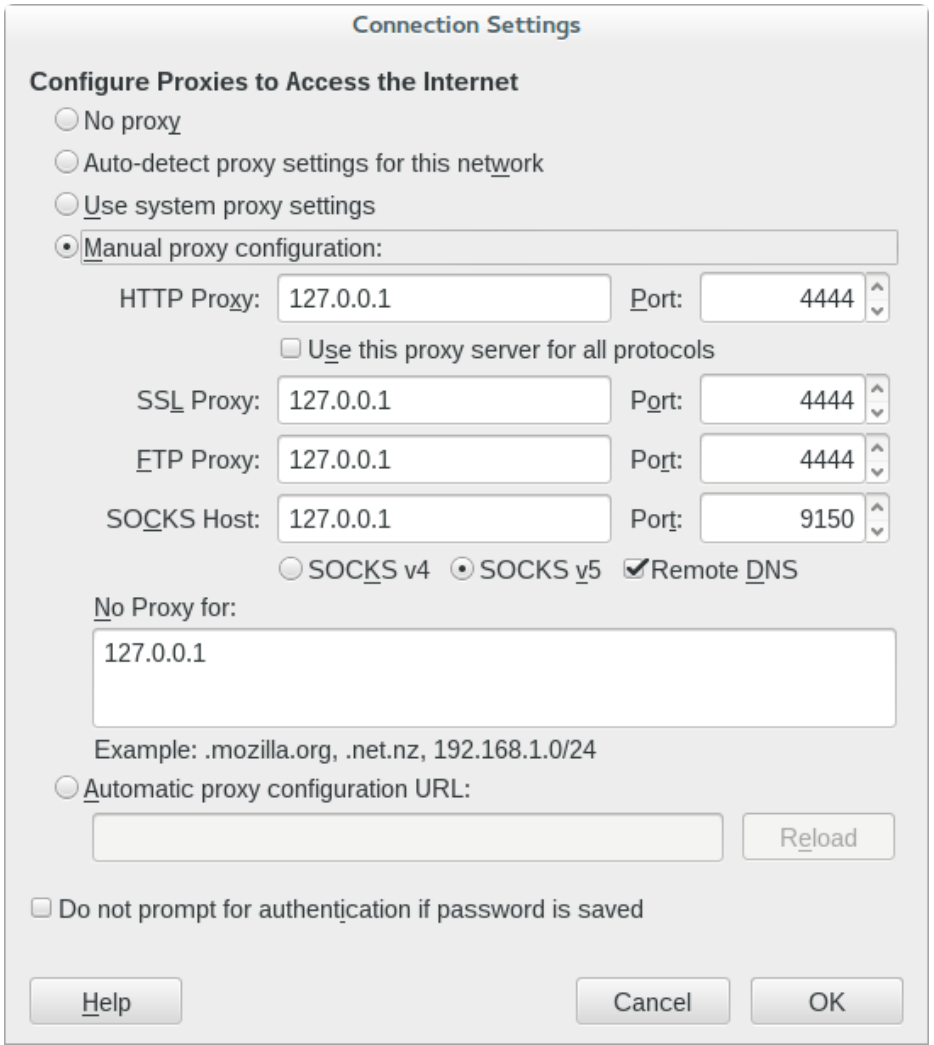
Continue to bote.i2p without saving

Save bote.i2p to router address book and continue to website

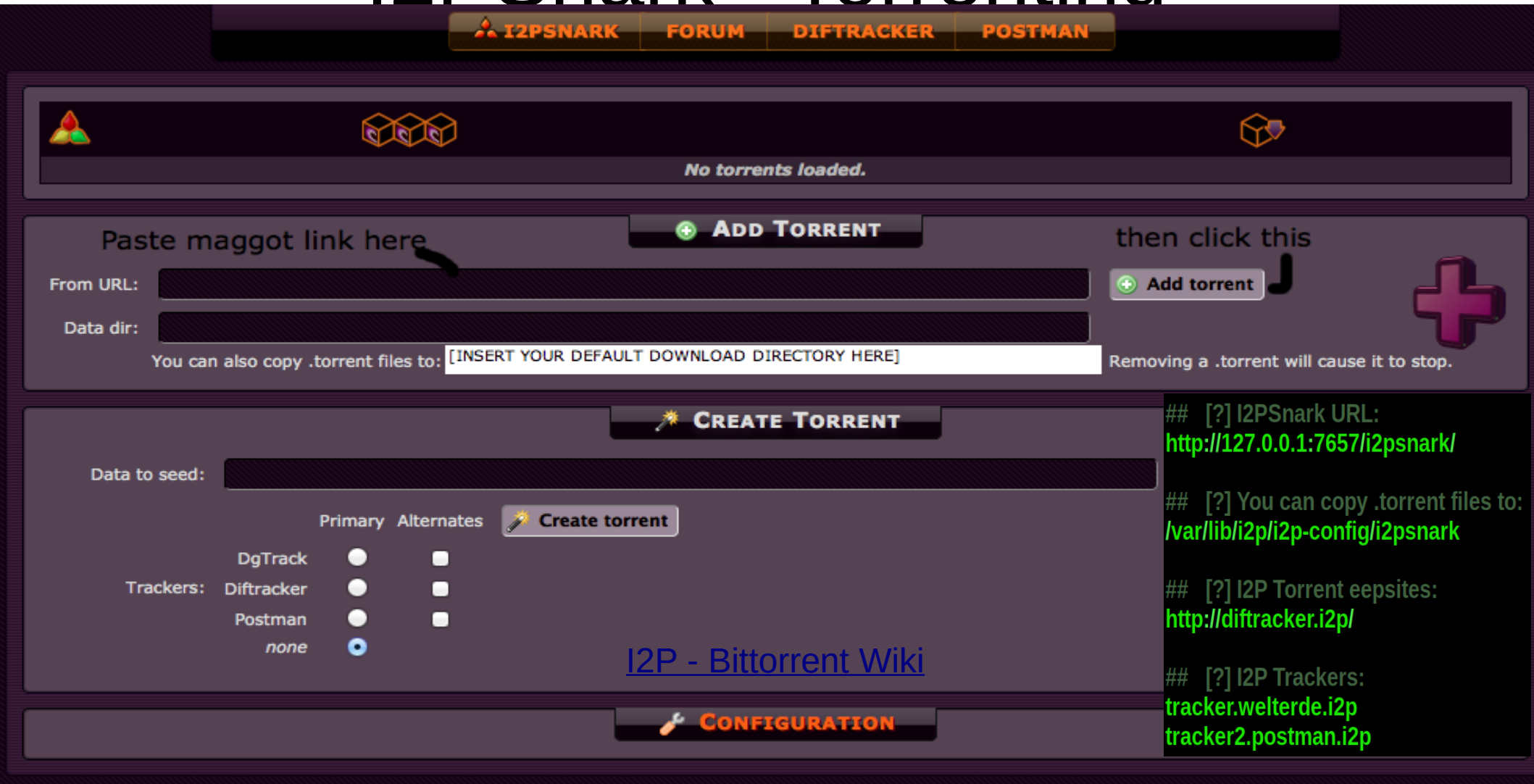
Save bote.i2p to master address book and continue to website

Save bote.i2p to private address book and continue to website

I2P – Firefox Settings



I2PSnark - Torrenting



I2PSnark - Torrenting

```
##   [?] I2PSnark URL:
http://127.0.0.1:7657/i2psnark/

##   [?] You can copy .torrent files to:
/var/lib/i2p/i2p-config/i2psnark

##   [?] I2P Torrent eepsites:
http://diftracker.i2p/

##   [?] I2P Trackers:
tracker.welterde.i2p
tracker2.postman.i2p
```

DHT

Unlike standard DHT, I2P DHT does not use a bit in the options handshake, or the PORT message. It is advertised with an extension message, identified as "i2p_dht" in the extension handshake. It contains a bencoded dictionary with two keys, "port" and "rport", both integers.

The UDP (datagram) port listed in the compact node info is used to receive repliable (signed) datagrams. This is used for queries, except for announces. We call this the "query port". This is the "port" value from the extension message. Queries use I2CP protocol number 17.

In addition to that UDP port, we use a second datagram port equal to the query port + 1. This is used to receive unsigned (raw) datagrams for replies, errors, and announces. This port provides increased efficiency since replies contain tokens sent in the query, and need not be signed. We call this the "response port". This is the "rport" value from the extension message. It must be 1 + the query port. Responses and announces use I2CP protocol number 18.

Compact peer info is 32 bytes (32 byte SHA256 Hash) instead of 4 byte IP + 2 byte port. There is no peer port. In a response, the "values" key is a list of strings, each containing a single compact peer info.

Compact node info is 54 bytes (20 byte SHA1 Hash + 32 byte SHA256 Hash + 2 byte port) instead of 20 byte SHA1 Hash + 4 byte IP + 2 byte port. In a response, the "nodes" key is a single byte string with concatenated compact node info.

Secure node ID requirement: To make various DHT attacks more difficult, the first 4 bytes of the Node ID must match the first 4 bytes of the destination Hash, and the next two bytes of the Node ID must match the next two bytes of the destination hash exclusive-ORed with the port.

In a torrent file, the trackerless torrent dictionary "nodes" key is TBD. It could be a list of 32 byte binary strings (SHA256 Hashes) instead of a list of lists containing a host string and a port integer. Alternatives: A single byte string with concatenated hashes, or a list of strings alone.

I2P - Bittorrent Wiki

Destination Enforcement

Some, but not all, I2P bittorrent clients announce over their own tunnels. Trackers may choose to prevent spoofing by requiring this, and verifying the client's Destination using HTTP headers added by the I2PTunnel HTTP Server tunnel. The headers are X-I2P-DestHash, X-I2P-DestB64, and X-I2P-DestB32, which are different formats for the same information. These headers cannot be spoofed by the client. A tracker enforcing destinations need not require the ip announce parameter at all.

I2P – AppArmor Profiles

```
capability sys_ptrace,

/usr/bin/i2prouter                  r,

@{PROC}/1/comm                      r,
owner @{PROC}/[0-9]*/               r,
owner @{PROC}/[0-9]*/stat           r,
owner @{PROC}/[0-9]*/cmdline        r,
@{PROC}/uptime                      r,
@{PROC}/sys/kernel/pid_max          r,
owner /{,lib/live/mount/overlay/}var/lib/i2p/** rwk,
owner /{,lib/live/mount/overlay/}var/lib/i2p/i2p-config/eepsite/cgi-bin rix,
owner /{,lib/live/mount/overlay/}var/log/i2p/* rw,

owner /{,var/}run/i2p/{i2p,routerjvm}.pid rw,
owner /{,var/}run/i2p/router.ping rw,
# Needed by Java
@{PROC}                                     r,
owner @{PROC}/[0-9]*/                       r,
owner @{PROC}/[0-9]*/cgroup                 r,
owner @{PROC}/[0-9]*/mountinfo              r,
owner @{PROC}/[0-9]*/status                 r,
@{PROC}/[0-9]*/net/ipv6_route               r,
@{PROC}/[0-9]*/net/if_inet6                 r,
@{HOME}/.java/fonts/**              r,
owner  @{HOME}/.i2p/                rw,
owner  @{HOME}/.i2p/**              rwk,
owner  @{HOME}/.i2p/eepsite/cgi-bin/** rix,

# Prevent spamming the logs
deny owner @{HOME}/.java/           wk,
deny @{HOME}/.fontconfig/           wk,
deny @{HOME}/.java/fonts/**         wk,
```

I2P – Firejail Profile

```
caps.drop all
ipc-namespace
machine-id
netfilter
no3d
nodvd
nogroups
nonewprivs
nosound
notv
nou2f
novideo
protocol unix,inet
seccomp
shell none
```

I2P – Firejail Profile 2

```
disable-mnt
private-cache
private-dev
private-etc alternatives,ca-certificates,crypto-policies,dconf,
group,hostname,hosts,i2p,java-10-openjdk,java-11-openjdk,java-1
2-openjdk,java-13-openjdk,java-8-openjdk,java-9-openjdk,java-op
enjdk,ld.so.cache,localtime,machine-id,nsswitch.conf,passwd,pki
,resolv.conf,ssl
private-tmp
noblacklist ${HOME}/.config/i2p
noblacklist ${HOME}/.i2p
noblacklist ${HOME}/.local/share/i2p
noblacklist ${HOME}/i2p
mkdir ${HOME}/.config/i2p
mkdir ${HOME}/.i2p
mkdir ${HOME}/.local/share/i2p
mkdir ${HOME}/i2p
whitelist ${HOME}/.config/i2p
whitelist ${HOME}/.i2p
whitelist ${HOME}/.local/share/i2p
whitelist ${HOME}/i2p
```

What is Tor?

- Tor is a SOCKS5 encryption protocol.
- Tor tunnels all traffic running across the users network anonymously.
- Tor conceals a user's location and network data from anyone monitoring the user locally, and remotely.

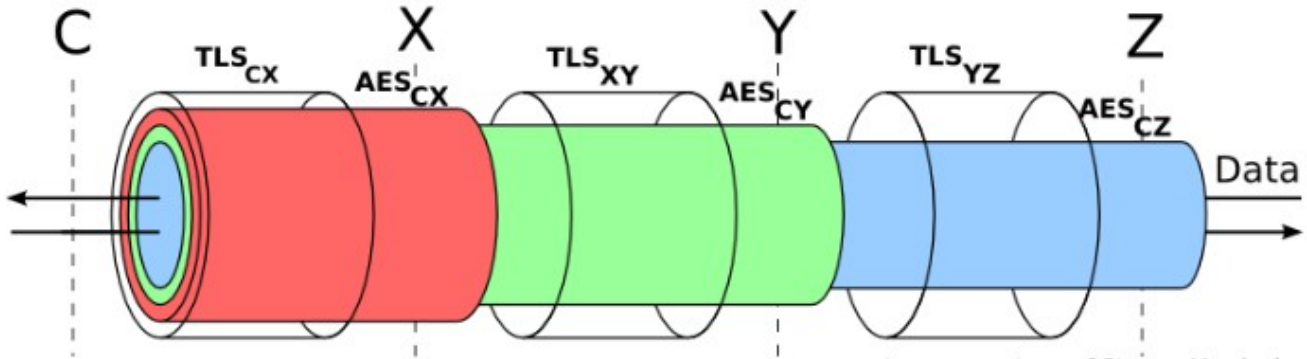
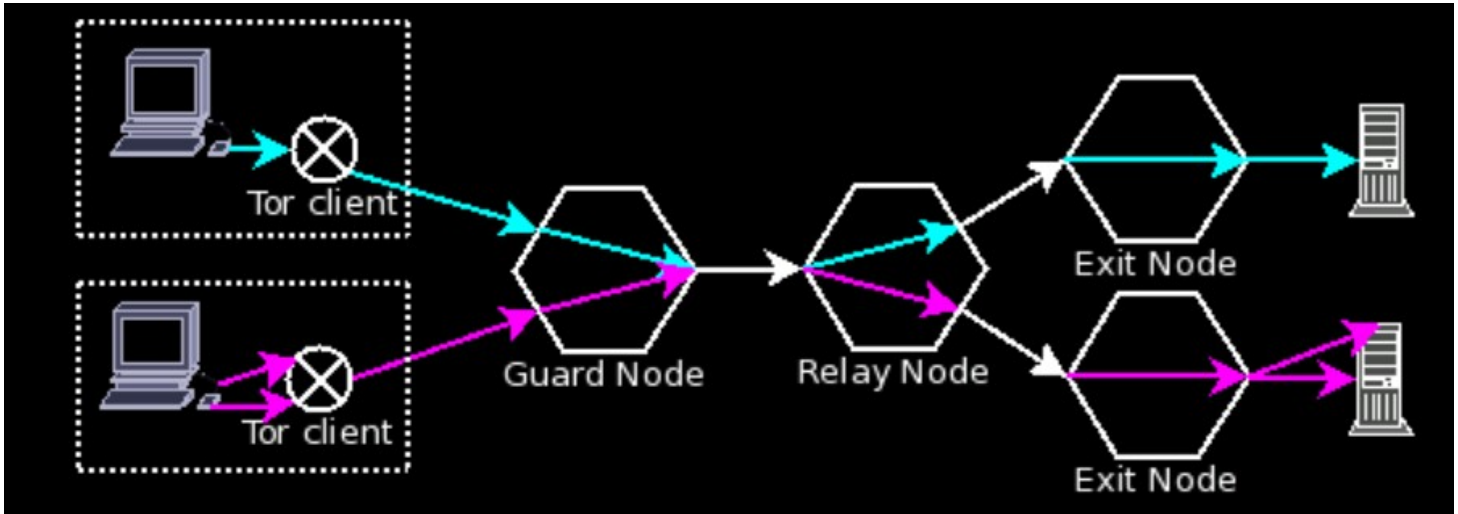


Illustration 2: Tor Encryption

Xe1phix-AnonSurf-Wiki.txt

Tor Protocol Details

- The Tor protocol works by:
- Multiplexing multiple “circuits” over a single node-to-node TLS connection.
- Tor traffic is routed through 3 nodes by default:
- Guard
- relay
- exit



Xe1phix-AnonSurf-Wiki.txt

How Tor Works

Tor passes data packets through a series of nodes. That path consists of three types of nodes: an entry node, a middle node, and an exit node. First, the traffic is encrypted and sent to the entry node. Next, layers of the data packet are stripped off each time it passes through one of the middle nodes. Finally, the exit node uses an unencrypted link to communicate with the target server outside the Tor network.

What is a Tor hidden service?

Tor hidden services provides a means of hosting services on the Tor network. Any type of network service may be hosted as a hidden service (such as web servers, file shares, and instant messaging servers).

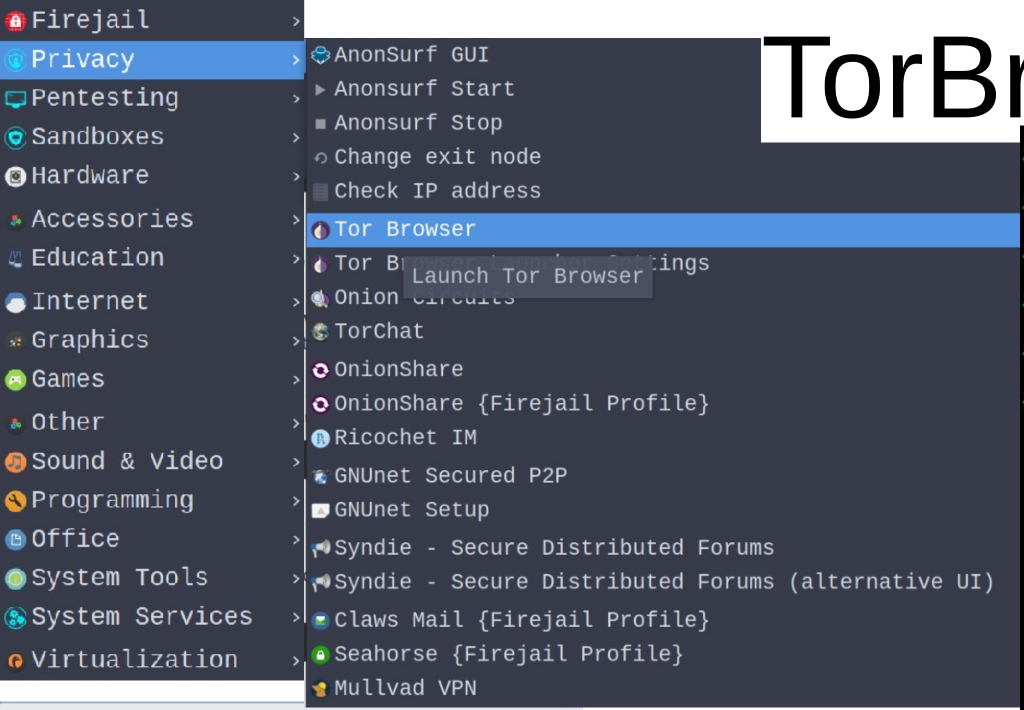
Instead of using an IP address or domain name, Tor hidden services are accessed by their *.onion* address. The *.onion* address is an automatically generated name that is derived from the public key of the hidden service.

.onion addresses are only accessible over Tor. **Tor Browser** is one way to access *.onion* addresses. In Subgraph OS, any application can access *.onion* addresses because all applications are routed through Tor.

Tor hidden services provide privacy and anonymity for both the server and the client. Tor hidden services have the following benefits over regular network services:

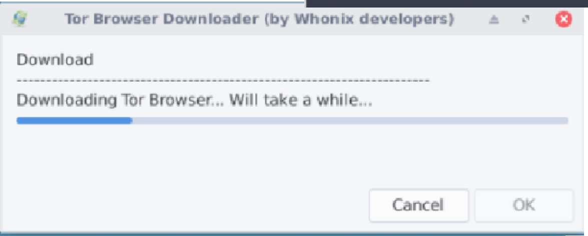
1. Neither the client nor the server need to know the network location (IP address) of each other. Tor routes traffic through a series of rendezvous points that hide the client IP address from the server. The server's network location (IP address) is also hidden from the client, who connects to the *.onion* address of the server.
2. All traffic between the client and server is end-to-end encrypted. Traffic never leaves the Tor network, meaning that it is only decrypted on either end of the transaction. When Tor is used to connect to the regular Internet, traffic is only encrypted until the *exit-node*. Without using another layer of encryption such as HTTPS, exit nodes can observe traffic. Tor hidden services are not affected by this limitation.

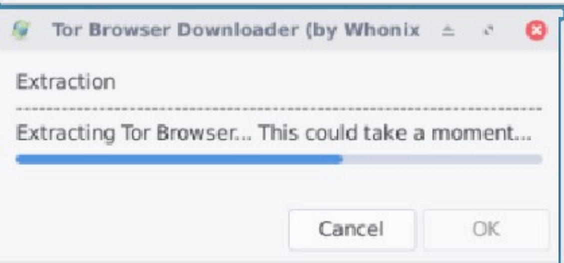
TorBrowser

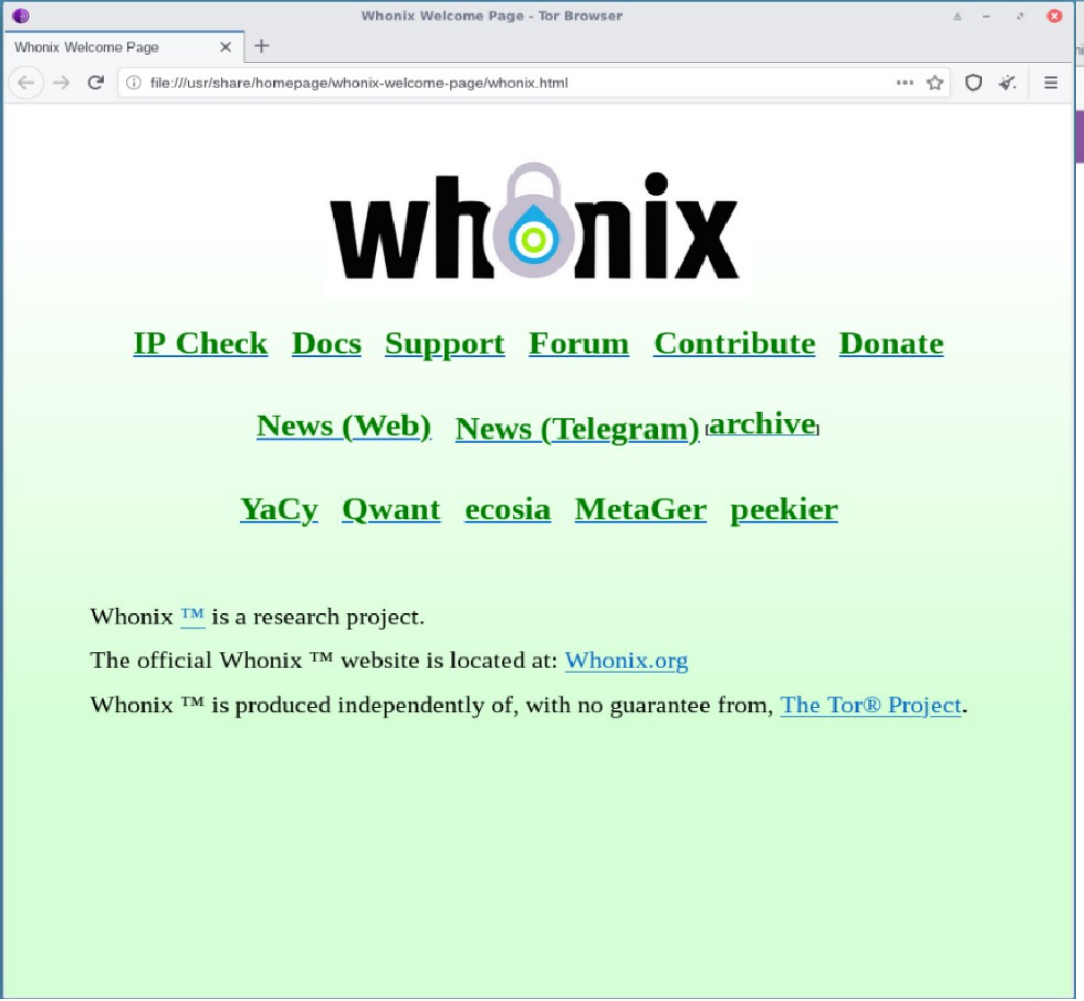


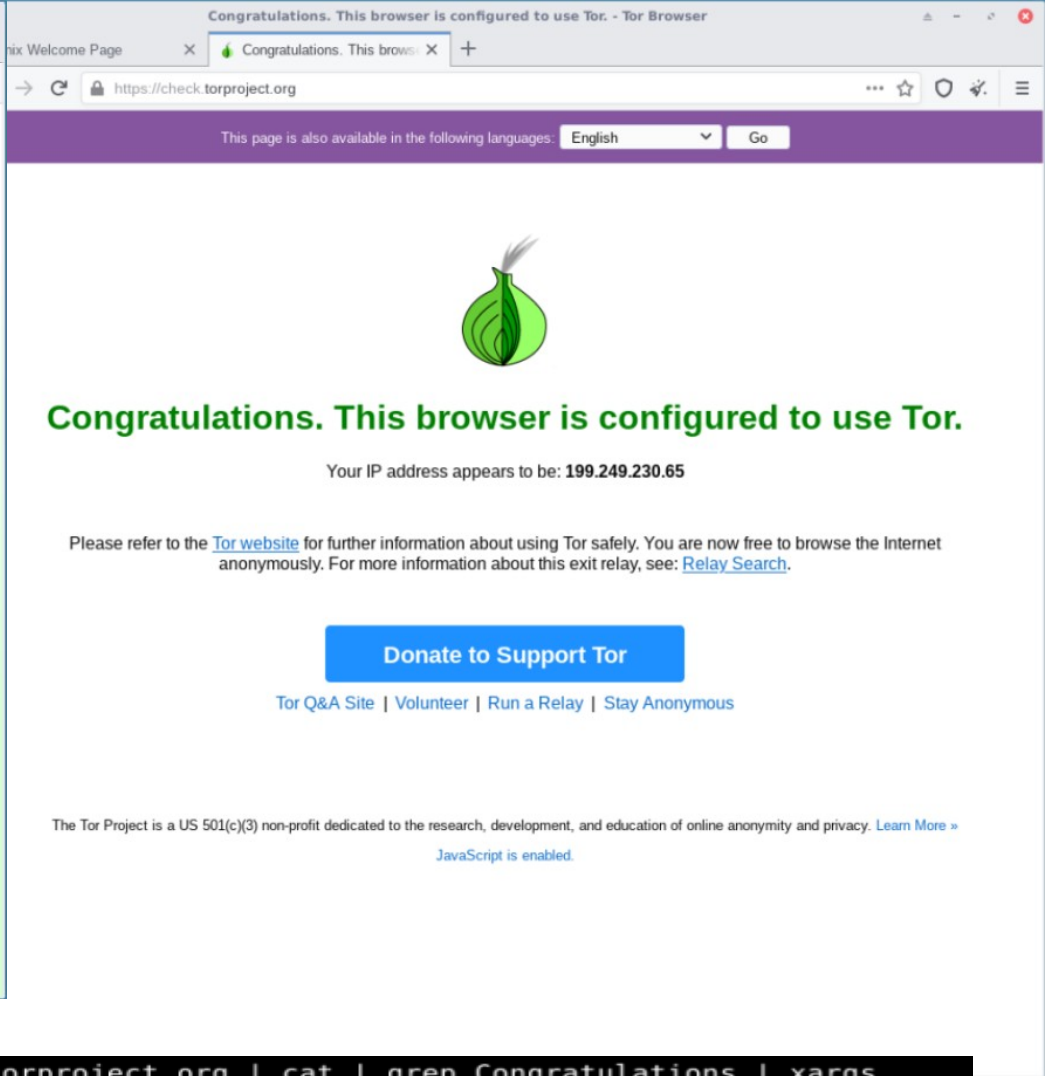
##~=-=-=-=-=-=--##
[+] TorBrowser Launcher Technical Details:
##~=-=-=-=-=-=--##
--
[?] When you start tor browser this is what happens:
--

[1] Tor version check is performed
[2] Tor browser signature is downloaded to verify no MITM attempts were performed.
[3] Tor tarball is downloaded
[4] The tor tarball is verified by the developers signature
[5] Either a GOOD SIGNATURE notification is presented or a SIGNATURE VERIFICATION FAILED.
[6] Tor is then decompressed and launched.










```
user@host:~$ curl --socks5 localhost:9050 https://check.torproject.org | cat | grep Congratulations | xargs
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  5072    0  5072    0     0   4798      0 --:--:--  0:00:01 --:--:--  4798
Congratulations. This browser is configured to use Tor. Congratulations. This browser is configured to use Tor.
```

Tor – AppArmor Profiles

```
network netlink raw,
network tcp,

ptrace (trace) peer=@{profile_name},
signal (receive, send) set=("term") peer=@{profile_name},

deny /etc/host.conf r,
deny /etc/hosts r,
deny /etc/nsswitch.conf r,
deny /etc/resolv.conf r,
deny /etc/passwd r,
deny /etc/group r,
deny /etc/mailcap r,

/etc/machine-id r,
/var/lib/dbus/machine-id r,

/dev/ r,
/dev/shm/ r,
```

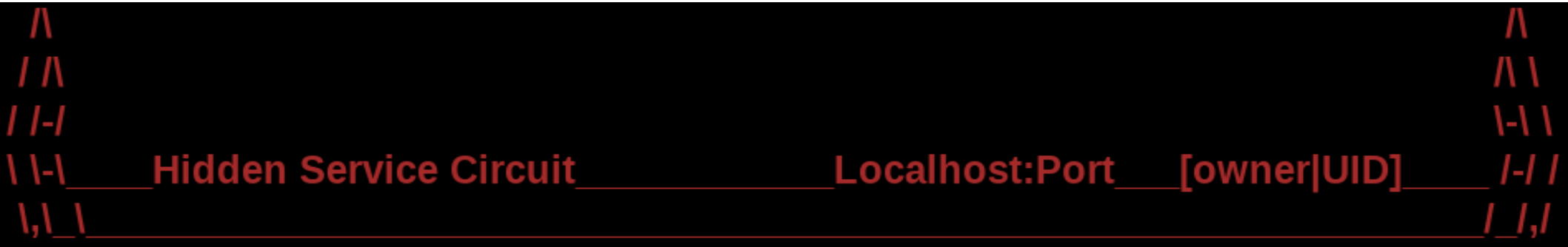
TorBrowser – AppArmor 2

```
  owner @{torbrowser_installation_dir}/ r,
  owner @{torbrowser_installation_dir}/* r,
  owner @{torbrowser_installation_dir}/** r,
  owner @{torbrowser_home_dir}/ rwk,
  owner @{torbrowser_home_dir}/.** rwk,
  owner @{torbrowser_home_dir}/ rw,
  owner @{torbrowser_home_dir}/** rwk,
  owner @{torbrowser_home_dir}.bak/ rwk,
  owner @{torbrowser_home_dir}.bak/** rwk,
  owner @{torbrowser_home_dir}/*.so mr,
  owner @{torbrowser_home_dir}/.cache/fontconfig/ rwk,
  owner @{torbrowser_home_dir}/.cache/fontconfig/** rwkl,
  owner @{torbrowser_home_dir}/browser/** r,
  owner @{torbrowser_home_dir}/{,browser/}components/*.so mr,
  owner @{torbrowser_home_dir}/Downloads/ rwk,
  owner @{torbrowser_home_dir}/Downloads/** rwk,
  owner @{torbrowser_home_dir}/firefox rix,
  owner @{torbrowser_home_dir}/{,TorBrowser/UpdateInfo/}updates
/[0-9]*/* rw,
  owner @{torbrowser_home_dir}/{,TorBrowser/UpdateInfo/}updates
/[0-9]*/{,MozUpdater/bgupdate/}updater ix,
```

```
mkdir ${HOME}/.config/torbrowser
mkdir ${HOME}/.local/share/torbrowser
whitelist ${DOWNLOADS}
whitelist ${HOME}/.config/torbrowser
whitelist ${HOME}/.local/share/torbrowser
include whitelist-common.inc
include whitelist-var-common.inc

caps.drop all
netfilter
nodvd
nogroups
nonewprivs
noroot
notv
nou2f
novideo
protocol unix,inet,inet6
seccomp !chroot
shell none
```

Tor Firejail Profiles

```
ignore noexec ${HOME}

noblacklist ${HOME}/.config/torbrowser
noblacklist ${HOME}/.local/share/torbrowser
  include disable-common.inc
  include disable-devel.inc
  include disable-exec.inc
  include disable-interpreters.inc
  include disable-passwdmgr.inc
  include disable-programs.inc
  include disable-xdg.inc

disable-mnt
private-bin bash,cat,cp,cut,dirname,env,
expr,file,gpg,grep,gxmessage,id,kdialog,
ln,mkdir,mv,python*,rm,sed,sh,tail,tar,t
clsh,test,tor-browser,tor-browser-en,tor
browser-launcher,update-desktop-database
,xmessage,xz,zenity
private-dev
private-etc alsa,alternatives,asound.conf,
ca-certificates,crypto-policies,fonts,ld.s
o.cache,ld.so.conf,ld.so.conf.d,ld.so.prel
oad,machine-id,pki,pulse,resolv.conf,ssl
private-tmp

dbus-user none
dbus-system none
```



[+] Tor HTTPProxy	127.0.0.1:80	
[+] Tor HTTPSProxy	127.0.0.1:443	
[+] Tor Transparent Proxy	127.0.0.1:9040	amnesia
[+] Tor SOCKS4a	127.0.0.1:1080	amnesia
[+] Tor SOCKS5 (Default)	127.0.0.1:9050	amnesia
[+] SocksPort for Tor Browser	127.0.0.1:9150	amnesia
[+] SocksPort for the MUA	127.0.0.1:9061	amnesia
[+] Tails-Specific Services SocksPort	127.0.0.1:9062	amnesia
[+] Tails Time Synchronization Service	127.0.0.1:9062	htp
[+] Tails System DNS	127.0.0.1:53	amnesia
[+] Torified DNS Socket	127.0.0.1:5353	amnesia
[+] Tor ControlPort	127.0.0.1:9051	root
[+] Tor Control Port Filter	127.0.0.1:9052	amnesia

Curl – Tor (SOCKS5)

```
curl --socks5-hostname 127.0.0.1:9050 -o $File $URL
curl --socks5-hostname 127.0.0.1:9150 -o $File $URL
curl --proxy "socks5h://localhost:9050" --tlsv1.2 $URL
curl --proxy "socks5h://localhost:9150" --tlsv1.2 $URL
```

```
curl --resolve 127.0.0.1:4444:http://killyourtv.i2p/killyourtv.asc

curl --resolve 127.0.0.1:9053:https://tails.boum.org/tails-signing.key
curl --socks5-hostname 127.0.0.1:9050 https://check.torproject.org
```

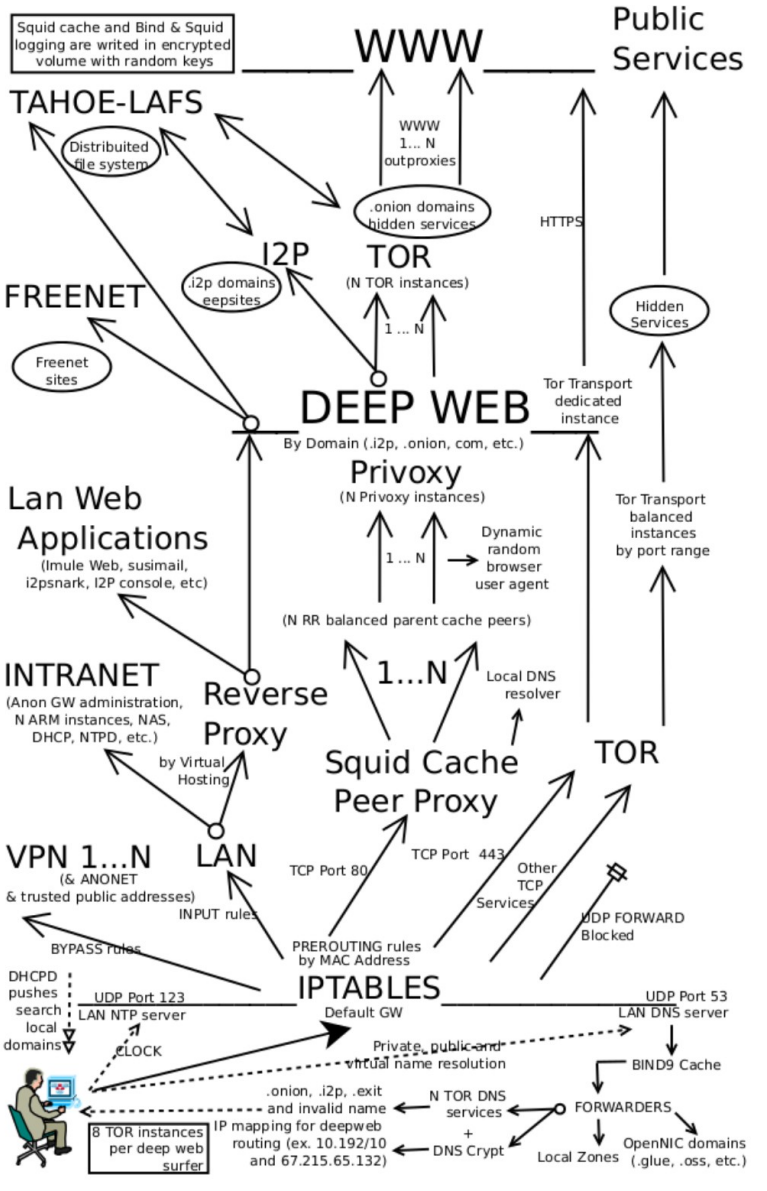
```
curl --proxy socks4a://
curl --proxy --socks4a
curl --proxy socks5://
curl --socks5 $HOST:$Port
curl --socks5 127.0.0.1:9150
curl --proxy "socks5h://localhost:9050"
```

The AnonGW provides:

- DHCP service ==> automatically dictates the network rules to use the deep web proxy.
- Time service (NTP server) ==> syncronizes LAN clocks, no UDP leaks out of the LAN.
- DNS service (bind9) ==> (I) manages local zones for LAN services, VPN networks and reverse proxy (II) forwards the queries to the TOR network and, for a full answer, to OpenDNS via DNSCrypt.
- filtering service (iptables) ==> activate/deactivate the deep web proxy filtering mac address (layer 2 oriented).
- firewalling all protocols except toward VPN's and Anonet (layer 3 oriented).
- routing tcp via proxies (layer 4 oriented).
- web traffic managed through HTTP headers via direct cache, reverse and filtering web proxies (layer 7 oriented).
- access anonymously to web (clearnet and hidden services) through TOR: (I) fast HTTP via a round robin clustered cache proxy (Squid3) over multiple privoxy and TOR instances (II) HTTPS and other TCP services directly via TOR (transport mode).
- access to I2P resources: (I) eeepsites via Squid and Privoxy (II) I2P web clients via reverse proxy (Apache2 virtual host).
- access to Freenet resources: via reverse proxy (virtual host).
- access to Tahoe-LAFS storage grids in I2P (or TOR or anything else): via reverse proxy (virtual host) or via SSHFS.
- access to OpenNIC domains but note: only few exit nodes resolve these domains.

[AnonGW I2P Eepsite]

[AnonGW Tor Onion]



Darknet – AnonGW Project

How it works:

1. Jon Do adds in the AnonGW the MAC address of his new laptop (or tablet or smartphone or anything else with the capability of web browsing and a known physical address...)

2. The laptop broadcasts its presence in the LAN, asking networking settings ==> the DHCP server (isc-dhcp) instructs which IP address, default gateway, NTP server, DNS server and which DNS domain to search are needed. The domain will be the local LAN domain and the VPN's domains (office, datacentre, etc.).

3. he laptop starts to synchronize its clock with the clock of the AnonGW. Jon Do searches "http://www.debian.org" in his browser and its DNS client queries AnonGW DNS server to resolve it. If the answer is not cached, the DNS server (bind9) forwards the query via TOR network or, as last attempt, via DnsCrypt on OpenDNS

4. HTTP request is intercepted by the cache proxy (Squid3) and "round robin" balanced over the N (ex. 8) Privoxy parent peers. The browser user agent are dinamically modified by a script (uagent.py) and forwarded via N (ex: 8) TOR instances.

5. Jon Do searches https://www.eff.org and open a ssh session on a public server: both connections are "transported" by two different instances of TOR.

6. Jon Do opens a new tab in his web browser and searches his Tor Mail "http://jhiwjjlqpyawmpjx.onion": the DNS server forwards directly to the TOR DNS service that resolves the ".onion" domain with a private address (ex: 10.192.0.1). HTTP request is intercepted by Squid and forwarded to Privoxy: it recognizes the "dot onion" domain and forwards it in the TOR network to find the "hidden service".

7. Jon Do opens again a new tab and searches "http://killyourtv.i2p.xyz/debian/": the DNS server forwards directly to the TOR DNS service that resolves the ".i2p.xyz" domain with a private address (ex: 10.192.0.2). HTTP request is always intercepted by Squid and forwarded to Privoxy: it recognizes the "dot i2p" domain and forwards it in the I2P network to find the "eepsite".

8. Entering only "i2p" in his address bar he has access to the i2p console or to a invisible torrent client on "https://i2p/i2psnark/": the DNS find in its local zones and resolves it with the AnonGW IP and the HTTP/HTTPS request is authenticated by the Apache2 I2P.JonDolocaldomain virtual host and forwarded via its reverse proxy to the i2p router.

9. Entering only "tahoe" in his address bar he has access to the Tahoe-LAFS welcome page: Jon Do can find the KYTV's Tahoe-LAFS debian repository on:

```
https://tahoe/uri/URI:DIR2-RO:mvp3so6kemo6fn6abddzjthnuu:jjg475us6hbmya3cc
```

1. Entering only "freenet" in his address bar he has access to the Freenet console and can read the Toad's Flog on:

```
https://freenet/freenet:USK@yGvITGZzrY1vUZK-4AaYLgcjZ7ysRqNTMfdcO8gS-LY,-a
```

1. Jon Do can access to the LAN of his office via VPN: he has configured the DNS zone and the VPN service on his Anonymous Gateway. Anonet is the same concept: he has configured a VPN but the DNS forwards the ".ano" queries to the Anonet DNS.

2. Jon Do needs to do a different research and removes his MAC address from the "Deep Web Proxy rules" from the AnonGW and search "http://grep.geek": the DNS server forwards the queries to the OpenNIC DNS servers. He can surf free TLDs and normal clearnet without anonymization.

Requirements:

1. A box with a dual core processor, two network interfaces and two hard drives.

2. Ubuntu server 12.04 64 bit.

3. RAID 1 (if paranoia/comfort > 1 then whole disk encrypted) and/or: random key encrypted swap, random key encrypted tmp logical volume and random key encrypted cache volume.

4. ISC DHCP, Bind9, DnsCrypt, NTPD, Squid3, N Privoxy instances, N TOR instances (where N is 8 * deep web surfer), I2P, Tahoe-LAFS-I2P, Freenet, Apache2, OpenVPN

I used an Ubuntu Gnu/Linux server 12.04 (LTS) with apparmor enabled and with all its standard repository packages, included I2P repository for "Precise" mantained by KillYourTV (thank you!).

40 privoxy and tor instances over a **40** round robin squid cache proxy peers, **2** TOR TransPort on different instances each user's MacAddress to torify the ports different from *80* tcp.

AnonGW I2P Eepsite

AnonGW Tor Onion

Securing IRC

- **SSL** - For more info, see Certificates. On the IndyMedia network that hosts the #riseup channel, you'll need to use port 6697.
- **SASL EXTERNAL** and **CertFP** - different but related mechanisms to authenticate to services using certificates. Used together, they can ensure that you are authenticated to services before the connection to the network is complete and without any passwords.
  - Register your nick: `/msg nickserv register <password> <email>`
  - Make yourself a certificate:
    `openssl req -x509 -new -newkey rsa:4096 -sha256 -days 1000 -nodes -out riseup.pem -keyout riseup.pem`
  - Set your client to connect with that certificate using SASL EXTERNAL: see OFTC for examples
  - Connect using that certificate
  - Check that your fingerprint with `/whois <your-nick>` is the same as your certificate's fingerprint. See Certificates for more info.
  - Add the CertFP (certificate fingerprint): `/msg nickserv cert add`
  - Guard the certificate as you would your password.

Hexchat - IRC Hardening

```
~/.config/hexchat/
~/.config/hexchat/certs/
~/.config/hexchat/ignore.conf
~/.config/hexchat/hexchat.conf

/ignore *!*@* CTCP DCC
/ignore * CTCP DCC
/ignore * CTCPS
/ignore * DCC
/ignore * CTCP DCC
/set irc_logging 1
/set identd OFF
/set dcc_auto_chat 0
/set dcc_auto_resume OFF
/set irc_hide_version on
```

```
##-============================-##
##    [+]  Darknet-[I2P]-IRC Servers
##-============================-##
##        >  irc.echelon.i2p
##        >  irc.postman.i2p
##        >  irc.killyourtv.i2p
##        >  irc.devfs.i2p
##        >  irc.dg.i2p
##        >  irc.undefined.i2p
##        >  irc.welterde.i2p
##-============================-##
```

```
##-============================-##
##    [+] net_proxy_use:
##-============================-##
## ------------------------------------------- ##
##    [?] What to use proxies for:
##          ____________________
##         ||  > 0 = All        ||
##         ||  > 1 = IRC Only   ||
##         ||  > 2 = DCC Only   ||
##          ^^^^^^^^^^^^^^^^^^^^
##    [?] 0 is not equal to "off"
##    [?] 0 stands for "All"
## ------------------------------------------- ##
net_proxy_use = 0
```

```
##-==============================-##
##    [+] net_proxy_type:
##-==============================-##
## ----------------------------------------------- ##
##    [?] Which type of proxy to use:
##          ____________________
##         ||  > 2 = Socks4     ||
##         ||  > 3 = Socks5     ||
##         ||  > 4 = HTTP       ||
##          ^^^^^^^^^^^^^^^^^^^^
## ----------------------------------------------- ##
net_proxy_type = 3
```

```
Ignore on * changed.
*!*@* added to ignore list.
Ignore on * changed.
dcc_auto_chat set to: 0 (was: 0)
dcc_auto_resume set to: 0 (was: 1)
irc_user_name set to: xelphix (was: Xelphix)
```

[List of Hexchat Settings.md]

[Hexchat-Hardening-Cheatsheet]

Hexchat – Register Your Nick

```
##  Set your nick:
/nick xe1phix

##  Register your IRC nick:
/msg NickServ REGISTER [redacted] xe1phix@protonmail.ch

##  Verify the IRC nick:
/msg NickServ VERIFY REGISTER xe1phix [redacted]

##  identify to your primary account:
/msg NickServ IDENTIFY xe1phix [redacted]
```

```
        *  You are now known as xe1phix
 >NickServ<  REGISTER [redacted] xe1phix@protonmail.ch
 -NickServ-  An email containing nickname activation instructions has been sent to xe1phix@protonmail.ch.
 -NickServ-  Please check the address if you don't receive it. If it is incorrect, DROP then REGISTER again.
 -NickServ-  If you do not complete registration within one day, your nickname will expire.
 -NickServ-  xe1phix is now registered to xe1phix@protonmail.ch.
 >NickServ<  VERIFY REGISTER xe1phix [redacted]
 -NickServ-  xe1phix has now been verified.
```

Generate A TLS Certificate

```
## ---------------------------------------- ##
##    [+] Generate A Certificate:
## ---------------------------------------- ##
openssl req -x509 -new -newkey rsa:4096 -sha256 -days 1096 -nodes -out libera.pem -keyout libera.pem

## ---------------------------------------- ##
##    [+] Inspecting your certificate
## ---------------------------------------- ##
openssl x509 -in libera.pem -noout -enddate

## ---------------------------------------- ##
##    [+] Check The fingerprint:
## ---------------------------------------- ##
openssl x509 -in libera.pem -noout -fingerprint -sha512 | awk -F= '{gsub(":","");print tolower ($2)}'
## ---------------------------------------- ##
##    [+] Move Certificate Into The Hexchat Directory:
## ---------------------------------------- ##
cp -v libera.pem ~/.config/hexchat/
```

```
-NickServ- Thank you for verifying your e-mail address! You have taken steps in ensuring that your registrations are not exploited.
         * [xe1phix] (~Xe1phix@173-22-75-86.client.mchsi.com): realname
         * [xe1phix] #cialug #parrotsec
         * [xe1phix] iridium.libera.chat :Stockholm, SE
         * [xe1phix] is using a secure connection [TLSv1.3, TLS_AES_256_GCM_SHA384]
         * [xe1phix] [redacted] :actually using host
         * [xe1phix] idle 00:07:17, signon: Sat Aug  7 16:40:27
         * [xe1phix] is logged in as xe1phix
         * [xe1phix] End of WHOIS list.
```

https://libera.chat/guides/certfp

Hexchat-TLS-Cheatsheet

IRC – Libera.chat TLS Log

```
* Looking up irc.libera.chat
* Connecting to irc.libera.chat ([redacted]:6697)
* * Subject: /C=US/O=Internet Security Research Group/CN=ISRG Root X1
* * Issuer: /C=US/O=Internet Security Research Group/CN=ISRG Root X1
* * Subject: /C=US/O=Let's Encrypt/CN=R3
* * Issuer: /C=US/O=Internet Security Research Group/CN=ISRG Root X1
* * Subject: /CN=iridium.libera.chat
* * Issuer: /C=US/O=Let's Encrypt/CN=R3
* * Certification info:
*   Subject:
*     CN=iridium.libera.chat
*   Issuer:
*     C=US
*     O=Let's Encrypt
*     CN=R3
*   Public key algorithm: rsaEncryption (4096 bits)
*   Sign algorithm sha256WithRSAEncryption
*   Valid since Aug  2 05:09:56 2021 GM to Oct 31 05:09:54 2021 GM
* * Cipher info:
*   Version: TLSv1.3, cipher TLS_AES_256_GCM_SHA384 (256 bits)
* Connected. Now logging in.
* *** Checking Ident
* *** Looking up your hostname...
* *** No Ident response
* *** Found your hostname: [redacted]
* Capabilities supported: account-notify away-notify chghost extended-join multi-prefix sasl=PLAIN,ECDSA-NIST256P-CHALLENGE,EXTERNAL tls account-tag cap-notify echo-message solanum.chat/identify-msg solanum.chat/realhost
* Capabilities requested: account-notify away-notify chghost extended-join multi-prefix cap-notify
* Capabilities acknowledged: account-notify away-notify chghost extended-join multi-prefix cap-notify
* Welcome to the Libera.Chat Internet Relay Chat Network Xe1phix
```

Add your fingerprint to NickServ

```
## ------------------------------------------------------------------------------------ ##
##    [+] Allow NickServ to recognise you based on your certificate
## ------------------------------------------------------------------------------------ ##
/whois xe1phix

## ---------------------------------------- ##
##    [+]
## ---------------------------------------- ##
/msg NickServ CERT LIST

## ------------------------------------------------------------------------------ ##
##    [+] Authorise your current certificate fingerprint:
## ------------------------------------------------------------------------------ ##
/msg NickServ CERT ADD [redacted]
```

```
>NickServ< CERT LIST
-NickServ- Fingerprint list for xe1phix:
-NickServ- - [redacted]
-NickServ- End of xe1phix fingerprint list.
```

```
>NickServ< CERT ADD [redacted]
-NickServ- Added fingerprint [redacted] to your fingerprint list.
```

[Hexchat-Add-Cheatsheet]

Accessing Libera.Chat Via Tor

Libera.Chat is reachable via Tor using our onion service.

Configuration requirements with details below:

- Update `torrc` configuration file to map to the onion service.
- Configure your client to use your Tor SOCKS proxy (typically `localhost:9050`).
- Configure public-key SASL authentication.
- Connect to `palladium.libera.chat`.

```
# torrc entry for libera.chat onion service
MapAddress palladium.libera.chat libera75jm6of4wxpxt4aynol3xjmbtxgfyjpu34ss4
```

This service requires public-key SASL authentication using either the `EXTERNAL` or `ECDSA-NIST256P-CHALLENGE` mechanisms. See our guide on setting up CertFP for more information.

Some clients lack SOCKS4a or later support. In this case you will need to change your `torrc` file to map a private IP address to the onion service address instead and disable TLS hostname verification in your client. Onion service names securely identify a service. The connection will still be secure.

I2P – IRC2P Protocols

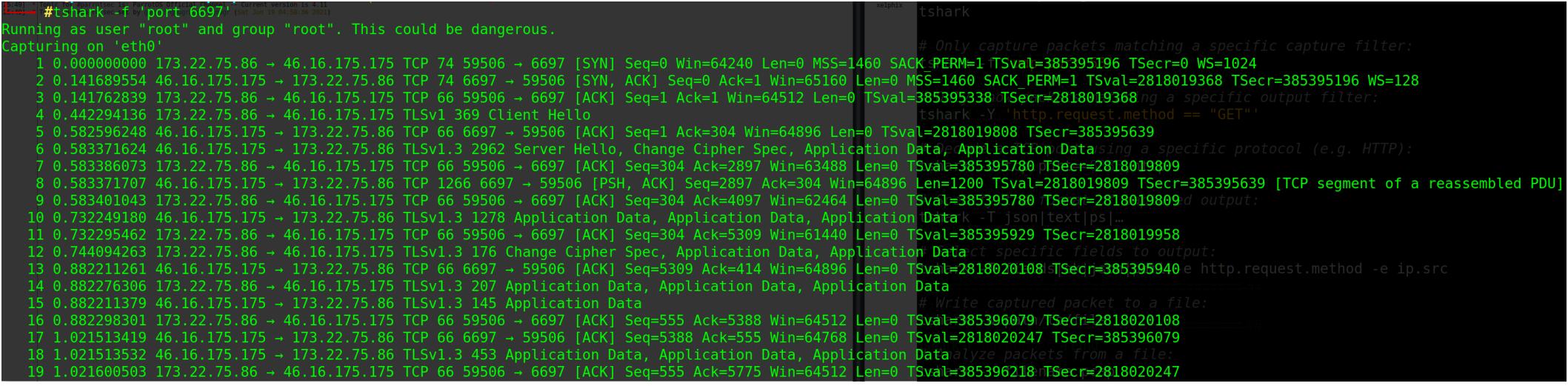
```
echo "      [+] I2pPostmanSTMP          127.0.0.1:7659
echo "      [+] I2pPostmanPop3          127.0.0.1:7660
echo "      [+] I2pIrc                  127.0.0.1:6668
echo "      [+] Irc2PPort               127.0.0.1:6668
```

```
##-============================-##
##    [+]  Darknet-[I2P]-IRC Servers
##-============================-##
##        >  irc.echelon.i2p
##        >  irc.postman.i2p
##        >  irc.killyourtv.i2p
##        >  irc.devfs.i2p
##        >  irc.dg.i2p
##        >  irc.undefined.i2p
##        >  irc.welterde.i2p
##-============================-##
```

IRC – I2P Connection Log

```
* Looking up 127.0.0.1
* Connecting to 127.0.0.1 (127.0.0.1) port 6669...
* Connected. Now logging in...
* *** Looking up your hostname...
* *** Found your hostname (cached)
* You have not registered
* You have not registered
* Welcome to the KillYourIRC IRC Network [redacted]!~[redacted]@6ntvfpuymt55avnvzalgl6k2mjarrtv7gn3btedcx74majtfhyaa.b32
  .i2p
* Your host is irc.killyourtv.i2p, running version Unreal3.2.9
* This server was created [redacted]
* irc.killyourtv.i2p Unreal3.2.9 [redacted]
* UHNAMES NAMESX SAFELIST HCN MAXCHANNELS=50 CHANLIMIT=#:50 MAXLIST=b:60,e:60,I:60 NICKLEN=30
  CHANNELLEN=32 TOPICLEN=307 KICKLEN=307 AWAYLEN=307 MAXTARGETS=20 :are supported by this server
* WALLCHOPS WATCH=128 WATCHOPTS=A SILENCE=15 MODES=12 CHANTYPES=# PREFIX=(qaohv)~&@%+
  CHANMODES=beI,kfL,lj,psmntirRcOAQKVCuzNSMTGZ NETWORK=KillYourIRC CASEMAPPING=ascii EXTBAN=~,qjncrR
  ELIST=MNUCT STATUSMSG=~&@%+ :are supported by this server
* EXCEPTS INVEX CMDS=KNOCK,MAP,DCCALLOW,USERIP :are supported by this server
* There are 1 users and 20 invisible on 2 servers
* 12 :operator(s) online
* 19 :channels formed
* I have 9 clients and 1 servers
* Current Local Users: 9  Max: 15
* Current Global Users: 21  Max: 21
```

IRC - TShark


#tshark -f 'port 6697'
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
1 0.000000000 173.22.75.86 → 46.16.175.175 TCP 74 59506 → 6697 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=385395196 TSecr=0 WS=1024
2 0.141689554 46.16.175.175 → 173.22.75.86 TCP 74 6697 → 59506 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2818019368 TSecr=385395196 WS=128
3 0.141762839 173.22.75.86 → 46.16.175.175 TCP 66 59506 → 6697 [ACK] Seq=1 Ack=1 Win=64512 Len=0 TSval=385395338 TSecr=2818019368
4 0.442294136 173.22.75.86 → 46.16.175.175 TLSv1 369 Client Hello
5 0.582596248 46.16.175.175 → 173.22.75.86 TCP 66 6697 → 59506 [ACK] Seq=1 Ack=304 Win=64896 Len=0 TSval=2818019808 TSecr=385395639
6 0.583371624 46.16.175.175 → 173.22.75.86 TLSv1.3 2962 Server Hello, Change Cipher Spec, Application Data, Application Data
7 0.583386073 173.22.75.86 → 46.16.175.175 TCP 66 59506 → 6697 [ACK] Seq=304 Ack=2897 Win=63488 Len=0 TSval=385395780 TSecr=2818019809
8 0.583371707 46.16.175.175 → 173.22.75.86 TCP 1266 6697 → 59506 [PSH, ACK] Seq=2897 Ack=304 Win=64896 Len=1200 TSval=2818019809 TSecr=385395639 [TCP segment of a reassembled PDU]
9 0.583401043 173.22.75.86 → 46.16.175.175 TCP 66 59506 → 6697 [ACK] Seq=304 Ack=4097 Win=62464 Len=0 TSval=385395780 TSecr=2818019809
10 0.732249180 46.16.175.175 → 173.22.75.86 TLSv1.3 1278 Application Data, Application Data, Application Data
11 0.732295462 173.22.75.86 → 46.16.175.175 TCP 66 59506 → 6697 [ACK] Seq=304 Ack=5309 Win=61440 Len=0 TSval=385395929 TSecr=2818019958
12 0.744094263 173.22.75.86 → 46.16.175.175 TLSv1.3 176 Change Cipher Spec, Application Data, Application Data
13 0.882211261 46.16.175.175 → 173.22.75.86 TCP 66 6697 → 59506 [ACK] Seq=5309 Ack=414 Win=64896 Len=0 TSval=2818020108 TSecr=385395940
14 0.882276306 173.22.75.86 → 46.16.175.175 TLSv1.3 207 Application Data, Application Data, Application Data
15 0.882211379 46.16.175.175 → 173.22.75.86 TLSv1.3 145 Application Data
16 0.882298301 173.22.75.86 → 46.16.175.175 TCP 66 59506 → 6697 [ACK] Seq=555 Ack=5388 Win=64512 Len=0 TSval=385396079 TSecr=2818020108
17 1.021513419 46.16.175.175 → 173.22.75.86 TCP 66 6697 → 59506 [ACK] Seq=5388 Ack=555 Win=64768 Len=0 TSval=2818020247 TSecr=385396079
18 1.021513532 46.16.175.175 → 173.22.75.86 TLSv1.3 453 Application Data, Application Data, Application Data
19 1.021600503 173.22.75.86 → 46.16.175.175 TCP 66 59506 → 6697 [ACK] Seq=555 Ack=5775 Win=64512 Len=0 TSval=385396218 TSecr=2818020247

Hexchat – Firejail Profile

```
include allow-python2.inc
include allow-python3.inc

# Allow perl (blacklisted by disable-interpreters.inc)
include allow-perl.inc

include disable-common.inc
include disable-devel.inc
include disable-exec.inc
include disable-interpreters.inc
include disable-passwdmgr.inc
include disable-programs.inc
include disable-shell.inc
include disable-xdg.inc

mkdir ${HOME}/.config/hexchat
whitelist ${HOME}/.config/hexchat
include whitelist-common.inc
include whitelist-var-common.inc
netfilter
no3d
nodvd
nogroups
nonewprivs
noroot
notv
nou2f
novideo
protocol unix,inet,inet6
seccomp
shell none
tracelog

disable-mnt
# debug note: private-bin requi
 some systems
private-bin hexchat,python*
private-dev
#private-lib - python problems
private-tmp
```

OnionShare lets you anonymously share files, or receive files. From version 2.2, OnionShare can also be used as a webserver through which static HTML files can be served. OnionShare has four different operating modes: the default for sharing files, receive mode for receiving files, public mode for sharing files with a large audience, and website mode.

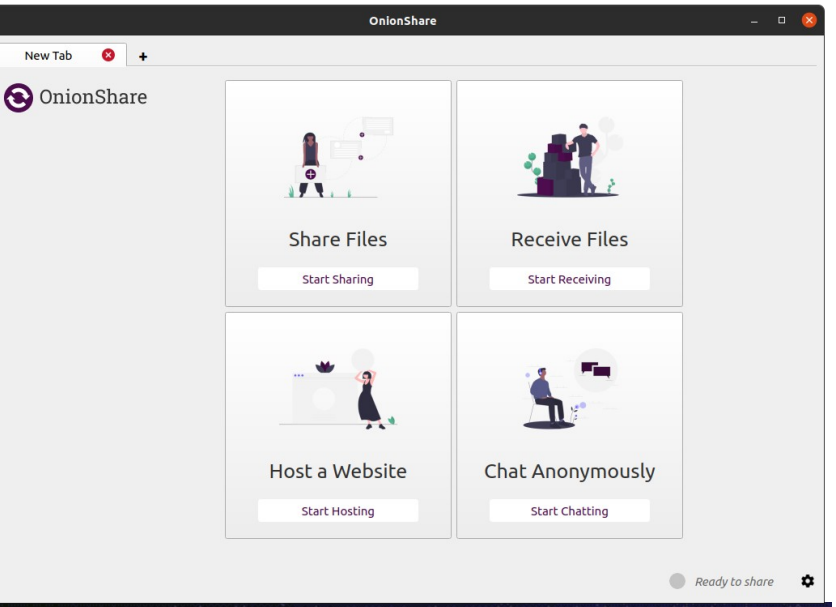
Public mode In case of sharing the unguessable URL with many peers, please enable "Public mode", otherwise OnionShare might interpret multiple accesses as an attack and shut down the hidden service.

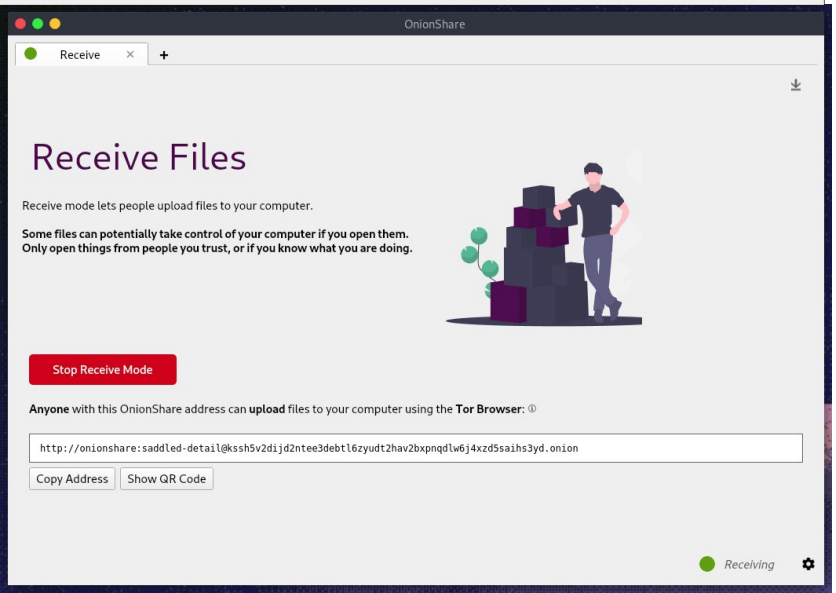
Receive Mode lets people upload files to your OnionShare via Tor Browser.

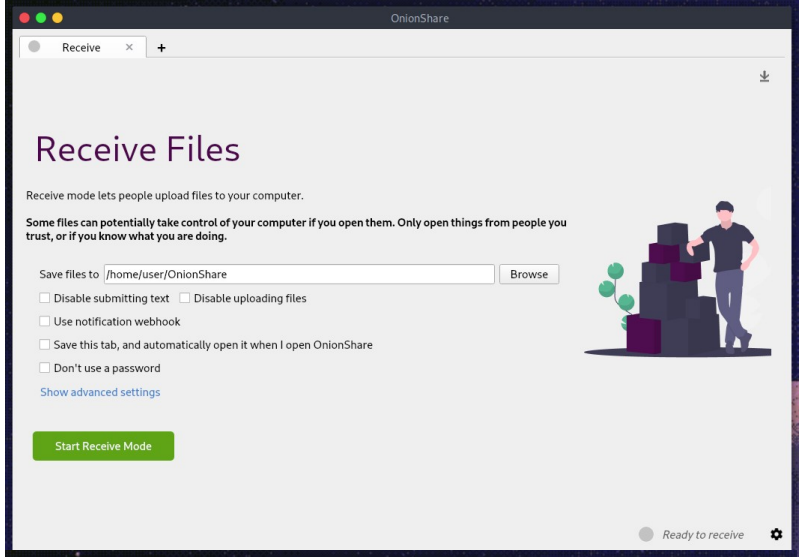
Website Mode OnionShare allows you to publish a website as an onion service by sharing an entire path to a local directory. When the directory contains an index.html file (along with any static files such as images, CSS etc), then the site will be viewable on the Tor network as an onion service as the actual website itself (not as an OnionShare interface). The onion service will render HTML in Tor Browser just like any other website. If you don't have an index.html, OnionShare will show a directory listing of your files instead. In directory listing, each file can be opened, and there is no download button for the compressed archive. OnionShare does not share your original website files. Instead, it creates a copy of those files in a temporary folder and shares those copies.

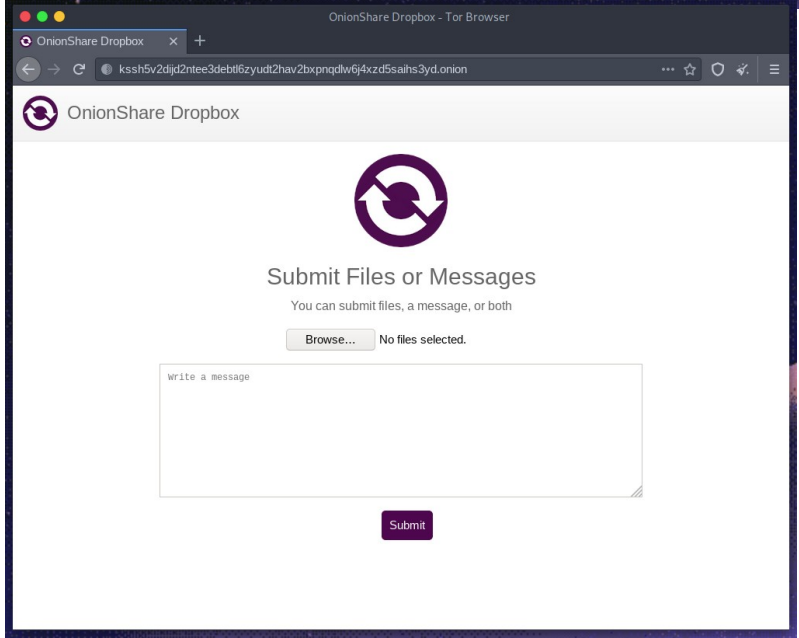
When using OnionShare, a web server is started, making OnionShare accessible as a Tor Onion Service, over the Internet. An unguessable address is generated and can be shared for the recipient to open in a tor client, such as Tor Browser, to download, upload, or consult files. Traffic between their device and your onion service end-to-end encrypted. No separate server or third party file-sharing service required. You host the files on your own computer, acting as a server, for as long as the program is running.

OnionShare
Receive Files

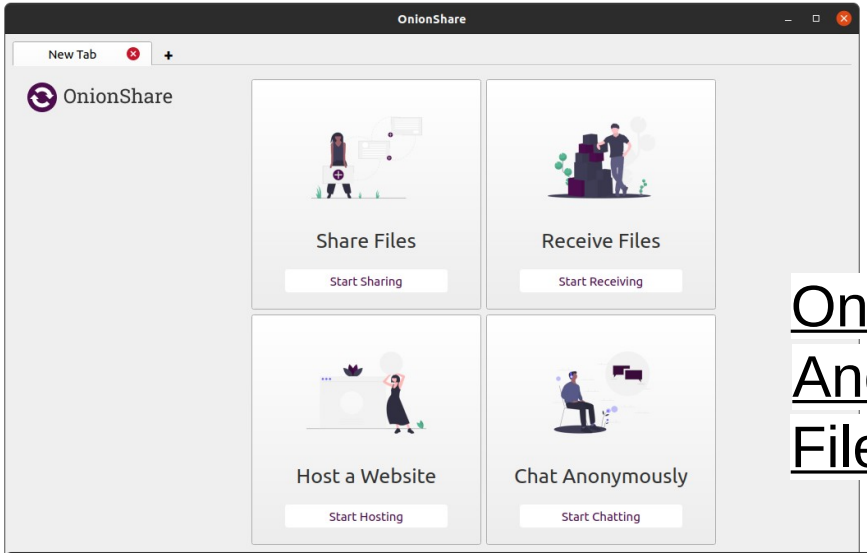


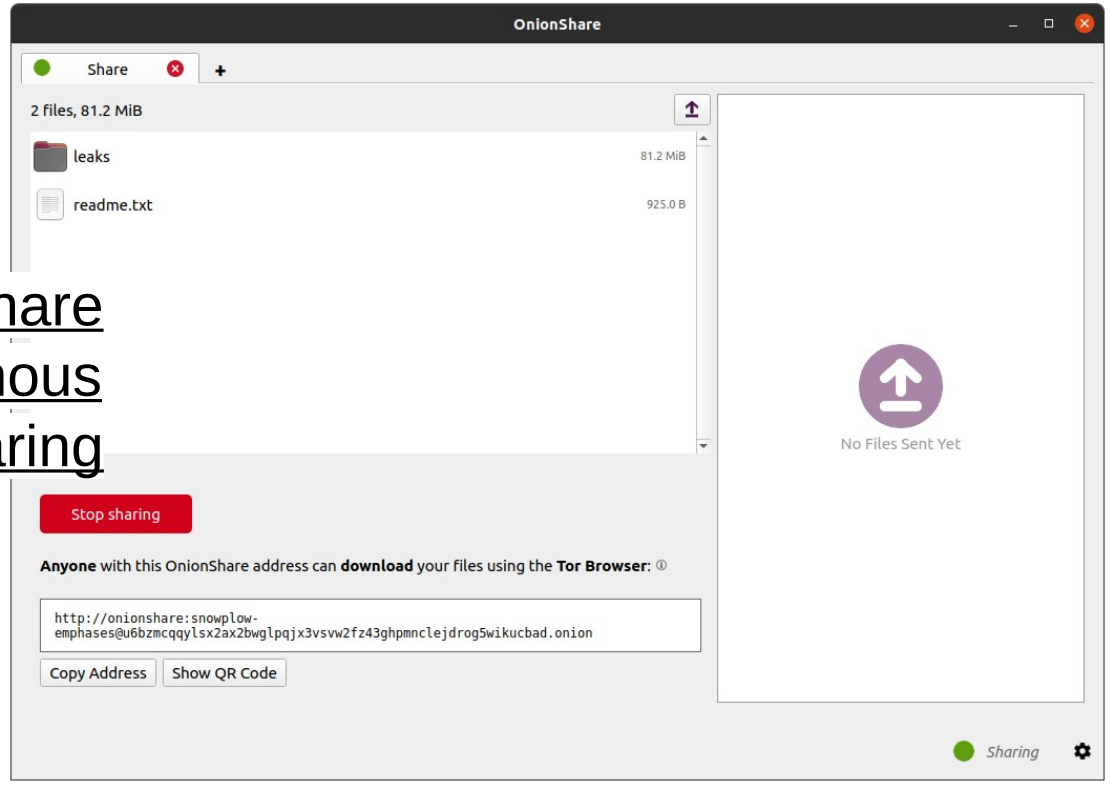


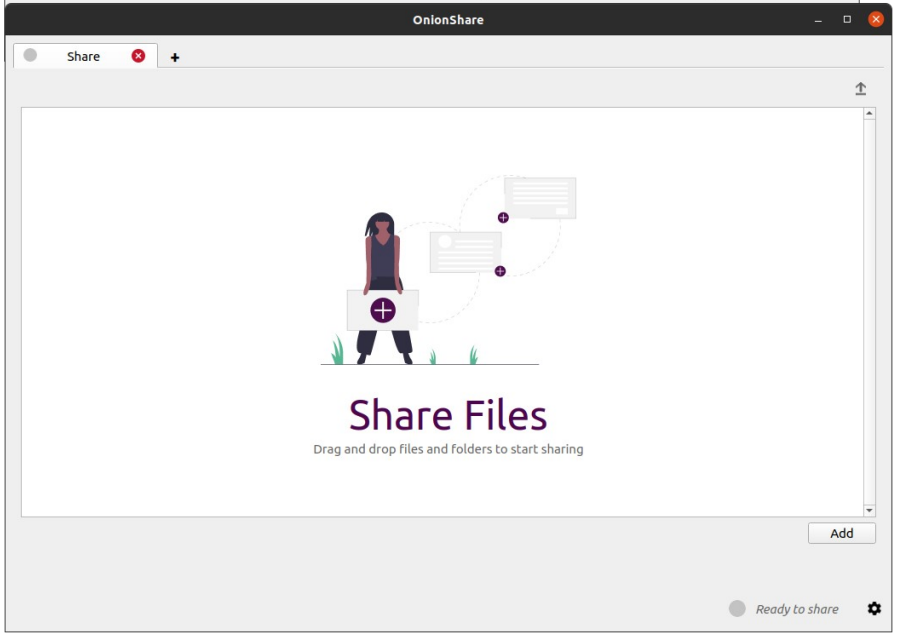


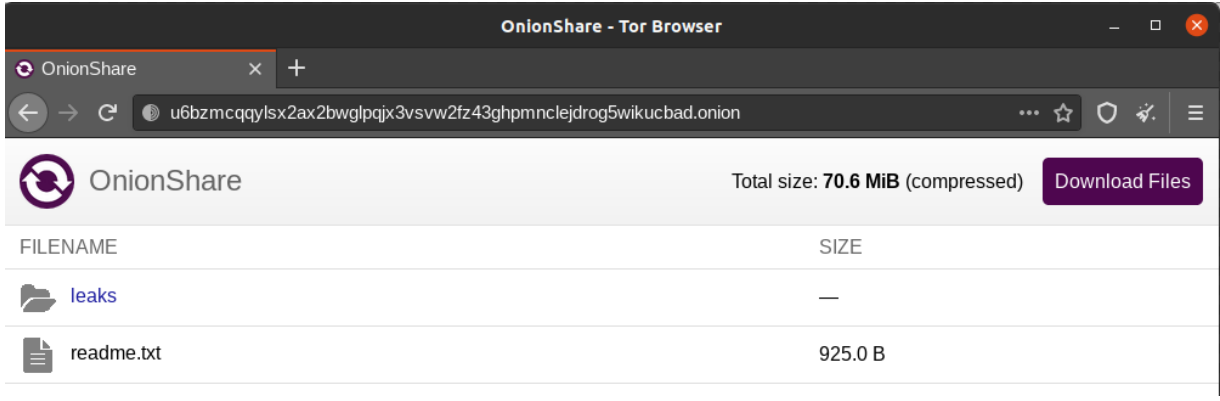


OnionShare
Anonymous
File Sharing

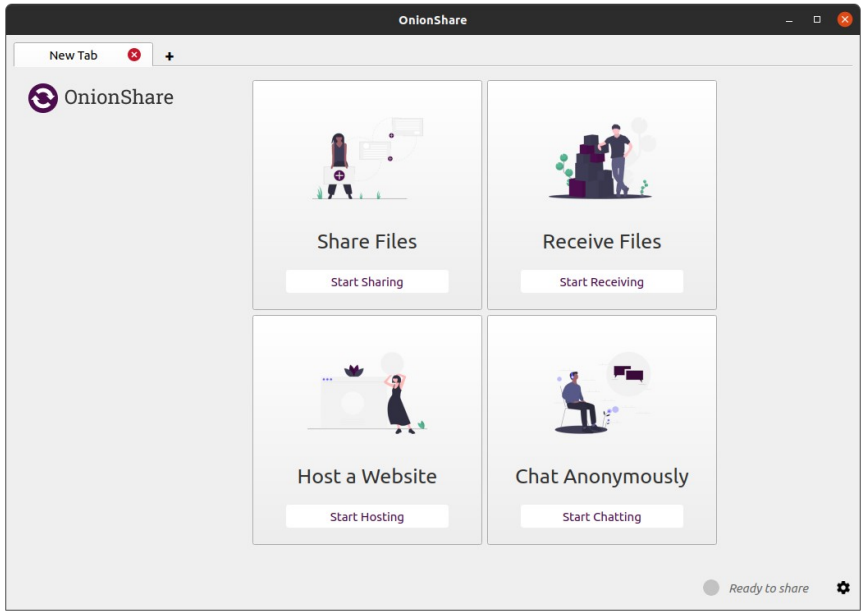


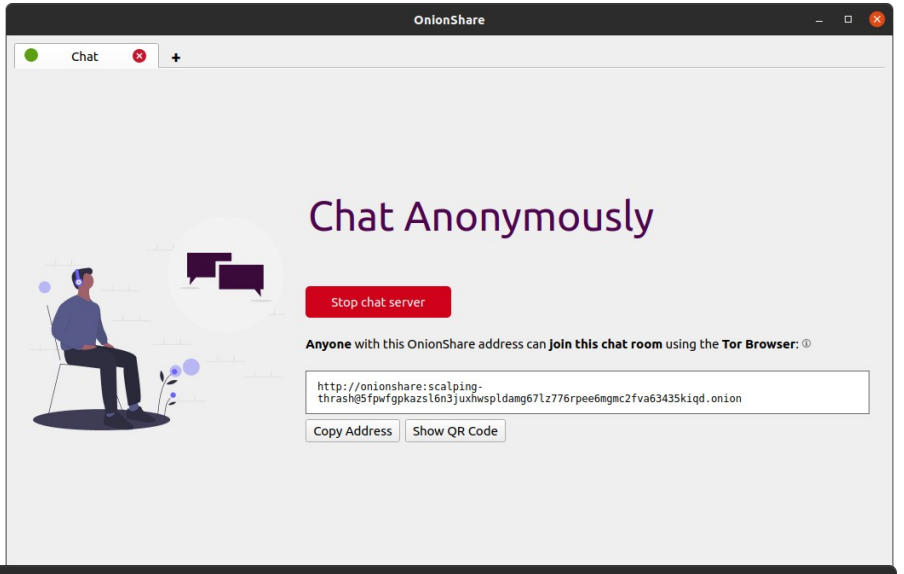


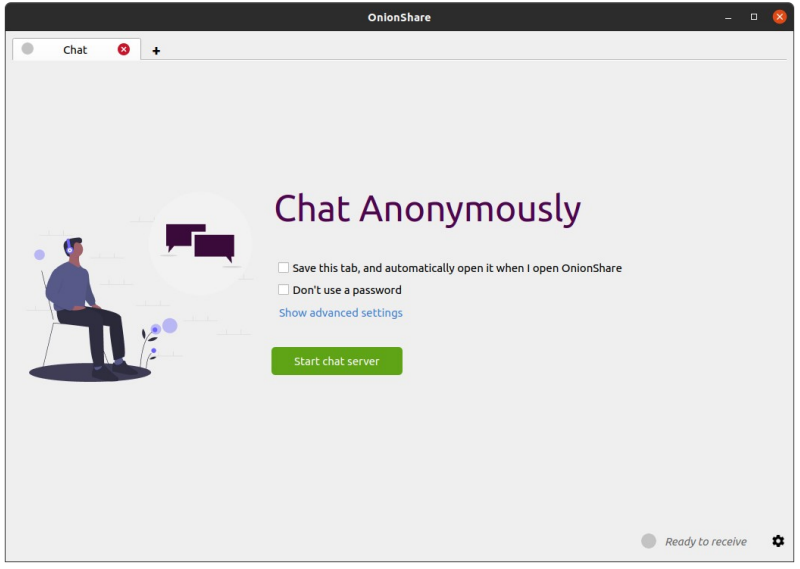


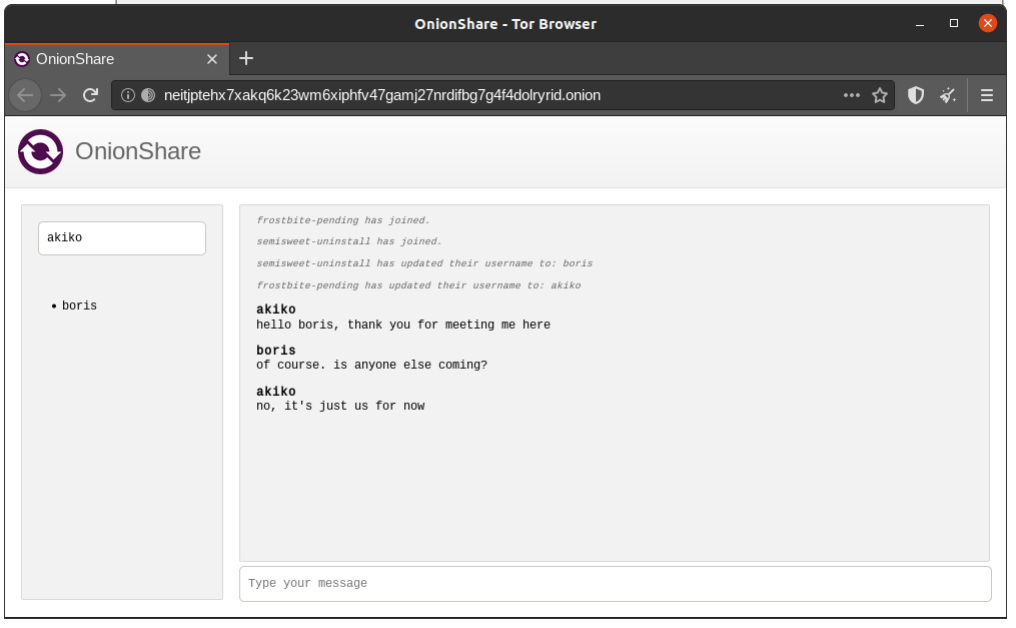


OnionShare Anonymous Chat

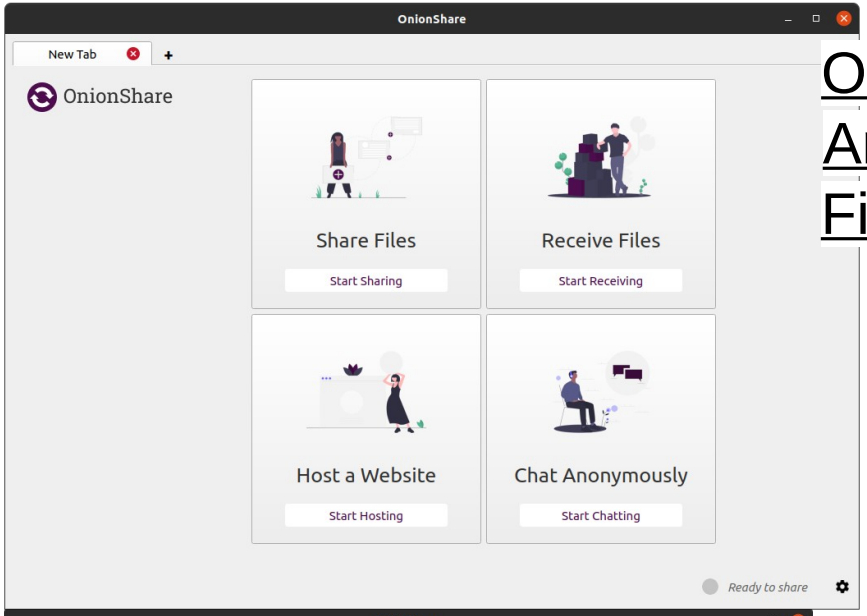


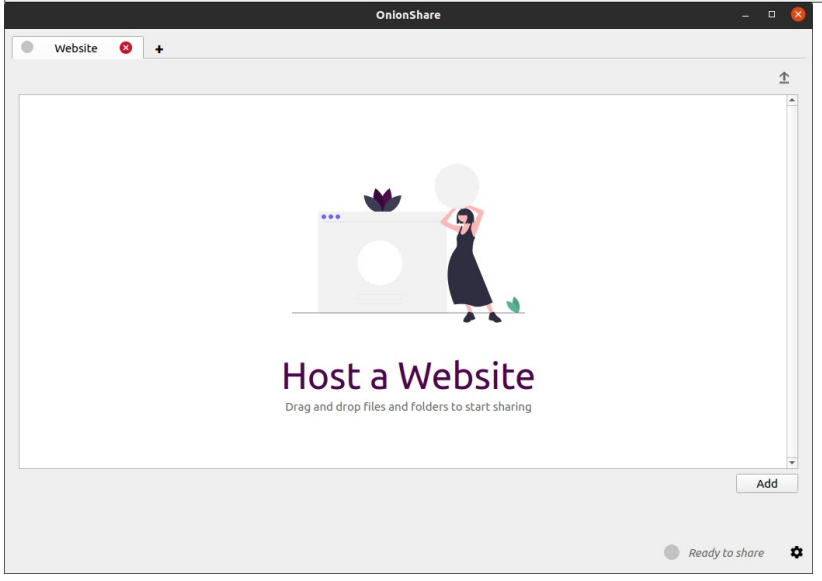


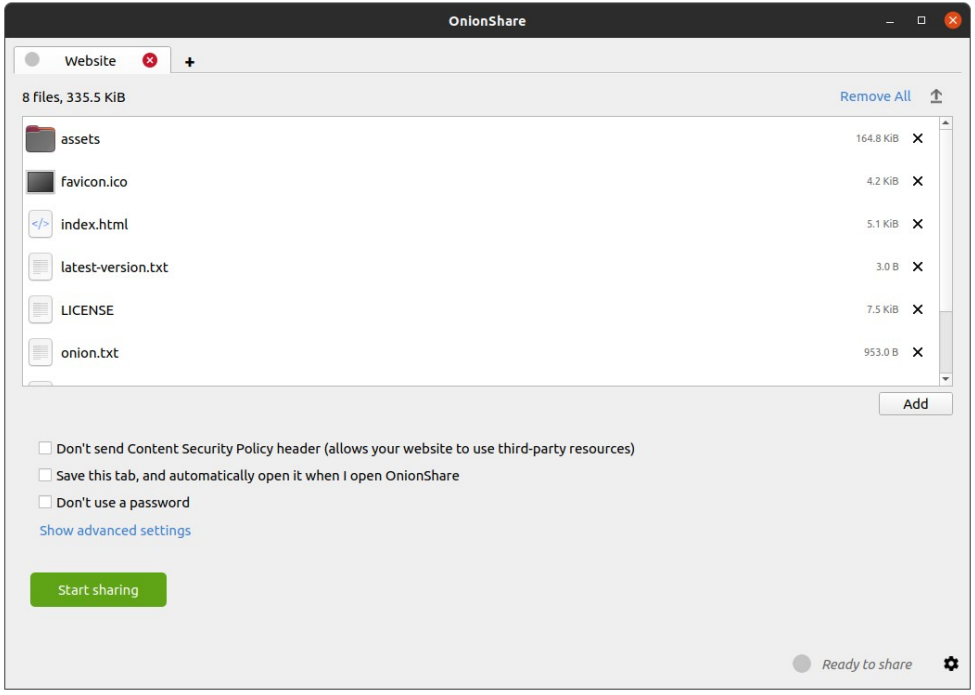




OnionShare Anonymous File Sharing







OnionShare – Firejail Profile

```
caps.drop all
ipc-namespace
netfilter
no3d
nodvd
nogroups
nonewprivs
noroot
nosound
notv
nou2f
novideo
protocol unix,inet
seccomp
shell none

private-dev
private-tmp

memory-deny-write-execute
```

Ricochet – Tor Messenger

Anonymous instant messaging for **real** privacy

Ricochet is a different approach to instant messaging that **doesn't trust anyone** in protecting your privacy.

- *Eliminate metadata*. Nobody knows who you are, who you talk to, or what you say.
- *Stay anonymous*. Share what you want, without sharing your identity and location.
- *Nobody in the middle*. There are no servers to monitor, censor, or hack.
- *Safe by default*. Security isn't secure until it's automatic and easy to use.

How it works

Ricochet uses the **Tor network** to reach your contacts without relying on messaging servers. It creates a **hidden service**, which is used to rendezvous with your contacts without revealing your location or IP address.

Overview

Ricochet is a peer-to-peer instant messaging system built on anonymity networks. This document defines the communication protocol between two Ricochet instances, as carried out over a Tor hidden service connection.

The protocol is defined in three layers:

The **connection layer** describes the use of an anonymized TCP-style connection for peer-to-peer communication.

The **packet layer** separates the connection into a series of *packets* delivered to *channels*. This allows multiplexing different operations on the same connection, and packetizes data for channel-level parsing.

The **channel layer** parses and handles packets according to the *channel type* and the state of that specific channel.

Hidden services

Ricochet uses Tor hidden services as a transport; the reader should be familiar with that architecture and the properties it provides. In particular:

- The hostname is calculated from a hash of the server's public key, and serves to authenticate the server without relying on a third party
- Connections are encrypted end-to-end, using the server's key and a DHE handshake to provide forward secrecy
- Both ends of a connection are anonymous in that neither peer should be able to identify or locate the other, and no relay should be able to connect an identity to the requests it makes
- Impersonating a server without its private key requires an 80-bit SHA1 collision using a valid RSA key

Ricochet – AppArmor Profile

```
# Allow Ricochet to exec tor
/usr/bin/tor rix,
# Tor in turn needs various
/usr/share/tor/geoip  r,
/usr/share/tor/geoip6 r,
/proc/sys/kernel/random/uuid r,
/sys/devices/system/cpu/ r,
# Allow Ricochet to read tor daemons auth cookie
/run/tor/control.authcookie r,

# Allow Ricochet to read itself
/usr/bin/ricochet r,
/proc/[0-9]*/cmdline r,
/proc/[0-9]*/environ r,
# Allow Ricochet to look up all your machine's PII
# Why does it need this stuff? BAD NEWS BEARS
/etc/machine-id r,
/var/lib/dbus/machine-id r,

owner @{HOME}/.local/share/Ricochet/ rw,
owner @{HOME}/.local/share/Ricochet/** mrwk,
```

Ricochet – Firejail Profile

```
caps.drop all
ipc-namespace
netfilter
no3d
nodvd
nogroups
nonewprivs
noroot
notv
nou2f
novideo
protocol unix,inet,inet6
seccomp
shell none

disable-mnt
private-bin ricochet,tor
private-dev
#private-etc alternatives,alternatives,ca-certificates,
crypto-policies,fonts,pki,ssl,tor,X11
```

Journalctl

```
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #systemctl list-unit-files --type=service |  grep enabled
networking.service                          disabled        enabled
NetworkManager-dispatcher.service           enabled         enabled
NetworkManager-wait-online.service          enabled         enabled
NetworkManager.service                      enabled         enabled
┌──[✗]─[root@parrot]─[/home/parrotsec-kiosk]
└─── #journalctl --since "yesterday"
```

```
└─── #journalctl -u NetworkManager.service
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #journalctl -u openvpn.service
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #journalctl -u systemd-resolved.service
┌──[root@parrot]─[/home/parrotsec-kiosk]
└─── #journalctl --disk-usage
Archived and active journals take up 4.1G in the file system.
```

```
## Display 500 most recent journalctl entries (with paging)
journalctl -n 500
```

Journalctl Cheatsheet

```
journalctl --list-boots | head          ## ------------------------------------------------ ##
                                        ## Show a list of IDs, and the timestamps of boots
                                        ## ------------------------------------------------ ##
journalctl -k                           ## kernel messages
                                        ## ------------------------------------------------ ##
journalctl -k -f                        ## Follow kernel messages
                                        ## ------------------------------------------------ ##
journalctl -u NetworkManager.service    ## Show messages for the unit NetworkManager
                                        ## ------------------------------------------------ ##
journalctl -f -u NetworkManager.service ## Follow service
                                        ## ------------------------------------------------ ##
journalctl -u httpd.service             ## Show messages for the unit httpd
                                        ## ------------------------------------------------ ##
journalctl -k -b -1                     ## view the boot logs
                                        ## ------------------------------------------------ ##
journalctl /dev/sda                     ## All logs of the kernel device node `/dev/sda`
                                        ## ------------------------------------------------ ##
journalctl -u systemd-networkd          ## Show messages for the unit systemd-networkd
                                        ## ------------------------------------------------ ##
journalctl -u auditd.service            ## Show messages for the unit auditd
                                        ## ------------------------------------------------ ##
journalctl --list-boots                 ## check only boot messages
                                        ## ------------------------------------------------ ##
journalctl -b $BootID                   ## Show boot messages for a selected boot ID
                                        ## ------------------------------------------------ ##
```

Syslog Cheatsheet

/etc/syslog.conf

```
# facility.level                    action
*.info;mail.none;authpriv.none      /var/log/messages
authpriv.*                          /var/log/secure
mail.*                              /var/log/maillog
*.alert                             root
*.emerg                             *
local5.*                            @10.7.7.7
local7.*                            /var/log/boot.log
```

Facility Creator of the message	**Level** Severity of the message	**Action** Destination of the message	
`auth` or `security`†	`emerg` or `panic`† (highest)	`filename`	message is written into a logfile
`authpriv`	`alert`	`@hostname`	message is sent to a logger server (via UDP port 514)
`cron`	`crit`	`user1,user2,user3`	message is sent to users' consoles
`daemon`	`err` or `error`†	`*`	message is sent to all logged-in users' consoles
`kern`	`warning` or `warn`†		
`lpr`	`notice`		
`mail`	`info`		
`mark` (for syslog internal use)	`debug` (lowest)		
`news`			
`syslog`	`none` (facility disabled)		
`user`			
`uucp`			
`local0` ... `local7` (custom)			

[Xe1phix-[Syslog]-Cheatsheet]

Lsof Cheatsheet 2

Show process that use internet connection at the moment

```
lsof -P -i -n
```

Show process that use specific port number

```
lsof -i tcp:443
```

Lists all listening ports together with the PID of the associated process

```
lsof -Pan -i tcp -i udp
```

List all open ports and their owning executables

```
lsof -i -P | grep -i "listen"
```

Show all open ports

```
lsof -Pnl -i
```

```
└─ #lsof +D /var/log
COMMAND      PID USER   FD   TYPE DEVICE  SIZE/OFF   NODE NAME
systemd-j    466 root   63u   REG   0,26 134217728 885160 /var/log/journal/52e38e44c61345b7b00c11e48e79f827/system.journal
systemd-j    466 root   66u   REG   0,26   8388608 885161 /var/log/journal/52e38e44c61345b7b00c11e48e79f827/user-1000.journal
fail2ban-   1073 root    3w   REG   0,26      2338 725109 /var/log/fail2ban.log
lightdm     1074 root    6w   REG   0,26      5750 777051 /var/log/lightdm/lightdm.log
Xorg        1091 root    1w   REG   0,26     17757 777052 /var/log/lightdm/x-0.log
Xorg        1091 root    2w   REG   0,26     17757 777052 /var/log/lightdm/x-0.log
Xorg        1091 root    6w   REG   0,26     69772 777055 /var/log/Xorg.0.log
mullvad-d 436402 root    3w   REG   0,26      5920 884127 /var/log/mullvad-vpn/daemon.log
```

```
└─ #lsof -iTCP -sTCP:ESTABLISHED
COMMAND     PID          USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
Telegram   2560 parrotseckiosk   42u  IPv4 5786431      0t0  TCP 192.168.1.101:36390->149.154.175.51:https (ESTABLISHED)
Telegram   2560 parrotseckiosk   47u  IPv4 6757561      0t0  TCP 192.168.1.101:51636->149.154.175.50:https (ESTABLISHED)
Telegram   2560 parrotseckiosk   57u  IPv4 6128828      0t0  TCP 192.168.1.101:37756->149.154.175.51:https (ESTABLISHED)
firefox  582761 parrotseckiosk  122u  IPv4 6967670      0t0  TCP 192.168.1.101:33956->ec2-35-155-44-228.us-west-2.compute.amazonaws.com:https (ESTABLISHED)
firefox  582761 parrotseckiosk  144u  IPv4 7188861      0t0  TCP 192.168.1.101:38414->185-70-42-42.protonmail.ch:https (ESTABLISHED)
firefox  582761 parrotseckiosk  169u  IPv4 7271230      0t0  TCP 192.168.1.101:43238->ec2-18-200-77-145.eu-west-1.compute.amazonaws.com:https (ESTABLISHED)
firefox  582761 parrotseckiosk  170u  IPv4 7271235      0t0  TCP 192.168.1.101:43240->ec2-18-200-77-145.eu-west-1.compute.amazonaws.com:https (ESTABLISHED)
firefox  582761 parrotseckiosk  312u  IPv4 7217745      0t0  TCP 192.168.1.101:33368->ec2-54-232-131-104.sa-east-1.compute.amazonaws.com:https (ESTABLISHED)
```

```
└─ #lsof -c Telegram
COMMAND   PID          USER   FD      TYPE             DEVICE  SIZE/OFF   NODE NAME
Telegram 2560 parrotseckiosk  cwd       DIR               0,36        52  11522 /home/parrotseckiosk/.local/share/TelegramDesktop
Telegram 2560 parrotseckiosk  rtd       DIR               0,26       312    256 /
Telegram 2560 parrotseckiosk  txt       REG               0,36 105740848 313881 /home/parrotseckiosk/Downloads/Telegram/Telegram
Telegram 2560 parrotseckiosk  mem       REG               0,24           313881 /home/parrotseckiosk/Downloads/Telegram/Telegram
```

```
└─ #lsof -p $(pgrep firefox)
COMMAND    PID          USER   FD      TYPE             DEVICE SIZE/OFF    NODE NAME
firefox 582761 parrotseckiosk  cwd       DIR               0,65      300       2 /home/parrotseckiosk
firefox 582761 parrotseckiosk  rtd       DIR               0,26      312     256 /
firefox 582761 parrotseckiosk  txt       REG               0,26   658808   23747 /usr/lib/firefox/firefox
firefox 582761 parrotseckiosk  mem       REG               0,24            23747 /usr/lib/firefox/firefox
```

```
└─ #lsof -p $(pgrep qbittorrent)
COMMAND     PID          USER   FD      TYPE             DEVICE SIZE/OFF   NODE NAME
qbittorre 11156 parrotseckiosk  cwd       DIR              0,133      240      2 /home/parrotseckiosk
qbittorre 11156 parrotseckiosk  rtd       DIR               0,26      312    256 /
qbittorre 11156 parrotseckiosk  txt       REG              0,115  8911176     38 /usr/bin/qbittorrent
```

Fuser Cheatsheet

```
## ------------------------------------------------------------------------------------------------ ##
    fuser /var/log/daemon.log       ## Show which processes use the file
## ------------------------------------------------------------------------------------------------ ##
    fuser -v /home/$User            ## Show which processes are used in the home directory
## ------------------------------------------------------------------------------------------------ ##
    fuser -ki $File                 ## Kills a process that is locking a file
## ------------------------------------------------------------------------------------------------ ##
    fuser -k -HUP $File             ## Kills a process with specific signal
## ------------------------------------------------------------------------------------------------ ##
    fuser -v 53/udp                 ## Show what PID is listening on specific port
## ------------------------------------------------------------------------------------------------ ##
    fuser -mv /var/www              ## Show all processes using the named filesystems or block device
## ------------------------------------------------------------------------------------------------ ##
```

Xe1phix-[FUser]-Cheatsheet

Kill Cheatsheet

```
killall -1 $Service
kill -SIGHUP $PID
pkill -9 $Service
kill -SIGKILL $PID
killall -9 $Service
kill -9 $$
kill -s TERM $PID
pkill -TERM -u $User
kill -SIGTERM $PID
fuser -k -TERM -m /home
```

```
## ------------------------------------------------------ ##
## [ 1] HUP (hang up)
## ------------------------------------------------------ ##
## [ 1] HUP (hang up)
## ------------------------------------------------------ ##
## [ 9] KILL (noncatchable, nonignorable
## ------------------------------------------------------ ##
## [ 9]  SIGKILL (Kill signal)
## ------------------------------------------------------ ##
## [ 9]  SIGKILL (Kill signal)
## ------------------------------------------------------ ##
## [ 9] Kill current session
## ------------------------------------------------------ ##
## [15] TERM (software termination signal)
## ------------------------------------------------------ ##
## [15] TERM (software termination signal)
## ------------------------------------------------------ ##
## [15] SIGTERM (Termination signal.)
## ------------------------------------------------------ ##
## [15] kill every process accessing /home (to umount)
## ------------------------------------------------------ ##
```

PID Cheatsheet

```
## ------------------------------------------------------------------------------------------ ##
    pgrep -u $User          ## Find the process ID of user
## ------------------------------------------------------------------------------------------ ##
    pcat -v $PID            ## Displays the location of each memory region that is being copied
## ------------------------------------------------------------------------------------------ ##
    pmap -d $PID            ## Provide Libraries loaded by a running process with pmap
## ------------------------------------------------------------------------------------------ ##
    pidstat -p $PID         ## Gather resource consumption details for a specific target process
## ------------------------------------------------------------------------------------------ ##
```

Systemd Cheatsheet

Action	Command	Note
Analyzing the system state		
Show system status	`systemctl status`	
List running units	`systemctl` or `systemctl list-units`	
List failed units	`systemctl --failed`	
List installed unit files[1]	`systemctl list-unit-files`	
Show process status for a PID	`systemctl status pid`	**cgroup slice**, memory and parent
Checking the unit status		
Show a manual page associated with a unit	`systemctl help unit`	as supported by the unit
Status of a unit	`systemctl status unit`	including whether it is running or not
Check whether a unit is enabled	`systemctl is-enabled unit`	
Starting, restarting, reloading a unit		
Start a unit immediately	`systemctl start unit` as root	
Stop a unit immediately	`systemctl stop unit` as root	
Restart a unit	`systemctl restart unit` as root	
Reload a unit and its configuration	`systemctl reload unit` as root	
Reload systemd manager configuration[2]	`systemctl daemon-reload` as root	scan for new or changed units
Enabling a unit		
Enable a unit to start automatically at boot	`systemctl enable unit` as root	
Enable a unit to start automatically at boot and **start** it immediately	`systemctl enable --now unit` as root	
Disable a unit to no longer start at boot	`systemctl disable unit` as root	
Reenable a unit[3]	`systemctl reenable unit` as root	i.e. disable and enable anew
Masking a unit		
Mask a unit to make it impossible to start[4]	`systemctl mask unit` as root	
Unmask a unit	`systemctl unmask unit` as root	

Systemctl + Service Cheatsheet

```
chkconfig --list
service --status-all | grep '+'
service --status-all | grep running
service --status-all | grep running... | sort
systemctl list-units | grep .service
systemctl list-units | grep .target
systemctl list-unit-files --type=service
systemctl list-unit-files --type=target
systemctl list-unit-files --type service --state enabled
systemctl list-unit-files --type service --state running
systemctl list-unit-files --type=service | grep -v disabled
systemctl list-unit-files --type=service | grep -v masked
```

STrace Cheatsheet

```
## ------------------------------------------------------------------------------------ ##
   strace -T                        ## Display syscall duration In the output
## ------------------------------------------------------------------------------------ ##
   strace -c                        ## See what time is spend and where
## ------------------------------------------------------------------------------------ ##
   strace -f                        ## Track process including forked child processes
## ------------------------------------------------------------------------------------ ##
   strace -e open                   ## Monitor Opening of Files
## ------------------------------------------------------------------------------------ ##
   strace -P $Path                  ## Track a process when interacting with a path
## ------------------------------------------------------------------------------------ ##
   strace -o $File.txt              ## Log strace output to a file
## ------------------------------------------------------------------------------------ ##
   strace -e trace=$File -p $PID    ## Trace File Activity
## ------------------------------------------------------------------------------------ ##
   strace -e trace=$Desc -p $PID    ## Trace File Descriptor
## ------------------------------------------------------------------------------------ ##
   strace -e trace=$Ipc             ## Track communication between processes (IPC)
## ------------------------------------------------------------------------------------ ##
   strace -e trace=$Signal          ## Track process signal handling (like HUP, exit)
## ------------------------------------------------------------------------------------ ##
   strace -e trace=$File            ## Track file related syscalls
## ------------------------------------------------------------------------------------ ##
   strace -e trace=process          ## Track process calls (like fork, exec)
## ------------------------------------------------------------------------------------ ##
   strace -e trace=memory           ## Track Memory syscalls
## ------------------------------------------------------------------------------------ ##
   strace -e trace=network          ## Track Network syscalls
## ------------------------------------------------------------------------------------ ##
```

SS – Socket Statistics

```
┌─[root@parrotseckiosk-optiplex990]─[/home/parrotseckiosk/Downloads/PDFs/Archive.org/LinuxBooks]
└──╼ #ss -o state established '( dport = :https or sport = :https )'
Netid  Recv-Q   Send-Q     Local Address:Port       Peer Address:Port   Process
tcp    0        0          192.168.1.101:34252      185.70.42.42:https
tcp    0        0          192.168.1.101:47300     52.37.190.150:https   timer:(keepalive,5min15sec,0)
tcp    0        0          192.168.1.101:38692     89.187.183.13:https
tcp    0        0          192.168.1.101:57918      3.211.86.134:https   timer:(keepalive,7min9sec,0)
 192.168.1.101:42748        149.154.175.51:https     users:(("Telegram",pid=2560,fd=59))
 192.168.1.101:33266       142.250.191.206:https     users:(("firefox",pid=582761,fd=153))
 192.168.1.101:33956         35.155.44.228:https     users:(("firefox",pid=582761,fd=122))
 192.168.1.101:40238       173.194.162.198:https     users:(("firefox",pid=582761,fd=233))
 192.168.1.101:33368        54.232.131.104:https     users:(("firefox",pid=582761,fd=312))
 192.168.1.101:54940        149.154.175.50:https     users:(("Telegram",pid=2560,fd=46))
 192.168.1.101:38414          185.70.42.42:https     users:(("firefox",pid=582761,fd=144))
 192.168.1.101:42674        149.154.175.51:https     users:(("Telegram",pid=2560,fd=42))
```

```
 #ss -tlup --ipv4
     Local Address:Port                  Peer Address:Port          Process
          0.0.0.0:5355                       0.0.0.0:*              users:(("systemd-resolve",pid=711,fd=11))
     127.0.0.53%lo:domain                    0.0.0.0:*              users:(("systemd-resolve",pid=711,fd=16))
192.168.1.101%eth0:49213                     0.0.0.0:*              users:(("qbittorrent",pid=11156,fd=26))
      127.0.0.1%lo:49213                     0.0.0.0:*              users:(("qbittorrent",pid=11156,fd=23))
     127.0.0.53%lo:domain                    0.0.0.0:*              users:(("systemd-resolve",pid=711,fd=17))
192.168.1.101%eth0:49213                     0.0.0.0:*              users:(("qbittorrent",pid=11156,fd=25))
      127.0.0.1%lo:49213                     0.0.0.0:*              users:(("qbittorrent",pid=11156,fd=20))
          0.0.0.0:5355                       0.0.0.0:*              users:(("systemd-resolve",pid=711,fd=12))
```

PS Cheatsheet

```
##-==========================================-##
##   [+] show top 10 process eating memory
##-==========================================-##
ps auxf | sort -nr -k 4 | head -10
```

```
##-==========================================-##
##   [+] show top 10 process eating CPU
##-==========================================-##
ps auxf | sort -nr -k 3 | head -10
```

```
##-==============================================-##
##   [+] Print User, Service And Num of Processes
##-==============================================-##
ps -ef | awk '{print $1}' | sort | uniq -c | sort -nr
```

```
##-==========================================-##
##   [+] Find Processes Being Run As Root
##-==========================================-##
ps -ef | awk '$1 == "root" && $6 != "?" {print}'
```

Xe1phix–[PS]-Cheatsheet

```
##-----------------------------------------##
##   [+] show top 10 process eating memory
##-----------------------------------------##
ps auxf | sort -nr -k 4 | head -10


##-----------------------------------------##
##   [+] show top 10 process eating CPU
##-----------------------------------------##
ps auxf | sort -nr -k 3 | head -10


##-----------------------------------------##
##   [+] Show Every Process Running As Root
##-----------------------------------------##
ps -U root -u root u


##-------------------------------------##
##   [+] List All Threads For A process:
##-------------------------------------##
ps -C firefox -L -o pid,tid,pcpu,state
```

```
##-------------------------------------------------##
##   [+] Print User, Service And Num of Processes
##-------------------------------------------------##
ps -ef | awk '{print $1}' | sort | uniq -c | sort -nr


##-------------------------------------##
##   [+] Print The Process ID of Rsyslogd
##-------------------------------------##
ps -C rsyslogd -o pid


##-----------------------------##
##   [+] Print sshd Daemon PID
##-----------------------------##
ps -ef | awk '/sshd/ {print $2}'


##-------------------------------------##
##   [+] Find The sshd Daemon, And Kill It
##-------------------------------------##
kill $(ps -ef | awk '/sshd/ {print $2}')


##-----------------------------------------##
##   [+] PSTree - Graphical list of processes
##-----------------------------------------##
pstree --arguments --show-pids --show-pgids --show-parents


##-------------------------------------------------##
##   [+] Print User, Service And Num of Processes
##-------------------------------------------------##
ps -ef | awk '{print $1}' | sort | uniq -c | sort -nr
```

```
##-------------------------------------##
##   [+] Find Processes Being Run As Root
##-------------------------------------##
ps -ef | awk '$1 == "root" && $6 != "?" {print}'


##-------------------------------##
##   [+] Indepth Group Statistics
##-------------------------------##
ps -eo pid,command,size,vsize,%mem,gid,sgid,egid,fgid,sgroup,rgroup,group,fgroup,egroup,tpgid,tgid,flags


##-----------------------------------------------##
##   [+] Stack Statistics, esp eip nwchan etc...
##-----------------------------------------------##
ps -eo pid,uid,user,command,vsize,esp,eip,stackp,nwchan,lwp,psr,nlwp,flags


##-------------------------------------##
##   [+] Verbose User & UID Statistics
##-------------------------------------##
ps -eo uid,fuid,suid,ruid,euid,fuser,suser,user,uname,ruser,euser,command


##-------------------------------------##
##   [+] Verbose Mem & Cpu Statistics
##-------------------------------------##
ps -eo uid,gid,user,group,command,size,vsize,sz,%mem,%cpu,flags
##-------------------------------------------------##
##   [+] Display any tcp connections to apache
##-------------------------------------------------##
for i in `ps aux | grep httpd | awk '{print $2}'`; do lsof -n -p $i | grep ESTABLISHED; done;
```

[Xe1phix-[PS]-Cheatsheet]

```
## ------------------------------ ##
##    [?] Extract PCAP Data:
## ------------------------------ ##
capinfos $File.pcap
tcpslice -r $File.pcap
tcpstat $File.pcap
tcpprof -S lipn -P 30000 -r $File.pcap
tcpflow -r $File.pcap
tcpxtract -f $File.pcap -o $Dir/
tcpick -a -C -r $File.pcap
ngrep -I $File.pcap
nfdump -r $File.pcap
chaosreader -ve $File.pcap
tshark -r $File.pcap
tcpdump -r $File.pcap
bro -r $File.pcap
snort -r $File.pcap
```

Xe1phix-[PCAP]-Cheatsheet

tcpflow: Reassemble input packet data to TCP data segments

This utility will perform TCP reassembly, then output each side of the TCP data flows to separate files. This is essentially a scalable, command-line equivalent to Wireshark's "Follow | TCP Stream" feature. Additionally, tcpflow can perform a variety of decoding and post-processing functions on the resulting flows.

Usage:

```
$ tcpflow <options> -r <input file> ⏎
  -o <output path>
```

Common command-line parameters:

-r	Read from specified pcap file (can be used multiple times for multiple files)
-l	Read from multiple pcap files (with wildcards)
-o	Place output files into specified directory

Examples:

```
$ tcpflow -r infile.pcap  -o /tmp/output/
$ tcpflow -l *.pcap -o /tmp/output/
```

ngrep: Display metadata and context from packets that match a specified regular expression pattern

While grep is a very capable tool for ASCII input, it does not understand the pcap file format. ngrep performs the same function but against the Layer 4 – Layer 7 payload in each individual packet. It does not perform any TCP session reassembly, so matches are made against individual packets only.

Usage:

```
$ ngrep -I <input file> <options> ⏎
  <pattern> <bpf filter>
```

Common command-line parameters:

-I	Read from specified pcap file
-O	Write matching packets to specified pcap file
-i	Case-insensitive search
-v	Invert match – only show packets that do not match the search pattern
-t	Show timestamp from each matching packet

Note: The BPF filter is an optional parameter

Examples:

```
$ ngrep -I infile.pcap 'RETR' 'tcp and port 21'
$ ngrep -I infile.pcap -i 'l33tAUTH'
```

nfdump: Process NetFlow data from nfcapd-compatible files on disk

Files created by nfcapd (live collector) or nfpcapd (pcap-to-NetFlow distillation) are read, parsed, and displayed by nfdump. Filters include numerous observed and calculated fields, and outputs can be customized to unique analysis requirements.

Usage:

```
$ nfdump (-R <input directory path> |
          -r <nfcapd file> ) ⏎
  <options> <filter>
```

Common command-line parameters:

-r	Read from the specified single file
-R	Recursively read from the specified directory tree
-t	Specify time window in which to search (Use format: YYYY/MM/DD.hh:mm:ss-YYYY/MM/DD.hh:mm:ss)
-o	Output format to use (line, long, extended, or custom with fmt:<format string>)
-O	Output sort ordering (tstart, bytes, packets, more)
-a	Aggregate output on source IP+port, destination IP+port, layer 4 protocol
-A	Comma-separated custom aggregation fields

Filter syntax:

host	IP address or FQDN
net	Netblock in CIDR notation
proto	Layer 4 protocol (tcp, udp, icmp, etc)
as	Autonomous System number

Parameters such as host, net, and port can be applied in just one direction with the src or dst modifiers. Primitives can be combined with and, or, or not, and order can be enforced with parenthesis.

Filter examples:

- proto tcp and port 80
- proto udp and dst host 8.8.8.8
- src host 1.2.3.4 and (dst net 10.0.0.0/8 or dst net 172.16.0.0/12)
- src as 32625 (Note: Not all collections include ASNs)

Custom output formatting:

Format strings for the custom output format option (-o 'fmt:<format string>')consist of format tags, including but not limited to those below.

%ts	Start time	%sa	Source IP address
%te	End time	%da	Destination IP address
%td	Duration (In seconds)	%sp	Source port (TCP or UDP)
%pr	Layer 4 protocol		
%dp	Destination port (TCP or UDP; formatted as type.code for ICMP)		
%sap	Source IP address and port		
%dap	Destination IP address and port		
%pkt	Packet count		
%byt	Byte count		
%flg	TCP flags (sum total for flow)		
%bps	Bits per second (average)		
%pps	Packets per second (average)		
%bpp	Bytes per packet (average)		

Custom aggregation:

Records displayed can be aggregated (tallied) on user-specified fields including but not limited to those below:

proto	Layer 4 protocol
srcip	Source IP address
dstip	Destination IP address
srcport	TCP or UDP source port
dstport	TCP or UDP destination port
srcnet	Source netblock in CIDR notation
dstnet	Destination netblock in CIDR notation

Examples:

```
$ nfdump -r nfcapd.201703271745 ⏎
  -o long 'proto tcp and port 53'
$ nfdump -R /var/log/netflow/2017/03/ ⏎
  -o 'fmt:%sa %da %pr' -A srcip,dstip,proto
  'dst net 66.35.59.0/24'
$ nfdump -R /var/log/netflow/2016/ ⏎
  -O tstart 'proto tcp and port 4444'
```

tshark: Command-line access to nearly all Wireshark features

For all of Wireshark's features, the ability to access them from the command line provides scalable power to the analyst. Whether building repeatable commands into a script, looping over dozens of input files, or performing analysis directly within the shell, `tshark` packs nearly all of Wireshark's features in a command-line utility.

Usage:

```
$ tshark -n -r <input file> <options> ⏎
  -Y '<display filter>'
```

Common command-line parameters:

- `-n` Prevent DNS lookups on IP addresses
- `-r` Read from specified pcap file
- `-w` Write packet data to a file
- `-Y` Specify Wireshark-compatible display filter
- `-T` Specify output mode (`fields`, `text` (default), `pdml`, etc.)
- `-e` When used with `-T fields`, specifies a field to include in output tab-separated values (can be used multiple times)
- `-G` Specify glossary to display (`protocols`, `fields`, etc.) – shows available capabilities via command line, suitable for `grep`'ing, etc.

Display filter resources:

See the `wireshark-filter` man page for more command-line details on how to construct display filters.

Examples:

```
$ tshark -n -r infile.pcap ⏎
   -Y 'http.host contains "google"' ⏎
   -T fields -e ip.src -e http.host ⏎
   -e http.user_agent
$ tshark -n -r infile.pcap ⏎
   -Y 'ssl.handshake.certificates' ⏎
   -w just_certificates.pcap
```

tcpxtract: Carve reassembled TCP streams for known header and footer bytes to attempt file reassembly

This is the TCP equivalent to the venerable `foremost` and `scalpel` disk/memory carving utilities. `tcpxtract` will reassemble each TCP stream, then search for known start/end bytes in the stream, writing out matching sub-streams to disk. It is not protocol-aware, so it cannot determine metadata such as filenames and cannot handle protocol content consisting of non-contiguous byte sequences. Notably, `tcpxtract` cannot parse SMB traffic, encrypted payload content, or chunked-encoded HTTP traffic. Parsing compressed data requires signatures for the compressed bytes rather than the corresponding plaintext.

Usage:

```
$ tcpxtract -r <input file> <options>
```

Common command-line parameters:

- `-f` Read from specified pcap file
- `-c` Configuration (signature) file to use
- `-o` Place output files into specified directory

Signature format:

- `file_ext(max_size, start_bytes, end_bytes);`

Signature examples:

- `gif(3000000, \x47\x49\x46\x38\x37\x61, \x00\x3b);`
- `rpm(400000000, \xed\xab\xee\xdb);`

Example:

- `tcpxtract -f infile.pcap ⏎`
  `-c rpm-tcpxtract.conf -o ./`

capinfos: Calculate and display high-level summary statistics for an input pcap file

This utility displays summary metadata from one or more source pcap files. Reported metadata includes but is not limited to start/end times, hash values, packet count, and byte count.

Usage:

```
$ capinfos <options> <input file 1> ⏎
  <input file 2> <...>
```

Common command-line parameters:

- `-A` Generate all available statistics
- `-T` Use "table" output format instead of list format

Examples:

```
$ capinfos -A infile.pcap
$ capinfos -A -T infile2.pcap
$ capinfos -A *.pcap
```

mergecap: Merge two or more pcap files

When faced with a large number of pcap files, it may be advantageous to merge a subset of them to a single file for more streamlined processing. This utility will ensure the packets written to the output file are chronological.

Usage:

```
$ mergecap <options> -w <output file> ⏎
  <input file 1> <input file 2> ⏎
  <input file n>
```

Common command-line parameters:

- `-w` New pcap file to create, containing merged data
- `-s` Number of bytes per packet to retain

Example:

```
$ mergecap -w new.pcap infile1.pcap ⏎
  infile2.pcap
```

Switch	Syntax	Description
-i any	tcpdump -i any	Capture from all interfaces
-i eth0	tcpdump -i eth0	Capture from specific interface (Ex Eth0)
-c	tcpdump -i eth0 -c 10	Capture first 10 packets and exit
-D	tcpdump -D	Show available interfaces
-A	tcpdump -i eth0 -A	Print in ASCII
-w	tcpdump -i eth0 -w tcpdump.txt	To save capture to a file
-r	tcpdump -r tcpdump.txt	Read and analyze saved capture file
-n	tcpdump -n -I eth0	Do not resolve host names
-nn	tcpdump -n -i eth0	Stop Domain name translation and lookups (Host names or port names)
tcp	tcpdump -i eth0 -c 10 -w tcpdump.pcap tcp	Capture TCP packets only
port	tcpdump -i eth0 port 80	Capture traffic from a defined port only
host	tcpdump host 192.168.1.100	Capture packets from specific host
net	tcpdump net 10.1.1.0/16	Capture files from network subnet
src	tcpdump src 10.1.1.100	Capture from a specific source address
dst	tcpdump dst 10.1.1.100	Capture from a specific destination address
<service>	tcpdump http	Filter traffic based on a port number for a service
<port>	tcpdump port 80	Filter traffic based on a service
port range	tcpdump portrange 21-125	Filter based on port range
-S	tcpdump -S http	Display entire packet
ipv6	tcpdunp -IPV6	Show only IPV6 packets
-d	tcpdump -d tcpdump.pcap	display human readable form in standard output
-F	tcpdump -F tcpdump.pcap	Use the given file as input for filter
-I	tcpdump -I eth0	set interface as monitor mode
-L	tcpdump -L	Display data link types for the interface
-N	tcpdump -N tcpdump.pcap	not printing domian names
-K	tcpdump -K tcpdump.pcap	Do not verify checksum
-p	tcpdump -p -i eth0	Not capturing in promiscuous mode

Protocols		
arp	ip6	slip
ether	link	tcp
fddi	ppp	tr
icmp	radio	udp
ip	rarp	wlan

TCP Flags	
tcp-urg	tcp-rst
tcp-ack	tcp-syn
tcp-psh	tcp-fin

ICMP Types
icmp-routeradvert
icmp-routersolicit
icmp-timxceed
icmp-paramprob
icmp-tstamp
icmp-tstampreply
icmp-ireq
icmp-ireqreply
icmp-maskreq
icmp-maskreply
icmp-echoreply
icmp-unreach
icmp-sourcequench
icmp-redirect
icmp-echo

Xe1phix-TCPDump-Cheatsheet-[v5.7.8].sh

TCPDump

```
┌──[✗]─[root@parrotseckiosk-optiplex990]─[/home/parrotseckiosk/Downloads]
└─── #tcpdump -D
1.eth0 [Up, Running, Connected]
2.wg-mullvad [Up, Running]
3.any (Pseudo-device that captures on all interfaces) [Up, Running]
4.lo [Up, Running, Loopback]
```

```
┌──[✗]─[root@parrotseckiosk-optiplex990]─[/home/parrotseckiosk/Downloads]
└─── #tcpdump -vn -i 2 -w Wireguard2.pcap
tcpdump: listening on wg-mullvad, link-type RAW (Raw IP), snapshot length 262144 bytes
^C1522 packets captured
1522 packets received by filter
0 packets dropped by kernel
```

[Xe1phix-TCPDump-Cheatsheet-[v5.7.8].sh]

TShark

```
└─ #tshark -f 'udp port 53'
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
    1 0.000000000 173.22.75.86 → 172.98.193.62 DNS 82 Standard query 0x8ec9 A mullvad.net OPT
    2 0.000146254 173.22.75.86 → 172.98.193.62 DNS 82 Standard query 0xb449 A mullvad.net OPT
    3 3.485878486 172.98.193.62 → 173.22.75.86 DNS 98 Standard query response 0xb449 A mullvad.net A 45.83.220.101 OPT
    4 3.485878539 172.98.193.62 → 173.22.75.86 DNS 98 Standard query response 0x8ec9 A mullvad.net A 45.83.220.101 OPT
    5 5.685797805 173.22.75.86 → 172.98.193.62 DNS 92 Standard query 0x7433 A ipv4.am.i.mullvad.net OPT
    6 5.685958160 173.22.75.86 → 172.98.193.62 DNS 92 Standard query 0xefef A ipv4.am.i.mullvad.net OPT
    7 6.244646320 172.98.193.62 → 173.22.75.86 DNS 108 Standard query response 0x7433 A ipv4.am.i.mullvad.net A 193.138.218.116 OPT
    8 6.244831467 172.98.193.62 → 173.22.75.86 DNS 108 Standard query response 0xefef A ipv4.am.i.mullvad.net A 193.138.218.116 OPT
```

```
└─ #tshark -f 'port 6697'
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
    1 0.000000000 173.22.75.86 → 46.16.175.175 TCP 74 59506 → 6697 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=385395196 TSecr=0 WS=1024
    2 0.141689554 46.16.175.175 → 173.22.75.86 TCP 74 6697 → 59506 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2818019368 TSecr=385395196 WS=128
    3 0.141762839 173.22.75.86 → 46.16.175.175 TCP 66 59506 → 6697 [ACK] Seq=1 Ack=1 Win=64512 Len=0 TSval=385395338 TSecr=2818019368
    4 0.442294136 173.22.75.86 → 46.16.175.175 TLSv1 369 Client Hello
```

```
└─ #tshark -i any -f 'udp port 1194'
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'
    1 0.000000000 173.22.75.86 → 193.138.218.136 OpenVPN 58 MessageType: P_CONTROL_HARD_RESET_CLIENT_V2
    2 0.128643529 193.138.218.136 → 173.22.75.86 OpenVPN 70 MessageType: P_CONTROL_HARD_RESET_SERVER_V2
    3 0.128865762 173.22.75.86 → 193.138.218.136 OpenVPN 66 MessageType: P_ACK_V1
    4 0.128944171 173.22.75.86 → 193.138.218.136 TLSv1 267 Client Hello
    5 0.272074537 193.138.218.136 → 173.22.75.86 TLSv1.3 1244 Server Hello, Change Cipher Spec, Application Data, Application Data
    6 0.272276797 193.138.218.136 → 173.22.75.86 TLSv1.3 1232 Continuation Data
    7 0.272276872 193.138.218.136 → 173.22.75.86 TLSv1.3 1232 Continuation Data
    8 0.272329335 193.138.218.136 → 173.22.75.86 TLSv1.3 851 Continuation Data
    9 0.272372003 173.22.75.86 → 193.138.218.136 OpenVPN 66 MessageType: P_ACK_V1
   10 0.272395651 173.22.75.86 → 193.138.218.136 OpenVPN 66 MessageType: P_ACK_V1
   11 0.272421831 173.22.75.86 → 193.138.218.136 OpenVPN 66 MessageType: P_ACK_V1
   12 0.273627595 173.22.75.86 → 193.138.218.136 TLSv1.3 85 Application Data, Application Data, Application Data
   13 0.404691384 193.138.218.136 → 173.22.75.86 TLSv1.3 85 Application Data
```

tshark CheatSheet

Wireshark/Tcpdump Resources

[?] Capture filters (like tcp port 80) are not to be confused with display filters (like tcp.port == 80)

- Capture Filters - Wireshark Wiki
- Display Filters - Wireshark Wiki
-
-

```
└─$capinfos gitlab.pcap
File name:           gitlab.pcap
File type:           Wireshark/tcpdump/... - pcap
File encapsulation:  Ethernet
File timestamp precision:  microseconds (6)
Packet size limit:   file hdr: 262144 bytes
Number of packets:   756
File size:           517kB
Data size:           505kB
Capture duration:    65.180906 seconds
First packet time:   2021-07-02 22:31:36.270989
Last packet time:    2021-07-02 22:32:41.451895
Data byte rate:      7,754 bytes/s
Data bit rate:       62kbps
Average packet size: 668.54 bytes
Average packet rate: 11 packets/s
SHA256:              0d8a0d8329709b3394284d9ab61be00c3b6bbbd6f4a4ccb82abcf9587d
c29dc0
RIPEMD160:           693518853c755af82544f942e0cde81979b90adc
SHA1:                d96caa950814f2530ed28055419f7489b0cfd969
Strict time order:   False
Number of interfaces in file: 1
Interface #0 info:
                     Encapsulation = Ethernet (1 - ether)
                     Capture length = 262144
                     Time precision = microseconds (6)
                     Time ticks per second = 1000000
                     Number of stat entries = 0
                     Number of packets = 756
```

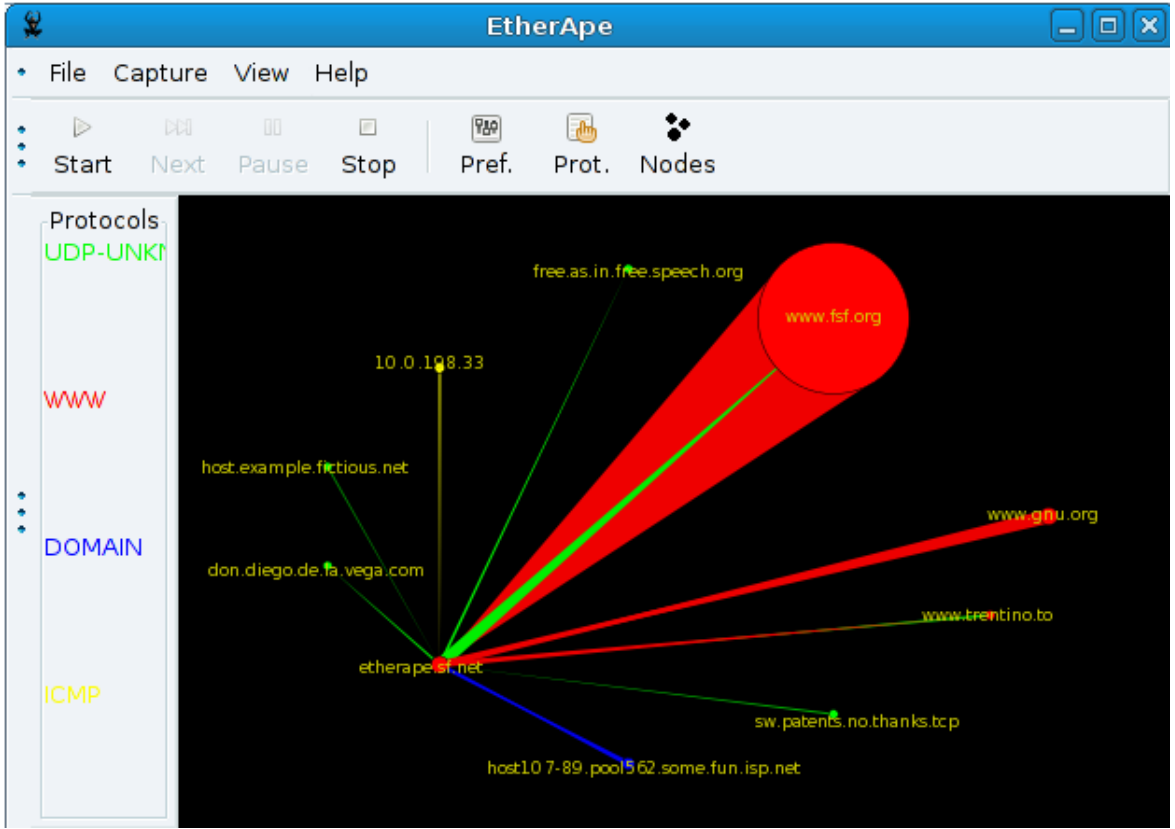
Xe1phix - TCPick - Cheatsheet

```
tcpick -i eth0 -C                                  ## display the connection status:
tcpick -i eth0 -C -yP -h -a                        ## display the payload and packet headers:
tcpick -i eth0 -C -bCU -T1 "port 25"               ## display client data only of the first smtp connection:
tcpick -i eth0 -wR "port ftp-data"                 ## download a file passively:
tcpick -i eth0 "port 80" -wRub                     ## log http data in unique files
```

[Xe1phix-[TCPick]-Cheatsheet]

```
192.168.1.101:58180 A > ics144-205.icsincorporated.com:https (0)
sn-pjnpu-5hfe.googlevideo.com..............@..E.....)........
sn-pjnpu-5hfe.googlevideo.com..@.......@..E.....)............@.".r2
192.168.1.101:44128 S > ics144-205.icsincorporated.com:5355 (0)
8      SYN-SENT       192.168.1.101:44128 > ics144-205.icsincorporated.com:5355
./...........205.144.5.69.in-addr.arpa..............
ics144-205.icsincorporated.com...)........
.x...........205.144.5.69.in-addr.arpa..............
ics144-205.icsincorporated.com...)........
.............205.144.5.69.in-addr.arpa.....
.............205.144.5.69.in-addr.arpa.....
192.168.1.101:51330 S > 193.138.218.74:domain (0)
9      SYN-SENT       192.168.1.101:51330 > 193.138.218.74:domain
193.138.218.74:domain AS > 192.168.1.101:51330 (0)
9      SYN-RECEIVED   192.168.1.101:51330 > 193.138.218.74:domain
192.168.1.101:51330 A > 193.138.218.74:domain (0)
9      ESTABLISHED    192.168.1.101:51330 > 193.138.218.74:domain
192.168.1.101:51330 AP > 193.138.218.74:domain (45)
.+.............205.144.5.69.in-addr.arpa.....
193.138.218.74:domain A > 192.168.1.101:51330 (0)
192.168.1.101:54356 AP > 149.154.167.151:http (225)
```

Etherape – Traffic Graphs



Protocol detail dialog

EtherApe: Protocols

Protocol	Port	Inst Traffic	Accum Traffic	Avg Size	Last Heard	Packets
DOMAIN	53	0 bps	1,21 Kbytes	207 bytes	13" ago	6
ICMP	-	2,35 Kbps	784 bytes	98 bytes	0" ago	8
UDP-UNKNOWN	-	16,89 Kbps	54,74 Kbytes	132 bytes	0" ago	425
WWW	80	264 bps	54,60 Kbytes	478 bytes	2" ago	117

Node detail dialog

etherape.sf.net

Resolved Name: etherape.sf.net
Numeric Name: 192.168.1.22

	Total	Inbound	Outbound
Instantaneous	31,53 Kbps	14,52 Kbps	17,01 Kbps
Accumulated	503,83 Kbytes	411,83 Kbytes	92,00 Kbytes
Average size	397 bytes	680 bytes	139 bytes

Protocol	Port	Inst Traffic	Accum Traffic	Avg Size	Last Heard	Packets
DOMAIN	53	2,87 Kbps	7,17 Kbytes	99 bytes	1" ago	74
HTTP-ALT	8080	0 bps	56 bytes	56 bytes	24" ago	1
NTP	123	736 bps	368 bytes	92 bytes	1" ago	4
WWW	80	27,92 Kbps	496,24 Kbytes	416 bytes	0" ago	1221

Xe1phix - Darkstat Cheatsheet

```
##-================================================-##
##   [+] Gather statistics on the eth0 interface:
##-================================================-##
darkstat -i eth0


##-============================-##
##   [+] Ignore ARP traffic:
##-============================-##
darkstat -i eth0 -f "not arp"


##-====================-##
##   [+] SSH traffic:
##-====================-##
darkstat -i eth0 -f "port 22"


##-========================-##
##   [+] OpenVPN traffic:
##-========================-##
darkstat -i eth0 -f "port 1194"


##-============================================-##
##   [+] Show hex dumps of received traffic
##-============================================-##
darkstat --verbose -i eth0 --hexdump
```

[Xe1phix-[Darkstat]-Cheatsheet]

```
## ------------------------------------------------------------ ##
##  [?] account for traffic on the Internet-facing  interface,
##  [?] but only  serve  web pages to our private local network
##  [?] where we have the IP address 192.168.0.1:
## ------------------------------------------------------------ ##
darkstat -i eth0 -b 192.168.0.1



##-================================================-##
##   [+] serve web pages on the standard HTTP port:
##-================================================-##
darkstat -i eth0 -p 80



##-============================================================-##
##   [+] don't account for traffic between internal IPs:
##-============================================================-##
darkstat -i eth0 -f "not (src net 192.168.0 and dst net 192.168.0)"



## ------------------------------------------------------------------ ##
##  [?] We have a network consisting of a gateway server (192.168.1.1)
##  [?] and a few  workstations (192.168.1.2,  192.168.1.3,  etc.)
## ------------------------------------------------------------------ ##
##  [?] graph all traffic entering and leaving the local network,
##  [?] not just the gateway server (which is running darkstat):
## ------------------------------------------------------------------ ##
darkstat -i eth0 -l 192.168.1.0/255.255.255.0
```

Conntrack Cheatsheet

- List all currently tracked connections:

```
conntrack --dump
```

- Display a real-time event log of connection changes:

```
conntrack --event
```

- Display a real-time event log of connection changes and associated timestamps:

```
conntrack --event -o timestamp
```

- Display a real-time event log of connection changes for a specific IP address:

```
conntrack --event --orig-src $IP
```

- Delete all flows for a specific source IP address:

```
conntrack --delete --orig-src $IP
```

Curl – Follow Redirect

```
└──╼ $curl -Iks --location -X GET -A "x-agent" https://bit.ly/3n4epen
HTTP/2 301
server: nginx
date: Tue, 06 Oct 2020 10:43:35 GMT
content-type: text/html; charset=utf-8
content-length: 146
cache-control: private, max-age=90
content-security-policy: referrer always;
location: https://youtube.com/channel/UC4rzx4VToyHJDWbAEJ5cMxQ/videos
referrer-policy: unsafe-url
set-cookie: _bit=k96aHz-a49934bc17ed744380-00y; Domain=bit.ly; Expires
4 Apr 2021 10:43:35 GMT
via: 1.1 google
alt-svc: clear
```

```
└──╼ $curl -sI https://bit.ly/3n4epen | sed -n 's/location: *//p'
https://youtube.com/channel/UC4rzx4VToyHJDWbAEJ5cMxQ/videos
```

Xe1phix-Curl-Cheatsheet.sh

Curl – Saving and loading cookies

```
##-======================-##
##    [+] Save Cookie to file:
##-======================-##
curl -c $File.txt $Domain

##-========================-##
##    [+] Load Cookie from file:
##-========================-##
curl -b $File.txt $Domain
```

Xe1phix-Curl-Cheatsheet.sh

Connect To IRC Using OpenSSL

```
┌─[✗]─[parrotsec-kiosk@parrot]─[~]
└──╼ $openssl s_client -showcerts -connect chat.freenode.net:6697
CONNECTED(00000003)
depth=2 O = Digital Signature Trust Co., CN = DST Root CA X3
verify return:1
depth=1 C = US, O = Let's Encrypt, CN = R3
verify return:1
depth=0 CN = verne.freenode.net
verify return:1
---
Certificate chain
 0 s:CN = verne.freenode.net
   i:C = US, O = Let's Encrypt, CN = R3
-----BEGIN CERTIFICATE-----
```

Connect To SMTP Using OpenSSL

```
##-====================================-##
##    [+] Connect To A SMTP Server
##    [+] Securing The Connection Using A CA:
##-====================================-##
openssl s_client -starttls smtp -CApath $Dir/ -connect 127.0.0.1:25
openssl s_client -starttls smtp -CApath $Dir/ -connect $Domain:25
openssl s_client -CAfile $CAFile -starttls smtp -connect 127.0.0.1:25
openssl s_client -CAfile $CAFile -starttls smtp -connect $Domain:25
openssl s_client -CAfile $CAFile -starttls smtp -connect $Domain --port 25
openssl s_client -starttls smtp -CApath /etc/postfix/certs/ -connect 127.0.0.1:25
openssl s_client -starttls smtp -CApath /etc/postfix/certs/ -connect $Domain:25
```

Xe1phix-[OpenSSL-s_client]-Cheatsheet

Connect to Gmail using IMAP

```
└─ $openssl s_client -tls1_3 -connect imap.gmail.com:993
CONNECTED(00000003)
depth=2 C = US, O = Google Trust Services LLC, CN = GTS Root R1
verify return:1
depth=1 C = US, O = Google Trust Services LLC, CN = GTS CA 1C3
verify return:1
depth=0 CN = imap.gmail.com
verify return:1
---
Certificate chain
 0 s:CN = imap.gmail.com
   i:C = US, O = Google Trust Services LLC, CN = GTS CA 1C3
 1 s:C = US, O = Google Trust Services LLC, CN = GTS CA 1C3
   i:C = US, O = Google Trust Services LLC, CN = GTS Root R1
 2 s:C = US, O = Google Trust Services LLC, CN = GTS Root R1
   i:C = BE, O = GlobalSign nv-sa, OU = Root CA, CN = GlobalSign Root CA
```

```
subject=CN = imap.gmail.com

issuer=C = US, O = Google Trust Services LLC, CN = GTS CA 1C3

---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: ECDSA
Server Temp Key: X25519, 253 bits
---
SSL handshake has read 4293 bytes and written 318 bytes
Verification: OK
---
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
Server public key is 256 bit
```

[Xe1phix-[OpenSSL-s_client]-Cheatsheet]

Connect To POP3 Using OpenSSL

```
##-==========================================-##
##   [+] Connect To POP3 Using OpenSSL
##-==========================================-##
openssl s_client -crlf -connect $Domain:110 -starttls pop3
```

Xe1phix-[OpenSSL-s_client]-Cheatsheet

Connect Using TLSv1.2 + SSLv3

```
##-============================-##
##   [+] Connect Using TLSv1.2
##-============================-##
openssl s_client -tls1_2 -connect $Domain:443


##-=======================================-##
##   [+] Connect Using A SSLv3 Connection
##-=======================================-##
openssl s_client -connect $Domain:443 -ssl3
```

Xe1phix-[OpenSSL-s_client]-Cheatsheet

OpenSSL – Connecting Via TLS

```
└──╼ $echo | openssl s_client -tls1_2 -connect mullvad.net:443
CONNECTED(00000003)
depth=2 O = Digital Signature Trust Co., CN = DST Root CA X3
verify return:1
depth=1 C = US, O = Let's Encrypt, CN = Let's Encrypt Authority X3
verify return:1
depth=0 CN = mullvad.net
verify return:1
---
Certificate chain
 0 s:CN = mullvad.net
   i:C = US, O = Let's Encrypt, CN = Let's Encrypt Authority X3
 1 s:C = US, O = Let's Encrypt, CN = Let's Encrypt Authority X3
   i:O = Digital Signature Trust Co., CN = DST Root CA X3
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIFXzCCBEegAwIBAgISA9Ms3l7BxSL1O5s/4TGmRWY/MA0GCSqGSIb3DQEBCwUA
```

OpenSSL – Connect Over Port 443, Printing the Transaction

```
└──╼ $echo | openssl s_client -connect mullvad.net:443 -showcerts
CONNECTED(00000003)
depth=2 O = Digital Signature Trust Co., CN = DST Root CA X3
verify return:1
depth=1 C = US, O = Let's Encrypt, CN = Let's Encrypt Authority X3
verify return:1
depth=0 CN = mullvad.net
verify return:1
---
Certificate chain
 0 s:CN = mullvad.net
   i:C = US, O = Let's Encrypt, CN = Let's Encrypt Authority X3
-----BEGIN CERTIFICATE-----
```

Print Certificate Fingerprints

- Calculate the fingerprint of Mullvad-n.pem

```
└─ $openssl x509 -sha256 -in mullvad-n.pem -noout -fingerprint | grep "SHA256 Fingerprint"
SHA256 Fingerprint=2C:B1:B6:E7:17:92:10:C5:19:64:20:AB:D0:46:C4:4A:A8:EF:64:C6:1D:58:E1:4F:78:19:70:2
A:27:59:0F:DB
```

```
└─ $certtool --certificate-info < mullvad-n.pem | grep sha256
            sha256:2cb1b6e7179210c5196420abd046c44aa8ef64c61d58e14f7819702a27590fdb
            sha256:ee8ccc72d52981e4957e5a60f161011dce3afce1cd7e17a6ebeeb44ca6243a94
            pin-sha256:7ozMctUpgeSVflpg8WEBHc46/OHNfhem6+60TKYkOpQ=
```

- Extract the dates, issuer, subject and fingerprint:

```
└─ $echo | openssl s_client -connect mullvad.net:443 2>/dev/null | openssl
x509 -noout -issuer -subject -fingerprint -dates
issuer=C = US, O = Let's Encrypt, CN = Let's Encrypt Authority X3
subject=CN = mullvad.net
SHA1 Fingerprint=A7:4C:5A:95:AC:07:20:36:F0:0F:00:E9:A1:67:42:D3:B6:2F:79:CA
notBefore=Sep 14 06:32:35 2020 GMT
notAfter=Dec 13 06:32:35 2020 GMT
```

GnuTLS – Connecting Over Port 443

```
└──╼ $gnutls-cli -p 443 mullvad.net
Processed 128 CA certificate(s).
Resolving 'mullvad.net:443'...
Connecting to '45.83.220.101:443'...
- Certificate type: X.509
- Got a certificate list of 2 certificates.
- Certificate[0] info:
 - subject `CN=mullvad.net', issuer `CN=Let's Encrypt Authority X3,O=Let's En
crypt,C=US', serial 0x03d32cde5ec1c522f5439b3fe131a645663f, RSA key 2048 bits
, signed using RSA-SHA256, activated `2020-09-14 06:32:35 UTC', expires `2020
-12-13 06:32:35 UTC', pin-sha256="7ozMctUpgeSVflpg8WEBHc46/OHNfhem6+60TKYkOpQ
="
        Public Key ID:
                sha1:861dc86bce901560911caa987aff4be699cfc015
                sha256:ee8ccc72d52981e4957e5a60f161011dce3afce1cd7e17a6ebeeb4
4ca6243a94
        Public Key PIN:
                pin-sha256:7ozMctUpgeSVflpg8WEBHc46/OHNfhem6+60TKYkOpQ=
```

GnuTLS – Print Cert

```
└─── $gnutls-cli --print-cert mullvad.net
Processed 128 CA certificate(s).
Resolving 'mullvad.net:443'...
Connecting to '45.83.220.101:443'...
- Certificate type: X.509
- Got a certificate list of 2 certificates.
- Certificate[0] info:
 - subject `CN=mullvad.net', issuer `CN=Let's Encrypt Authority X3,O=Let's En
crypt,C=US', serial 0x03d32cde5ec1c522f5439b3fe131a645663f, RSA key 2048 bits
, signed using RSA-SHA256, activated `2020-09-14 06:32:35 UTC', expires `2020
-12-13 06:32:35 UTC', pin-sha256="7ozMctUpgeSVflpg8WEBHc46/OHNfhem6+60TKYkOpQ
="
        Public Key ID:
                sha1:861dc86bce901560911caa987aff4be699cfc015
                sha256:ee8ccc72d52981e4957e5a60f161011dce3afce1cd7e17a6ebeeb4
- Status: The certificate is trusted.
- Description: (TLS1.2-X.509)-(ECDHE-X25519)-(RSA-SHA512)-(CHACHA20-POLY1305)
- Session ID: 13:58:8D:22:9A:63:C5:90:82:F2:5F:68:12:2E:B5:63:29:4F:5A:0C:B5:
1D:96:04:DA:3C:41:1F:C5:49:9E:8B
- Options: extended master secret, safe renegotiation, OCSP status request,
- Handshake was completed
```

Xe1phix-[GnuTLS]-Cheatsheet

GnuTLS – Save Cert

```
└──╼ $gnutls-cli --save-cert=mullvad.pem mullvad.net:443
Processed 128 CA certificate(s).
Resolving 'mullvad.net:443'...
Connecting to '45.83.220.101:443'...
- Certificate type: X.509
- Got a certificate list of 2 certificates.
- Certificate[0] info:
 - subject `CN=mullvad.net', issuer `CN=Let's Encrypt Authority X3,O=Let's En
crypt,C=US', serial 0x03d32cde5ec1c522f5439b3fe131a645663f, RSA key 2048 bits
, signed using RSA-SHA256, activated `2020-09-14 06:32:35 UTC', expires `2020
-12-13 06:32:35 UTC', pin-sha256="7ozMctUpgeSVflpg8WEBHc46/OHNfhem6+60TKYkOpQ
="
```

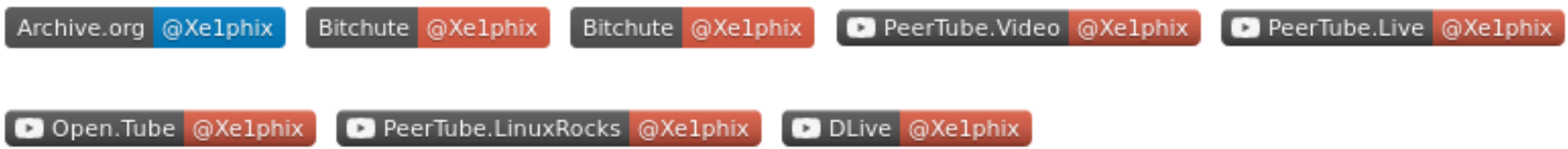
Xe1phix-[GnuTLS]-Cheatsheet

Worth Mentioning

- GNUnet
- Tinc Mesh VPN
- eBPF Filtering
-

Xe1phix Tutorial Videos

- https://archive.org/details/@xe1phix
- https://youtube.com/channel/UC4rzx4VToyHJDWbAEJ5cMxQ/videos
- https://www.bitchute.com/channel/U0QCI90XuSH9/



Using Mullvad OpenVPN SOCKS5 Proxy With Telegram Video Links

- https://www.youtube.com/watch?v=itFSRk7FffQ
- https://archive.org/details/UsingMullvadOpenVPNSOCKS5ProxyWithTelegram
- https://www.bitchute.com/video/itFSRk7FffQ/

Xe1phix Cheatsheet URLs:

- Secure-Linux-Networking-[Cheatsheets]:
  - https://gitlab.com/xe1phix/ParrotSecWiki/-/tree/InfoSecTalk/Xe1phix-InfoSec-Talk-Materials/Secure-Linux-Networking-%5BCornCon-2020%5D/Secure-Linux-Networking-%5BCheatsheets%5D
- Xe1phix-Curl-Cheatsheet-v2.5.sh
  - https://gitlab.com/xe1phix/ParrotLinux-Public-Kiosk-Project/-/blob/master/Xe1phix-Curl/Xe1phix-Curl-Cheatsheet-v2.5.sh

Xe1phix Cheatsheet URLs:

- Xe1phix-qBittorrent-IPfilter-Fetch.sh
  - https://gitlab.com/xe1phix/ParrotSecWiki/-/raw/InfoSecTalk/Xe1phix-InfoSec-Talk-Materials/Secure-Linux-Networking-%5BCornCon-2020%5D/Secure-Linux-Networking-%5BCheatsheets%5D/qBittorrent-Cheatsheets/Xe1phix-qBittorrent-IPfilter-Fetch.sh
-

Xe1phix Cheatsheet URLs:

- Xe1phix-TCPDump-Cheatsheet-[v5.3.2].sh
  - https://gitlab.com/xe1phix/ParrotSecWiki/-/blob/InfoSecTalk/Xe1phix-InfoSec-Talk-Materials/Secure-Linux-Networking-%5BCornCon-2020%5D/Secure-Linux-Networking-%5BCheatsheets%5D/TCPDump-Cheatsheet/Xe1phix-TCPDump-Cheatsheet-%5Bv5.3.2%5D.sh
- GnuPG-MullvadTrustVerifiedSignatures.sh
  - https://gitlab.com/xe1phix/ParrotSecWiki/-/blob/InfoSecTalk/Xe1phix-InfoSec-Talk-Materials/Secure-Linux-Networking-%5BCornCon-2020%5D/Secure-Linux-Networking-%5BCheatsheets%5D/Mullvad-Cheatsheets/GnuPG-MullvadTrustVerifiedSignatures.sh

Xe1phix Cheatsheet URLs:

- Connect-To-Telegram-Using-SOCKS5-Proxy-Connections-[URLs].txt
  - https://gitlab.com/xe1phix/ParrotSecWiki/-/blob/InfoSecTalk/Xe1phix-InfoSec-Talk-Materials/Secure-Linux-Networking-%5BCornCon-2020%5D/Secure-Linux-Networking-%5BCheatsheets%5D/Telegram-Cheatsheets/Connect-To-Telegram-Using-SOCKS5-Proxy-Connections-_URLs_.txt
- Xe1phix-[OpenVPN]+[Wireguard]-SOCKS5-Virtual-Tunnel-Interface-[Address]+[Port]-Information-[.v4.5.8.].txt
  - https://gitlab.com/xe1phix/ParrotSecWiki/-/blob/InfoSecTalk/Xe1phix-InfoSec-Talk-Materials/Secure-Linux-Networking-%5BCornCon-2020%5D/Secure-Linux-Networking-%5BCheatsheets%5D/Telegram-Cheatsheets/Xe1phix-%5BOpenVPN%5D+%5BWireguard%5D-SOCKS5-Virtual-Tunnel-Interface-%5BAddress%5D+%5BPort%5D-Information-%5B.v4.5.8.%5D.txt

CommandLineFu

- curl http://www.commandlinefu.com/commands/browse/sort-by-votes/plaintext/[0-2500:25] | >> CommandLineFu.txt

Mullvad References

-
- https://mullvad.net/en/check/
-