

Scripts and Configs

Torrc Examples

The following torrc config files are downsized blancos.

Exit

```
Address IP/HOSTNAME
BandwidthRate 800 # expressed in Kb
MyFamily $HASH1, $HASH2, $HASH3, $HASH4
ContactInfo 0x02225522 Frenn vun der Enn (FVDE) <info AT enn DOT lu>
DirPortFrontPage /etc/tor/tor-exit-notice.html
ORPort 9001
ControlPort 9051
DirPort 443
BandwidthBurst 900 # expressed in Kb
Nickname Foobar
ExitPolicy reject *:25
ExitPolicy reject *:587
ExitPolicy reject *:465
ExitPolicy accept *:.*
HashedControlPassword HASH:HASH
NumCPUs 2
# HardwareAccel 1 # in newer versions obsolete
# AccelName aesni
DisableDebuggerAttachment 0
```

Relay

```
Address IP/HOSTNAME
BandwidthRate 800 # expressed in Kb
MyFamily $HASH1, $HASH2, $HASH3, $HASH4
ContactInfo 0x02225522 Frenn vun der Enn (FVDE) <info AT enn DOT lu>
ORPort 9001
ControlPort 9051
DirPort 443
BandwidthBurst 900 # expressed in Kb
Nickname Foobar
ExitPolicy reject *:.*
HashedControlPassword HASH:HASH
NumCPUs 2
# HardwareAccel 1 # in newer versions obsolete
```

```
# AccelName aesni
DisableDebuggerAttachment 0
```

Bridge

```
Address IP/HOSTNAME
BandwidthRate 800 # expressed in Kb
ContactInfo 0x02225522 Frenn vun der Enn (FVDE) <info AT enn DOT lu>
ORPort 9001
ControlPort 9051
BandwidthBurst 900 # expressed in Kb
Nickname Foobar
HashedControlPassword HASH:HASH
NumCPUs 2
# HardwareAccel 1 # in newer versions obsolete
# AccelName aesni
DisableDebuggerAttachment 0
BridgeRelay 1
ExitPolicy reject *: *
ServerTransportPlugin obfs3,scramblesuit exec /usr/bin/obfsproxy managed
```

Tor AutoConfig

Tor AutoConfig is a bunch of scripts that autoconfigs your exit nodes. This TorAutoConfigscript? bundle is part of “EnnStatus?” and can only work in combination with it. Get those script via mercurial on bitbucket!

hg clone <https://bitbucket.org/fvde/tor-autoconfig>

AutoConfig

Autoconfiger is the main script which creates your personal torrc file.

Run like this: perl autoconf.pl [SERVER-TYPE] [YOUR-NODE-NICKNAME] [NETWORK SPEED] [METERED|UNMETERED] ([TRAFFIC LIMIT])

Examples:

- perl autoconf.pl exit foobar 1Gbit unmetered
- perl autoconf.pl relay foobar 100mbit metered 15TB
- perl autoconf.pl bridge foobar 10mbit metered 500GB

If you want a htmlized version of this scripts POD, use the following command:
pod2html -infile=POD/autoconf.pod -outfile=POD/autoconf.html

Family Updater

Because your Tor Node family grows with time it is a pain in the ass to keep it up2date by hand. Now you can simply run this script as a cronjob and it keeps your Family Hashes up2date.

Create a cronjob

```
*/10 * * * * perl /root/family_updater.pl
```

If you want a htmlized version of this scripts POD, use the following command:

```
pod2html -infile=POD/family_updater.pod -outfile=POD/family_updater.html
```

Network Statistics

This tiny script gives you the amount of traffic pushed through every exit node of us. And it shows you the total amount. Get this script via mercurial or wget from our repos.

wget

```
https://projects.c3l.lu/FVDE/Scripts/rawfile/207c124454d0faa0590d88188da4d2ceb2d1ad53/Exit-Network-Stats.pl
```

```
hg clone https://projects.c3l.lu/FVDE/Scripts
```

sthttpd

This is a fork of Jef Poskanzer's popular thttpd server, which you can read about on his acme.com page. The project got named sthttpd because practically every other name was taking. So something like "supported" thttpd made sense to me. Except for that change, it aims to be a drop in replacement for thttpd.

config

```
# sthttpd config - enn.lu

dir=/var/www
port=80

user=thttpd

logfile=/dev/null
pidfile=/var/run/thttpd.pid

charset=utf-8
```

Save it as /etc/thttpd.conf

init script

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          Kontroll iwwert thttpd
# Required-Start:
# Required-Stop:
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Einfach service Skript zum starten/stoppen/neistarten
#                   vun thttpd
# Description:       War halt neideg ;)
### END INIT INFO
# Author: virii <virii@tormail.org>

# Aktiounen

pid=`pidof thttpd`
case "$1" in
    start)
        thttpd -C /etc/thttpd.conf
        ;;
    stop)
        kill $pid
        ;;
    restart)
        kill $pid
        thttpd -C /etc/thttpd.conf
        ;;
esac

exit 0
```

Save it as /etc/init.d/thttpd and chmod it to 0755

unbound

Unbound is a validating, recursive, and caching DNS resolver. The C implementation of Unbound is developed and maintained by NLnet Labs. It is based on ideas and algorithms taken from a java prototype developed by Verisign labs, Nominet, Kirei and ep.net. Unbound is designed as a set of modular components, so that also DNSSEC (secure DNS) validation and stub-resolvers (that do not run as a server, but are linked into an application) are easily possible.

config

```
# Unbound configuration file for Debian.
```

```
#  
# See the unbound.conf(5) man page.  
#  
# See /usr/share/doc/unbound/examples/unbound.conf for a commented  
# reference config file.  
#  
# The following line includes additional configuration files from the  
# /etc/unbound/unbound.conf.d directory.  
  
include: "/etc/unbound/unbound.conf.d/*.conf"  
  
num-threads: 8  
  
do-ip4: yes  
do-ip6: yes
```

Change num-threads to the number of cpu cores your machine has. Save it as
/etc/unbound/unbound.conf

You don't have to install unbound on every Tor node. But you really should if you are running very high traffic nodes because most provider DNS server get buggy if they are confronted whit a shitload of DNS requests coming from only one server.

When unbound is running, edit /etc/resolv.conf

```
nameserver 127.0.0.1  
nameserver 8.8.8.8  
nameserver 8.8.4.4
```

8.8.8.8 and **8.8.4.4** are Google DNS servers which can handle some more traffic and function as fallback.

From:
<http://qzz2cvotstk357oj.onion/> - **Frënn vun der Ënn A.S.B.L.**

Permanent link:
<http://qzz2cvotstk357oj.onion/doku.php?id=scripts-config>

Last update: **2016/10/19 18:35**

