Tor (Русский)

- Page
- Discussion

- Read
- View source
- View history

Состояние перевода: На этой странице представлен перевод статьи **Tor**. Дата последней синхронизации: 11 июля 2021. Вы можете **помочь** синхронизировать перевод, если в английской версии произошли **изменения**.

Ссылки по теме

GNUnet
I2P (Русский)
Freenet

Tor Project (*T*he *o*nion *r*outing) — открытая реализация **луковой маршрутизации**, предоставляющая доступ к анонимной прокси-сети. Основная цель — сохранить **анонимность** пользователя в интернете, обеспечив защиту от **анализа трафика**.

Пользователи сети Tor запускают на своих машинах "луковый прокси" (onion proxy), который предоставляет программам-клиентам SOCKS-интерфейс. Прокси подключается к сети Tor, периодически согласовывая виртуальный канал через неё. Tor использует многослойную криптографию (отсюда "луковые" аналогии), последовательно обеспечивая секретность на каждом промежуточном маршрутизаторе.

В процессе работы луковый прокси управляет сетевым трафиком, обеспечивая анонимность конечного пользователя. Трафик зашифровывается, пересылается

через узлы сети Tor и дешифруется на последнем узле перед отправкой на целевой сервер. Цена обеспечения анонимности — низкая скорость работы по сравнению с обычным прямым соединением из-за большого количества перенаправлений трафика. Кроме того, хотя Tor обеспечивает защиту от анализа трафика, невозможно избежать подтверждения трафика на границах сети Tor (т.е. в точках входа и выхода из сети). Подробнее см. **Wikipedia:ru:Tor**.

Примечание: Сам по себе Tor **не** обеспечивает полную анонимность. Существует целый ряд проблем и возможных ошибок (см. **Want Tor To Really Work?**).

Установка

Установите пакет **`tor`**.

Как правило, с его помощью осуществляется **#Веб-сёрфинг**.

Nyx — консольная утилита для мониторинга Tor. Позволяет отслеживать использование пропускной способности, детали соединений, а также редактировать настройки "на лету". Для его использования **установите** пакет **nyx**.

Использование

Запустите/включите службу `tor.service`. В качестве альтернативы можно запустить Tor командой `sudo -u tor /usr/bin/tor`.

Чтобы программа работала через Tor, настройте её использовать `127.0.0.1` или `localhost` в качестве SOCKS5-прокси на порте 9050 (для Tor со стандартными настройками).

Прокси позволяет удалённо выполнять разрешение доменных имён: используйте `socks5h://localhost:9050` для отправки DNS-запросов с выходного узла (вместо `socks5` для локальных запросов).

Проверить работу Tor можно на странице **https://check.torproject.org/** или **https://torcheck.xenobite.eu/**[**устаревшая ссылка** 2023-07-30 ⓘ].

Настройка

По умолчанию Tor считывает настройки из файла `/etc/tor/torrc`, или, если последний не обнаружен, из `$HOME/.torrc`. Настройки объясняются в руководстве **`tor(1)`** и на **сайте Tor**. Настройки по умолчанию должны работать для большинства пользователей.

После изменения настроек службу `tor.service` необходимо **перезагрузить**.

Настройка Tor Relay

Максимальное количество дескрипторов файлов, одновременно открытых Tor, задаётся параметром `LimitNOFILE` в файле `tor.service`. Для быстрых ретрансляторов имеет смысл увеличить это значение.

Если на вашей системе не запущен веб-сервер и вы не задавали значение `AccountingMax` (определяет максимальный объём передаваемых данных), рассмотрите возможность установки параметра `ORPort` в значение `443` и/или `DirPort` в значение `80`. Многие пользователи Tor находятся за жёсткими межсетевыми экранами, которые разрешают им только веб-сёрфинг, и такая настройка позволит им использовать ваш ретранслятор. Если порты `80` и `443` уже

заняты, то подойдут `22`, `110`, `143` и `9001` [1]. Порты с номерами до 1024 являются системными и при работе через них Tor должен быть запущен с привилегиями суперпользователя; задайте параметры `User=root` и `User tor` в файлах `tor.service` и `torrc` соответственно.

Также будет полезно прочитать статью о **жизненном цикле** ретрансляторов в документации Tor.

Tor ControlPort

Как правило, особой необходимости открывать *ControlPort* не возникает, но некоторым программам это может потребоваться для получения низкоуровневого доступа к узлу.

Через открытый ControlPort внешние приложения смогут отслеживать состояние вашего узла, корректировать настройки, а также получать информацию о состоянии сети Tor и виртуальных каналов.

Добавьте следующую строку в файл `torrc`:

```
ControlPort 9051
```

Разумеется, доступ к ControlPort должен предоставляться только доверенным пользователям. Ограничение доступа осуществляется либо с помощью cookie-файла, либо паролем, либо обоими способами одновременно.

Cookie-файл Tor Control

Добавьте к файлу `torrc` следующие строки:

```
CookieAuthentication 1
```

```
CookieAuthFile /var/lib/tor/control_auth_cookie

CookieAuthFileGroupReadable 1

DataDirectoryGroupReadable 1
```

Доступ к ControlPort будет ограничен набором файловых разрешений cookie-файла и каталога data. Доступ к cookie-файлу Tor Control получат все пользователи группы `tor`.

Добавьте пользователя в группу `tor`:

```
# usermod -a -G tor пользователь
```

Перезагрузите настройки группы:

```
$ newgrp tor
```

Перезапустите Tor:

```
# systemctl restart tor
```

Теперь *пользователь* имеет доступ к файлу cookie. Команда

```
$ stat -c%a /var/lib/tor /var/lib/tor/control_auth_cookie
```

должна вывести значения `750` и `640`.

Пароль Tor Control

Преобразуйте пароль из представления в виде открытого текста в хэш:

```
# set +o history                          # отключить историю команд bash

# tor --hash-password пароль

# set -o history                          # включить историю команд bash
```

Добавьте этот хэш к файлу `torrc`:

```
HashedControlPassword хэш
```

Команда `set +o history` отключает сохранение истории в файл `$HISTFILE`, чтобы при выполнении команды `tor --hash-password` в нём не сохранилось значение пароля открытым текстом.

Tor ControlSocket

Tor ControlSocket имеет примерно то же назначение, что и **#Tor ControlPort**, с той лишь разницей, что прослушивается не TCP-сокет, а **сокет домена** Unix.

Если какой-то программе нужен доступ к Tor ControlSocket, добавьте следующие строки к файлу `torrc`:

```
ControlSocket /var/lib/tor/control_socket

ControlSocketsGroupWritable 1

DataDirectoryGroupReadable 1

CacheDirectoryGroupReadable 1              # обходное решение для бага #26913
```

Добавьте запускающего программу пользователя в группу `tor`:

```
# usermod -a -G tor пользователь
```

Перезагрузите настройки группы:

```
$ newgrp tor
```

Затем **перезапустите** Tor

```
# systemctl restart tor
```

и перезапустите программу.

Чтобы проверить состояние контрольного сокета, выполните

```
# stat -c%a /var/lib/tor /var/lib/tor/control_socket
```

Команда должна вывести значения `750` и `660`.

Проверка Tor Control

Чтобы проверить настройки ControlPort, используйте входящую в пакет **gnu-netcat** утилиту *nc*:

```
$ echo -e 'PROTOCOLINFO\r\n' | nc 127.0.0.1 9051
```

Также запустите **socat**:

```
$ echo -e 'PROTOCOLINFO\r\n' | sudo -u пользователь socat -
UNIX-CLIENT:/var/lib/tor/control_socket
```

Обе команды должны вывести

```
250-PROTOCOLINFO 1

250-AUTH METHODS=COOKIE,SAFECOOKIE,HASHEDPASSWORD
COOKIEFILE="/var/lib/tor/control_auth_cookie"

250-VERSION Tor="0.3.4.8"

250 OK

514 Authentication required.
```

Дополнительную информацию можно найти в описании **протокола Tor Control**.

Запуск Tor в Chroot

Важно: Подключение по telnet на локальный ControlPort окажется невозможным, если Tor был запущен в chroot.

По соображениям безопасности желательно запускать Tor в **chroot**. Следующий скрипт создаст подходящее окружение chroot в каталоге `/opt/torchroot`:

~/torchroot-setup.sh

```
#!/bin/sh
export TORCHROOT=/opt/torchroot


mkdir -p $TORCHROOT
mkdir -p $TORCHROOT/etc/tor
mkdir -p $TORCHROOT/dev
mkdir -p $TORCHROOT/usr/bin
mkdir -p $TORCHROOT/usr/lib
mkdir -p $TORCHROOT/usr/share/tor
mkdir -p $TORCHROOT/var/lib
mkdir -p $TORCHROOT/var/log/tor/


ln -s /usr/lib  $TORCHROOT/lib
cp /etc/hosts           $TORCHROOT/etc/
cp /etc/host.conf       $TORCHROOT/etc/
cp /etc/localtime       $TORCHROOT/etc/
cp /etc/nsswitch.conf   $TORCHROOT/etc/
cp /etc/resolv.conf     $TORCHROOT/etc/
```

```
cp /usr/bin/tor         $TORCHROOT/usr/bin/

cp /usr/share/tor/geoip* $TORCHROOT/usr/share/tor/

cp /lib/libnss* /lib/libnsl* /lib/ld-linux-*.so* /lib/libresolv* /lib/libgcc_s.so*
$TORCHROOT/usr/lib/

cp $(ldd /usr/bin/tor | awk '{print $3}'|grep --color=never "^/")
$TORCHROOT/usr/lib/


## /var/log/tor/notices.log необходим только в том случае, если вы запускаете
hidden service

# cp /var/log/tor/notices.log $TORCHROOT/var/log/tor/


cp -r /var/lib/tor      $TORCHROOT/var/lib/

cp /etc/tor/torrc       $TORCHROOT/etc/tor/


chown tor:tor $TORCHROOT

chmod 700 $TORCHROOT

chown -R tor:tor $TORCHROOT/var/lib/tor

chown -R tor:tor $TORCHROOT/var/log/tor


sh -c "grep --color=never ^tor /etc/passwd > $TORCHROOT/etc/passwd"

sh -c "grep --color=never ^tor /etc/group > $TORCHROOT/etc/group"


mknod -m 644 $TORCHROOT/dev/random c 1 8

mknod -m 644 $TORCHROOT/dev/urandom c 1 9

mknod -m 666 $TORCHROOT/dev/null c 1 3


if [ "$(uname -m)" = "x86_64" ]; then
```

```
  cp /usr/lib/ld-linux-x86-64.so* $TORCHROOT/usr/lib/.
  ln -sr /usr/lib64 $TORCHROOT/lib64
  ln -s $TORCHROOT/usr/lib ${TORCHROOT}/usr/lib64
fi
```

Выполнив скрипт от пользователя root, вы можете запустить Tor в окружении chroot командой:

```
# chroot --userspec=tor:tor /opt/torchroot /usr/bin/tor
```

Или же, если вы используете systemd, то можете создать **drop-in файл** для службы `tor.service`:

```
/etc/systemd/system/tor.service.d/chroot.conf
[Service]
User=root
ExecStart=
ExecStart=/usr/bin/sh -c "chroot --userspec=tor:tor /opt/torchroot /usr/bin/tor -f /etc/tor/torrc"
KillSignal=SIGINT
```

Запуск Tor в контейнере systemd-nspawn

В этом примере мы создадим контейнер systemd-nspawn с виртуальным сетевым macvlan-интерфейсом. Контейнер будет называться `tor-exit`.

В статьях **systemd-nspawn** и **systemd-networkd** можно найти полную информацию о работе и настройке соответствующих программ.

Установка и настройка хоста

Контейнер будет размещаться в каталоге `/srv/container`:

```
# mkdir -p /srv/container/tor-exit
```

Установите пакет **arch-install-scripts**, чтобы получить доступ к утилите *pacstrap*.

С помощью *pacstrap* установите в каталог контейнера пакеты **base**, **tor** и **nyx** (подробнее см. статью об **установке Arch Linux в контейнер**):

```
# pacstrap -ci /srv/container/tor-exit base tor nyx
```

Если зарегистрировать контейнер на хосте, то в дальнейшем с ним можно будет взаимодействовать извне с помощью команды `machinectl`. Для этого необходимо создать символическую ссылку на контейнер в каталоге `/var/lib/container/`. Создайте каталог, если он отсутствует:

```
# mkdir -p /var/lib/container
```

и поместите в него символическую ссылку на контейнер (см. **systemd-nspawn#Management**):

```
# ln -s /srv/container/tor-exit /var/lib/container/tor-exit
```

Виртуальный сетевой интерфейс

Создайте drop-in-файл настроек контейнера:

```
/etc/systemd/nspawn/tor-exit.nspawn
```

```
[Service]

MACVLAN=интерфейс
```

```
[Exec]
LimitNOFILE=32768
```

Опция `MACVLAN=`*`интерфейс`* создаёт macvlan-интерфейс с названием `mv-`*`интерфейс`* и привязывает его к контейнеру, подробнее см. **systemd-nspawn#Use a "macvlan" or "ipvlan" interface**. Это полезно с точки зрения безопасности, поскольку контейнеру можно назначить приватный IP-адрес, а настоящий адрес машины из контейнера виден не будет. Так можно, например, скрыть DNS-запросы.

Опция `LimitNOFILE=32768` позволит открывать большее **#Количество соединений** одновременно.

Наконец, сохраните сетевые настройки **systemd-networkd** в файле `/srv/container/tor-exit/etc/systemd/network/mv-`*`интерфейс`*`.network`.

Запуск и включение systemd-nspawn

Запустите/включите службу `systemd-nspawn@tor-exit.service`.

Настройка контейнера

Чтобы войти в контейнер, выполните (см. **systemd-nspawn#machinectl**):

```
# machinectl login tor-exit
```

Если выполнить вход не удаётся, см. **systemd-nspawn#Root login fails**.

Запуск и включение systemd-networkd

Запустите/включите службу `systemd-networkd.service`. Команда `networkctl` отобразит список сетевых интерфейсов контейнера, если `systemd-networkd` настроен корректно.

Настройка Tor

Смотри раздел **#Запуск сервера Tor**.

Совет: Файлы контейнера удобнее редактировать с хоста, вашим любимым редактором.

Веб-сёрфинг

Единственный способ оставаться анонимным при просмотре страниц в интернете — использовать *Tor Browser Bundle*, который использует пропатченную версию **Firefox**. Его можно установить с пакетом **`torbrowser-launcher`** или **`tor-browser`**AUR.

Также можно использовать Tor с обычными браузерами: в разделах **#Firefox** и **#Chromium** объясняется, как настроить их на работу через Tor. Обратите внимание, что даже в режиме приватного просмотра это не гарантирует анонимности: отпечатки, плагины, DNS-утечки и прочие дефекты могут стать причиной разглашения вашего IP-адреса или личности **[2]**.

Совет: Чтобы makepkg мог проверить подпись загруженного из AUR tar-архива с исходниками *Tor Browser*, импортируйте **ключи подписей Tor Project** по инструкции из статьи **GnuPG#Use a keyserver**.

Firefox

Preferences > *General* > *Network Settings* > *Settings...* , выберите пункт *Manual proxy configuration*, после чего укажите: SOCKS Host `localhost` на порте `9050` (SOCKS v5). Чтобы перенаправить все DNS-запросы в сеть Tor, выберите пункт *Proxy DNS when using SOCKS v5*.

Chromium

Запустите Chromium:

```
$ chromium --proxy-server="socks5://мой-прокси:8080" --host-resolver-rules="MAP * ~NOTFOUND , EXCLUDE мой-прокси"
```

Флаг `--proxy-server="socks5://мой-прокси:8080"` означает, что все `http://` и `https://` запросы будут посылаться через прокси-сервер `"мой-прокси:8080"` посредством протокола SOCKS пятой версии. Разрешение имён для этих запросов будет выполняться прокси-сервером, а не браузером локально.

Примечание: Проксирование `ftp://` через SOCKS-прокси на данный момент не реализовано [3].

Флаг `--proxy-server` влияет только на загрузку URL-страниц. Однако в Chromium есть и другие компоненты, которые могут попытаться выполнить DNS-разрешение напрямую. Наиболее важный их этих компонентов — *DNS-prefetcher*. Если DNS-prefetcher не отключён, то браузер будет посылать DNS-запросы напрямую, минуя SOCKS5-сервер. Prefetcher и другие компоненты можно отключить, но такой подход неудобен и ненадёжен, поскольку придётся отслеживать каждый элемент Chromium, который может захотеть посылать DNS-запросы самостоятельно.

Для комплексного решения этой проблемы используется флаг `--host-resolver-rules="MAP * ~NOTFOUND , EXCLUDE мой-прокси"`, который представляет собой ловушку для посылаемых через обычную сеть DNS-запросов. Каждое выполяемое локально разрешение DNS теперь будет привязано к (нерабочему) адресу `~NOTFOUND` (можно представить его как адрес `0.0.0.0`). Указание `"EXCLUDE"` создаёт исключение для прокси-сервера `"мой-прокси"`, потому что без этого Chromium не сможет выполнять разрешение адреса самого прокси-сервера SOCKS и все запросы будут завершаться неудачей с ответом `PROXY_CONNECTION_FAILED`.

Также, чтобы предотвратить **утечки WebRTC**, можно установить расширение браузера **WebRTC Network Limiter**.

Отладка

В случае возникновения каких-либо проблем в первую очередь нужно проверить настройки прокси, введя адрес `chrome://net-internals/#proxy`.

Затем нужно изучить вкладку настроек DNS, чтобы убедиться, что Chromium не выполняет локальное разрешение DNS: `chrome://net-internals/#dns`.

Расширения

Как и для Firefox, вы можете установить удобный переключатель прокси вроде **Proxy SwitchySharp**.

После установки перейдите на его панель настроек. Под вкладкой *Proxy Profiles* добавьте новый профиль *Tor*, уберите отметку с опции *Use the same proxy server for*

all protocols, затем добавьте *localhost* в качестве хоста SOCKS, порт *9050*, и выберите *SOCKS v5*.

При желании можно включить опцию быстрого переключения на вкладке настроек *General*. Тогда переключаться между нормальной навигацией и сетью Tor можно будет одним кликом на иконке Proxy SwitchySharp.

Luakit

Примечание: Вас будет легко опознать по редкой строке `user-agent`; также могут возникнуть проблемы с Flash, JavaScript и т.д.

Выполните:

```
$ torsocks luakit
```

HTTP-прокси

В Tor может работать через встроенный туннелированный HTTP-прокси или сторонний прокси вроде **Privoxy**. Тем не менее, разработчики Tor рекомендуют использовать библиотеку SOCKS5, если ваш браузер её поддерживает.

Tor

Добавьте следующую строку в файл `torrc`, чтобы использовать порт `8118` на `localhost` в качестве http-прокси:

```
HTTPTunnelPort 127.0.0.1:8118
```

Подробнее смотрите **руководство Tor**.

Firefox

Расширение браузера **FoxyProxy** позволяет назначить прокси-сервер как для всех HTTP-запросов в целом, так и для обращения по отдельным веб-адресам. После установки расширения перезапустите браузер и вручную настройте использование прокси по адресу `localhost:8118`, где должен работать **Privoxy**. Эти настройки находятся в меню *Add* > *Standard proxy type*. Выберите метку прокси-сервера (например, `Tor`) и введите хост и порт в поля *HTTP Proxy* и *SSL Proxy*. Для проверки правильности работы Tor посетите страницу **Tor Check**.

Privoxy

Privoxy можно использовать для обмена сообщениями (**Jabber**, **IRC**) и других приложений. Приложения, которые поддерживают HTTP-прокси, можно подключить к Privoxy (например, на адрес `127.0.0.1:8118`). Чтобы использовать SOCKS-прокси, направьте ваше приложение в Tor (адрес `127.0.0.1:9050`). Следует иметь в виду, что приложение может самостоятельно выполнять DNS-разрешение, что приведет к утечке информации. В этом случае можно попробовать использовать SOCKS4A (например, с Privoxy).

Обмен мгновенными сообщениями

Для использования мессенджера через Tor не нужен HTTP-прокси вроде **Privoxy**. Мы используем напрямую демон Tor, по умолчанию прослушивающий порт 9050.

Pidgin

Pidgin можно настроить на работу через Tor глобально или для отдельных аккаунтов. Для глобальных настроек используйте пункт меню *Tools -> Preferences ->*

Proxy. Чтобы настроить использование Tor для одного аккаунта, перейдите в *Accounts* > *Manage Accounts*, выберите нужный аккаунт, нажмите *Modify*, после чего на вкладке *Proxy* укажите следующие параметры:

```
Proxy type SOCKS5

Host 127.0.0.1

Port 9150
```

Заметьте, что **в 2013 году** значение Port для Tor Browser Bundle изменилось с 9050 на 9150. Если вы получили ошибку *Connection refused*, попробуйте изменить номер порта на прежний.

Irssi

 **This article or section is out of date.** 

Reason: `cap_sasl.pl` сломался с обновлением *perl* 5.20; кроме того, SSL не работает с `torsocks`. (Discuss in **Talk:Tor (Русский)**)

Libera Chat рекомендует подключаться напрямую к `.onion`. Для этого также потребуется SASL для идентификации NickServ в процессе соединения; подробности можно найти в статье **Irssi#Authenticating with SASL**. Запустите irssi:

```
$ torsocks irssi
```

Задайте ваши индентификационные данные для сервиса **nickserv**, которые будут считываться при создании соединения. Поддерживаются механизмы аутентификации ECDSA-NIST256P-CHALLENGE (см. **ecdsatool**) и PLAIN. DH-BLOWFISH больше **не поддерживается**.

```
/sasl set сеть имя-пользователя пароль механизм
```

Отключите CTCP и DCC и задайте фальшивое имя хоста, чтобы чтобы скрыть настоящее:

```
/ignore * CTCPS

/ignore * DCC

/set hostname фальшивый_хост
```

Подключитесь к Libera Chat:

```
/connect -network network
libera75jm6of4wxpxt4aynol3xjmbtxgfyjpu34ss4d7r7q2v5zrpyd.onion
```

Дополнительную информацию можно найти в статьях **Accessing Libera.Chat Via Tor**, **Using SASL** и **IRC/SILC Wiki article**.

Pacman

Через сеть Tor можно выполнять загрузочные операции **pacman** — синхронизировать базы данных репозиториев, скачивать пакеты и открытые ключи.

Преимущества:

- Если кто-то отслеживает Интернет-соединение вашей машины, то он не увидит выполняемые обновления. Соответственно, нельзя будет определить, какие пакеты установлены, какие из них устарели и как часто выполяется обновление. Но следует учитывать, что атакующий всё ещё может опознать используемое программное обеспечение и узнать его версию другими способами. Например, проанализировав исходящие пакеты вашего HTTP-сервера и просканировав его порт можно будет определить, что а) это работающий HTTP-сервер и б) его версию.

- Если зеркало для скачивания находится не в onion-домене, то скомпроментированный выходной узел может обнаружить попытку обновления и решить атаковать вашу машину. Однако при этом атакующий скорее всего не будет знать, какую именно машину он атакует.
- Атакующий может попытаться убедить вашу машину, что доступных обновлений нет, чтобы не позволить применить обновления безопасности. При обновлении через Tor сделать это будет крайне непросто, поскольку в анонимной сети почти невозможно определить конкретно вашу машину.

  Недостатки:

- Более длительное время обновления из-за высокого значения задержки и низкой пропускной способности сети Tor. Это может быть значительным недостатком с точки зрения безопасности, если обновление нужно выполнить как можно быстрее, особенно на машинах, непосредственно подключённых к сети Интернет. Например, если уязвимость серьёзная и её легко обнаружить и использовать, то злоумышленники часто стараются атаковать как можно больше уязвимых систем максимально быстро, чтобы успеть до применения обновлений безопасности.

  Надёжность обновлений через Tor:

- Не нужно использовать DNS.
- Вы зависите от состояния сети Tor и особенно выходных узлов (например, они могут блокировать обновления).
- Вы зависите от правильной работы демона Tor. Например, если на диске закончилось свободное место, то демон Tor может отказать работать. "Reserved

blocks gid:" в ext4, квоты на использование дискового пространства и некоторые другие меры помогут решить эту проблему.

- Если вы находитесь в стране, где Tor блокируется, или в которой почти или совсем нет пользователей Tor, то вам придётся использовать мост (Tor bridge).

  Замечание по поводу gpg:

  Pacman считает надёжными только те ключи, которые подписаны либо лично вами (делается командой `pacman-key --lsign-key`), либо тремя из пяти мастер-ключей Arch. Если вредоносная выходная нода попробует заменить пакет на другой, подписаный её ключом, pacman не позволит пользователю установить такой пакет.

  Примечание: Для других дистрибутивов на основе Arch, а также неофициальных репозиториев и AUR это утверждение может оказаться неверным.

  ```
  /etc/pacman.conf
  ...
  XferCommand = /usr/bin/curl --socks5-hostname localhost:9050 --continue-at - --fail
  --output %o %u
  ...
  ```

  Примечание: Во время обработки подписей баз данных иногда можно получить ошибку 404. Это зависит от **настроек pacman** и как правило безвредно.

Java

Чтобы заставить приложение Java **проксировать** все соединения через Tor, задайте следующую опцию командной строки:

```
export JAVA_OPTIONS="$JAVA_OPTIONS -DsocksProxyHost=localhost -
DsocksProxyPort=9050"
```

Запуск сервера Tor

Сеть Tor существует благодаря пользователям, которые создают и обслуживают узлы сети, предлагая свою пропускную способность, и запускают onion-сервисы. Есть несколько способов внести свой вклад в работу сети.

Мост

Мост Tor (Tor bridge) — передающий узел сети, адрес которого не содержится в открытом каталоге узлов Tor. По этой причине мост останется доступным для желающих подключиться к Tor, даже если правительство или интернет-провайдер блокирует все публичные передатчики. На странице **https://bridges.torproject.org/** объясняется, как узнать адреса мостов.

Для запуска моста файл `torrc` должен содержать только следующие четыре строки (см. также **[4]**):

```
SOCKSPort 0

ORPort 443

BridgeRelay 1

Exitpolicy reject *:*
```

Передающий узел

В режиме передатчика (relay) ваша машина будет работать в качестве входного (guard relay) или промежуточного (middle relay) узла сети. В отличие от моста, адрес передающей машины будет опубликован в каталоге узлов Tor. Задача

передающего узла заключается в пересылке пакетов к другим передатчикам или выходным узлам, но не в сеть Интернет.

Файл настроек передающего узла с пропускной способностью не менее 20 Кбит/с должен выглядеть так:

```
Nickname название-узла-Tor

ORPort 9001                    # Этот TCP-порт должен быть открыт или проброшен в
вашем сетевом экране

BandwidthRate 20 KB            # Ограничить скорость передачи величиной 20 Кбит/с

BandwidthBurst 50 KB           # Но разрешить пиковые значения до 50 Кбит/с

ExitPolicy reject *:*          # Запретить отправку пакетов в обычную сеть
```

Выходной узел

Чтобы запрос из сети Tor попал в обычную сеть Интернет, необходим выходной узел (exit relay). Важно понимать, что кто бы ни посылал запрос, для получателя всё будет выглядеть так, будто отправителем является именно выходной узел. Поэтому запуск выходной ноды считается наименее безопасным с точки зрения законности. Если вы размышляете над запуском выходного узла, то стоит изучить **советы и рекомендации** от создателей проекта.

Настройка

В файле `torrc` можно настроить перечень разрешённых на выходном узле сервисов. Например, разрешить весь трафик:

```
ExitPolicy accept *:*
```

Разрешить только соединения через IRC-порты 6660-6667 и запретить всё остальное:

```
ExitPolicy accept *:6660-6667,reject *:*
```

По умолчанию Tor настроен блокировать определённые порты. В файле `torrc` можно внести корректировки в этот список:

```
ExitPolicy accept *:119
```

Пример настройки 100-мегабитного выходного узла

Ниже даны рекомендации по настройке быстрого (более 100 Мбит/с) выходного узла Tor, на котором устанавливается межсетевой экран **iptables**, **Haveged** для повышения энтропии системы и DNS-кэш **pdnsd**. Настоятельно рекомендуется *предварительно* изучить статью **о настройке выходного узла**.

Примечание: В разделе **#Запуск Tor в контейнере systemd-nspawn** описана установка Tor в контейнер `systemd-nspawn`. **Haveged** должен быть установлен на хосте.

Tor

Количество соединений

По умолчанию, Tor может обрабатывать до 8192 соединений одновременно. Можно увеличить это значение до 32768 **[5]**:

```
/etc/systemd/system/tor.service.d/increase-file-limits.conf
[Service]

LimitNOFILE=32768
```

Также нужно скорректировать системный параметр `nofile`:

```
/etc/security/limits.conf
...
tor     soft    nofile    32768
tor     hard    nofile    32768
@tor    soft    nofile    32768
@tor    hard    nofile    32768
```

Узнать текущее значение `nofile` можно командой `# sudo -u tor 'ulimit -Hn'` (или разбив её на две: `# sudo -u tor bash` и `# ulimit -Hn`).

Привилегированные порты

Чтобы разрешить Tor использовать привилегированные порты, службу `tor.service` нужно запускать от root. Не забудьте также добавить параметр `User tor` в файле `/etc/tor/torrc`, чтобы понизить привилегии после запуска.

```
/etc/systemd/system/tor.service.d/start-as-root.conf
[Service]
User=root
```

Конфигурация

Образец файла настроек выходного узла Tor:

```
/etc/tor/torrc
SOCKSPort 0                                          ## Обычный передатчик без использования локального прокси-сервера SOCKS


Log notice stdout                                    ## Стандартное поведение Tor


ControlPort 9051                                     ## Для соединения nyx
```

```
CookieAuthentication 1                                  ## Для соединения nyx
DisableDebuggerAttachment 0                             ## Для соединения nyx


ORPort 443                                              ## Служба tor.service должна быть
запущена от root


Address $IP                                             ## IP-адрес или FQDN
Nickname $NICKNAME                                      ## Название узла


RelayBandwidthRate 500 Mbits                            ##
bytes/KBytes/MBytes/GBytes/KBits/MBits/GBits
RelayBandwidthBurst 1000 MBits                          ##
bytes/KBytes/MBytes/GBytes/KBits/MBits/GBits


ContactInfo $E-MAIL - $BTC-ADDRESS                      ## Контактная информация


DirPort 80                                              ## Служба tor.service должна быть
запущена от root
DirPortFrontPage /etc/tor/tor-exit-notice.html          ## Демонстрация веб-страницы на
порте 80


MyFamily $($KEYID),$($KEYID)...                         ## Не забудьте знак $ перед keyid
;)


ExitPolicy reject XXX.XXX.XXX.XXX/XX:*                  ## Заблокировать отдельный домен
или IP-адрес в дополнение к стандартной политике
```

```
User tor                                                ## Изменяет пользователя на tor,
если служба была запущена от root


### Производительность ###
AvoidDiskWrites 1                                       ## Уменьшение износа SSD
DisableAllSwap 1                                        ## Служба tor.service должна быть
запущена от root
HardwareAccel 1                                         ## Использование аппаратной
поддержки OpenSSL
NumCPUs 2                                               ## Запуск в двух потоках
```

Информацию об использованных здесь опциях можно найти в **руководстве Tor**.

Tor по умолчанию запускает SOCKS-прокси на порте 9050 — даже если вы не не говорили ему этого делать. Укажите параметр `SOCKSPort 0`, если планируете использовать Tor только в качестве передатчика без запуска работающих через прокси-сервер локальных приложений.

`Log notice stdout` перенаправляет логи в поток стандартного вывода stdout, что тоже является настройкой Tor по умолчанию.

`ControlPort 9051`, `CookieAuthentication 1` и `DisableDebuggerAttachment 0` позволит использовать **nyx** для мониторинга.

`ORPort 443` и `DirPort 80` говорит Tor прослушивать порты 443 и 80, а `DirPortFrontPage` выводит **страницу приветствия** при установлении соединения на порте 80.

`ExitPolicy reject XXX.XXX.XXX.XXX/XX:*` позволяет заблокировать соединения с определённого адреса или домена; можно использовать для

блокировки соседних с выходным узлом хостов, чтобы злоумышленник не смог воспользоваться ими для обхода правил сетевого экрана.

`AvoidDiskWrites 1` уменьшает количество записей на диск и износ SSD.

`DisableAllSwap 1` заблокирует все текущие и будущие страницы памяти, чтобы её нельзя было выгрузить.

Если команда `grep aes /proc/cpuinfo` возвращает какой-то результат, то ваш CPU поддерживает AES-инструкции. Если соответствующий модуль был загружен (`lsmod | grep aes`), то параметр `HardwareAccel 1` включит встроенное аппаратное ускорение шифрования **[6]**.

`ORPort 443`, `DirPort 80` и `DisableAllSwap 1` требуют запуска службы Tor от root, как описано в разделе **#Привилегированные порты**.

nyx

Если в файле `/etc/tor/torrc` указаны параметры `ControlPort 9051` и `CookieAuthentication 1`, то можно запустить **nyx** командой `sudo -u tor nyx`.

Чтобы просматривать список соединений Tor с помощью **nyx**, нужно также указать параметр `DisableDebuggerAttachment 0`.

Чтобы запустить `nyx` от другого пользователя (не `tor`), изучите раздел **#Cookie-файл Tor Control**.

iptables

Установите и изучите работу с **iptables**. Вместо использования **экрана с контекстной фильтрацией**, которому на выходном узле пришлось бы

отслеживать тысячи соединений, мы настроим межсетевой экран без отслеживания контекста.

```
/etc/iptables/iptables.rules
```

```
*raw
-A PREROUTING -j NOTRACK
-A OUTPUT -j NOTRACK
COMMIT


*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -p tcp ! --syn -j ACCEPT
-A INPUT -p udp -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -i lo -j ACCEPT
COMMIT
```

`-A PREROUTING -j NOTRACK` и `-A OUTPUT -j NOTRACK` отключит отслеживание соединений в таблице `raw`.

`:INPUT DROP [0:0]` — цель (target) цепочки `INPUT` по умолчанию; отбрасывает входящий трафик, который не был отдельно разрешён опцией `ACCEPT`.

`:FORWARD DROP [0:0]` — цель цепочки `FORWARD` по умолчанию; используется для обычных маршрутизаторов, но не подходит для "луковых".

`:OUTPUT ACCEPT [0:0]` — цель цепочки `OUTPUT` по умолчанию; разрешает все исходящие соединения.

`-A INPUT -p tcp ! --syn -j ACCEPT` разрешает уже установленные по перечисленным ниже правилам входящие TCP соединения, а также TCP соединения от выходного узла.

`-A INPUT -p udp -j ACCEPT` разрешает все входящие UDP соединения, т.к. мы не используем отслеживание соединений (connection tracking).

`-A INPUT -p icmp -j ACCEPT` разрешает **ICMP**-пакеты.

`-A INPUT -p tcp --dport 443 -j ACCEPT` разрешает входящие соединения на `ORPort`.

`-A INPUT -p tcp --dport 80 -j ACCEPT` разрешает входящие соединения на `DirPort`.

`-A INPUT -i lo -j ACCEPT` разрешает входящие соединения на петлевой интерфейс.

Haveged

В статье **Haveged** описано, как определить, генерирует ли ваша система достаточно энтропии для управления большим количеством OpenSSL соединений; документация и советы: **[7]**, **[8]**.

pdnsd

Важно: Предполагается, что у вас доверенный (не цензурируемый) DNS-сервер.

Вы можете использовать **pdnsd** для локального кэширования DNS-запросов, тогда выходной узел сможет быстрее выполнять разрешение и посылать меньшее количество запросов внешнему DNS-серверу.

```
/etc/pdnsd.conf
```

```
...
perm_cache=102400                       ## (Значение по умолчанию)*100 = 1 Мбайт *
100 = 100 Мбайт
...
server {
    label= "resolvconf";
    file = "/etc/pdnsd-resolv.conf";    ## Чтобы не использовать файл
/etc/resolv.conf
    timeout=4;                          ## Период ожидания ответа сервера; может
быть значительно меньше, чем глобальное время ожидания
    uptest=query;                       ## Проверять доступность сервера, посылая
пустые DNS-запросы
    query_test_name=".";                ## Используется, если удалённый сервер
игнорирует пустые запросы
    interval=10m;                       ## Выполнять тестирование каждые 10 минут
    purge_cache=off;                    ## Игнорировать парамет TTL
    edns_query=yes;                     ## Использовать EDNS для исходящих
запросов, чтобы разрешить UDP-сообщения длиннее 512 байт; может вызвать проблемы на
некоторых устаревших системах
    preset=off;                         ## Перед проверкой состояния сервера
предполагать, что он выключен
 }
```

```
...
```

Такая настройка позволит кэшировать локально до 100 Мбайт DNS-запросов.

Uncensored DNS

Если ваш обычный DNS-сервер каким-то образом цензурируется или работает нестабильно, в статье **Alternative DNS services** описаны альтернативные сервера; добавьте нужное в отдельный server-разделе файла `/etc/pdnsd.conf` (см. **Pdnsd#DNS servers**).

TorDNS

Начиная с версий 0.2.x в Tor появился механизм перенаправления DNS-запросов. Чтобы его включить, добавьте строки ниже в файл настроек и **перезапустите** демон Tor:

```
/etc/tor/torrc
DNSPort 9053

AutomapHostsOnResolve 1

AutomapHostsSuffixes .exit,.onion
```

Теперь Tor будет принимать запросы на порт 9053 как обычный DNS-сервер и выполнять разрешение доменов через сеть Tor. К сожалению, через Tor выполняется разрешение только A-записей; MX и NS запросы будут проигнорированы. Дополнительную информацию можно найти в **документации TorDNS** для Debian.

Кроме того, появилась возможность посылать DNS-запросы посредством командного интерпретатора, командой `tor-resolve`. Например:

```
$ tor-resolve archlinux.org

66.211.214.131
```

Перенаправление DNS-запросов через TorDNS

Вашу систему можно настроить посылать *все* запросы через TorDNS вне зависимости от того, используется ли Tor для соединения с конечной целью. Для этого настройте систему использовать адрес `127.0.0.1` в качестве DNS-сервера и отредактируйте строку `DNSPort` в файле `/etc/tor/torrc`:

```
DNSPort 53
```

Альтернативное решение — использовать локальный кэширующий DNS-сервер, вроде **dnsmasq** или **pdnsd**. Кэш DNS позволит несколько компенсировать низкую скорость TorDNS по сравнению с обычными DNS-серверами. Далее приведены инструкции по настройке *dnsmasq*.

Установите пакет **dnsmasq** и укажите Tor прослушивать DNS запросы на порте 9053.

Задайте следующую конфигурацию dnsmasq:

```
/etc/dnsmasq.conf
no-resolv

port=53

server=127.0.0.1#9053

listen-address=127.0.0.1
```

Теперь dnsmasq будет ожидать локальных запросов и использовать TorDNS в качестве посредника. Отредактируйте файл `/etc/resolv.conf`, чтобы система

опрашивала только сервер dnsmasq:

```
/etc/resolv.conf
nameserver 127.0.0.1
```

Запустите демон dnsmasq.

Наконец, если вы используете **dhcpcd**, укажите ему не изменять настройки в файле `resolv.conf`:

```
/etc/dhcpcd.conf
nohook resolv.conf
```

Если строка `nohook` уже есть, то добавьте **resolv.conf** через запятую.

Torsocks

`torsocks` позволяет заставить приложение работать через сеть Tor без необходимости корректировать настройки самого приложения. Выдержка из руководства:

torsocks — обёртка между библиотекой torsocks и приложением с целью сделать каждое Интернет-соединение проходящим через сеть Tor.

Примеры использования:

```
$ torsocks elinks checkip.dyndns.org

$ torsocks wget -qO- https://check.torproject.org/ | grep -i congratulations
```

"Торификация"

В некоторых случаях более безопасно (и часто проще) выполнить сквозную "торифицикацию" всей системы вместо настройки отдельных приложений на

использование SOCKS-порта Tor и отслеживания утечек DNS. Для полной торификации сетевой экран **iptables** настраивают на пересылку всех исходящих пакетов (помимо собственно трафика Tor) на *TransPort*. В этом случае приложения не нужно настраивать для работы через Tor, хотя работа через *SOCKSPort* всё ещё возможна. Торификация также работает и для DNS (*DNSPort*), но при этом нужно учитывать, что Tor поддерживает только протокол TCP, и, за исключением DNS-запросов, UDP-пакеты через Tor посылаться не могут. Следовательно, они должны блокироваться для предотвращения утечек.

Торификация через iptables даёт сравнительно надёжную защиту, но она не является полноценной заменой приложениям виртуализированной торификации вроде Whonix или TorVM **[9]**. Торификация также не позволяет скрыть отпечаток браузера (fingerprint), поэтому рекомендуется воспользоваться "амнезийным" решением вроде **Tails**. Приложения всё ещё могут узнать имя хоста, MAC-адрес, серийный номер, временную зону вашего компьютера и другие данные, а с root-привилегиями смогут даже отключить сетевой экран целиком. Другими словами, полная торификация через iptables защищает от случайных соединений и DNS-утечек неправильно настроенного программного обеспечения, но не от вредоносных программ или ПО с серьёзными уязвимостями.

При работе через **прозрачный прокси-сервер** можно запустить сеанс Tor дважды, на клиенте и на прокси, в режиме "Tor поверх Tor". Такая схема работы обладает непредсказуемым поведением и потенциально небезопасна. В теории, трафик пользователя будет выполнять шесть прыжков вместо обычных трёх, но нет никакой гарантии, что дополнительные прыжки не совпадут с тремя изначальными — например, в другом порядке. Разработчики Tor Project считают, что это небезопасно **[10]**, **[11]**.

Ниже приведён файл настроек для утилит `iptables-restore` и `ip6tables-restore` (используются **службами** `iptables.service` и `ip6tables.service` соответственно).

Примечание: Правила в таблице nat принудительно перенаправляют исходящие соединения на TransPort или DNSPort, и блокируют всё, что торифицировать не удаётся.

- Флагами `--ipv6` и `--ipv4` задаются правила для конкретных версий протокола **IP**. Таким образом, утилиты `iptables-restore` и `ip6tables-restore` могут использовать один и тот же файл настроек.
- В случае явного указания флага `--ipv6` или `--ipv4`, утилита `ip*tables-restore` будет игнорировать правило, если оно установлено для другой версии протокола.
- `ip6tables` не поддерживает флаг `--reject-with`.

Убедитесь, что в файле `torrc` есть следующие строки:

```
SOCKSPort 9050

DNSPort 5353

TransPort 9040
```

Подробнее смотрите руководство **iptables(8)**.

Примечание: Если вы получили ошибку `iptables-restore: unable to initialize table 'nat'`, то нужно загрузить некоторые модули ядра:

```
# modprobe ip_tables iptable_nat ip_conntrack iptable-filter ipt_state
```

/etc/iptables/iptables.rules

```
*nat

:PREROUTING ACCEPT [6:2126]

:INPUT ACCEPT [0:0]

:OUTPUT ACCEPT [17:6239]

:POSTROUTING ACCEPT [6:408]



-A PREROUTING ! -i lo -p udp -m udp --dport 53 -j REDIRECT --to-ports 5353

-A PREROUTING ! -i lo -p tcp -m tcp --tcp-flags FIN,SYN,RST,ACK SYN -j REDIRECT --
to-ports 9040

-A OUTPUT -o lo -j RETURN

--ipv4 -A OUTPUT -d 192.168.0.0/16 -j RETURN

-A OUTPUT -m owner --uid-owner "tor" -j RETURN

-A OUTPUT -p udp -m udp --dport 53 -j REDIRECT --to-ports 5353

-A OUTPUT -p tcp -m tcp --tcp-flags FIN,SYN,RST,ACK SYN -j REDIRECT --to-ports 9040

COMMIT



*filter

:INPUT DROP [0:0]

:FORWARD DROP [0:0]

:OUTPUT DROP [0:0]



-A INPUT -i lo -j ACCEPT

-A INPUT -p icmp -j ACCEPT

-A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

--ipv4 -A INPUT -p tcp -j REJECT --reject-with tcp-reset
```

```
--ipv4 -A INPUT -p udp -j REJECT --reject-with icmp-port-unreachable
--ipv4 -A INPUT -j REJECT --reject-with icmp-proto-unreachable
--ipv6 -A INPUT -j REJECT
--ipv4 -A OUTPUT -d 127.0.0.0/8 -j ACCEPT
--ipv4 -A OUTPUT -d 192.168.0.0/16 -j ACCEPT
--ipv6 -A OUTPUT -d ::1/8 -j ACCEPT
-A OUTPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -m owner --uid-owner "tor" -j ACCEPT
--ipv4 -A OUTPUT -j REJECT --reject-with icmp-port-unreachable
--ipv6 -A OUTPUT -j REJECT
COMMIT
```

Данный файл также подходит для утилиты `ip6tables-restore`. Создайте символическую ссылку на него:

```
# ln -s /etc/iptables/iptables.rules /etc/iptables/ip6tables.rules
```

Убедитесь, что Tor работает, и **запустите/включите** службы `iptables` и `ip6tables`.

При необходимости можно **добавить указания** `Requires=iptables.service` и `Requires=ip6tables.service` к любому юниту systemd, чтобы выбранный процесс запускался строго после включения межсетевого экрана.

Советы и рекомендации

Возможности ядра

Если необходимо запустить Tor от обычного пользователя, и в то же время использовать порты меньше 1024, то можно воспользоваться функциональностью

ядра для выдачи процессу `/usr/bin/tor` разрешения выполнять привязку привилегированных портов:

```
# setcap CAP_NET_BIND_SERVICE=+eip /usr/bin/tor
```

Примечание: После обновления пакета Tor эти разрешения сбросятся. Создайте **хук** для автоматической выдачи разрешений после обновлений.

Если вы используете службу systemd, то выдать Tor соответствующие разрешения можно также через настройки системного демона. Это удобно тем, что разрешения не нужно возобновлять после каждого обновления Tor:

```
/etc/systemd/system/tor.service.d/netcap.conf
[Service]
CapabilityBoundingSet=
CapabilityBoundingSet=CAP_NET_BIND_SERVICE
AmbientCapabilities=
AmbientCapabilities=CAP_NET_BIND_SERVICE
```

Подробности можно найти на странице **superuser.com**.

Решение проблем

Проблема с параметром User

Если не удалось запустить демон Tor, выполните следующую команду от root (или воспользуйтесь утилитой **sudo**):

```
# tor
```

Ошибка

```
May 23 00:27:24.624 [warn] Error setting groups to gid 43: "Operation not
permitted".

May 23 00:27:24.624 [warn] If you set the "User" option, you must start Tor as
root.

May 23 00:27:24.624 [warn] Failed to parse/validate config: Problem with User
value. See logs for details.

May 23 00:27:24.624 [err] Reading config failed--see warnings above.
```

означает проблемы с параметром User. Скорее всего, один или несколько файлов/каталогов в каталоге `/var/lib/tor` не принадлежат пользователю `tor`. Это можно определить с помощью команды *find*:

```
find /var/lib/tor/ ! -user tor
```

Для всех выведенных файлов и каталогов нужно поменять параметр владельца. Можно сделать это индивидуально для каждого файла

```
# chown tor:tor /var/lib/tor/имя_файла
```

или отредактировать все сразу командой

```
# chown -R -v tor:tor /var/lib/tor
```

Теперь Tor должен запуститься без проблем.

Если запустить службу всё же не удаётся, запустите её от root. Измените имя пользователя в файле `/etc/tor/torrc`:

```
User tor
```

Затем отредактируйте файл службы systemd `/usr/lib/systemd/system/tor.service`:

```
[Service]

User=root

Group=root

Type=simple
```

В дальнейшем процесс будет запускаться от пользователя `tor`, поэтому измените UID и GID файлов на `tor` и добавьте разрешение на запись:

```
# chown -R tor:tor /var/lib/tor/

# chmod -R 700 /var/lib/tor
```

Сохраните изменения:

```
# systemctl --system daemon-reload
```

Наконец, **запустите** `tor.service`.

Проблемы с прокси tor-browser

`tor-browser`AUR обычно нормально работает без дополнительных настроек. Если установленный/настроенный прокси выдаёт ошибку `proxy server is refusing connections` при любой попытке установить соединение, попробуйте сбросить настройки, переместив или удалив каталог `~/.tor-browser`.

Пустой чёрный экран в tor-browser

При работе с **AppArmor** для доступа к ресурсам необходимо внести изменения в профиль Tor Browser **[12]**, **[13]**:

```
/etc/apparmor.d/local/torbrowser.Browser.firefox
owner /{dev,run}/shm/org.mozilla.*.* rw,
```

Смотрите также

- **Запуск клиента Tor на Linux/BSD/Unix**
- **Список статей о Tor для Unix**
- **Программы с интеграцией Tor**
- **Как включить Tor *Hidden Service***
- **Pluggable Transports для обфускации трафика Tor**