# Installing things (distilled)

## Installing: shadowsocks-libev with mbed TLS (formerly known as PolarSSL) in CentOS.

Install necessary packages:-

```
  yum install nano gcc gcc-c++ automake autoconf libtool make autoconf libtool
 git curl curl-devel zlib-devel perl perl-devel cpio gettext-
devel libxml2 libxml2-devel libxslt libxslt-devel
```

Download, compile and install mbed TLS and shadowsocks-libev:-

Using PolarSSL 1.3.13

```
  cd /usr/src/  wget https://tls.mbed.org/download/mbedtls-1.3.13-gpl.tgz  tar
 -xvf mbedtls-1.3.13-gpl.tgz  cd mbedtls-1.3.13  make && make install  cd /usr
/src/  git clone https://github.com/shadowsocks/shadowsocks-libev shadowsocks-
libev  cd shadowsocks-libev  ./configure --with-crypto-library=polarssl --with-
polarssl=/usr/src/mbedtls-1.3.13 --prefix=/usr && make && make install
```

Using mbedTLS 2.1.1

```
  cd /usr/src/  wget https://tls.mbed.org/download/mbedtls-2.1.1-gpl.tgz  tar
-xvf mbedtls-2.1.1-gpl.tgz  cd mbedtls-2.1.1  make && make install  cd /usr/sr
c/  git clone https://github.com/shadowsocks/shadowsocks-libev shadowsocks-lib
ev  cd shadowsocks-libev  ./configure --with-crypto-library=mbedtls --with-mbe
dtls=/usr/src/mbedtls-2.1.1 --with-mbedtls-include=/usr/src/mbedtls-2.1.1/incl
ude/mbedtls --with-mbedtls-
lib=/usr/src/mbedtls-2.1.1/library --prefix=/usr && make && make install
```

Make configuration files:-

```
  mkdir /etc/shadowsocks-libev/  nano /etc/shadowsocks-libev/config.json
```

Copy paste the lines below into config.json, and edit it further to suit your server:-

```
  {  "server":"your server IP",  "server_port":443,  "local_port":1080,  "pass
word":"your desired password",  "timeout":600,  "method":"aes-128-cfb",  "name
server":"209.244.0.3"  }
```

If your server CPU supports AES-NI, use `aes-128-cfb` for “method”, if it doesn’t, use `chacha20`. For “nameserver”, you can use Level3 or Google’s nameservers.
If you want the daemon to bind to all available IPv4 and IPv6 addresses you have, remove the “server” option. If you use a NAT-ed VPS, use your assigned RFC 1918 or use ‘0.0.0.0’.

Download the daemon init script:-

```
  cd /etc/init.d/  wget -O /etc/init.d/shadowsocks-libev https://raw.githubuse
rcontent.com/shadowsocks/shadowsocks-libev/master/rpm/SOURCES/etc/init.d/shado
wsocks-libev  chmod +x shadowsocks-libev  chkconfig --add shadowsocks-libev
```

Start the server:-

```
  service shadowsocks-libev start
```

When updating from git:-

Using PolarSSL 1.3.13

```
  cd /usr/src/shadowsocks-libev  make distclean  git pull  ./configure --with-
crypto-library=polarssl --with-polarssl=/usr/src/mbedtls-1.3.13 --prefix=/usr
&& make && make install  service shadowsocks-libev restart
```

Using mbedTLS 2.1.1

```
  cd /usr/src/shadowsocks-libev  make distclean  git pull  ./configure --with-
crypto-library=mbedtls --with-mbedtls=/usr/src/mbedtls-2.1.1 --with-mbedtls-in
clude=/usr/src/mbedtls-2.1.1/include/mbedtls --with-mbedtls-lib=/usr/src/mbedt
ls-2.1.1/library --prefix=/usr && make && make install  service shadowsocks-
libev restart
```

Proxy chaining with haproxy:-

Situation:-

Client —> haproxy VPS —> shadowsocks VPS.

In the shadowsocks VPS, install and configure shadowsocks proxy as usual.

Then in haproxy VPS, install haproxy and enable the daemon.

```
yum -y install haproxy  chkconfig haproxy on
```

Now edit /etc/haproxy/haproxy.cfg

```
nano /etc/haproxy/haproxy.cfg
```

Modify it so that it looks like below:-

```
    #---------------------------------------------------------------------  #
Example configuration for a possible web application.  See the  # full configu
ration options online.  #  #   http://haproxy.1wt.eu/download/1.4/doc/configur
ation.txt  #  #---------------------------------------------------------------
------    #---------------------------------------------------------------------
--  # Global settings  #---------------------------------------------------------
---------------  global  # to have these messages end up in /var/log/haproxy.l
og you will  # need to:  #  # 1) configure syslog to accept network log events
.  This is done  #    by adding the '-r' option to the SYSLOGD_OPTIONS in  #
  /etc/sysconfig/syslog  #  # 2) configure local2 events to go to the /var/log
/haproxy.log  #   file. A line like the following can be added to  #   /etc/sy
sconfig/syslog  #  #    local2.*                       /var/log/haproxy.log  #
  log         127.0.0.1 local2    chroot      /var/lib/haproxy  pidfile     /v
ar/run/haproxy.pid  maxconn     4000  user        haproxy  group       haproxy
  daemon  ulimit-n  51200    # turn on stats unix socket  stats socket /var/li
b/haproxy/stats    #---------------------------------------------------------
-----------  # common defaults that all the 'listen' and 'backend' sections wi
ll  # use if not designated in their block  #-------------------------------
---------------------------------  #defaults  #    mode
http  #    log                     global  #    option                  httplo
g  #    option                  dontlognull  #    option http-server-close  #
   option forwardfor       except 127.0.0.0/8  #    option                  re
dispatch  #    retries                 3  #    timeout http-request    10s  #
   timeout queue           1m  #    timeout connect         10s  #    timeout
client          1m  #    timeout server          1m  #    timeout http-keep-al
ive 10s  #    timeout check           10s  #    maxconn                 3000
  defaults  log global  mode    tcp  option  dontlognull  timeout connect
    20s  timeout client           2m  timeout server          2m    #--------
-------------------------------------------------------------  # main frontend
 which proxys to the backends  #-------------------------------------------
----------------------  #frontend  main *:5000  #    acl url_static       pat
h_beg       -i /static /images /javascript /stylesheets  #    acl url_static
     path_end       -i .jpg .gif .png .css .js    #    use_backend static
     if url_static  #    default_backend             app    frontend ss-in01
bind *:443  default_backend ss-out01      #-----------------------------------
---------------------------------  # static backend for serving up images, st
ylesheets and such  #-----------------------------------------------------
------------  #backend static  #    balance     roundrobin  #    server      s
tatic 127.0.0.1:4331 check    #-------------------------------------------
----------------------  # round robin balancing between the various backends
```

```
#---------------------------------------------------------------------  #backe
nd app  #    balance     roundrobin  #    server  app1 127.0.0.1:5001 check  #
    server  app2 127.0.0.1:5002 check  #    server  app3 127.0.0.1:5003 check
 #    server  app4 127.0.0.1:5004 check    backend ss-out01          server se
rver1 YOUR_SHADOWSOCKS_VPS_IP_SERVER:443 maxconn 20480
```

Save the haproxy.cfg file, then start the daemon.

```
service haproxy start
```

## Install:utorrent server on CentOS x64

Download the approriate .tar.gz file from http://www.utorrent.com/downloads/linux to your user home directory (you are not going to run this as root, but as ‘Joe’).

Untar the said file:-

```
    cd /home/Joe  tar -xvf utserver.tar.gz
```

Move ‘utserver’ and ‘webgui.zip’ file to the home directory, then chmod it to be executable.

```
    chmod +x utserver
```

Compile and install OpenSSL 0.9.8

```
    yum groupinstall "Development Tools"  cd /usr/src  wget http://www.openssl
.org/source/openssl-0.9.8zc.tar.gz  untar -xvf openssl-0.9.8zc.tar.gz  cd open
ssl-0.9.8zc  ./config shared  make && make install
```

Link to the compiled OpenSSL

```
    ln -s /usr/local/ssl/lib/libssl.so /usr/lib64/libssl.so.0.9.8  ln -s /usr/
local/ssl/lib/libcrypto.so /usr/lib64/libcrypto.so.0.9.8
```

Start utorrent :-

```
    su Joe -c /home/Joe/utserver
```

Open utorrent in your web browser:-

```
http://[your-IP-here]:8080/gui
```

username: admin
empty password

## Install: transmission-daemon on CentOS.

Install necessary packages:-

```
yum install nano gcc make openssl-devel curl-devel intltool gcc-c++ m4 automake libtool gettext
```

Compile and install libevent2:-

```
cd /usr/src wget http://downloads.sourceforge.net/project/levent/libevent/libevent-2.0/libevent-2.0.22-stable.tar.gz tar zxvf libevent-2.0.22-stable.tar.gz cd libevent-2.0.22-stable ./configure --prefix=/opt/libevent make make install
```

Compile and install transmission-daemon:-

```
cd /usr/src wget http://download.transmissionbt.com/files/transmission-2.82.tar.xz tar xvf transmission-2.82.tar.xz cd transmission-2.82 export PKG_CONFIG_PATH=/opt/libevent/lib/pkgconfig ./configure --prefix=/opt/transmission make && make install
```

Initial configuration:-

```
su Joe -c "/opt/transmission/bin/transmission-daemon" killall transmission-daemon nano /home/Joe/.config/transmission-daemon/settings.json
```

Edit settings.json to fit your preferences. See https://trac.transmissionbt.com/wiki/EditConfigFiles for more information.

Restart transmission-daemon:-

```
su Joe -c "/opt/transmission/bin/transmission-daemon"
```

Make transmission-daemon start at boot:-

```
nano /etc/rc.local
```

Paste the *su Joe -c "/opt/transmission/bin/transmission-daemon"* command just above the "Exit 0!" line.

## Install: Hentai@Home client on CentOS headless VPS.

This assumes that you already have been approved (you must have Client ID and Client Key). Do not run as root, but as 'Joe'.

Install OpenJDK.

```
yum install java-1.7.0-openjdk
```

Download and install Hentai@Home client.

```
cd /home/Joe  # Download latest version at http://g.e-hentai.org/hentaiathome.php  wget http://hentaiathome.net/get/HentaiAtHome_1.2.5.zip  unzip HentaiAtHome_1.2.5.zip  chown -R Joe:Joe /home/Joe
```

Start the Hentai@Home client.

```
cd /home/Joe  su -c Joe 'java -jar /home/Joe/HentaiAtHome.jar'
```

Enter Client ID and Client Key (need to be only done once) and you should be done. Client configuration is done at the e-hentai website.

### Share this:

- Facebook
- Google
- Twitter
- Pinterest
-

### Like this:

Like Loading...