shadowsocks-libev

Intro

Shadowsocks-libev is a lightweight secured SOCKS5 proxy for embedded devices and low-end boxes.

It is a port of Shadowsocks created by @clowwindy, and maintained by @madeye and @linusyang.

Current version: 3.1.3 | Changelog

Travis CI: 

Features

Shadowsocks-libev is written in pure C and depends on libev. It's designed to be a lightweight implementation of shadowsocks protocol, in order to keep the resource usage as low as possible.

For a full list of feature comparison between different versions of shadowsocks, refer to the Wiki page.

Prerequisites

Get the latest source code

To get the latest source code, you should also update the submodules as following:

```
git clone https://github.com/shadowsocks/shadowsocks-libev.git
cd shadowsocks-libev
git submodule update --init --recursive
```

Build and install with recent libsodium

You have to install libsodium at least 1.0.8, but recommended 1.0.12 or later version before building. See Directly build and install on UNIX-like system.

Installation

Distribution-specific guide

- Debian & Ubuntu
  - Install from repository
  - Build deb package from source
  - Configure and start the service
- Fedora & RHEL
  - Build from source with centos
  - Install from repository
- Archlinux
- NixOS
- Nix
- Directly build and install on UNIX-like system
- FreeBSD
- OpenWRT
- OS X
- Windows (MinGW)
- Docker

Pre-build configure guide

For a complete list of available configure-time option, try `configure --help`.

Debian & Ubuntu

Install from repository

Shadowsocks-libev is available in the official repository for following distributions:

- Debian 8 or higher, including oldstable (jessie), stable (stretch), testing (buster) and unstable (sid)
- Ubuntu 16.10 or higher

```
sudo apt update
sudo apt install shadowsocks-libev
```

For **Debian 8 (Jessie)** users, please install it from `jessie-backports-sloppy`: We strongly encourage you to install shadowsocks-libev from `jessie-backports-sloppy`. For more info about backports, you can refer Debian Backports.

```
sudo sh -c 'printf "deb http://deb.debian.org/debian jessie-backports main\n" > /etc/apt/sources.list.d/jessie-backpc
sudo sh -c 'printf "deb http://deb.debian.org/debian jessie-backports-sloppy main" >> /etc/apt/sources.list.d/jessie-
sudo apt update
sudo apt -t jessie-backports-sloppy install shadowsocks-libev
```

For **Debian 9 (Stretch)** users, please install it from `stretch-backports`: We strongly encourage you to install shadowsocks-libev from `stretch-backports`. For more info about backports, you can refer Debian Backports.

```
sudo sh -c 'printf "deb http://deb.debian.org/debian stretch-backports main" > /etc/apt/sources.list.d/stretch-backpc
sudo apt update
sudo apt -t stretch-backports install shadowsocks-libev
```

For **Ubuntu 14.04 and 16.04** users, please install from PPA:

```
sudo apt-get install software-properties-common -y
sudo add-apt-repository ppa:max-c-lv/shadowsocks-libev -y
sudo apt-get update
sudo apt install shadowsocks-libev
```

Build deb package from source

Supported distributions:

- Debian 8, 9 or higher
- Ubuntu 14.04 LTS, 16.04 LTS, 16.10 or higher

You can build shadowsocks-libev and all its dependencies by script:

```
mkdir -p ~/build-area/
cp ./scripts/build_deb.sh ~/build-area/
cd ~/build-area
./build_deb.sh
```

For older systems, building `.deb` packages is not supported. Please try to build and install directly from source. See the Linux section below.

Note for Debian 8 (Jessie) users to build their own deb packages:

We strongly encourage you to install shadowsocks-libev from `jessie-backports-sloppy`. If you insist on building from source, you will need to manually install libsodium from `jessie-backports-sloppy`, **NOT** libsodium in main repository.

For more info about backports, you can refer Debian Backports.

```
cd shadowsocks-libev
sudo sh -c 'printf "deb http://deb.debian.org/debian jessie-backports main" > /etc/apt/sources.list.d/jessie-backport
sudo sh -c 'printf "deb http://deb.debian.org/debian jessie-backports-sloppy main" >> /etc/apt/sources.list.d/jessie-
sudo apt-get install --no-install-recommends devscripts equivs
mk-build-deps --root-cmd sudo --install --tool "apt-get -o Debug::pkgProblemResolver=yes --no-install-recommends -y"
./autogen.sh && dpkg-buildpackage -b -us -uc
cd ..
sudo dpkg -i shadowsocks-libev*.deb
```

Note for Debian 9 (Stretch) users to build their own deb packages:

We strongly encourage you to install shadowsocks-libev from `stretch-backports`. If you insist on building from source, you will need to manually install libsodium from `stretch-backports`, **NOT** libsodium in main repository.

For more info about backports, you can refer Debian Backports.

```
cd shadowsocks-libev
sudo sh -c 'printf "deb http://deb.debian.org/debian stretch-backports main" > /etc/apt/sources.list.d/stretch-backpc
sudo apt-get install --no-install-recommends devscripts equivs
mk-build-deps --root-cmd sudo --install --tool "apt-get -o Debug::pkgProblemResolver=yes --no-install-recommends -y"
./autogen.sh && dpkg-buildpackage -b -us -uc
cd ..
sudo dpkg -i shadowsocks-libev*.deb
```

Configure and start the service

```
# Edit the configuration file
sudo vim /etc/shadowsocks-libev/config.json

# Edit the default configuration for debian
sudo vim /etc/default/shadowsocks-libev

# Start the service
sudo /etc/init.d/shadowsocks-libev start    # for sysvinit, or
sudo systemctl start shadowsocks-libev      # for systemd
```

Fedora & RHEL

Supported distributions:

- Recent Fedora versions (until EOL)
- RHEL 6, 7 and derivatives (including CentOS, Scientific Linux)

Build from source with centos

If you are using CentOS 7, you need to install these prequirement to build from source code:

```
yum install epel-release -y
yum install gcc gettext autoconf libtool automake make pcre-devel asciidoc xmlto c-ares-devel libev-devel libsodium-c
```

Install from repository

Enable repo via `dnf`:

```
su -c 'dnf copr enable librehat/shadowsocks'
```

Or download yum repo on Fedora Copr and put it inside `/etc/yum.repos.d/`. The release `Epel` is for RHEL and its derivatives.

Then, install `shadowsocks-libev` via `dnf`:

```
su -c 'dnf update'
su -c 'dnf install shadowsocks-libev'
```

or `yum`:

```
su -c 'yum update'
su -c 'yum install shadowsocks-libev'
```

The repository is maintained by @librehat, any issues, please report here

Archlinux

```
sudo pacman -S shadowsocks-libev
```

Please refer to downstream PKGBUILD script for extra modifications and distribution-specific bugs.

NixOS

```
nix-env -iA nixos.shadowsocks-libev
```

Nix

```
nix-env -iA nixpkgs.shadowsocks-libev
```

Linux

In general, you need the following build dependencies:

- autotools (autoconf, automake, libtool)
- gettext
- pkg-config
- libmbedtls
- libsodium
- libpcre3 (old pcre library)
- libev
- libc-ares
- asciidoc (for documentation only)
- xmlto (for documentation only)

Notes: Fedora 26 libsodium version >= 1.0.12, so you can install via dnf install libsodium instead build from source.

If your system is too old to provide libmbedtls and libsodium (later than **v1.0.8**), you will need to either install those libraries manually or upgrade your system.

If your system provides with those libraries, you **should not** install them from source. You should jump this section and install them from distribution repository instead.

For some of the distributions, you might install build dependencies like this:

```
# Installation of basic build dependencies
## Debian / Ubuntu
sudo apt-get install --no-install-recommends gettext build-essential autoconf libtool libpcre3-dev asciidoc xmlto lib
## CentOS / Fedora / RHEL
sudo yum install gettext gcc autoconf libtool automake make asciidoc xmlto c-ares-devel libev-devel
## Arch
sudo pacman -S gettext gcc autoconf libtool automake make asciidoc xmlto c-ares libev

# Installation of Libsodium
export LIBSODIUM_VER=1.0.13
wget https://download.libsodium.org/libsodium/releases/libsodium-$LIBSODIUM_VER.tar.gz
tar xvf libsodium-$LIBSODIUM_VER.tar.gz
pushd libsodium-$LIBSODIUM_VER
./configure --prefix=/usr && make
sudo make install
popd
sudo ldconfig
```

```
# Installation of MbedTLS
export MBEDTLS_VER=2.6.0
wget https://tls.mbed.org/download/mbedtls-$MBEDTLS_VER-gpl.tgz
tar xvf mbedtls-$MBEDTLS_VER-gpl.tgz
pushd mbedtls-$MBEDTLS_VER
make SHARED=1 CFLAGS=-fPIC
sudo make DESTDIR=/usr install
popd
sudo ldconfig

# Start building
./autogen.sh && ./configure && make
sudo make install
```

You may need to manually install missing softwares.

FreeBSD

```
su
cd /usr/ports/net/shadowsocks-libev
make install
```

Edit your config.json file. By default, it's located in /usr/local/etc/shadowsocks-libev.

To enable shadowsocks-libev, add the following rc variable to your /etc/rc.conf file:

```
shadowsocks_libev_enable="YES"
```

Start the Shadowsocks server:

```
service shadowsocks_libev start
```

OpenWRT

The OpenWRT project is maintained here: openwrt-shadowsocks.

OS X

For OS X, use Homebrew to install or build.

Install Homebrew:

```
ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Install shadowsocks-libev:

```
brew install shadowsocks-libev
```

Windows (MinGW)

To build Windows native binaries, the recommended method is to use Docker:

- On Windows: double-click `make.bat` in `docker\mingw`
- On Unix-like system:

```
cd shadowsocks-libev/docker/mingw
make
```

A tarball with 32-bit and 64-bit binaries will be generated in the same directory.

You could also manually use MinGW-w64 compilers to build in Unix-like shell (MSYS2/Cygwin), or cross-compile on Unix-like systems (Linux/MacOS). Please refer to build scripts in `docker/mingw`.

Currently you need to use a patched libev library for MinGW:

- https://github.com/shadowsocks/libev/archive/mingw.zip

Notice that TCP Fast Open (TFO) is only available on **Windows 10**, **1607** or later version (precisely, build >= 14393). If you are using **1709** (build 16299) or later version, you also need to run the following command in PowerShell/Command Prompt **as Administrator** and **reboot** to use TFO properly:

```
netsh int tcp set global fastopenfallback=disabled
```

Docker

As you expect, simply pull the image and run.

```
docker pull shadowsocks/shadowsocks-libev
docker run -e PASSWORD=<password> -p<server-port>:8388 -p<server-port>:8388/udp -d shadowsocks/shadowsocks-libev
```

More information about the image can be found here.

Usage

For a detailed and complete list of all supported arguments, you may refer to the man pages of the applications, respectively.

```
ss-[local|redir|server|tunnel|manager]

   -s <server_host>           host name or ip address of your remote server

   -p <server_port>           port number of your remote server

   -l <local_port>            port number of your local server

   -k <password>              password of your remote server

   -m <encrypt_method>        Encrypt method: rc4-md5,
                              aes-128-gcm, aes-192-gcm, aes-256-gcm,
                              aes-128-cfb, aes-192-cfb, aes-256-cfb,
                              aes-128-ctr, aes-192-ctr, aes-256-ctr,
                              camellia-128-cfb, camellia-192-cfb,
                              camellia-256-cfb, bf-cfb,
                              chacha20-poly1305, chacha20-ietf-poly1305
                              salsa20, chacha20 and chacha20-ietf.

   [-f <pid_file>]            the file path to store pid

   [-t <timeout>]             socket timeout in seconds

   [-c <config_file>]         the path to config file

   [-i <interface>]           network interface to bind,
                              not available in redir mode

   [-b <local_address>]       local address to bind

   [-u]                       enable udprelay mode,
                              TPROXY is required in redir mode

   [-U]                       enable UDP relay and disable TCP relay,
                              not available in local mode

   [-L <addr>:<port>]         specify destination server address and port
                              for local port forwarding,
                              only available in tunnel mode

   [-6]                       Resovle hostname to IPv6 address first.

   [-d <addr>]                setup name servers for internal DNS resolver,
                              only available in server mode
```

```
    [--reuse-port]            Enable port reuse.

    [--fast-open]             enable TCP fast open,
                              only available in local and server mode,
                              with Linux kernel > 3.7.0

    [--acl <acl_file>]        config file of ACL (Access Control List)
                              only available in local and server mode

    [--manager-address <addr>] UNIX domain socket address
                              only available in server and manager mode

    [--mtu <MTU>]             MTU of your network interface.

    [--mptcp]                 Enable Multipath TCP on MPTCP Kernel.

    [--no-delay]              Enable TCP_NODELAY.

    [--executable <path>]     path to the executable of ss-server
                              only available in manager mode

    [--plugin <name>]         Enable SIP003 plugin. (Experimental)
    [--plugin-opts <options>] Set SIP003 plugin options. (Experimental)

    [-v]                      verbose mode
```

Transparent proxy

The latest shadowsocks-libev has provided a *redir* mode. You can configure your Linux-based box or router to proxy all TCP traffic transparently, which is handy if you use a OpenWRT-powered router.

```
# Create new chain
iptables -t nat -N SHADOWSOCKS
iptables -t mangle -N SHADOWSOCKS

# Ignore your shadowsocks server's addresses
# It's very IMPORTANT, just be careful.
iptables -t nat -A SHADOWSOCKS -d 123.123.123.123 -j RETURN

# Ignore LANs and any other addresses you'd like to bypass the proxy
# See Wikipedia and RFC5735 for full list of reserved networks.
# See ashi009/bestroutetb for a highly optimized CHN route list.
iptables -t nat -A SHADOWSOCKS -d 0.0.0.0/8 -j RETURN
iptables -t nat -A SHADOWSOCKS -d 10.0.0.0/8 -j RETURN
iptables -t nat -A SHADOWSOCKS -d 127.0.0.0/8 -j RETURN
iptables -t nat -A SHADOWSOCKS -d 169.254.0.0/16 -j RETURN
iptables -t nat -A SHADOWSOCKS -d 172.16.0.0/12 -j RETURN
iptables -t nat -A SHADOWSOCKS -d 192.168.0.0/16 -j RETURN
iptables -t nat -A SHADOWSOCKS -d 224.0.0.0/4 -j RETURN
iptables -t nat -A SHADOWSOCKS -d 240.0.0.0/4 -j RETURN

# Anything else should be redirected to shadowsocks's local port
iptables -t nat -A SHADOWSOCKS -p tcp -j REDIRECT --to-ports 12345

# Add any UDP rules
ip route add local default dev lo table 100
ip rule add fwmark 1 lookup 100
iptables -t mangle -A SHADOWSOCKS -p udp --dport 53 -j TPROXY --on-port 12345 --tproxy-mark 0x01/0x01

# Apply the rules
iptables -t nat -A PREROUTING -p tcp -j SHADOWSOCKS
iptables -t mangle -A PREROUTING -j SHADOWSOCKS

# Start the shadowsocks-redir
ss-redir -u -c /etc/config/shadowsocks.json -f /var/run/shadowsocks.pid
```

Shadowsocks over KCP

It's quite easy to use shadowsocks and KCP together with kcptun.

The goal of shadowsocks over KCP is to provide a fully configurable, UDP based protocol to improve poor connections, e.g. a high packet loss 3G network.

Setup your server

```
server_linux_amd64 -l :21 -t 127.0.0.1:443 --crypt none --mtu 1200 --nocomp --mode normal --dscp 46 &
ss-server -s 0.0.0.0 -p 443 -k passwd -m chacha20 -u
```

Setup your client

```
client_linux_amd64 -l 127.0.0.1:1090 -r <server_ip>:21 --crypt none --mtu 1200 --nocomp --mode normal --dscp 46 &
ss-local -s 127.0.0.1 -p 1090 -k passwd -m chacha20 -l 1080 -b 0.0.0.0 &
ss-local -s <server_ip> -p 443 -k passwd -m chacha20 -l 1080 -U -b 0.0.0.0
```

Security Tips

Although shadowsocks-libev can handle thousands of concurrent connections nicely, we still recommend setting up your server's firewall rules to limit connections from each user:

```
# Up to 32 connections are enough for normal usage
iptables -A INPUT -p tcp --syn --dport ${SHADOWSOCKS_PORT} -m connlimit --connlimit-above 32 -j REJECT --reject-
with tcp-reset
```

License