En Français

Lines aggregation

You can now use OpenMPTCProuter. Based on OpenWrt/LEDE with MPTCP support and Shadowsocks, this permit to aggregate Internet connections easily.

The goal is to aggregate 2 lines and have a fixed IPv4 / IPv6 output. The operation is similar to OVH's OverTheBox.

In this tutorial we will use MultiPath TCP (MPTCP) as well as proxy socks shadowsocks in order to aggregate two lines (for example an ADSL line and a 4G line).

A remote server is required, a Kimsufi VPS SSD 1 under Debian will suffice, the addition of the flow of the 2 lines is less than 100Mb/s.

A raspberry pi 3 is used as a router. A modem / box is considered to have the address 192.168.0.254 and another one has the address 192.168.2.254, the raspberry having address 192.168.1.1

This guide if for Debian Jessie

Server configuration

MPTCP

Just use apt, as shown here:

```
sudo apt-key adv --keyserver hkp://keys.gnupg.net --recv-keys 379CE192D401AB61
echo "deb https://dl.bintray.com/cpaasch/deb jessie main" >> /etc/apt/sources.list
sudo apt-get update
sudo apt-get install linux-mptcp
```

Shadowsocks

Shadowsocks is a socks proxy with mptcp support.

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

You should follow recommendation here to optimize Shadowsocks.

An example of working */etc/sysctl.d/local.conf*:

```
# max open files
fs.file-max = 51200
# max read buffer
net.core.rmem_max = 67108864
# max write buffer
net.core.wmem_max = 67108864
# default read buffer
net.core.rmem_default = 65536
# default write buffer
net.core.wmem_default = 65536
# max processor input queue
net.core.netdev_max_backlog = 4096
# max backlog
net.core.somaxconn = 4096
# resist SYN flood attacks
net.ipv4.tcp_syncookies = 1
# reuse timewait sockets when safe
net.ipv4.tcp_tw_reuse = 1
# turn off fast timewait sockets recycling
net.ipv4.tcp_tw_recycle = 0
# short FIN timeout
net.ipv4.tcp_fin_timeout = 30
# short keepalive time
net.ipv4.tcp_keepalive_time = 1200
# outbound port range
net.ipv4.ip_local_port_range = 10000 65000
# max SYN backlog
net.ipv4.tcp_max_syn_backlog = 4096
# max timewait sockets held by system simultaneously
net.ipv4.tcp_max_tw_buckets = 5000
# turn on TCP Fast Open on both client and server side
net.ipv4.tcp_fastopen = 3
# TCP receive buffer
net.ipv4.tcp_rmem = 4096 87380 67108864
# TCP write buffer
net.ipv4.tcp_wmem = 4096 65536 67108864
# turn on path MTU discovery
net.ipv4.tcp_mtu_probing = 1
# for high-latency network
net.ipv4.tcp_congestion_control = iloa
```

The configuration file */etc/shadowsocks-libev/config.json* should look like this:

```
{
    "server":["[::0]", "0.0.0.0"],
    "server_port":8388,
    "local_port":1080,
    "password":"yourpassword",
    "timeout":600,
    "method":"aes-256-cfb",
    "verbose":2,
    "prefer_ipv6": true
}
```

To launch it manually:

```
ss-server -c /etc/shadowsocks-libev/config.json --mptcp -u --fast-open
```

To laucnh it automatically, modify */etc/default/shadowsocks-libev* and add **-u --fast-open** in **DAEMON_ARGS** option.

Firewall

It is more than recommended to set up a firewall on the server, but this is outside the scope of this tutorial. Port 8388 must be opened in TCP and UDP to the IPs of the aggregated lines.

Router configuration

Raspbian

First step it to put Rapsbian on an SD card.

Download Raspbian at https://www.raspberrypi.org/downloads/raspbian/ then copy it on the sd card:

```
dd bs=4M if=2017-04-10-raspbian-jessie.img of=/dev/sdx
```

To enable SSH:

```
mount /dev/sdx1 /mnt
touch ssh /mnt/boot
```

MPTCP

Once the Rapsberry is booted on the sd card, you must add akernel with MPTCP support:

```
cd /usr/src
wget https://github.com/Ysurac/raspberry_kernel_mptcp/archive/4.4.50_mptcp0.92.tar.gz
tar xzvf 4.4.50_mptcp0.92.tar.gz
```

Configure the kernel:

```
apt-get install libncurses5-dev
KERNEL=kernel7
make bcm2709_defconfig
make menuconfig
```

Don't put IPv6 as module, and add olia TCP congestion. Check also that TPROXY is enabled as module.

You also use this .config.

Compile the kernel:

```
make -j4 zImage modules dtbs
sudo make modules_install
sudo cp arch/arm/boot/dts/*.dtb /boot/
sudo cp arch/arm/boot/dts/overlays/*.dtb* /boot/overlays/
sudo cp arch/arm/boot/dts/overlays/README /boot/overlays/
sudo cp arch/arm/boot/zImage /boot/kernel7.img
```

Network

You have to create as many virtual interfaces as lines to aggregate.

Here is an example of */etc/network/interfaces* file:

```
auto lo
iface lo inet loopback
auto eth0
iface eth0 inet static
 address 192.168.1.1
 netmask 255.255.255.0

auto eth0:line1
iface eth0:line1 inet static
 address 192.168.0.2
 netmask 255.255.255.0
 gateway 192.168.0.254
 post-up /opt/mptcp/rules.sh

auto eth0:line2
iface eth0:line2 inet static
 address 192.168.2.2
 netmask 255.255.255.0
 gateway 192.168.2.254
```

With the file */opt/mptcp/rules.sh*:

```
ip rule add from 192.168.0.2 table 1
ip rule add from 192.168.2.2 table 2
ip route add 192.168.0.0/24 dev eth0:line1 scope link table 1
ip route add default via 192.168.0.254 dev eth0:line1 table 1
ip route add 192.168.2.0/24 dev eth0:line2 scope link table 2
ip route add default via 192.168.2.254 dev eth0:line2 table 2
ip route add default scope global nexthop via 192.168.0.254 dev eth0:line1

ss-nat -s ipduserveur -l 1080 -u
#ss-nat6 -s ipv6duserveur -l 1080 -u
```

Shadowsocks

```
sudo apt-get install git ipset devscripts equivs
sudo sh -c 'printf "deb http://httpredir.debian.org/debian jessie-backports main" > /etc/apt/sources.list.d/jessie-backports.list'
sudo apt install libsodium-dev
```

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

cd shadowsocks-libev
mk-build-deps --root-cmd sudo --install --tool "apt-get -o Debug::pkgProblemResolver=yes --no-install-recommends -y –force-yes"
./autogen.sh && dpkg-buildpackage -b -us -uc
cd ..
sudo dpkg -i shadowsocks-libev*.deb
```

You should follow recommendation here to optimize Shadowsocks.

An example of working */etc/sysctl.d/local.conf*:

```
# max open files
fs.file-max = 51200
# max read buffer
net.core.rmem_max = 67108864
# max write buffer
net.core.wmem_max = 67108864
# default read buffer
net.core.rmem_default = 65536
# default write buffer
net.core.wmem_default = 65536
# max processor input queue
net.core.netdev_max_backlog = 4096
# max backlog
net.core.somaxconn = 4096
# resist SYN flood attacks
net.ipv4.tcp_syncookies = 1
# reuse timewait sockets when safe
net.ipv4.tcp_tw_reuse = 1
# turn off fast timewait sockets recycling
net.ipv4.tcp_tw_recycle = 0
# short FIN timeout
net.ipv4.tcp_fin_timeout = 30
# short keepalive time
net.ipv4.tcp_keepalive_time = 1200
# outbound port range
net.ipv4.ip_local_port_range = 10000 65000
# max SYN backlog
net.ipv4.tcp_max_syn_backlog = 4096
# max timewait sockets held by system simultaneously
net.ipv4.tcp_max_tw_buckets = 5000
# turn on TCP Fast Open on both client and server side
net.ipv4.tcp_fastopen = 3
# TCP receive buffer
net.ipv4.tcp_rmem = 4096 87380 67108864
# TCP write buffer
net.ipv4.tcp_wmem = 4096 65536 67108864
# turn on path MTU discovery
net.ipv4.tcp_mtu_probing = 1
# for high-latency network
net.ipv4.tcp_congestion_control = iloa
```

Config file */etc/shadowsocks-libev/config.json*:

```
{
    "server":"ipduserveur",
    "server_port":8388,
    "local":"::",
    "local_port":1080,
    "password":"votrepassword",
    "timeout":600,
    "method":"aes-256-cfb",
    "verbose":2
}
```

To launch it manually:

```
ss-redir -c /etc/shadowsocks-libev/config.json --mptcp -u --fast-open
```

To run it automatically, modify */etc/default/shadowsocks-libev* and add **-u --fast-open** to **DAEMON_ARGS** option, and replace ss-server in the **DAEMON** option by **ss-redir** in */etc/init.d*.