
title: 使用DigitalOcean搭建ShadowSocks及其优化

copyright: true

date: 2017-12-03 10:13

tags:

categories:

前言

在github的教育资助下买了DigitalOcean的服务器，送了50刀，不错啊，其实用别人的邀请码可以再多10刀的。

一直看朋友自己搭梯子，挺好用的，刚好买了vps，就琢磨着干脆自己来搭个梯子。

本文将包括一下内容：

1. DigitalOcean注册与配置
2. 服务器端ss配置
3. bbr加速

一. 概念介绍

DigitalOcean（以下简称DO）是美国的一家云主机提供商，其目前拥有包括美国（纽约、旧金山）、荷兰阿姆斯特丹、新加坡、英国伦敦、德国法兰克福、加拿大多伦多和印度班加罗尔共8个城市12个可选机房节点。其采用KVM虚拟机，配置高性能的SSD做储存，拥有IPv6可选项，加上服务器配备的是1000M端口，以512M内存为起点，再加上DO较低的价格\$5起/月（而且还有其他方式可以获得优惠，下面会讲到），受到很多的追捧。选用这么一个主机不仅可以用来搭建ss，有一定网页制作基础的同学也可以搭建自己的个人网站，甚至可以选择DO提供的高级套餐来进行机器学习等大规模云计算。

Shadowsocks 是由大神 clowwindy（传说是妹纸）开发的一款基于SSH的通讯方式，或者理解为SSH的一个变种。简单来说，其原理就是在墙的两边分别建立起ssh-local和ssh-server两个端口，用户PC首先基于Socks5协议与ssh-local进行通讯，其信息由ssh-local进行多种可选方式的加密后通过墙与另一侧的ssh-server通讯，该信息由ssh-server解密后再对真正的目标地址进行请求。因为ssh-local与ssh-server之间的加密理论无法解密且不含特征信息，就是普通的TCP包，因此无法被捕获或拦截。由于该程序实在太过优秀，因此作者本人被请去喝了茶后就把github上的项目删去了，只留下一句：

Removed according to regulations.

ps: 以上是cope的。

二. DigitalOcean注册与购买

DigitalOcean的官网目前在中国大陆可以正常访问，欢迎通过我的[邀请链接进行注册](#).即获得10美元的优惠。现在最低价为5美元/月，因此可以说得到了免费两个月的额度。之后如果您付款超过\$25美元，我也会得到网站一定的奖励，是个互惠互利的事儿:)

1.首先在主页即可注册: [DigitalOcean官网](#) 我的邀请链接可以得到10美金



注测后添加付款方式：需要绑定“VISA信用卡”或“PayPal账户”用于付款。采用绑定“VISA信用卡”的方式，会先从账户中扣除1刀，10分钟左右返还，确定账户的可以付款，不需要提前付费；采用绑定“PayPal账户”的方式，必须首先至少充值5刀。

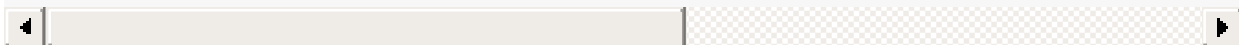
2.创建“Droplet”虚拟机

- 推荐的配置如下：

1. 镜像和大小的选择



注：DigitalOcean是Shadowsocks Wiki上推荐的VPS之一，
同时Ubuntu是推荐的系统，
详情参见「Shadowsocks」([https://github.com/shadowsocks/shadowsocks/wiki/Shadowssocks-%E7%](https://github.com/shadowsocks/shadowsocks/wiki/Shadowssocks-%E7%BB%A8%E6%B0%B8)



2.选择服务器所在地区

服务器地址可从[DigitalOcean测速](#)网址进行测速。



3.其他



- 勾选IPv6。IPv6可用于连接教育网的PT站等。
- 添加SSH公钥，用于SSH远程连接主机。生成密钥的命令为ssh-keygen -t rsa -C “yourmail”。

4.最后

可以取个名字



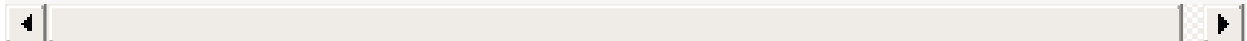
二、服务器端ss配置

如果没有配置ssh 你会发现自己的邮箱收到如下的一封邮件告诉了你IP地址（IP Address）、用户名（Username）和初始密码（Password）
如果配置了ssh，就可以直接登陆。

```
ssh root@IP
```

1.服务器端安装SS

```
apt-get update // 更新源中包列表
apt-get install python-pip // 安装pip
pip install --upgrade pip // 更新pip
apt-get install git
pip install git+https://github.com/shadowsocks/shadowsocks.git@master // 安装SS
```



2.安装chacha20加密

虽然shadowsocks的作者推荐使用 `aes-256-cfb` 的加密算法，但是这个算法对手机或者openwrt路由器的压力还是不小的，于是后来Google推出的 `chacha20` 算法因为其安全性和惊人的速度被现在更多人使用。

```
apt-get install build-essential
wget https://github.com/jedisct1/libsodium/releases/download/1.0.8/libsodium-1.0.8
tar xf libsodium-1.0.8.tar.gz && cd libsodium-1.0.8
./configure && make -j2
make install
ldconfig
```

3.服务器端配置

- 配置shadowsocks文件

```
vim /etc/shadowsocks.json
```

这是shadowsocks.json的详情：

```
{
    "server": "服务器的ip",
    "server_port": 8388,
    "local_address": "127.0.0.1",
    "local_port": 1080,
    "password": "你的密码",
    "timeout": 300,
    "method": "chacha20",
    "fast_open": false
}
```

- 服务器端启动与停止(一直启动即可)

```
ssserver -c /etc/shadowsocks.json -d start
ssserver -c /etc/shadowsocks.json -d stop
```

三、BBR加速

TCP-BBR: Linux的内核必须是4.9.0-rc8 内核及以后的版本 才能开启, 而锐速并不支持这个最新的内核版本, 所以TCP-BBR和锐速是不能共存的。

1.查看系统

```
uname -m
```

使用这个命令会返回你的系统位数, 如果是 x86_64 那就是64位, 如果是 i386 或 386 或 i686 或 686 那就是32位。

2.下载内核文件

- 64位系统

```
wget -O linux-image-4.10.1-amd64.deb http://kernel.ubuntu.com/~kernel-ppa/mainline/v
```



- 32位系统

```
wget -O linux-image-4.10.1-i386.deb http://kernel.ubuntu.com/~kernel-ppa/mainline/v
```



下载完是这样的:

VPS提示 (64位)

```
root@bbr1:~# wget -O linux-image-4.10.1-amd64.deb http://kernel.ubuntu.com/~kerne
converted 'http://kernel.ubuntu.com/~kernel-ppa/mainline/v4.10.1/linux-image-4.10
--2016-12-08 15:31-- http://kernel.ubuntu.com/~kernel-ppa/mainline/v4.9/linux-i
Resolving kernel.ubuntu.com (kernel.ubuntu.com)... 91.189.94.216
Connecting to kernel.ubuntu.com (kernel.ubuntu.com)|91.189.94.216|:80... connecte
HTTP request sent, awaiting response... 200 OK
Length: 60489324 (58M) [application/x-debian-package]
Saving to: 'linux-image-4.10.1-amd64.deb'
```

```
linux-image-4.10.1-040900 100%[=====>] 57.69M 3
```

```
2016-12-08 15:31 (4.05 MB/s) - 'linux-image-4.10.1-amd64.deb' saved [60489324/60
```



3.安装内核

- 64位系统

```
dpkg -i linux-image-4.10.1-amd64.deb
```

- 32位系统

```
dpkg -i linux-image-4.10.1-i386.deb
```

安装成功：

VPS提示

```
root@bbr1:~# dpkg -i linux-image-4.10.1-amd64.deb
Selecting previously unselected package linux-image-4.10.1-040900rc8-generic.
(Reading database ... 25141 files and directories currently installed.)
Preparing to unpack linux-image-4.10.1-040900rc8-generic_4.10.1-040900rc8.201612051443.deb ...
Done.
Unpacking linux-image-4.10.1-040900rc8-generic (4.10.1-040900rc8.201612051443) ..
Setting up linux-image-4.10.1-040900rc8-generic (4.10.1-040900rc8.201612051443) .
Running depmod.
update-initramfs: deferring update (hook will be called later)
Examining /etc/kernel/postinst.d.
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.10.1-040900rc8-generic
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.10.1-040900rc8-generic
update-initramfs: Generating /boot/initrd.img-4.10.1-040900rc8-generic
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 4.10.1-040900rc8-generic
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.10.1-040900rc8-generic
Found initrd image: /boot/initrd.img-4.10.1-040900rc8-generic
Found linux image: /boot/vmlinuz-3.16.0-4-amd64
Found initrd image: /boot/initrd.img-3.16.0-4-amd64
done
```

4.列出并删除旧内核

列出VPS 除了刚才安装的内核以外的 所有内核名称：

ps:这个代码是排除了刚才安装的内核，所以不会显示刚才安装的那个内核。删除了多余的内核后，就会什么都不显示！

```
dpkg -l|grep linux-image | awk '{print $2}' | grep -v 'linux-image-[4-9].[0-9]*.'
```

```
root@bbr1:~# dpkg -l|grep linux-image | awk '{print $2}' | grep -v 'linux-image-[4-5]
linux-image-3.16.0-4-amd64
linux-image-amd64
```

删除旧内核：

```
apt-get purge 旧内核名称
```

```
# !!!!! 示例!!!!
# 比如我是有这两个内核，所以我把这两个删掉，不要照着我的这个步骤做，自己换成你VPS的内核名称!!!
apt-get purge linux-image-3.16.0-4-amd64
apt-get purge linux-image-amd64
# !!!!! 示例!!!!
```

5.更新 grub 系统引导文件并重启VPS

```
# 更新系统引导文件
update-grub
# 重启VPS，执行后SSH链接会断开，等十几秒再重新连接继续执行下面的启动命令。
reboot
```

```
# VPS提示
```

```
root@bbr1:~# update-grub
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.10.1-040900rc8-generic
Found initrd image: /boot/initrd.img-4.10.1-040900rc8-generic
done
```

6.开启bbr

```
echo "net.core.default_qdisc=fq" >> /etc/sysctl.conf
echo "net.ipv4.tcp_congestion_control=bbr" >> /etc/sysctl.conf
# 执行这个保存生效更改。
sysctl -p
```

VPS提示

```
root@bbr1:~# sysctl -p
net.ipv6.conf.all.accept_ra = 2
net.core.default_qdisc = fq
net.ipv4.tcp_congestion_control = bbr
```

7.关闭bbr

```
sed -i '/net\.core\.default_qdisc=fq/d' /etc/sysctl.conf && sed -i '/net\.ipv4\.tc
sysctl -p
```

#执行完上面的代码，就使用reboot重启VPS后才能关闭bbr，重启后再用下面的查看bbr状态代码，查看是否reboot

8.查看bbr是否开启

- 执行下面命令，如果结果中有bbr，即证明你的内核已开启bbr。

```
sysctl net.ipv4.tcp_available_congestion_control
```

- 执行下面命令，看到有 tcp_bbr 模块，即说明bbr已启动

```
lsmod | grep bbr
```

VPS提示

```
root@bbr1:~# sysctl net.ipv4.tcp_available_congestion_control
net.ipv4.tcp_available_congestion_control = bbr cubic reno
```

=====

```
root@bbr1:~# lsmod | grep bbr
tcp_bbr                20480  0
```

这样就可以快乐的翻墙出去观看世界了 ^_^