

1. 注册谷歌云，账号类型选择个人；注册地址：
2. 国家我选的中国，付款信用卡必须是visa或者mastercard,JCB
3. 填写好的信息截图：

Google Cloud Platform

免费试用 Cloud Platform Google

客户信息

帐号类型 ⓘ 个人

姓名和地址 ⓘ 中国

邮政编码: 000000

+86

付款选项

自动付款

您只有在产生费用之后才需要为此服务付款。当帐号费用达到结算起付金额，或距离您上次自动付款已有 30 天（二者取其先）时，我们的系统将自动向您收取费用。

付款方式 ⓘ

卡号 月份 年份 信用卡

#

持卡人姓名

☒ 信用卡或借记卡帐单邮寄地址与上述地址相同

START MY FREE TRIAL

可使用所有 Cloud Platform 产品
获得开发和运行应用、网站及服务所需的一切，包括 Firebase 和 Google Maps API。

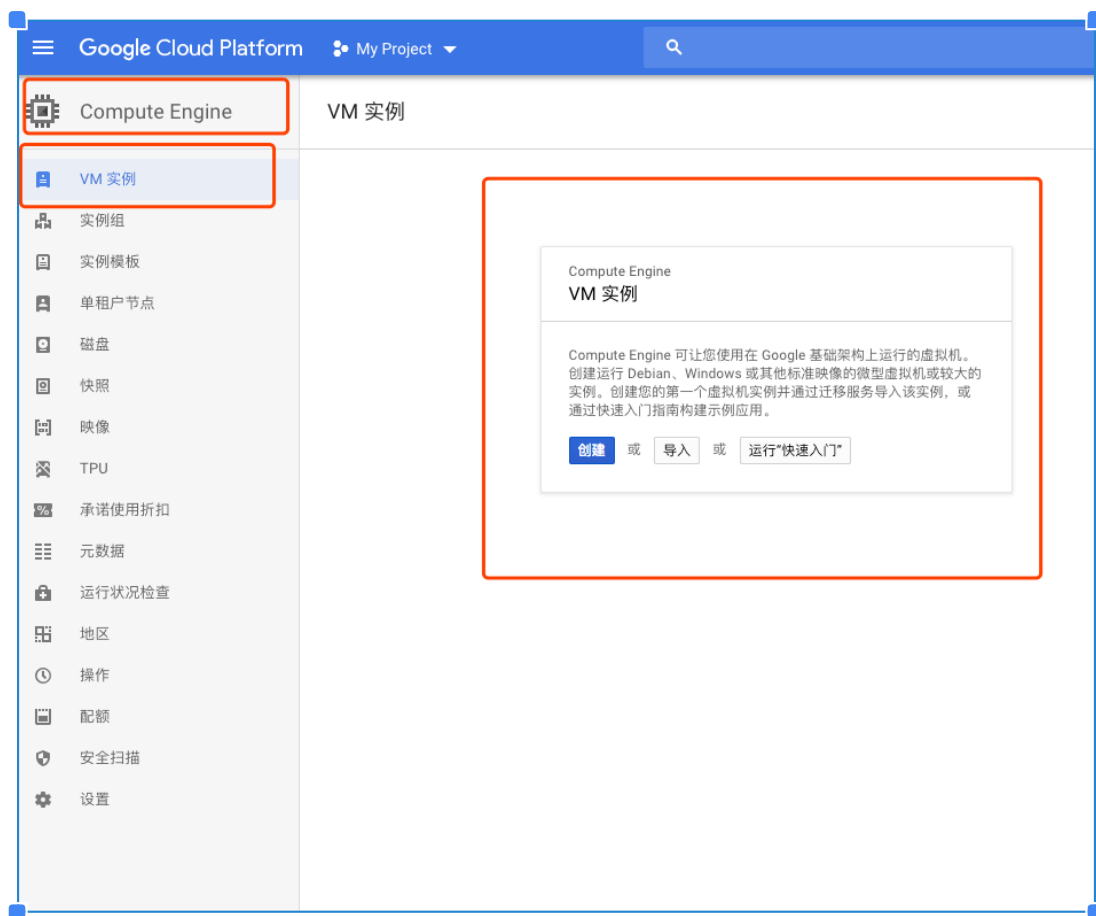
免费获得 \$300 赠金
注册即可获得 \$300 赠金。未来 12 个月内可在 Google Cloud Platform 上随心使用。

免费试用结束后不会自动收费
我们要求您提供信用卡信息是为了确认您不是机器人。除非您手动升级为付费帐号，否则我们不会向您收费。

4.注册成功截图



5. 创建VM实例



6.配置实例

Google Cloud Platform

My Project

Compute Engine

VM 实例

实例组

实例模板

单租户节点

磁盘

快照

映像

TPU

承诺使用折扣

元数据

运行状况检查

地区

操作

配额

安全扫描

设置

创建实例

名称

instance-1

区域

us-west1 (俄勒冈州)

地区

us-west1-b

机器类型

微型 (1 个共享 ...)

0.6 GB 内存

自定义

容器

☐ 将一个容器映像部署到此 VM 实例。[了解详情](#)

启动磁盘

新的 10 GB 标准永久性磁盘

映像

Debian GNU/Linux 9 (stretch)

更改

身份和 API 访问权限

服务帐号

Compute Engine default service account

访问权限范围

☒ 允许默认访问权限

☐ 允许所有 Cloud API 的全面访问权限

☐ 针对每个 API 设置访问权限

防火墙

添加标记和防火墙规则。允许来自互联网的特定网络流量

☐ 允许 HTTP 流量

☐ 允许 HTTPS 流量

Management, security, disks, networking, sole tenancy

您的免费试用赠金 (如果有) 将用于抵扣此实例的费用

创建

取消

等效 REST 或命令行

\$4.28/月 估算值

有效的每小时费率为 \$0.006 (每月 730 小时)

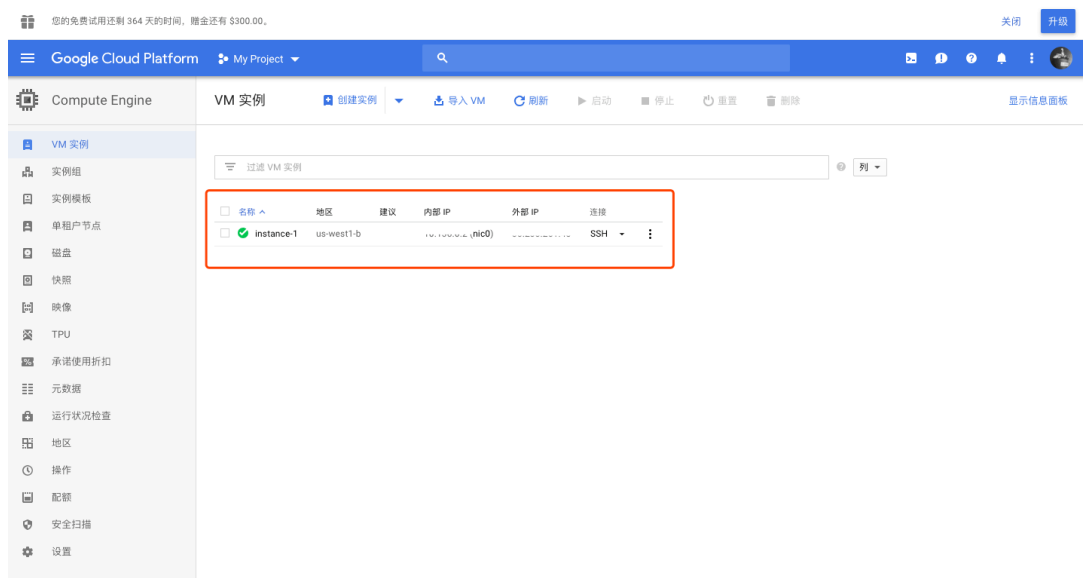
您在本月使用“f1-micro”实例的前 744 个小时是免费的。[了解详情](#)

内容	估算费用
1 个共享 vCPU + 0.6 GB 内存	\$5.55/月
10 GB 标准永久性磁盘	\$0.40/月
持续使用折扣	-\$1.66/月
总计	\$4.28/月

[Compute Engine 定价](#)

[收起](#)

7.创建好后的实例



8. 申请保留静态IP



9. 创建防火墙规则

Google Cloud Platform

My Project

VPC 网络

VPC 网络

外部 IP 地址

防火墙规则

路由

VPC 网络对等互连

共享的 VPC

← 创建防火墙规则

防火墙规则用于控制传入实例和从实例传出的流量。默认情况下，系统会阻止从您的网络之外传入的流量。[了解详情](#)

名称 ?

default

说明 (可选)

网络 ?

default

优先级 ?

优先级可以从 0 到 65535 [检查其他防火墙规则的优先级](#)

1000

流量方向 ?

入站

出站

对匹配项执行的操作 ?

允许

拒绝

目标 ?

网络中的所有实例

来源过滤条件 ?

IP 地址范围

来源 IP 地址范围 ?

0.0.0.0/0

次要来源过滤条件 ?

无

协议和端口 ?

全部允许

指定的协议和端口

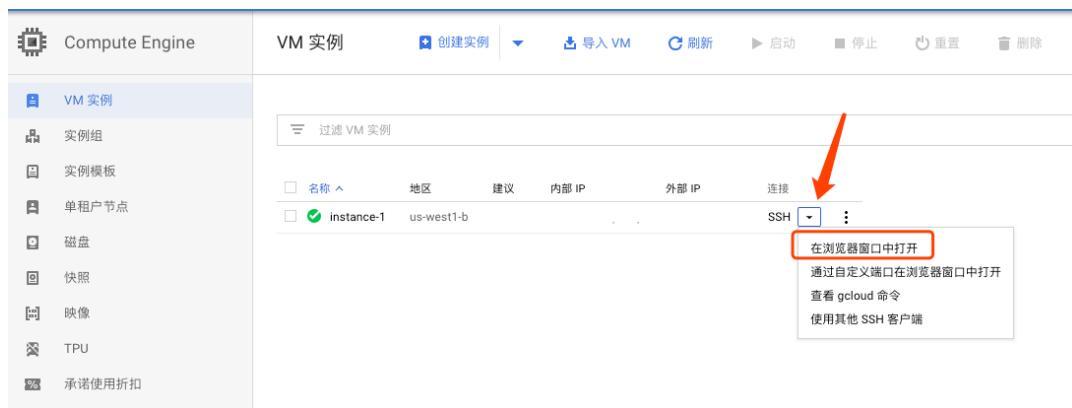
停用规则

创建

取消

等效 [REST](#) 或 [命令行](#)

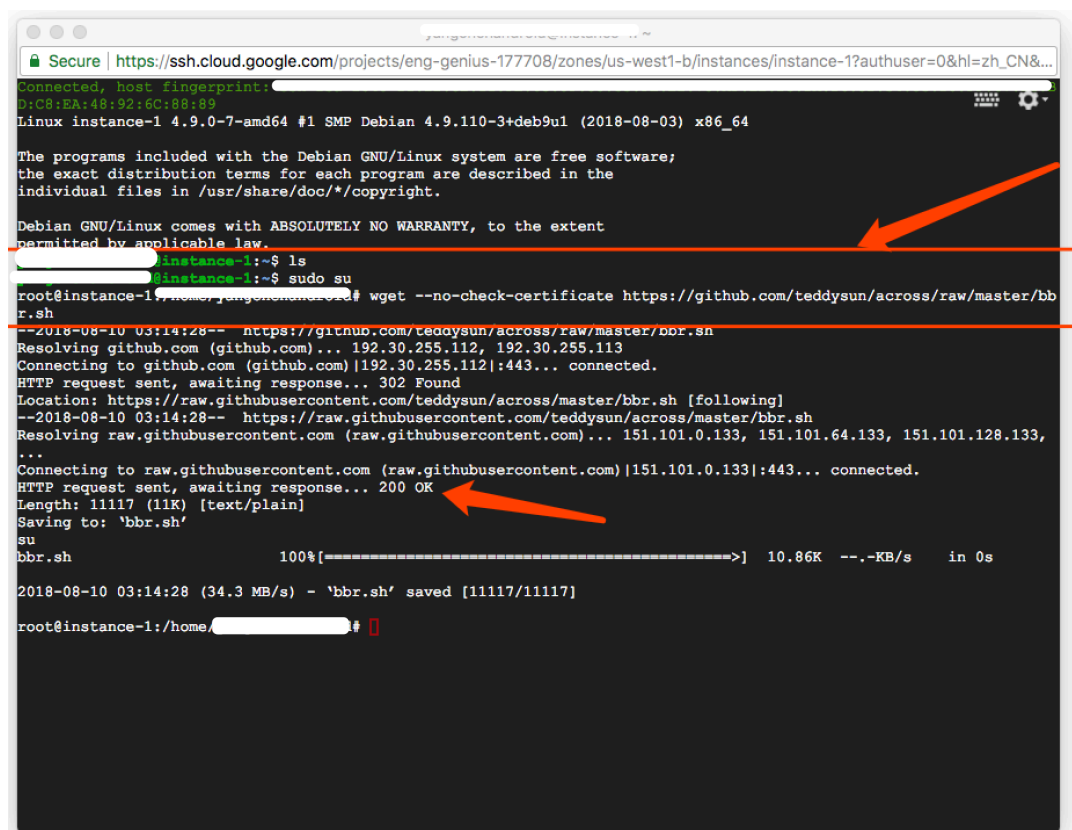
10.安装BBR内核 启动命令行窗口



11 debian command:

`wget --no-check-certificate`

`https://github.com/teddysun/across/raw/master/bbr.sh`



b. `chmod +x bbr.sh`

c. `./bbr.sh`

```

----- System Information -----
OS      : Debian GNU/Linux 9
Arch    : x86_64 (64 Bit)
Kernel  : 4.9.0-7-amd64
-----

Auto install latest kernel for TCP BBR

URL: https://teddysun.com/489.html
-----

Press any key to start...or Press Ctrl+C to cancel

```

完成后：

```

----- System Information -----
OS      : Debian GNU/Linux 9
Arch    : x86_64 (64 Bit)
Kernel  : 4.9.0-7-amd64
-----

Auto install latest kernel for TCP BBR

URL: https://teddysun.com/489.html
-----

Press any key to start...or Press Ctrl+C to cancel

Info: Your kernel version is greater than 4.9, directly setting TCP BBR...
Info: Setting TCP BBR completed...

```

d. `uname -r`

```

Linux instance-1 4.9.0-7-amd64 #1 SMP Debian 4.9.110-3+deb9u1 (2018-08-03) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Aug 10 03:11:36 2018 from 173.194.94.98
@instance-1:~$ uname -r
4.9.0-7-amd64
@instance-1:~$

```

e.验证BBR是否安装成功，先进入root账户

`sudo su`

`sysctl net.ipv4.tcp_available_congestion_control`

返回值：`net.ipv4.tcp_available_congestion_control = bbr cubic reno`

```
Linux instance-1 4.9.0-7-amd64 #1 SMP Debian 4.9.110-3+deb9u1 (2018-08-03) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Aug 10 03:11:36 2018 from 173.194.94.98
instance-1:~$ uname -r
4.9.0-7-amd64
instance-1:~$ sysctl net.ipv4.tcp_available_congestion_control
-bash: sysctl: command not found
yangchenandroid@instance-1:~$ sudo so
sudo: so: command not found
instance-1:~$ sudo su
root@instance-1:/home/instance-1:~# sysctl net.ipv4.tcp_available_congestion_control
net.ipv4.tcp_available_congestion_control = bbr cubic reno
root@instance-1:/home/instance-1:~#
```

f. sysctl net.ipv4.tcp_congestion_control

返回结果：net.ipv4.tcp_congestion_control = bbr

```
instance-1:~$ sudo so
sudo: so: command not found
instance-1:~$ sudo su
root@instance-1:/home/instance-1:~# sysctl net.ipv4.tcp_available_congestion_control
net.ipv4.tcp_available_congestion_control = bbr cubic reno
root@instance-1:/home/instance-1:~# sysctl net.ipv4.tcp_congestion_control
net.ipv4.tcp_congestion_control = bbr
root@instance-1:/home/instance-1:~#
```

g. sysctl net.core.default_qdisc

返回值：net.core.default_qdisc = fq

```
root@instance-1:/home/instance-1:~# sysctl net.ipv4.tcp_available_congestion_control
net.ipv4.tcp_available_congestion_control = bbr cubic reno
root@instance-1:/home/instance-1:~# sysctl net.ipv4.tcp_congestion_control
net.ipv4.tcp_congestion_control = bbr
root@instance-1:/home/instance-1:~# sysctl net.core.default_qdisc
net.core.default_qdisc = fq
root@instance-1:/home/instance-1:~#
```

h. lsmod | grep bbr

返回值有 tcp_bbr 模块即说明bbr已启动。

```
instance-1:~$ sudo su
root@instance-1:/home/instance-1:~# sysctl net.ipv4.tcp_available_congestion_control
net.ipv4.tcp_available_congestion_control = bbr cubic reno
root@instance-1:/home/instance-1:~# sysctl net.ipv4.tcp_congestion_control
net.ipv4.tcp_congestion_control = bbr
root@instance-1:/home/instance-1:~# sysctl net.core.default_qdisc
net.core.default_qdisc = fq
root@instance-1:/home/instance-1:~# lsmod | grep bbr
tcp_bbr          20480  7
```


i. 安装server ss

```
wget --no-check-certificate -O shadowsocks-all.sh
```

https://raw.githubusercontent.com/teddysun/shadowsocks_install/master/shadowsocks-all.sh

```
root@instance-1:/home/ # wget --no-check-certificate -O shadowsocks-all.sh https://raw.githubusercontent.com/teddysun/shadowsocks_install/master/shadowsocks-all.sh
--2018-08-10 03:33:20-- https://raw.githubusercontent.com/teddysun/shadowsocks_install/master/shadowsocks-all.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.0.133, 151.101.64.133, 151.101.128.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.0.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46132 (45K) [text/plain]
Saving to: 'shadowsocks-all.sh'

shadowsocks-all.sh      100%[=====>] 45.05K  --.-KB/s   in 0.01s
2018-08-10 03:33:20 (3.26 MB/s) - 'shadowsocks-all.sh' saved [46132/46132]
```

j. `chmod +x shadowsocks-all.sh`

```
2018-08-10 03:33:20 -- https://raw.githubusercontent.com/teddysun/shadowsocks_install/master/shadowsocks-all.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.0.133, 151.101.64.133, 151.101.128.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.0.133|:443... connected
HTTP request sent, awaiting response... 200 OK
Length: 46132 (45K) [text/plain]
Saving to: 'shadowsocks-all.sh'

shadowsocks-all.sh      100%[=====>] 45.05K  --.-KB/s   in 0.01s
2018-08-10 03:33:20 (3.26 MB/s) - 'shadowsocks-all.sh' saved [46132/46132]
root@instance-1:/home/ # chmod +x shadowsocks-all.sh
```

k. `./shadowsocks-all.sh 2>&1 | tee shadowsocks-all.log`

```
Which Shadowsocks server you'd select:
1) Shadowsocks-Python
2) ShadowsocksR
3) Shadowsocks-Go
4) Shadowsocks-libev
Please enter a number (Default Shadowsocks-Python):
```

选择GO版本

```
Which Shadowsocks server you'd select:
1) Shadowsocks-Python
2) ShadowsocksR
3) Shadowsocks-Go
4) Shadowsocks-libev
Please enter a number (Default Shadowsocks-Python):3

You choose = Shadowsocks-Go

Please enter password for Shadowsocks-Go
(Default password: teddysun.com):
```

```
Which Shadowsocks server you'd select:
1) Shadowsocks-Python
2) ShadowsocksR
3) Shadowsocks-Go
4) Shadowsocks-libev
Please enter a number (Default Shadowsocks-Python):3

You choose = Shadowsocks-Go

Please enter password for Shadowsocks-Go
(Default password: teddysun.com): 
password = 

Please enter a port for Shadowsocks-Go [1-65535]
(Default port: 14111):15000
```

```
Please select stream cipher for Shadowsocks-Go:
1) aes-256-cfb
2) aes-192-cfb
3) aes-128-cfb
4) aes-256-ctr
5) aes-192-ctr
6) aes-128-ctr
7) chacha20-ietf
8) chacha20
9) salsa20
10) rc4-md5
Which cipher you'd select(Default: aes-256-cfb):1

cipher = aes-256-cfb

Press any key to start...or Press Ctrl+C to cancel
```

(此过程需要10分钟左右) 完成SS安装后的截图

```
Starting Shadowsocks-go success

Congratulations, Shadowsocks-Go server install completed!
Your Server IP      : 
Your Server Port    : 15000
Your Password       : 
Your Encryption Method: aes-256-cfb

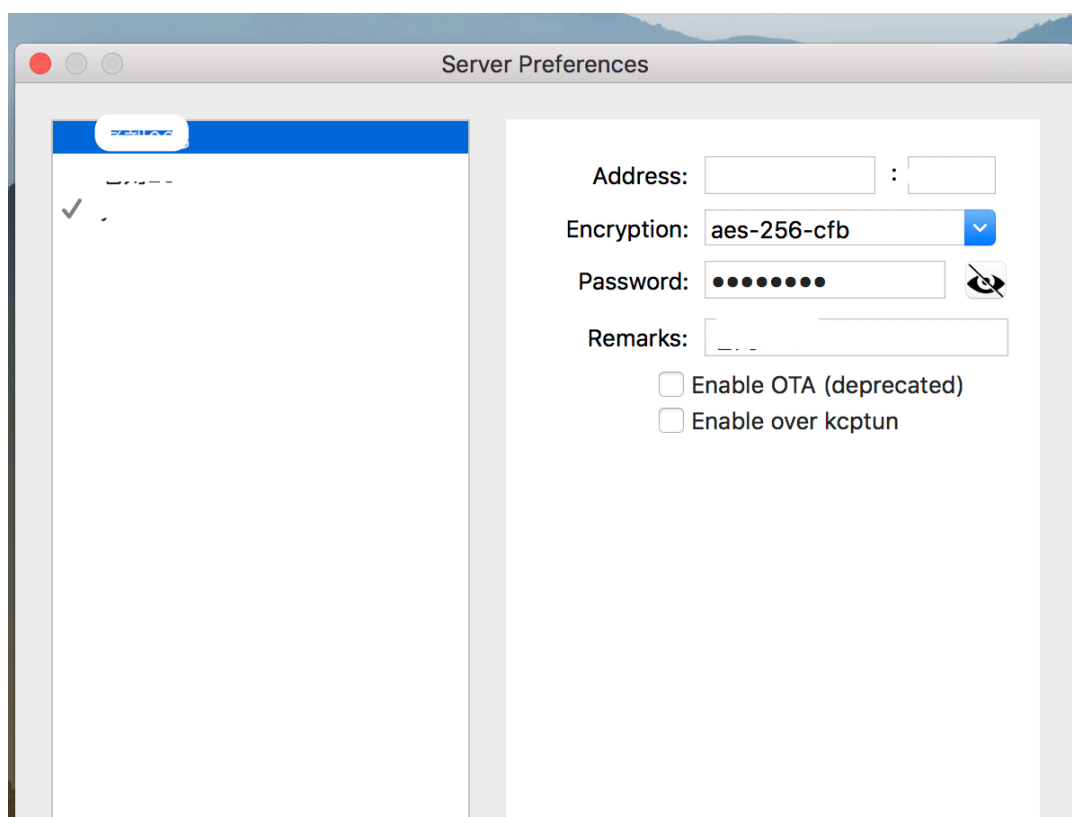
Your QR Code: (For Shadowsocks Windows, OSX, Android and iOS clients)
Your QR Code has been saved as a PNG file path:
/home/ /shadowsocks_go_qr.png

Welcome to visit: https://teddysun.com/486.html
Enjoy it!
```

到这一步其实就能访问到远程的服务器了。

可以在shadowsocks client里面配置下上面对应的ip和密码 端口号。

shadowsocks github: <https://github.com/shadowsocks>



当然我们可以配置多用户访问，设置配置文件

```
sudo su
```

```
vi /etc/shadowsocks-go/config.json
```

复制粘贴

```
{  
  "server": "0.0.0.0",  
  "local_port": 1080,  
  "port_password": {  
    "8989": "password0",  
    "9001": "password1",  
    "9002": "password2",  
    "9003": "password3",  
    "9004": "password4"  
  },  
  "timeout": 300,  
  "method": "aes-256-cfb",  
}
```

注意json格式有无错误，检查双引号是不是英文模式下的。

修改端口号（1-65536）和密码（自己随意），修改完后按
esc

然后按下shift+:,输入wq!,回车，就保存了
最后重新启动

```
/etc/init.d/shadowsocks-go restart
```

可以看到重启成功，大功告成！！

