Ghetto Forensics

CLI compromise assessment of Linux OS based systems

petar@securityops.ai

Agenda

What will we cover

- Importance of Command Line Investigations
- Ground Truths About Investigations
- Commands to Use with Clarifications and Examples.
- Q&A

Why command line investigations ?

Why is this a thing ?

- It is the default Linux interface!
- Even while using some EDR system (like Defender for endpoint or Crowd Strike) if you are investigating something suspicious you will have to drop to “live response” component and this is again working like a limited command shell.
- You can do response via orchestration tools such as Ansible, Salt Stack etc.
- “Cloud” is just someone else’s computer. Cloud VM’s are in full use and exposed on the internet in most cases!

Some Ground rules to guide you

Things I learned the hard way.

- Probably it’s nothing. Listen anyways!
- Ask Questions. A question well put is a problem half solved.
- Greed is good. Most of the hacking today is financially motivated, keep that in mind when looking for signs of compromise.
- Don't worry about APT’s. Attribution is HARD. Focus on what you see.
- Get observables as fast as possible - focus on things you can search for: IP’s, domains, process trees, hashes, strings.
- Compromise on linux systems most often comes from things exposed on the network (Web applications, API interfaces, SSH management)
- No matter how advanced a malware, it will attempt to do 4 things, RUN, Communicate, START, Attempt to hide.
- DO NOT make it worse!!!! - No slide here, it should be common sense.

Possibly it is nothing. Listen Anyways!

No matter where the report came from, listen/read the details.

- 90+ percent of the alerts you are going to see and 85 percent of the user reports are false positives.
- It is much easier to prove that something is wrong than to prove it is correct.
- Details are important - write them down.
- Try to get specifics. Ask the reporter for specific observables. When did the slow down start. ? Why do you think the computer was hacked. Those things will give you starting points to focus on.

Ask questions of your data and users!

- Fundamentally an investigation is just that a series of questions.
- |====> something happened =====> You are here=====>possible outcomes.
- In order to do that you need to answer specific questions.
- Good questions are the difference between 1 hour and 1 day of investigation.
- Example: The computer is hacked - Ok what where the specific observables that make you think the machine is hacked ?
- Stick to facts and observables.

Greed is good. Most of the hacking today is financially motivated. Keep that in mind when looking for signs of compromise.

- When thinking how did we get hacked, most of the time you are asking how are they **making money** out of this.
- Or how is this helping them **make money**?
- No matter what you are investigating, keep in mind that bad guys have objective and usually that objective is **making money**.
- Examples: Crypto Mining, Botnet Services, Extortion, Ransomware.
- **THAT IS GOOD FOR US**. Objectives mean deadlines, they are looking for results (money) faster. Your red team might spend 3 months researching and implementing super stealthy root kit. Bad guys will do the minimum effort possible (they are agile:).

Don't worry about APT’s. Focus on what you see.

- Most of the times you are dealing with PT persistent threats. Not very advanced thou.
- Still smart enough to bypass Antivirus.
- If they got in, it was probably the most obvious way possible.
- Bad guys are lazy (Even the good ones - North Korean APT’s where they were re using infrastructure IP’s attributed to them years ago)

Get observables as fast as possible - focus on things you can search for: IP’s, domains, process trees, strings.

- If you arrive to the conclusion that indeed there is an ongoing attack, you need to move fast. **All your fancy forensicating ain't worth a dime, if the environment gets encrypted.**
- Atomic indicators such as ip’s domains process names, process execution trees, string data are your best bet at first.
- Do assessment | Get-Observables | Create-Signatures | Scan/search with tool of choice supporting your signatures | Rinse and repeat. | Until all hosts that have been compromised are found.

Compromise on linux systems most often comes from things exposed on the network. (Web applications, API interfaces, SSH management)

- The bad guys are connecting somehow!
- When you are examining front end server, don’t worry about things not listening or talking on the network.
- Question is can you see this communication on host level. This is why PCAP or at least net flow data becomes important.
- Focus on External communications. I know it is stupid but look at stuff calling outside your networks first.
- Ask the question “should it do that” not “is this malicious”

No matter how advanced a malware, it will attempt to do 4 things: RUN, COMMUNICATE, START, ATTEMPT TO HIDE.

- Don't worry about what might happen and focus on what has to happen.
- Eg: To log on in to SSH you need a user. To exploit an application remotely it needs to be exposed on the network, etc.
- The pillars of Examination Malware will:
- RUN - as such there is going to be observables such as “process running” “legitimate executable hijacking” and many others.
- Communicate - it is going to be observable on network layer.
- Start - it will create a persistence point one way or another.
- Attempt to hide (often in plain sight) Malware will try to mask its presence and start up routines. The more masking is done the more initial noise will be raised. Look for deleted logs, strange ports etc.

Boring part is OVER!!!!

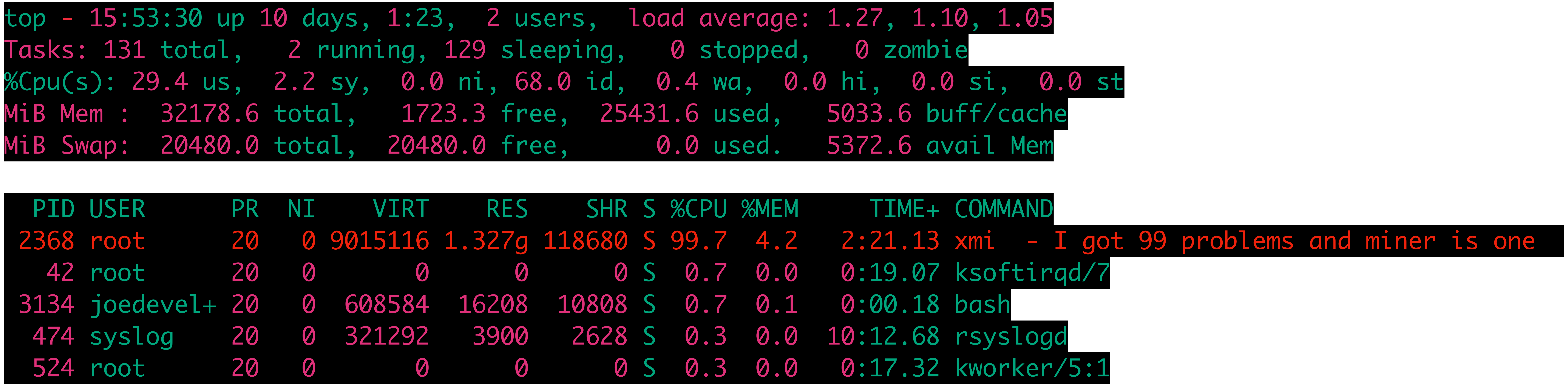
Some hands on approach commands and one liners you can use.

- Ladies and gentleman, I give you drumroll please:

top

Yes that is correct we are looking for system load first.

- Most Miners or bot's will use CPU as much as they can….
- Kind of obvious but look for the highest CPU load.



```
top - 15:53:30 up 10 days, 1:23,  2 users,  load average: 1.27, 1.10, 1.05
Tasks: 131 total,   2 running, 129 sleeping,   0 stopped,   0 zombie
%Cpu(s): 29.4 us,  2.2 sy,  0.0 ni, 68.0 id,  0.4 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  32178.6 total,   1723.3 free,  25431.6 used,   5033.6 buff/cache
MiB Swap:  20480.0 total,  20480.0 free,      0.0 used.   5372.6 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 2368 root      20   0 9015116 1.327g 118680 S 99.7  4.2   2:21.13 xmi  - I got 99 problems and miner is one
   42 root      20   0       0      0      0 S  0.7  0.0   0:19.07 ksoftirqd/7
 3134 joedevel+ 20   0  608584  16208  10808 S  0.7  0.1   0:00.18 bash
  474 syslog    20   0  321292   3900   2628 S  0.3  0.0  10:12.68 rsyslogd
  524 root      20   0       0      0      0 S  0.3  0.0   0:17.32 kworker/5:1
```

ps -auxw

Process Tree

```
USER         PID  %CPU %MEM    VSZ    RSS TTY      STAT START   TIME COMMAND
root           1  0.1  0.2 168116 10592 ?        Ss   Jan12   2:51 /sbin/init
root        1661  0.0  0.0  17468   952 ?        Ss   Jan12   0:00 /usr/sbin/apache2 -k start
www-data    1672  0.0  0.0 398320  6228 ?        Sl   Jan12   0:01  \_ /usr/sbin/apache2 -k start
www-data    1673  0.0  0.0 398320  6260 ?        Sl   Jan12   0:01  \_ /usr/sbin/apache2 -k start
www-data    1674  0.0  0.0 398320  6228 ?        Sl   Jan12   0:01  \_ /usr/sbin/apache2 -k start
totallyl+   3179  0.5  1.1 2332940 71148 ?       S    Mar02  64:06  \_ /bin/bash /var/www/html/xmi_miner.sh
totallyl+   3180  3.9  3.3 1760332 206304 ?      R    Mar02 494:32      \_ xmrig -o pool.supportxmr.com:9000 -u
42GXJa7cMZ4x1hM4weDfaUNPfktaV7SvX9NffVQJYgZuxNc1bxDhRYEhsa1SVfZcoLtpcQ3qX9JjC8DpZ1mWoV7LGvU5W6 -p x -t 4 <==== Seems
Legit
```

Explanation

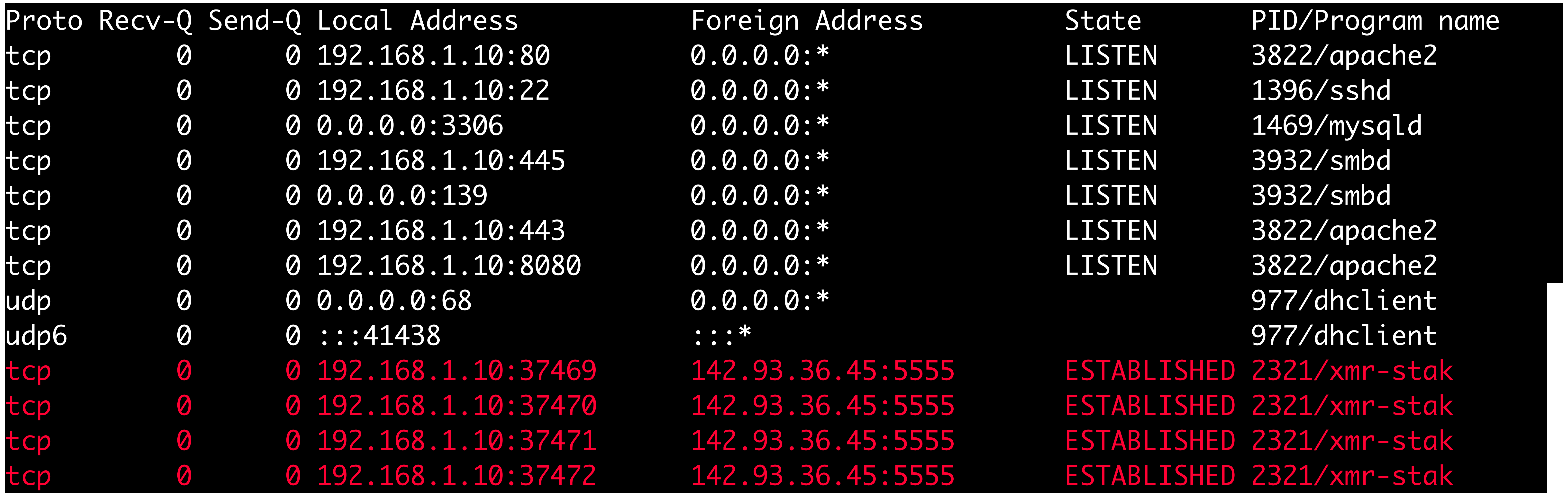
`-auxwf`: Flags used to display all processes owned by all users, in a tree structure, and with extra wide output to ensure all information is shown.

netstat -nalp

```
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program
name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1234/sshd
tcp6       0      0 :::80                   :::*                    LISTEN      4321/
apache2
tcp6       0      0 :::22                   :::*                    LISTEN      1234/sshd
tcp6       0      0 :::443                  :::*                    LISTEN      4321/
apache2
tcp6       0      0 192.0.2.10:4444         203.0.113.2:12345       ESTABLISHED 5678/nc
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           4321/avahi-
daemon:
```

Explanation: This command displays all active network connections and their associated processes. The `-n` *option disables hostname resolution, while the* `-a` *option displays all listening and non-listening sockets. The* `-l` *option displays only listening sockets, and the* `-p` *option shows the PID and process name of the associated process.*

netstat -plant



```
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 192.168.1.10:80         0.0.0.0:*               LISTEN      3822/apache2
tcp        0      0 192.168.1.10:22         0.0.0.0:*               LISTEN      1396/sshd
tcp        0      0 0.0.0.0:3306            0.0.0.0:*               LISTEN      1469/mysqld
tcp        0      0 192.168.1.10:445        0.0.0.0:*               LISTEN      3932/smbd
tcp        0      0 0.0.0.0:139             0.0.0.0:*               LISTEN      3932/smbd
tcp        0      0 192.168.1.10:443        0.0.0.0:*               LISTEN      3822/apache2
tcp        0      0 192.168.1.10:8080       0.0.0.0:*               LISTEN      3822/apache2
udp        0      0 0.0.0.0:68              0.0.0.0:*                           977/dhclient
udp6       0      0 :::41438                :::*                                977/dhclient
tcp        0      0 192.168.1.10:37469      142.93.36.45:5555       ESTABLISHED 2321/xmr-stak
tcp        0      0 192.168.1.10:37470      142.93.36.45:5555       ESTABLISHED 2321/xmr-stak
tcp        0      0 192.168.1.10:37471      142.93.36.45:5555       ESTABLISHED 2321/xmr-stak
tcp        0      0 192.168.1.10:37472      142.93.36.45:5555       ESTABLISHED 2321/xmr-stak
```

Explanation: The `netstat` command displays active network connections and the programs they are associated with. The `-p` option displays the process ID and name, while the `-a` option shows both listening and non-listening sockets. The `-l` option displays only listening sockets, and the `-n` option shows IP addresses and port numbers in numerical format.

ss -a -e -i

```
State      Recv-Q Send-Q Local Address:Port              Peer Address:Port          Process
ESTAB      1024   2048   192.168.1.10:37469              142.93.36.45:5555         ino:123456    sk:2   <->
LISTEN     0      128    127.0.0.1:9000                              *:*         ino:123457    sk:3   <->
ESTAB      0      0      192.168.1.10:ssh                192.168.1.100:54658        ino:16429     sk:6d   <->
ESTAB      0      0      192.168.1.10:37470              142.93.36.45:5555         ino:123458    sk:4   <->
LISTEN     0      80     192.168.1.10:80                              *:*         ino:123459    sk:5   <->
```

Explanation: The* ss *command is used to display active network socket connections. The* -a *option shows both listening and non-listening sockets, while the* -e *option displays extended socket information, and the* -i *option shows socket statistics.

ls -alR /proc/*/exe 2> /dev/null | grep deleted
Deleted binaries still running

```
lr-x------ 1 root root 64 Mar  1 09:00 1923/exe ->
/usr/lib/libudev.so (deleted)
```

Explanation: The ***ls -alR /proc/*/exe*** *command is used to display information about running processes and their corresponding executable files. The* ***2> /dev/null*** *portion of the command redirects error messages to* ***/dev/null,*** *effectively suppressing any "Permission denied" errors. The* ***grep deleted*** *part of the command filters the output to show only processes with deleted executable file. Bonus points who can name the malware....*

The /proc filesystem or figuring a process out

```
# Check the command name and command line of the XMR miner process
# Command Name and Arguments:
strings /proc/2321/comm
string /proc/2321/cmdline
# Output:
# xmr-stak ./xmr-stak --currency cryptonight_v8 --noCPU --url stratum+tcp://xmr.pool.minergate.com:45700 --user email@example.com --pass x --rig-id rig001
COMPARE those two!

#Check Process Environment - who started it.
strings /proc/232/environ

Process parent Pid and other useful data
cat /proc/2321/status

Recover Binary
cp /proc/2321/exe /tmp/totallylegit_bin

What files are open - malware could be stashing something in files and loaded libraries
cat /proc/2321/maps

# Check the real process path of the XMR miner process
# Process Path:
ls -al /proc/2321/exe
# Output:
# lrwxrwxrwx 1 user user 0 Mar 13 01:03 /proc/2321/exe -> /usr/bin/xmr-stak

# Check the environment variables of the XMR miner process
# Environment Variables:
strings /proc/2321/environ
# Output:
# ...XMR_MINER_CONFIG=/etc/xmr-stak/config.txtXMR_MINER_CPU=1XMR_MINER_LOGFILE=/var/log/xmr-stak.log...

# Check the working directory of the XMR miner process
# Working Directory:
ls -al /proc/2321/cwd
# Output:
# lrwxrwxrwx 1 user user 0 Mar 13 01:03 /proc/2321/cwd -> /
```

Processes running from /temp and /dev

Temp and dev are favourite run directories of commodity malware.(Everyone and their dog can write there)

```
ls -alR /proc/*/cwd 2> /dev/null | grep tmp
ls -alR /proc/*/cwd 2> /dev/null | grep dev
```

```
# Check for processes running from the /tmp directory
ls -alR /proc/*/cwd 2> /dev/null | grep tmp
# Output:
# /proc/2321/cwd -> /tmp/xmr-stak

# Check for processes running from the /dev directory
ls -alR /proc/*/cwd 2> /dev/null | grep dev
# Output:
# /proc/2321/cwd -> /dev/shm
```

File operations

Some tips for investigating file structures

```
ls -lap
```

In this example, the `ls` command is used with the `-lap` options to list all files and directories under the current directory. The `-a` option is used to show hidden files and directories, the `-l` option is used to show the file type, permissions, and other details in a long format, and the `-p` option is used to show file type indicators (e.g. `/` for directories).

Targeted DIRS
targeted directories `/tmp`, `/var/tmp`, and `/dev/shm` are often used by malware because they are locations where temporary files and directories can be created and accessed by all users on the system. These directories typically have lax permission settings, allowing any user to write to them.

Additionally, `/var/run` and `/var/spool` are also targeted by malware as they contain system-related files and directories that can be used to escalate privileges or gain access to sensitive information.

Finally, user home directories are targeted as they often contain personal and sensitive information, such as login credentials and documents. If a malware can gain access to a user's home directory, it can potentially steal or manipulate this information.

Users and Persistence

```
# Find all ssh authorized_keys files
find / -name authorized_keys

# History files for users
find / -name .*history

# History files linked to /dev/null
ls -alR / 2> /dev/null | grep .*history | grep null


# Check sudoers file
cat /etc/sudoers
cat /etc/group

# Check scheduled tasks
crontab -l
atq
systemctl list-timers --all

# Persistence areas
cat /etc/rc.local
ls -al /etc/init.d
ls -al /etc/rc*.d
cat /etc/modules
ls -al /etc/cron*
ls -al /var/spool/cron/*
```

Logs and What not…

I am not going in to log review this is another lecture by itself…

- The basics:

```
c
- `ls -al /var/log/*`: Lists all files in the `/var/log` directory, displaying file sizes,
ownership, and permissions. Checking for zero-size logs can help detect malicious activity.

- `utmpdump /var/log/wtmp`: Dumps the contents of the `/var/log/wtmp` file, which contains user
login and logout information.

- `utmpdump /var/run/utmp`: Dumps the contents of the `/var/run/utmp` file, which contains
information about current user sessions.

- `utmpdump /var/log/btmp`: Dumps the contents of the `/var/log/btmp` file, which contains failed
login attempts.

- `last`: Displays a list of last logged in users.

- `lastb`: Displays a list of failed login attempts.
```

Q&A

Questions, comments, hate ???