DETECTING INTRUDERS IN YOUR SERVER

"The one where we catch them red handed"

October 17th, 2024

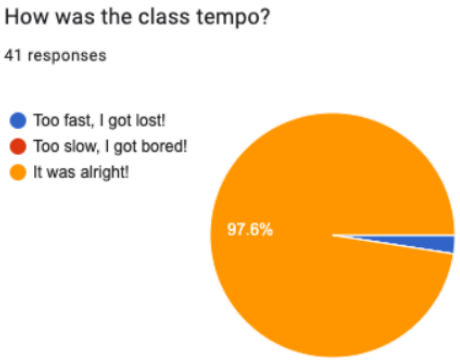
Credits

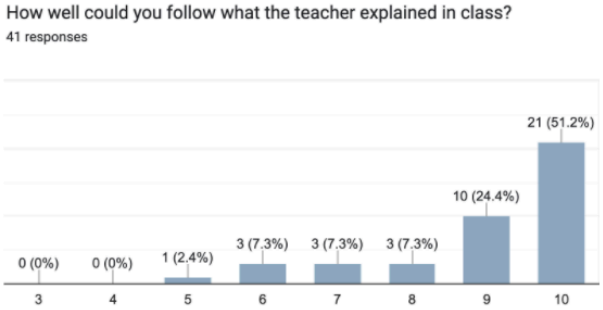
Content: Sebastian Garcia
Ondřej Lukáš
Maria Rigaki
Martin Řepa
Lukáš Forst
Veronica Valeros
Muris Sladić
Illustrations: Fermin Valeros
Design: Veronica Garcia, Veronica Valeros, Ondřej Lukáš
Music: Sebastian Garcia, Veronica Valeros, Ondřej Lukáš
CTU Video Recording: Jan Sláma, Václav Svoboda, Marcela Charvatová
Audio files, 3D prints, and stickers: Veronica Valeros

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

CLASS DOCUMENT	https://bit.ly/BSY2024-4
WEBSITE	https://cybersecurity.bsy.fel.cvut.cz/
CLASS MATRIX	https://matrix.bsy.fel.cvut.cz/
CLASS CTFD (CTU STUDENTS)	https://ctfd.bsy.fel.cvut.cz/
CLASS PASSCODE FORM (ONLINE STUDENTS)	https://bit.ly/BSY-VerifyClass
FEEDBACK	https://bit.ly/BSYFEEDBACK
LIVESTREAM	https://www.youtube.com/live/X4pbZf9L02M?si=ivetPnIvXOv37Ukr
INTRO SOUND	https://bit.ly/BSY-Intro
VIDEO RECORDINGS PLAYLIST	https://www.youtube.com/playlist?list=PLQL6z4JeTTQk_z3vwSIvn6wIHMeNQFU3d
CLASS AUDIO	https://audio.com/stratosphere

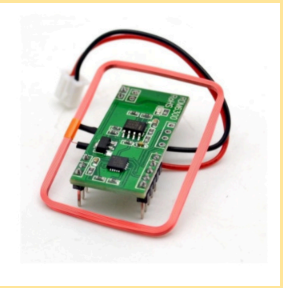
Results from last class survey (14:32, 1m)





This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

Pioneer Prize for Assignment 3 (14:33, 2m)

1st Place	2nd Place	3rd Place
		
Seeed Studio XIAO ESP32S3	RFID reader with antenna 125KHz EM4100 RDM6300	USB 2.0 isolator

Three main topics for this class:

1. How to find out if someone got into your computer?
2. How to secure your computer?
3. Host-Based Intrusion Detection Systems: AIDE

Protection is about layers. Like 🧅 (not 🍦) (14:35, 2m)

Goal: To understand how to protect a computer. To know how to find if there was an intruder inside a computer.

Security should be a layered approach:

1. Assume all other protection measurements **failed**, and each one is the **last** one.
2. Separate your defenses so they don't **depend** on each other.
3. Protect every aspect you can think about. And then again, some you didn't.
4. Typically separated into approximately these steps:

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

   a. **Data** protection (encryption, hiding, backup, etc.)
   b. **Application** protection (updates, monitoring, logs, processes, kernel, authorization, etc.)
   c. **Service** protection (DB, load balancers, etc.)
   d. **Endpoint** protection (FW, users, authorizations, monitoring, logs, AV, EDR, etc.)
   e. **Connection** protection (WiFi, VLANs, etc.)
   f. **Humans**
5. The famous Zero-trust terminology refers to this. No part of the system should trust any other part. Nor internal or external.
   a. Of course, impossible.

🔬 How to find out if someone got into your computer? (14:37, 1m)

Every access to a computer leaves **traces** in someplace. Attackers can get in using the standard authentication methods, exploits, backdoors, or just stealing your hard disk. You should be aware of them all.

Start by looking for signs of intrusions, backdoors, or anything out of place.

🪵 Check logins with standard methods (14:38, 5m)

1. Standard methods are those controlled by the operating system: SSH, Telnet, SMB, NFS, etc.
2. Non-standard methods are backdoors, exploits, etc. Those are not recorded by the system logs.
3. Check who is logged in now:
   a. `w`
4. Check the last logins from all computers:
   a. `last`
      i. List past logins on the system one after the other.
   b. This command reads log files that store login system information:

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

i. /var/log/utmp[1]: Binary file. A full accounting of the current status of the system, system boot time (used by uptime), user logins, logouts, etc. It may not exist.

ii. /var/log/wtmp: Binary file. Acts as a historical file to record all login sessions, including remote logins (via SSH), local logins, reboots, and shutdowns.

iii. To remember: **u**tmp (users), **w**tmp (who), **b**tmp (bad).

c. Logrotate

i. If logs are rotated by logrotate, then last shows the last one.

ii. To see an old log, specify the log file to read from with -f

1. `last -f /var/log/utmp.1`

iii. If they are older, logrotate can compress them. Check:

1. `vi /etc/logrotate.conf`
2. `ls /etc/logrotate.d/`
3. `vi /etc/logrotate.d/btmp` (and uncomment compress)

iv. If you have compressed logs, to see them, first uncompress them and then read them:

1. `gunzip -k /var/log/btmp.2.gz`
2. `zcat /var/log/btmp.2.gz`

5. Check failed attempts to log in

a. `lastb`

i. Uses /var/log/btmp: records failed login attempts

6. Another way is to see the list of users and then when each of them logged in for the last time:

a. `lastlog`

🚨 What is the disadvantage of using the commands w, last, and lastlog?

[1] B, M. (2016). Torvalds Tuesday: Logon History in the *tmp Files - Matt B - Medium. [online] Medium. Available at: https://bromiley.medium.com/torvalds-tuesday-logon-history-in-the-tmp-files-83530b2acc28 [Accessed 16 Oct. 2024].

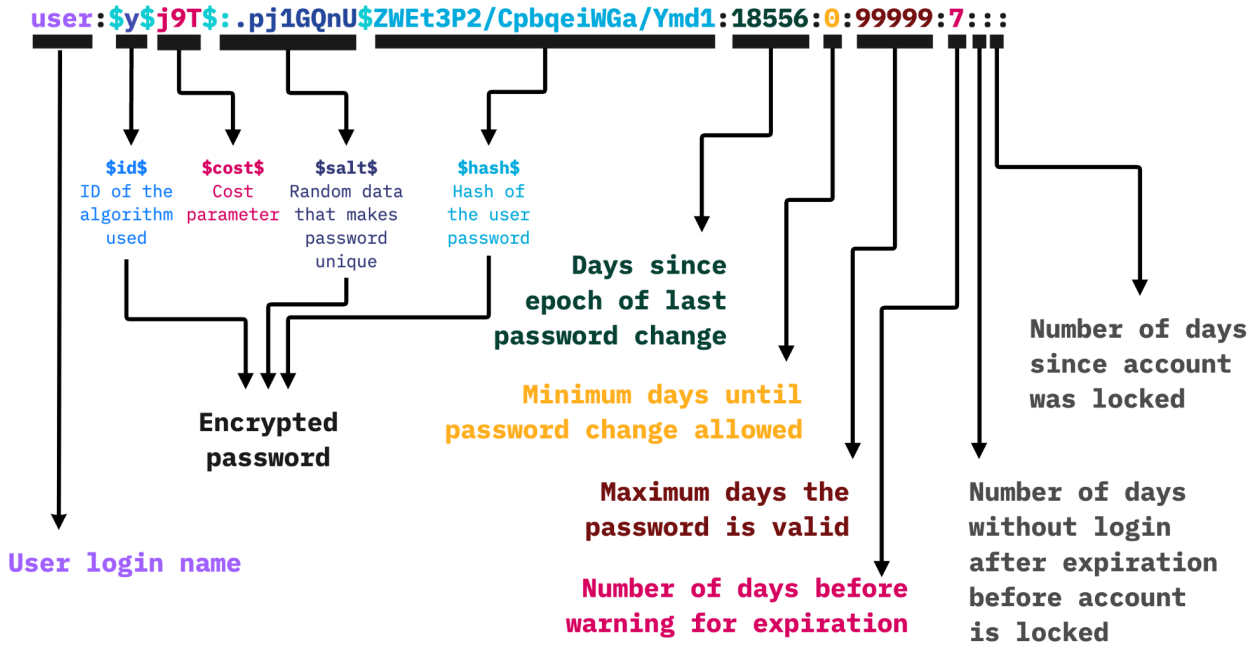
This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

🛂 Check the existing users in the system and password database (root needed) (14:43, 5m)

Users and passwords in the system are kept in the **/etc/shadow** file

Let's inspect it

- `cat /etc/shadow`



1. The first field is the **user name**.
2. The second field is important. It is a composed value **<ID>.<COST><SALT>.<HASH>**
    a. If the whole field is *****, it means the user does not have a password assigned.
    b. The **ID** field can have different values:
        i. 1 = MD5
        ii. 5 = SHA-256
        iii. 6 = SHA-512
        iv. $2a$, $2b$, $2y$: bcrypt (originally based on 'Blowfish', corrected, alternate)

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

v. y = yescrypt

1. Evolution of scrypt. Cost affects memory and CPU. Adjustable cost. GPU/FPGA resistant.

vi. 7 = scrypt

c. The **COST** field

i. How computationally expensive it is to compute the hash. (in yescrypt is for both CPU and Memory). Without storing the cost, the system wouldn't know how many rounds or how much computational effort to apply when re-hashing the user's input.

d. The **Salt** field

i. Adds a pseudo-random value that prevents the use of precomputed hashes (rainbow tables).

3. To quickly disable a user so it can no longer log in without forgetting the password, put a **! (exclamation mark)** as the first char of the password field:

a. `user:!$6$<string>:18071:0:99999:7:::`

4. Any invalid format means the user cannot log in.

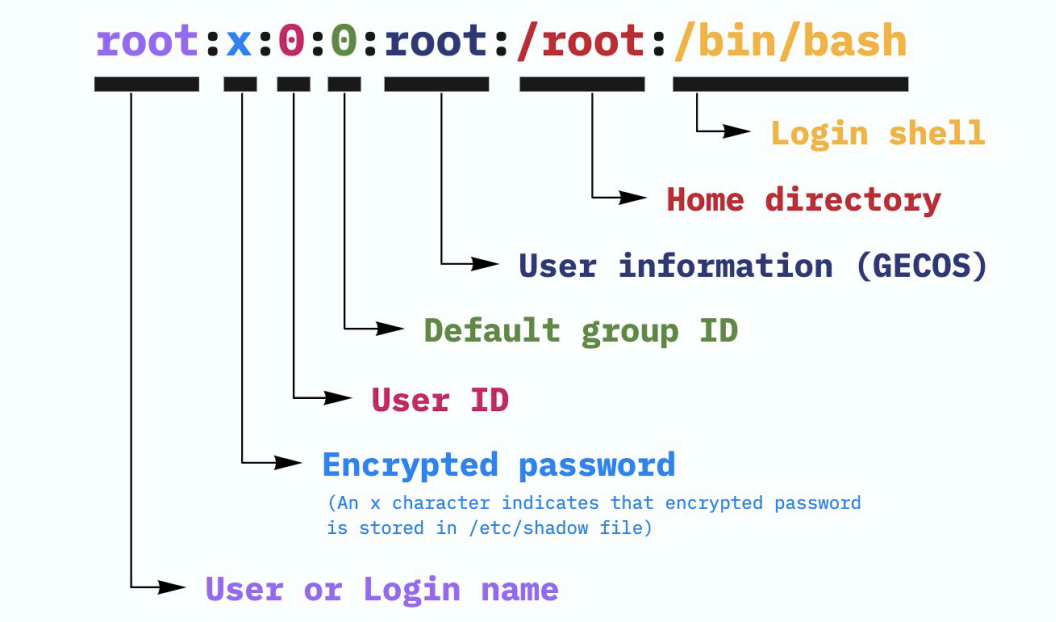
👩‍💻 Check the users in the system but only details of them (no need to be root) (14:48, 2m)

Some user information is stored in the /etc/passwd file. It is a file that has information about users but **not** passwords.

Let's inspect the content of the shadow file

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

- `cat /etc/passwd`



We will see later file permissions, but the /etc/passwd file has permission to be read by any user. Why?

☠️ Find and kill processes (14:50, 5m)

Attackers may execute code in the computer, and that code can be a process that is running in memory. Even if the binary file was deleted.

1. Search for their processes:
    a. `ps afxu |grep -v grep`
        i. -a: all users
        ii. -f: Display the uid, pid, parent pid, recent CPU usage, process start time, controlling tty, elapsed CPU usage, and the associated command
        iii. -x: include processes that do not have a controlling terminal
    b. `htop`
        i. Htop shows processes too, but it uses only /proc/[pid]/cmdline, while ps also uses /proc/[pid]/stat. So they may show different process names.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

c. `btop (already installed in the Dockers)`

i. Probably the most modern resource monitor in Linux.

ii. `apt install btop`

2. Kill process you know are suspicious

a. `kill -9 <process id>`

i. -9 is a signal sent to the process. -9 is special because it means 'die,' but it can not be intercepted by the process and stopped.

ii. -1 is 'die,' but the process can intercept it and choose to ignore it.

b. To kill all the processes of a user:

i. `killall -u <username>`

c. Let's try

i. Execute something forever as a new background process.

1. `nohup bash -c "while true; do echo '1' >> /tmp/file; sleep 5; done" &`

a. nohup: Invoke a utility immune to hangups by redirecting its standard output (terminal) to the file called "nohup".

b. bash -c: Executes this as a command in bash.

c. >>: Redirects the standard output of the command (echo in this case) to a file (/tmp/file) but it appends to previous content (Only one > is to rewrite).

d. &: Detach this command from the shell and puts it running in the background.

ii. Check it works

1. `tail -f /tmp/file`

a. This outputs the last line (by default 1 line) of the file (tail), and waits for more lines to come so it can output them. This 'follows' the new content of the file.

iii. Now, find the process

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

1. `ps afx`

iv. Now kill it

1. `kill -9 <process id (pid)>`

a. -9: Sends the signal KILL (that is not ignorable)

b. By default, kill sends -1 (HUP), but the process can ignore it.

🪵 Check log files (14:55, 2m)

All the logs of Linux systems are located in /var/log/. Let's see some of the logs that contain login information:

1. Authentication information is in /var/log/auth*

   a. `grep user /var/log/auth.log | grep -vi cron | less`

      i. grep -v cron: filter OUT the lines containing 'cron'

   b. If there are rotated files:

      i. `grep user /var/log/auth.log.1 | grep -vi cron | less`

   c. If compressed

      i. `zgrep user /var/log/auth.log.2.gz | grep -vi cron | less`

2. Find failed attempts to log in:

   a. `grep -i "Failed password" /var/log/auth.log`

3. The logs for all the system are in /var/log/syslog*

   a. `cat /var/log/syslog | less`

   b. To better see the /var/log/syslog, you can use (not in our dockers)

      i. `journalctl --since "24 hours ago"`

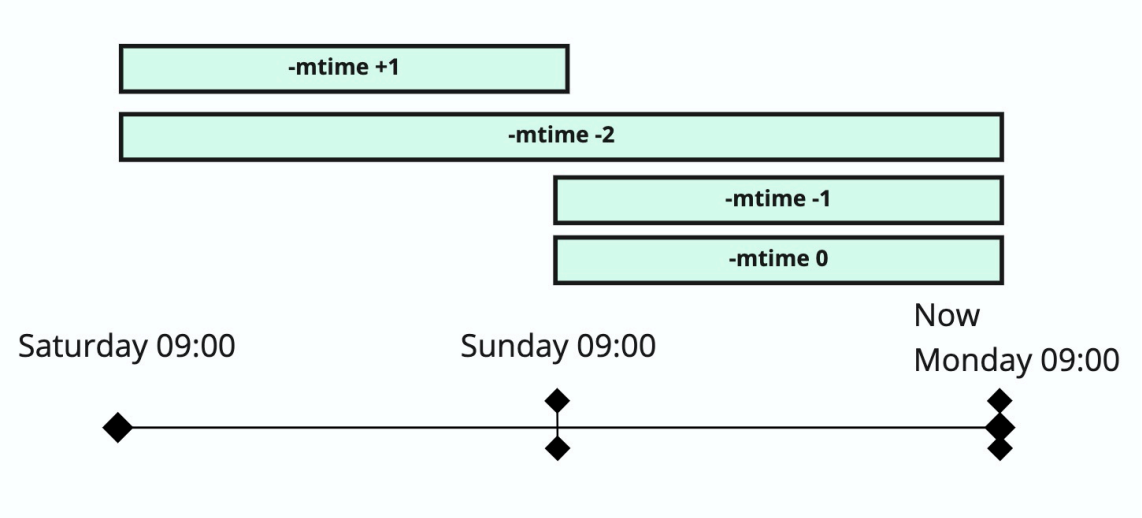
4. You can also try the `dmesg` command in normal Linux to print the kernel ring buffer (kernel logs), but **not** in the Dockers since this interface is not accessible!

🔬 Find modified files in the system (14:57, 4m)

It is critical to know what CHANGED after an attack or infection.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

1. `find`[2]: a powerful tool for searching for files and much more!
    a. The parameter is -mtime [+|-]N
        i. If the file was modified **more (+)** or **less (-)** than **N** days ago.
        ii. When 'find' figures out how many *24-hour periods* ago the file was last accessed, any fractional part is ignored.
        iii. So, for a file to match -mtime +1 (in your mind, "files modified more than 1 day ago"), a file has to have been accessed at **least 1 day** ago (because if not, its mtime can be, for example, 1.5days).
        iv. Without +|- it means **exactly** N*24hs ago. 0 means in the **last** 24hs
        v. Be careful. It is not super precise, and we are not sure why. 😕
            1. (wait, did you just italicized emojis? Yes, I did)
            2. (wait, can you say italicized? For sure, I can)
2. Understanding how `find` works with the -mtime parameter:



3. Examples of how to use `find` to check for modified files:
    a. Files that were modified 24 hours ago:
        i. `find / -mtime 0 -ls | grep -v "proc\|sys" | less`
    b. Files modified less than 48hs ago:
        i. `find / -mtime -2 -ls | grep -v "proc\|sys" | less`
    c. Files modified in the last week (less than 7 days ago):

[2] Cheatsheet for find linux (2024) [online] Gist. Available at: https://gist.github.com/gr1ev0us/3a9b9d9dbdd38f6379288eb2686fc538 [Accessed 17 Oct. 2024].

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

i. `find / -mtime -7 -ls | grep -v "proc\|sys" | less`

4. Alternative for individual files to know when it was last modified
    a. `stat /etc/passwd`

🎲 Miscellaneous checks (15:01, 3m)

1. Past commands. For this check ~/.bash_rc
    a. `less ~/.bash_history`
        i. ~ means 'the home folder of the user'.
        ii. If the first letter of the filename is '.' then the file does not show in the traditional 'ls', unless '-a' is used. AKA 'hidden'.
2. Installed programs with the date of installation
    a. `zgrep " install " /var/log/dpkg.log`
3. Open ports
    a. If an attacker gets in, a process may be listening for connections.
        i. `ss -anpult`
            1. Replaces old `netstat -anp`
            2. -a: All. Display both listening and not listening
            3. -n: Do not resolve DNS
            4. -p: Show the process using the socket.
            5. -l: Are listening for new connections
            6. -t: tcp
            7. -u: udp
    b. What is important? Search for
        i. LISTEN processes in TCP
        ii. Any UDP process
    c. How to find which process opened a port
        i. `ss -p` or `netstat -p`
        ii. If not `lsof -i :22`
        iii. (docker daemon uses the IP 127.0.0.11 and port 60879/UDP as an internal resolver, that is why it doesn't have a process)

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

4. Firewall
    a. A FW filters packets in and out of a system.
    b. We may offer an optional class on this.
    c. Check this for now: https://www.tecmint.com/linux-iptables-commands/

🚪🔙 Common backdoors? (15:04, 8m)

The idea is that attackers may have left behind ways to get into the system later. These are referred to as 'backdoors'.

This is not easy because there are many options.

- Inserted SSHd authentication keys
    - `cat ~/.ssh/authorized_keys`
        - These are the public keys authorized to log in without a password.
        - **!!!CTU Students: DO NOT remove the teachers' SSH keys.!!!**
- Processes executed at login. There are many places to check, some are:
    - `/etc/rc.local` (may not exist)
    - `/etc/profile`
    - `/etc/profile.d/*`
    - `~/.bash_rc`
    - `~/.zshrc`
    - All files in /etc/init.d/
        - They are executed according to the run levels.[3]
    - Using systemd

[3] GeeksforGeeks (2018). Run Levels in Linux. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/run-levels-linux/ [Accessed 17 Oct. 2024].

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

        - `systemctl list-unit-files --type=service | grep enabled`

- Cronjobs: `cron` is a process that executes commands at a certain time and date

```
# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
```

    - Usually abused to execute malware and other attacks when the attacker wants
    - What is the meaning of those * chars?
        - https://crontab.guru/
    - For current user
        - Read and list cron entries
            - `crontab -l`
        - Edit your crontab
            - `crontab -e`
    - For other users, check
        - `ll /var/spool/cron/crontabs/`
    - For the system
        - Master cron of the system
            - `cat /etc/crontab`
        - You can have specials for generic system things
            - `vi /etc/cron.d/logcheck`
        - For specific moments without configuration
            - /etc/cron.daily/
            - /etc/cron.hourly/
            - /etc/cron.monthly/
            - /etc/cron.weekly/

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

Recap: To check if your computer was attacked, you need to find modifications or additions that were not authorized. This is hard and changes over time.

How to secure your computer? (15:12, 0m)

Securing a computer is a constant process and is very, very hard. It requires monitoring of changes, and periodic tests. Companies actually do a penetration test as part of the process to determine what to secure.

🛡️ Hardening User access (15:12, 2m)

Let's harden the system so you have control over who logs in.

The best solution is to use SSH as the only entry point if you need remote access.

VPNs are the second best, but there are so many providers that they can have security issues. Remote desktops are usually good enough, but they are too complex and may have vulnerabilities.

🛜 How to configure SSH to manage users? (15:14, 7m)

SSHD can be configured in /etc/ssh/sshd.conf or /etc/ssh/sshd_config in these dockers.

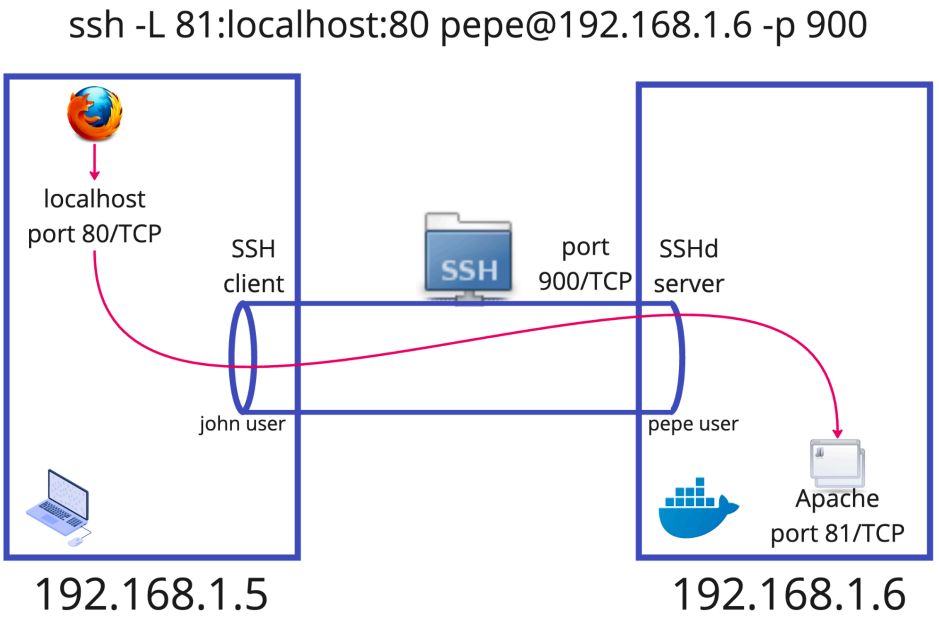
- Configure to forbid the use of text passwords and force only public keys.
    - Edit /etc/ssh/sshd_config, add
        - `PasswordAuthentication no`
    - 🚨ALWAYS CHECK IF IT WORKS **BEFORE** APPLYING!
    - 🚨ALWAYS HAVE A **BACKUP** CONNECTION WHILE YOU CHECK IN CASE IT FAILS!
        - Always create another user, put a password, add it to the sudoers file (next classes), and try to log in **BEFORE** you mess with your root account.
- How to choose which user can log in using SSH at all? Edit /etc/ssh/sshd_config

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

    - `AllowUsers <user to allow>`
    - `DenyUsers <user to deny>`
- Stop root from logging in to the system using SSH
    - `PermitRootLogin no`
        - This affects both with password and with public key.
- You can also allow access to all the users in a group.
    - `AllowGroups sshusers`
- ALWAYS reload SSH for the modifications to take effect. You may be kicked out!
    - `/etc/init.d/ssh reload`
    - ⚠️ **Be careful!** Don't do /etc/init.d/ssh restart . This will restart SSH, and you will get **disconnected,** and you will need to log in again. The docker is 'restarted'. But there is no data lost.
- One of the main problems is weak passwords and passwords by default. That is solved by checking that the passwords are not weak. No user can get created with bad passwords.
    - `vi /etc/pam.d/common-password`
        - `password required pam_cracklib.so retry=3 minlen=8 difok=3 dcredit=1 ucredit=1 lcredit=1`
            - retry=3: how many times the user gets to attempt again.
            - minlen=8: minimum length of the password
            - difok=3: checks the maximum number of reused characters compared to the user's old password.
            - dcredit=1: Minimum number of numerals
            - ucredit=1: Minimum number of uppercase characters
            - lcredit=1: Minimum number of lowercase characters.

👽 If you want to know how to use SSH even for other ports you need, like a web server. For this, use SSH local tunnels if needed.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License



🛡️ Hardening of files and directories (15:21, 12m)

Hardening means tightening and securing the permissions in order not to allow weak points to be exploited by attackers.

In a weak system, attackers may read, write, delete, or copy files that they should not.

Unfortunately, Linux does **not** come by default with a good files/directories/users policy, so most of Linux out there are badly configured.

- ⚠️ If you put good permissions, are your files going to be protected forever and perfectly?
    - Of course not. The hardening of files and directories only works if the operating system is controlling the access.
    - If the hard disk is disconnected and read in another computer, then the permissions don't matter. Only encryption will help.
    - We are **not** going to go into disk encryption but check **veracrypt** and others.
- Check the permissions in files

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

- `ls -alh /etc`

r: can read the file
w: can write the file
x: can execute the file

- Check the permissions in directories

r: can list files
w: can modify the structure of directory, like add or delete files
x: can 'get in' with cd

- For directory permission 'w,' you need also 'x' to make it work.
- There are some security permissions (setuid, setgid) that we are going to see in class 6. Not important for access restrictions to files.
- To change the permissions, use chmod
  - `chmod [u|g|o][+|-][r|w|x] name`
    - u: change the permissions of the user
    - g: change the permissions of the group
    - o: change the permissions of others
    - +: add permissions
    - -: delete permissions

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

        - r: permission r
        - w: permission w
        - x: permission x
        - name: name of file or directory
- Let's try! Remember you are root until the sudo command.
    - `useradd luna`
    - `mkdir /test`
    - `chown root.luna /test`
        - The user owner of /test is `root`
        - The group owner of /text is `luna`
    - `touch /test/file1`
        - As root, create a file
    - `touch /test/file2`
        - As root, create a file
    - `chown luna.luna /test/file2`
        - Make `luna` user and group own the second file
    - `chmod g-w /test`
        - Probably not necessary, given that by default, the group owner has no write anyway.
    - `ls -alh /test/`
        - Check permissions
        - Luna can not write in the /test folder
    - `sudo su - luna`
    - `id:` Test who you are

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

    - `ls -alh /test/`
    - `rm /test/file2`
        - `luna` **can not delete** the file that belongs to the `luna` user!
- This exemplifies the difference between being able to write in a directory or not.

You should check the permissions of:

- `/home`
- `/etc/shadow`
- `/etc/passwd`
- `/var/log`
- `/var/log/*`
- `/`
- `/tmp`
- `/home/*/.ssh/`

~~~~ 💗 First Break! 💗~~~~ (15:33, 10m)

🎭 Umask (15:43, 3m)

Umask is a system-global way to force a default set of permissions for newly created files and directories.

The mode mask contains the permission bits that should not be set on a newly created file. Hence, the mask acts as a filter to strip away default permission bits.

For **files,** the default permissions are **666,** and for **directories, 777**.

- Check the current umask
    - `umask`
- Example to change it
    - `umask 027`
        - This means that when a new file is created, the final permissions will be 666 'minus' 027. That is:
            - 740
            - -rw-r-----
    - You can think of 6-0=6, 6-2=4, and 6-7=0 (not -1)

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

- More easily, you can see what permissions the files **would** have with
    - `umask -S`

🗄️ File Attributes (15:46, 10m)

Attributes are **meta-data** stored in each file. It allows you to store **data** to modify what can be done or not done to the file.

Let's see some:

- `touch filetest`
- `lsattr filetest`

There are many attributes. The following are a few useful ones. (Not all filesystems support every attribute).

    - a - append-only: File can only be opened for appending.
    - c - compressed: Enable filesystem-level compression for the file.
    - i - immutable: Cannot be modified, deleted, renamed, or linked to. It can only be set by root.
    - j - data journaling: Use the journal for file data writes as well as metadata.
    - m - no compression: Disable filesystem-level compression for the file.
    - A - no atime update: The file's atime will not be modified.
    - C - no copy on write: Disable copy-on-write for filesystems that support it.
    - e: The file uses **extents** to map blocks on disk. Improve performance by allocating larger contiguous chunks.
- You can change them with
    - `chattr +c filetest`
        - +: to add
        - -: to delete
        - c: the attribute to change. In this case, on-the-fly compression.
- The complete list is
    - a: append only

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

    - A: no atime updates
    - **c: compressed**
    - C: no copy on write
    - d: no dump
    - D: synchronous directory updates
    - e: extent format
    - F: case-insensitive directory lookups
    - **i: immutable.** file cannot be modified, deleted, renamed or hard linked to. (not in docker with special docker capabilities)
    - j: data journaling
    - m: don't compress
    - P: project hierarchy
    - **s: secure deletion**
    - S: synchronous updates
    - t: no tail-merging
    - T: top of the directory hierarchy
    - **u: undeletable** (not in docker with special docker capabilities)
    - x: direct access to files
    - E: encrypted (only read-only)
    - I: indexed directory (only read-only)
    - N: inline data (only read-only)
    - V: verity (only read-only). Integrity protection, i.e. detection of accidental (non-malicious) corruption.
- We have a problem. In docker, some of them are restricted and need further permissions when docker is run.
    - In docker, you need to run `docker with --cap-add CAP_LINUX_IMMUTABLE`. This applies to `+a` (append-only). For `+j` (data journaling), you require the `CAP_SYS_RESOURCE` capability.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

- Let's try some more
    - `touch testfile`
    - `ls -alu --time-style=full-iso testfile`
        - `u`: show the time of last access and **not** the default time of the last modification.
    - `cat testfile`
        - i.e., access the file
    - `ls -alu --time-style=full-iso testfile`
    - `chattr +A testfile`
        - Put the attribute; do not update access time.
    - `lsattr testfile`
    - `cat testfile`
    - `ls -alu --time-style=full-iso testfile`
    - `chattr -A testfile`
        - Take it out
    - `lsattr testfile`
    - `cat testfile`
    - `ls -alu --time-style=full-iso testfile`

🗄️🗄️ Extended Attributes in files (15:56, 5m)

Apart from attributes to files, there are extended attributes to restrict how to interact with files and directories. Extended attributes are **"name:value"** pairs associated with files and directories. They are just **values** that other processes use.

Is like adding some meta-data value that is not restricted to just access in the filesystem.

There are four extended attribute classes/spaces: **security, system, trusted, and user**.

- To use the extended attributes, you need to install **attr** (already installed in our dockers)
- By default there are no extended attributes

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

    - `touch file.txt`
    - `getfattr file.txt`
    - `getfattr -n user.test file.txt`
        - `-n` name. Dump the value of the named extended attribute.
    - `getfattr -d file.txt`
        - `-d`, Dump the values of all matched extended attributes.

🚀 Use attributes in the User space (16:01, 6m)

User-extended attributes can be used to store arbitrary information about a file. It is up to you. Let's see:

- `touch file.txt`
- `setfattr -n user.note -v "this is mine" file.txt`
- `getfattr -n user.note file.txt`
- `getfattr -d -e base64 file.txt`
    - `-e` Encode values after retrieving them. Valid values of en are "text", "hex", and "base64".

The other attributes, **security**, **system,** and **trusted,** are used by the OS for other things, and you can not modify them by hand.

Trusted can be set and read only by the superuser, so a kind of secret message. Sorry, not in docker as it is.

🧢 Capabilities to processes (16:07, 15m)

Capabilities are a set of special permissions that the kernel allows to the **processes** created from executable **files**. Capabilities provide very fine-grained control over superuser and privileged permissions, **so you can avoid using root all the time**.

Traditionally, in UNIX systems, a process can be **privileged** (effective user ID 0) or **unprivileged** processes (eUID !=0).

- **Privileged** processes bypass all kernel permission checks.
- **Unprivileged** processes are fully checked in their permissions based on the process's credentials (usually effective UID, effective GID, and supplementary group list).

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

- After Linux 2.2, privileged processes can also be separated with capabilities.
- There are MANY capabilities[4]. For example:
    - CAP_NET_ADMIN. Perform various network-related operations
        - interface configuration
        - administration of IP firewall, masquerading, and accounting
        - modify routing tables
        - bind to any address for transparent proxying
    - CAP_NET_BIND_SERVICE
        - Bind a socket to Internet domain-privileged ports (port numbers less than 1024)
- Capabilities are implemented by using extended attributes in the security space!
- Example
    - `getcap /usr/bin/ping`
        - /usr/bin/ping = cap_net_raw+ep
        - Ping needs access to raw sockets to create the packets.
    - You can confirm it **is** a capability by trying to see the capabilities.
        - `getfattr -n security.capability /usr/bin/ping`
            - getfattr: Removing leading '/' from absolute path names
            - # file: usr/bin/ping
            - security.capability=0sAQAAAgAgAAAAAAAAAAAAAAAAAA=
    - Let's try to give a file permission to use RAW sockets in the network

[4] Archlinux.org. (2022). capabilities(7) — Arch manual pages. [online] Available at: https://man.archlinux.org/man/capabilities.7#File_capabilities [Accessed 17 Oct. 2024].

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

    - `touch testfile`
    - `setcap cap_net_raw+ep testfile`
    - `getcap testfile`
- Remove a capability
    - `setcap -r testfile`

🛗 Access Control Lists (ACL) in files (16:22, 15m)

ACLs allow us to apply a more specific set of permissions to a file or directory without (necessarily) changing the traditional ownership and permission letters. **Solves the issues of multiple groups doing different things to files.** It is more granular.

- ACL are implemented as system-extended **attributes**!
- If necessary, install acl. It should be in BSY dockers.
    - `apt-get install acl`
- You can list ACLs
    - `touch testfile`
    - `getfacl testfile`
        - # file: testfile
        - # owner: root
        - # group: root
        - user::rw-
        - group::r--
        - other::r–

Situation: We are super hackers, and we want our own folder only for root. But from time to time we want some individual user to write on it. But we **don't** want the user to be in the root group.

- Let's do it:

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

    - `mkdir /tmp/only-hackers`
    - `getfacl /tmp/only-hackers`
    - `adduser newbie`
        - Put a password
    - `tmux new -t newbie-session`
        - `su - newbie`
            - So we change users from root to newbie
        - `touch /tmp/only-hackers/mine`
            - *touch: cannot touch 'only-hackers/mine': No such file or directory*
        - `CTRL-b d` (get out of tmux)
- Let's make the newbie user to be able to write in the `only-hackers` folder
    - `setfacl -m newbie:rwx /tmp/only-hackers/`
        - -m is to modify
        - newbie: the username
        - rwx: what we want this user to be able to do
    - `getfacl /tmp/only-hackers/`
        - # file: /tmp/only-hackers/
        - # owner: root
        - # group: root
        - user::rwx
        - user:newbie:rwx
        - group::r--
        - mask::rwx
        - other::---

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

    - `tmux a -t newbie-session`
        - `touch /tmp/only-hackers/mine`
        - Should work!

Of course, we can always avoid the evil user to write on it, even if it is a member of the group. You need a user called 'evil'.

- `setfacl -m evil:- /tmp/only-hackers`

ACLs also allow you to force default users and groups for the new files created in a specific directory.

- If you create a file in /tmp/only-hackers, the default permissions are
    - `touch /tmp/only-hackers/test1`
    - `ll -alh /tmp/only-hackers/`
        - total 8.0K
        - drwxr-xr-x 2 root root 4.0K Oct 25 21:43 .
        - drwxrwxrwt 1 root root 4.0K Oct 25 21:43 ..
        - -rw-r--r-- 1 root root 0 Oct 25 21:43 test1

If we want all the files in the /tmp/only-hackers directory to be rw------- (600) by default. But also, if they are folders, we want rwx------ (700). And the group owner and others should not have any permission. The command is:

- `setfacl -dRm u::rwX,g::-,o::0 /tmp/only-hackers`
    - -d: default for directories
    - -f: default for files
    - -R: recursive
    - -m: modify
    - [user/group/other]:[uid/gid]:[perms]
    - X: capital X in the permissions means to apply the 'x' permission but only if it is a directory or if the permission is already there. This is to avoid making files executable automatically.
    - 0 (zero) are permissions in octal. The same is - as permissions.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

- `touch /tmp/only-hackers/test1`
- `ll -alh /tmp/only-hackers/`
    - total 8.0K
    - drwxr-xr-x+ 2 root root 4.0K Oct 25 21:45 .
    - drwxrwxrwt 1 root root 4.0K Oct 25 21:43 ..
    - -rw-r--r-- 1 root root   0 Oct 25 21:43 test1
    - -rw------- 1 root root   0 Oct 25 21:45 test2

~~~~ 💗 Second Break! 💗~~~~ (16:37, 10m)

🛡️ How to delete unwanted users from our system? (16:47, 0m)

Imagine that you found a suspicious user in the system, and you want to delete all the files owned by that user. How to do it?

📁 Delete files (16:47, 10m)

1. First, **backup** all their files in your system
    a. To test: create a dummy user and file
        i. `useradd -u 12000 luna`
        ii. `touch /lunalunera`
        iii. `chown luna.luna /lunalunera`
    d. Find files belonging to the user you want to search
        i. `find / -user luna`
    e. Copy and tar the files automatically
        i. `find / -user luna -exec tar -rvf /backup.tar {} \;`
    f. Change the owner of the backup tar files
        i. `chown root.root -R /backup.tar`
        ii. You can keep this file or move it to another system for analysis.
    g. Change permissions of tar file so only root can read it

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

i. `chmod 700 /backup.tar`

h. Alternity: Copy their files automatically for backup (you are the owner later)

i. `mkdir /backup`

ii. `find / -user luna -exec cp -r {} /backup \;`

iii. Delete backup files in the folder

1. `rm -rf /backup`

5. ☠️🚨 **Delete** their original files in your system from one user.

i. `find / -user luna -exec rm -rf {} \;`

📁 Delete user accounts (16:57, 5m)

1. Delete their lines in /etc/passwd and /etc/shadow
   a. Be careful with the editor's backup versions /etc/shadow- and /etc/passwd- if you edit them.
2. Delete their usernames in /etc/group (the system groups)
3. Remove their home folders (including their SSH keys)
   b. `rm -rf /home/<change the user name>`
   c. Check /home/<user>/.ssh
      i. /home/<user>/.ssh/authorized_keys
6. You can also use the system command (will not delete the home folder)
   a. `deluser luna`
   b. `userdel luna`
7. You can use the system command AND delete the home folder:
   a. `deluser --remove-home luna`

🛡️ Host-Based Intrusion Detection Systems: AIDE (17:02, 20m)

This is all good and pretty, but can't we just automate it? Yes, we can.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

1. **Online** Students only, install AIDE.
    a. `apt install aide`
2. What is AIDE (Advanced Intrusion Detection Environment)?
    a. It is a file integrity checker.
    b. It creates a snapshot database of files/directory properties and then checks the system against that database.
        i. For example: *file type, permissions, inode number, user, group, file size, mtime, ctime, atime, growing size, number of links, and link name.*
3. Let's try it
    a. Your configuration in BSY dockers is now modified to only monitor the /etc directory, so it's faster.
    b. In StratoCyberLab is left by default because it is a smaller system.
    c. To initialize the DB, run:
        i. `aide --init --config=/etc/aide/aide.conf`
        ii. A large system can take a long time.... Like hours. So check the conf file and /etc/aide/aide.conf.d/ files.
    d. Now copy the **newly** generated DB with the following command to be the official DB.:
        i. `cp /var/lib/aide/aide.db.new /var/lib/aide/aide.db`
4. Let's try to check modifications in your computer using the stored DB
    a. `aide --check --config=/etc/aide/aide.conf`
5. Aide check only generates files in stdout
    a. For example, if you still didn't get your output

```
Start timestamp: 2020-10-21 18:53:30 +0000 (AIDE 0.16.1)
AIDE found differences between database and filesystem!!
Verbose level: 6

Summary:
  Total number of entries:41171
  Added entries:          322
  Removed entries:        1
  Changed entries:        41
```


This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

```
---------------------------------------------------
Added entries:
---------------------------------------------------

f++++++++++++++++: /etc/cron.d/logcheck
d++++++++++++++++: /etc/logcheck/cracking.d

---------------------------------------------------
Removed entries:
---------------------------------------------------

s----------------: /run/screen/S-root/20447.aideinit
---------------------------------------------------
Changed entries:
---------------------------------------------------

f >....    . ..  : /dev/tty10
f >.... mc..C.. .: /etc/aliases
d =.... mc.. .. .: /etc/cron.d
d =.... mc.. .. .: /etc/cron.daily
---------------------------------------------------
Detailed information about changes:
---------------------------------------------------
File: /etc/aliases
  Size     : 185                              | 200
  Mtime    : 2020-10-20 12:16:58 +0000        | 2020-10-21 18:40:37 +0000
  Ctime    : 2020-10-20 12:16:59 +0000        | 2020-10-21 18:40:37 +0000
  RMD160   : Yc6MnN56GVq0H71ihHVwB19J9jE=     | FXrOEAKQXccBS/U3q9dh5RB5omE=
  TIGER    : aR6EXgbc5aUePggyMKc/4SzZti9LV+xB | onH7ExkjyZGDi9810iSjvxjYRVeF9alK
  SHA256   : ebB3R8fA5ZglxGFkZYDqOnMJstQe6BJD | aglClQYou6prqNv5a/iRlDzUlwhLlpOw
             Rl4xmo3p9ao=                     | OpVfqoN7BPY=
  SHA512   : qSE7OQJM1OsvrH3C4JqlGz5AxomUYv00 | 35CJbU4SlIA7GuDcZ3UL17P568Yr50/G
             sQi5VTUOnm8JoXXm7ix57wM+uMciyKcR | l92QkRca9r54pIuT+fFgTPl+cXjS2IIg
             p3dRXE/1Ofh5AzcIAlnx2A==         | 5tKtnv+dX9oJVQJWsX9yhw==
  CRC32    : M5knBg==                         | EV5Dsg==
  HAVAL    : pJIfRZB7Asu4Gm8gYsVUI7bbwT/hvgu9 | RGnmunBTYDhCumsCi8Urx4RjHbE/mN4P
             HOjC1jUouuA=                     | GyDvyco4790=
  GOST     : KitlDKOK7wz3yOCF4IXN88p40z1Y6knv | q7Iy8ZN3qYnZ0F4EQcY3GLY2YURdNI4R
             N2tuWT5WK8o=                     | MyWxhzkowEA=
```

6. AIDE gives the **added entries, removed entries, and changed entries.**
7. An example line is **“f++++++++++++++++”.** This string is formatted like “**YlZbpugamcinCAXS**”
    a. Y: type of file. Can be: f for a regular file, d for a directory, L for a symbolic link, D for a character device, B for a block device, F for a FIFO, s for a Unix socket, and ? otherwise
    b. In ‘*Changed Entries,*’ there is also the change indicator

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

i. The Z is replaced as follows: A = means that the size has not changed, a < reports a shrunk size and a > reports a grown size.

c. The other letters are the actual letters that will be output if the associated attribute has changed or:

i. **"."** for no change,

ii. "+" if the attribute has been added,

iii. "-" if it has been removed,

iv. ":" if the attribute is listed in ignore_list or

v. " " if the attribute has not been checked. (a newly created has all "+", removed file has all "-")

d. The attribute that is associated with each letter is as follows:

i. l → means that the link name has changed.

ii. b → means that the block count has changed.

iii. p → means that the permissions have changed.

iv. u → means that the uid has changed.

v. g → means that the gid has changed.

vi. a → means that the access time has changed.

vii. m → means that the modification time has changed.

viii. c → means that the change time has changed.

ix. i → means that the inode has changed.

x. n → means that the link count has changed.

xi. C → means that one or more checksums have changed.

xii. A → means that the access control list has changed.

xiii. X → means that the extended attributes have changed.

xiv. S → means that the SELinux attributes have changed.

8. If your check is ready, let's read it! Anything interesting? (17:30)

9. If you want to update the DB with the changes (not now, will take time)

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

    a. `aide --update --config=/etc/aide/aide.conf`

    b. `cp /var/lib/aide/aide.db.new /var/lib/aide/aide.db`

10. You can configure AIDE to run in CRON daily, and this generates the files in /var/log/aide/

11. Since any root user can modify the DB, you should copy it out.

🤖 Automated check of the security of a Linux (17:22, 10m)

There are some tools that go through the whole system, checking for vulnerabilities and security issues. Unfortunately, there are not many options.

One of the latest is done by a company (as open source) and is called Lynis[5]

- `git clone https://github.com/CISOfy/lynis.git`
- `cd lynis`
- `./lynis audit system`

🪵 Extra: Automatic analysis of log files (17:32, 5m)

1. `apt-get install logcheck` (already installed in Dockers)

    a. `su -s /bin/bash -c "/usr/sbin/logcheck -o -l /var/log/auth.log" logcheck`

        i. su: executes a command as another user

        ii. -s: run this shell /bin/bash

        iii. -c: command to run.

        iv. -o: don't send emails, just stdout

        v. Logcheck: Run as logcheck user

🚨 **CAREFUL**! When you run logcheck it marks the analyzed logs and does not show the past events upon the next runs.

[5] CISOfy (2024). Lynis. [online] Cisofy.com. Available at: https://cisofy.com/lynis/ [Accessed 17 Oct. 2024].

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

Parish Notices (17:37, 3m)

The next class (24.10.2024) will be about Deception methods by Muris. For the first time, we will have a guest lecturer for a part of the class (online):

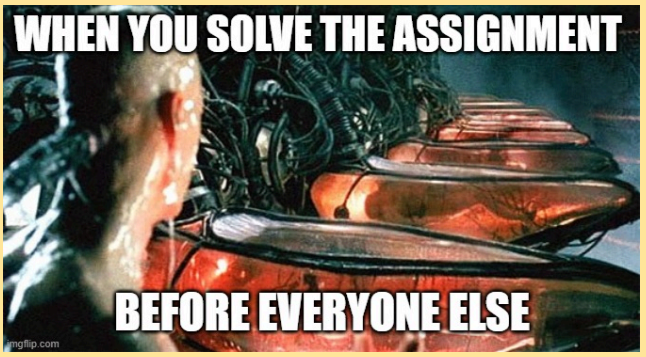
Tim Pappa is an Incident Response Engineer - Cyber Deception Strategy, Content Development, and Marketing, Cyber Deception Operations, Walmart Global Tech. Before Walmart, Tim was a Supervisory Special Agent and profiler with the Federal Bureau of Investigation's (FBI) Behavioral Analysis Unit (BAU), specializing in online influence and cyber deception. Tim is also a Senior Behavioral Consultant for Analyst1 and an Adjunct Professor at Capitol Technology University.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

Assignment 5 (17:40, 5m)

CTU Students - Assignment 5 (5 Points)

1. This week's assignment will open on Thursday 17th, 9PM CEST.
2. The assignment has several parts. You will need to login to your dockers, investigate, and answer the questions in CTFd.
3. Do not block user access by SSH (pass or public key).
4. Don't harden your dockers before the assignment starts.



Class Feedback

By giving us feedback after each class, we can make the next class even better!

bit.ly/BSYFeedback

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License