Source: *Stable Diffusion AI*

LABORATORY 5. LINUX OS SECURITY (NON-AUTOMATABLE)

DEPARTMENT OF COMPUTER SCIENCE. UNIVERSITY OF OVIEDO

Computer Security | 2022 – 2023 (V2.7 *"S-81 Isaac Peral"*)

CONTENT

THE INFRASTRUCTURE OF THIS LABORATORY

This laboratory only works with the *Ubuntu* virtual machine of the course. However, there are a series of considerations that you should read carefully.

- **We strongly recommend creating a snapshot of the VM first**.
- Multiple activities consist in going to the indicated section of a **CIS benchmark PDF** file and follow their detailed instructions. **The CIS benchmark file to use is provided along with this laboratory report**, so please download it. CIS benchmarks are described in the "*Security Policies*" theory topic. Do not worry if we have not reached this topic yet, for this laboratory, they are only a detailed set of instructions to follow to check or implement certain security operations. In other words, **it is your instruction manual** 😊. We are only applying part of it though.

GENERAL WARNING: This laboratory makes heavy usage of copying commands from PDF files. Please CHECK CAREFULLY what you copy, as Unicode characters like ", -, etc. may be considered syntactically wrong at the command line, as these are normally copied from files "as is", so you need to replace them by ASCII-equivalent ones. This is especially true with quotes. If the syntax of a command does not work, check carefully you command line for invalid characters copied accidentally. This is applicable from now on, even in future laboratories.

BLOCK 1: USER-RELATED SECURITY

Current security level evaluation

Practical application: *You need to know what the current security level of your Linux machine is*

Go to Laboratory 1 report and follow the `lynis` audit procedure to check the security of your VM. Store this result to compare it later with the hardened machine one.

Expected results: This activity is finished when you can evaluate the security of a VM with `lynis`.

Accounts and Access: Prevent single-user mode

Practical application: *You need to prevent anybody booting on your machine and acquiring root privileges by entering single-user (aka maintenance) mode*

One of the most important things you should do regarding account management is to **prevent access to single user mode**. Single user mode is used for disaster recovery when the system detects an issue during boot, or by manual selection from the bootloader. To prevent single user mode boot, follow *CIS benchmark* task ***1.5.3 Ensure authentication required for single user mode***.

Expected results: This activity will be finished when you can prevent non-authenticated boot into single user mode according to the CIS benchmark specifications.

Warning banners

Practical application: *You need to tell potential intruders to get off your lawn (like Clint Eastwood* 😊*)*

It is a good security practice to warn external users about the consequences of connecting to a system, in case they are not legitimate users. To put correct warning banners into the different access points of a system, follow *CIS benchmark* task ***1.8.1 Command Line Warning Banners***. Concretely, run only these tasks:

- ***1.8.1.1 Ensure message of the day is configured properly.***
- ***1.8.1.2 Ensure local login warning banner is configured properly.***
- ***1.8.1.3 Ensure remote login warning banner is configured properly.***

You can use any text for the warning messages, for example: https://www.tecmint.com/protect-ssh-logins-with-ssh-motd-banner-messages/

Expected results: This activity will be finished when you have proper warning banners configured in your system according to the CIS benchmark specifications.

Additional security for OpenSSH

Practical application: *You need to strengthen remote connections via SSH as much as possible*

In previous laboratories we installed *OpenSSH*, teach you to create secure passwords, and even provide a password-less method to connect through it. This enhances its security, but we can go much further following the *CIS benchmark* task ***5.2 SSH Server Configuration*** and most of its subtasks. However, it is recommended to **make a backup of the current configuration** in case something goes wrong and `ssh` is unavailable because of it. You can always login via the virtualization software console.

```
sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.factory-defaults
sudo chmod a-w /etc/ssh/sshd_config.factory-defaults
```

Once this is done, modify `/etc/ssh/sshd_config` with the values and directives indicated by the following CIS benchmark tasks. `Ctrl+W` is the search command in the `nano` editor if you decide to use it. If a directive name is not currently in the SSH configuration file, add a line with it and the recommended value.

- ***5.2.1 Ensure permissions on `/etc/ssh/sshd_config` are configured.***
- ***5.2.2 Ensure permissions on SSH private host key files are configured.***
- ***5.2.3 Ensure permissions on SSH public host key files are configured.***
- ***5.2.4 Ensure SSH `Protocol` is not set to 1.***
- ***5.2.5 Ensure SSH `LogLevel` is appropriate.***
- ***5.2.6 Ensure SSH X11 forwarding is disabled.***
- ***5.2.7 Ensure SSH `MaxAuthTries` is set to 4 or less.***
- ***5.2.8 Ensure SSH `IgnoreRhosts` is enabled.***
- ***5.2.9 Ensure SSH `HostbasedAuthentication` is disabled.***
- ***5.2.10 Ensure SSH root login is disabled.***
- ***5.2.11 Ensure SSH `PermitEmptyPasswords` is disabled.***
- ***5.2.12 Ensure SSH `PermitUserEnvironment` is disabled.***
- ***5.2.13 Ensure only strong Ciphers are used.***
- ***5.2.14 Ensure only strong MAC algorithms are used.***
- ***5.2.15 Ensure only strong Key Exchange algorithms are used.***
- ***5.2.16 Ensure SSH Idle Timeout Interval is configured.***
- ***5.2.17 Ensure SSH `LoginGraceTime` is set to one minute or less.***
- ***5.2.19 Ensure SSH warning banner is configured.***
- ***5.2.20 Ensure SSH PAM is enabled.***
- ***5.2.21 Ensure SSH `AllowTcpForwarding` is disabled.***
- ***5.2.22 Ensure SSH `MaxStartups` is configured.***
- ***5.2.23 Ensure SSH `MaxSessions` is set to 4 or less.***

Expected results: This activity will be finished when you strengthen the configuration of the `ssh` service according to the CIS benchmark specifications.

Secure PAM module configuration

Practical application: *You need to strengthen your password policies as much as possible*

PAM (*Pluggable Authentication Modules*) is a service that implements modular authentication modules on UNIX systems. PAM is implemented as a set of shared objects that are loaded and executed when a program needs to authenticate a user. Files for PAM are typically located in the `/etc/pam.d` directory. PAM must be carefully configured to secure system authentication, and we can do it following the *CIS benchmark* task ***5.3 Configure PAM*** and its subtasks.

- ***5.3.1 Ensure password creation requirements are configured.***
- ***5.3.2 Ensure lockout for failed password attempts is configured.***
- ***5.3.3 Ensure password reuse is limited.***
- ***5.3.4 Ensure password hashing algorithm is SHA-512.***

Expected results: This activity will be finished when you strengthen the system password policy according to the CIS benchmark specifications.

Password age and inactive accounts

Practical application: *You need to remove unused accounts and old passwords*

Password age (the number of days a password has been left unchanged) and inactive accounts may be a vector of security problems. We can control this vector by applying the *CIS benchmark* task ***5.4.1 Set Shadow Password Suite Parameters*** and its subtasks:

- ***5.4.1.1 Ensure password expiration is 365 days or less.***
- ***5.4.1.2 Ensure minimum days between password changes is configured.***
- ***5.4.1.3 Ensure password expiration warning days is 7 or more.***
- ***5.4.1.4 Ensure inactive password lock is 30 days or less.***

Expected results: This activity will be finished when you strengthen the system password expiration and inactivity policy according to the CIS benchmark specifications.

Restrict root and system accounts logins

Practical application: *You need to make sure root has no possibility of interactive login*

There are a series of measures required to harden `root` and system account logins or check that these are correctly set up. We can do that following the *CIS benchmark* task ***5.4.2 Ensure system accounts are secured*** and the task ***6.2.6 Ensure root is the only UID 0 account***.

Expected results: This activity will be finished when you strengthen the login policy of the `root` and the system accounts according to the CIS benchmark specifications.

Prevent using common passwords

Practical application: *You need to prevent your users to use common and easily crackable passwords*

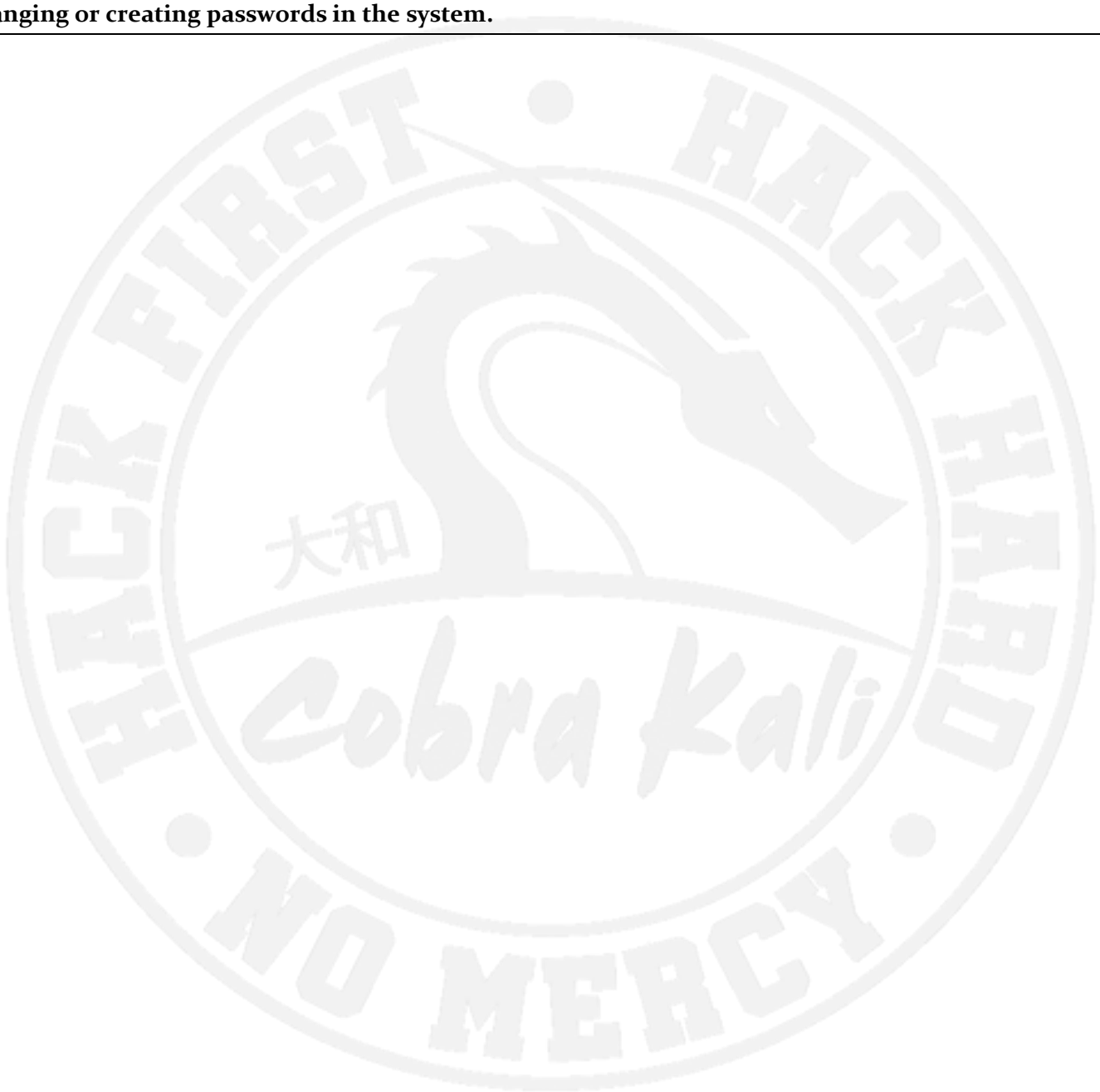
These operations help enforcing a proper password policy. It is not enough to enforce a strong password policy; we need to complement it with preventing common password usage to make sure unwanted logins are not done over our system. This activity prevents creating passwords that belong to a known list of the one million most used passwords on Internet: https://github.com/danielmiessler/SecLists.

- Install: `sudo apt-get install libpam-cracklib -y`

- Download the common password list: `sudo wget https://raw.githubusercontent.com/danielmiessler/SecLists/master/Passwords/xato-net-10-million-passwords-1000000.txt /usr/share/dict/ -O /usr/share/dict/million.txt`
- Enforce the list: `sudo create-cracklib-dict /usr/share/dict/million.txt`

You can check this trying to change the password of any user (or creating a new user) and using a word from this list as the password you intend to put. It should give you a warning.

Expected results: This activity will be finished when you prevent common password usage when changing or creating passwords in the system.



BLOCK 2: PROCESS-RELATED SECURITY

Compilers

Practical application: *You need to remove unused compilers that can be used to deploy malware*

If you are not really using them, you can disable common compilers installed in most *Ubuntu* systems by issuing the following commands. For simplicity, you can put them into a `.sh` file, give them executable permissions, and run it directly. **NOTE: Multiple "directory not found" errors are not a problem: they mean that a compiler is not installed**. Do not add more compilers to this script though, as some are needed for the whole OS to work and disabling them may leave you without a running system.

```
chmod 000 /usr/bin/byacc
chmod 000 /usr/bin/yacc
chmod 000 /usr/bin/bcc
chmod 000 /usr/bin/kgcc
chmod 000 /usr/bin/cc
chmod 000 /usr/bin/gcc
chmod 000 /usr/bin/*c++
chmod 000 /usr/bin/*g++
```

Expected results: This activity will be finished when you disable C/C++ compilers in the systems, if there are any.

Disable core dumps

Practical application: *You need to remove core dump files that can be used to obtain confidential information*

Follow the *CIS benchmark* task ***1.6.4 Ensure core dumps are restricted*** to disable this potential source of confidential information

Expected results: This activity will be finished when you prevent core dumps according to the CIS benchmark specifications if core dumps are activated in the base system.

AppArmor

Practical application: *You need to ensure that AppArmor MAC system is enabled and confine your Firefox browser*

AppArmor is a mandatory access control system (MAC) that limits programs to a set of resources they can use or access. Restricts programs to a set of files, attributes, and capabilities, so they are not able to do serious harm if somehow, they become or are malicious (unless given permission). *AppArmor* works at the kernel level and loads during boot. *AppArmor* manages permissions through *profiles*. These are a set of rules that determine what the program can and cannot do. There are two ways profiles can be applied:

- **Enforce**: Application of the policy defined in the profile and notifications of violation attempts of it.
- **Complain**: Only reports violation attempts of the policy but does not enforce it.

Most profiles are loaded into *Enforce* mode, although there may third-party profiles that are also loaded into *Complain* mode.

To check that *AppArmor* is correctly configured into a system, follow the *CIS benchmark* task ***1.7.1 Configure AppArmor***, but more precisely these two.

- ***1.7.1.1 Ensure AppArmor is installed***
- ***1.7.1.3 Ensure all AppArmor Profiles are in enforce or complain mode***

Search and apply disabled *AppArmor* profiles

Apart from the profiles that run on boot, there may be some of them that are disabled by default. We can check if these exist in the folder `/etc/apparmor.d/disable`. You may find there two disabled profiles: *Firefox* and *Rsyslogd* . As *Firefox* is a sensitive application, we should enable it following this procedure

- Install this package: `sudo apt install apparmor-utils`
- From a terminal, write: `sudo aa-enforce /etc/apparmor.d/usr.bin.firefox`
- If you want to disable it again, do:

```
sudo ln -s /etc/apparmor.d/usr.bin.firefox /etc/apparmor.d/disable/
sudo apparmor_parser -R /etc/apparmor.d/usr.bin.firefox
```

Expected results: This activity will be finished when you check that the status of *AppArmor* follows CIS benchmarks specifications and manage to enable an *AppArmor* profile for *Firefox*.

Install apt security software

Practical application: *You need to be sure not to install corrupted packages*

You can obtain more security in software installations through `apt` installing the packages `debsums` and `apt-show-versions`. These will avoid installing maliciously altered packages. To test this, follow these instructions: https://manpages.ubuntu.com/manpages/trusty/man1/debsums.1.html to check the integrity of some packages you choose

Expected results: This activity will be finished when the software is installed, and you check the integrity of some packages

Process accounting

Practical application: *You need to activate a log of process behavior for later (forensic) usage*

Process accounting enables logs of process behaviors that may be useful for forensic analysis and to understand failures. Installing it is quite easy:

- Install `acct` package: `sudo apt install acct`
- Create a log file: `sudo touch /var/log/pact`

- Activate process accounting: `sudo accton /var/log/pacct`

Expected results: This activity will be finished when the software is installed and running.

BLOCK 3: FILE-RELATED SECURITY

Session configuration files

Practical application: *You need to put the default file permission to the most adequate secure configuration*

New files may be created with a default permission configuration whose security can be improved. To prevent this, follow the *CIS benchmark* task ***5.4.4 Ensure default user umask is 027 or more restrictive***.

Expected results: This activity will be finished when the default `umask` is enforced to a secure value according to the CIS benchmarks specifications.

HIDS: Tripwire

Practical application: *You need to monitor any important file changes in your OS to detect potential unwanted behavior*

This activity covers the *CIS benchmark* task ***1.4 Filesystem Integrity Checking*** and its subtasks. Although the benchmark uses a different HIDS, this one covers the same requirements. This activity is based in the nature of the files in the hard disk; files on the hard drive can be divided into two categories:

- Those that will be typically modified frequently: User files, websites, logs…
- Files that are rarely modified: device drivers, system files…a modification of a file of this second group may indicate an intrusion into the server or the presence of certain types of malware (or also a system update 😊).

There is a way to detect intrusions that modify "sensitive" files with tools like `tripwire`: a *HOST-based Intrusion Detection System (HIDS)* that alerts the administrator when a file considered important (or that belong to the operating system) is modified, so actions can be taken to investigate the issue. The first step is installing this software on the VM (`sudo apt install tripwire`).

This installation will be a guided process in which the installer will ask us for required parameters . These are:

- Detected security incidents will be normally reported using an email program like `postfix`. In our tests we are not going to do this, so when prompted to configure the email we should configure it as "*Local only*" (this is not adequate for production environments!). We also leave the default email name.
- *Tripwire* requires two keys to perform a cryptographic signature of the files it monitors so changes are detected (remember the *Cryptography* topic). It is advisable to let the installer generate them. There are two keys: **site key** and **local key**, and we will be prompted for passwords protecting both. **These passwords are NOT stored, so please do not forget them!**
    - The site key sign files that are common to multiple systems
    - The local key signs files that are typical of the local OS
- *Tripwire* database is ciphered with the site key. If we change the configuration of `tripwire`, we should regenerate this database and the site key passphrase will be required.
- After generating the protected keys, a file with the system file monitoring policy will be created. This is the policy followed by `tripwire` on each execution (`twpol.txt`)
- Finally, to use *Tripwire* we need to initialize it (`sudo tripwire --init`): it will run through the filesystem, examine the files according to the policy, signing them and storing these signatures in an encrypted database. All the configuration is stored in `/etc/tripwire`

- Unfortunately, the policy that is installed by default typically returns a lot of errors when initialized. These are produced due inspecting files that change on each boot or do not exist. **We need to modify its policy** to exclude them and have a cleaner execution. To do that, replace the `/etc/tripwire/twpol.txt` file with the version we provide in the laboratory files, which is the same file with some lines that refer to these problematic files / paths commented.
- Once this is done, save the changes and recreate the *Tripwire* policy with: `sudo twadmin -m P /etc/tripwire/twpol.txt`. We need to introduce the site key.
- Finally, we can reinitialize `tripwire`: `sudo tripwire --init` to obtain an error-free system file status baseline, useful to detect unwanted changes.
- Once finished, we can check the filesystem with `sudo tripwire --check`

Expected results: This activity will be finished when you are able to install and use Tripwire correctly

BLOCK 4. HOST NETWORKING SECURITY

Network stack configuration of an individual server

Practical application: *You need to configure your network stack to the most secure configuration*

This part of the laboratory provides **a more secure configuration to the network stack of a host** via the *CIS Benchmarks***.** These are a series of settings that may configure a host network stack to avoid certain attack types. They may be done over individual servers no matter if there is a firewall active or not to strengthen the security configuration of the system. In other words, in our secure network architecture (SNI) of the theory topic 5, this may be done over each of its servers, no matter their subnet or functionality.

This activity consists of implementing the following *CIS benchmark security controls* on a file. Please note that almost all of them require adding or modifying lines in the same file, so you can make all of them together just following the instructions and changing the file with all the parameters accordingly in a single step.

This requires implementing these tasks from section ***3 Network Configuration*** of the provided *CIS Benchmark* file .

- ***3.1 Network Parameters (Host Only)***
  - *3.1.1 Ensure packet redirect sending is disabled*
  - *3.1.2 Ensure IP forwarding is disabled*
- ***3.2 Network Parameters (Host and Router)***
  - *3.2.1 Ensure source routed packets are not accepted*
  - *3.2.2 Ensure ICMP redirects are not accepted*
  - *3.2.3 Ensure secure ICMP redirects are not accepted*
  - *3.2.4 Ensure suspicious packets are logged*
  - *3.2.5 Ensure broadcast ICMP requests are ignored*
  - *3.2.6 Ensure bogus ICMP responses are ignored*
  - *3.2.7 Ensure Reverse Path Filtering is enabled*
  - *3.2.8 Ensure TCP SYN Cookies is enabled*
  - *3.2.9 Ensure IPv6 router advertisements are not accepted*

Expected results: This activity will be finished when you configure the indicated parameters of the network stack of your VM as specified by the CIS Benchmark.

Disable uncommon network protocols

Practical application: *You need to remove network protocols you do not use to minimize your exposure*

The requires implementing the following tasks of the section ***3.4 Uncommon Network Protocols*** from the *CIS Benchmark* file we provided.

- ***3.4 Uncommon Network Protocols***
  - *3.4.1 Ensure DCCP is disabled*
  - *3.4.2 Ensure SCTP is disabled*
  - *3.4.3 Ensure RDS is disabled*
  - *3.4.4 Ensure TIPC is disabled*

Expected results: This activity will be finished when you ensure that the indicated network protocols are disabled on your VM as specified by the CIS Benchmark.

BLOCK 5: LOGGING AND MONITORING

Configure audit rules

Practical application: *You need to enable a complete auditing profile in your Linux SO*

The requires implementing the following tasks of the section ***4 Logging and Auditing*** from the *CIS Benchmark* file we provided. However, this implementation can be much simplified thanks to the files included as part of this laboratory materials in the `audit` folder. These files are the implementation of each security control that involves editing or creating a file in the `/etc/audit/rules.d/` folder. Therefore, you only need to copy all these files to that folder to implement the remediation of all these security controls. This way most of the controls of this list will be automatically implemented:

- ***4 Logging and Auditing***
  - *4.1 Configure System Accounting (auditd)*
    - *4.1.1 Ensure auditing is enabled*
      - *4.1.1.1 Ensure auditd is installed*
      - *4.1.1.2 Ensure auditd service is enabled*
      - *4.1.1.3 Ensure auditing for processes that start prior to auditd is enabled*
      - *4.1.1.4 Ensure audit_backlog_limit is sufficient*
    - *4.1.2 Configure Data Retention*
      - *4.1.2.1 Ensure audit log storage size is configured*
      - *4.1.2.2 Ensure audit logs are not automatically deleted*
      - *4.1.2.3 Ensure system is disabled when audit logs are full*
    - *4.1.3 Ensure events that modify date and time information are collected*
    - *4.1.4 Ensure events that modify user/group information are collected (Scored)*
    - *4.1.5 Ensure events that modify the system's network environment are collected*
    - *4.1.6 Ensure events that modify the system's Mandatory Access Controls are*
    - *collected*
    - *4.1.7 Ensure login and logout events are collected*
    - *4.1.8 Ensure session initiation information is collected*
    - *4.1.9 Ensure discretionary access control permission modification events are*
    - *collected*
    - *4.1.10 Ensure unsuccessful unauthorized file access attempts are collected*
    - *4.1.11 Ensure use of privileged commands is collected*
    - *4.1.12 Ensure successful file system mounts are collected*
    - *4.1.13 Ensure file deletion events by users are collected*
    - *4.1.14 Ensure changes to system administration scope (sudoers) is collected*
    - *4.1.15 Ensure system administrator actions (sudolog) are collected*
    - *4.1.16 Ensure kernel module loading and unloading is collected*
    - *4.1.17 Ensure the audit configuration is immutable*

Install *sysstat* to see resource consumption

Practical application: *You need to monitor resource usage in your Linux OS*

Monitoring resource consumption in a system can identify suspicious behaviors that could be impairing the system performance. `sysstat` (https://github.com/sysstat/sysstat) is a popular package that contains various utilities to monitor system performance and usage activity that we can install with `sudo apt install sysstat`:

- `iostat` reports CPU and input/output statistics for block devices and partitions.
- `mpstat` reports individual or combined processor related statistics.

- `pidstat` reports statistics for Linux tasks (processes): I/O, CPU, memory, etc.
- `tapestat` reports statistics for tape drives connected to the system.
- `cifsiostat` reports CIFS (*Common Internet File System*) statistics.

Expected results: This activity will be finished when you are able to check the processor usage statistics using this package.

Install *glances*

Practical application: *You need to monitor CPU usage in your Linux OS*

Glances is another popular package that allow to monitor all resource usage in a system in a single screen, so they can be managed in real time easier. You can install it as a typical package: `sudo apt install glances` and use it running `glances` at command line.

Expected results: This activity will be finished when you are able to check the processor usage statistics using this package.

Automated log analysis: *LogCheck*

Practical application: *You need basic log checking capabilities in your Linux OS*

LogCheck is an automated log checking tool that may help us to detect unusual behavior in the system that should indicate suspicious activity. The optional ***Forensics*** course deals with these types of activities (and other related ones), so we are not going to explain this in detail, only show you how to perform basic detection of anomalies. To install it follow this procedure:

- Install it: `sudo apt install logcheck`
- Configure it as *Local only*, as we are not using emails to receive reports.
- Leave the default name.
- Check logs without sending email reports: `sudo -u logcheck -o -t`
- Check logs without sending email reports for login events: `sudo -u logcheck -o -t | grep login`

There are much more things that this software can do that can be consulted here. However, this is enough for this lab: http://somebooks.es/recibir-informes-sobre-sucesos-de-ubuntu-server-18-04-lts-con-logcheck/

Expected results: This activity will be finished when you are able to check login events using logs through this package.

Evaluate the final security level with Lynis

Practical application: *You know how to compare hardening indexes once hardening has been applied*

Once the manual security enforcement has been done, use `lynis` on the hardened system to evaluate their security level and see if we managed to improve it if compared with the initial evaluation we performed.

Expected results: This activity will be finished when you are able to obtain the new security level of the hardened VM and compare it with the original one to see if it has been improved.

BADGES AND SELF-ASSESSMENT

NOTE: You have a version of this badge table in an editable format available in the *Virtual Campus*. You can use this file to easily create a document in the format you want and take extended notes of your activities to create a log of what you did. Remember that this material is key to keep track of your self-evaluation.

Badge Level	Unlocked when	Unlocked?
	You can answer the following question: *What security advantages do you think you have by restricting access to single user mode?*	
	You know the meaning of the "password expiration" parameter	
	You know the meaning of the "password minimum days" parameter	
	You know the meaning of the "password expiration warning" parameter	
	You know the meaning of the "inactive password lock" parameter	
	You can answer the following question: *Do you think that warning banners are a real security measure? If not, why do you think they are used?*	
	You know how to put SHA-512 as the password hashing algorithm for `ssh`	
	You know how to limit password reuse	
	You can answer the following question: *What is the purpose of* `umask` *and why 027 is a good default secure value?*	
	You can answer the following question: *Why is having multiple* `root` *accounts a problem? What do you think it happened if you find multiple roots in your system, but you did not create them?*	
	Know where the SSH configuration file is and understand the usefulness of their different security parameters. You can answer the following question: *What security advantages do you think you have by changing the default SSH configuration?*	
	You can answer the following question: *If we change the default* `ssh` *port from 22 to another one, what security advantage you think we might have?*	
	Understand why being `root` is a strong security problem and why all the measures preventing unauthorized `root` logins exist.	
	Understand why system accounts should not be used for interactive logins. You can answer the following question: *if we suppose that by default no system account is configured for interactive logins, what could be the cause of suddenly finding one configured that way?*	
	You can answer the following questions: *Why preventing common passwords is recommendable as a complement of enforcing a complex password policy? What kind of attacks does this prevent? (remember past labs! 😊)*	
	You can answer the following question: *Why disabling core dumps is advisable from a security point of view?*	

	You can check the status of *AppArmor* and know how to enable profiles	
	You can answer the following question: *Why detecting an excessive resource consumption may indicate a security problem?*	
	You can answer the following question: Which resource monitor do you find more useful in different contexts? Which one would you use?	
	You can answer the following questions: *What specific legitimate operation do you think it may modify a system file and trigger a Tripwire security alert?* As these operations are legal, *what should we do if we have a lot of these alerts?*	
	You can improve the security of the network stack and optimize the supported protocols of a host to improve its security	
	You can enable and configure the *Linux* audit daemon (`auditd`)	
	You master password quality requirements: You understand the different parameters that controls the quality of a password in the PAM module.	
	You can answer the following question: *Why do you think not having a compiler in a local system is good from a security perspective?* (**TIP**: Malware are programs too, that have source code… 😜)	
	Understand the importance of checking logs and having tools able to detect unusual behaviors through them.	
	You can answer the following questions: *Is the security level of the hardened VM higher than the initial system? Do you think that* `lynis` *is aligned with the CIS (it checks the same security controls)?*	
	Install and use `tripwire` in a VM. Understand the usefulness of these tools. You can answer the following question: *What specific kind of malware do you think you can detect more easily with them?*	
	**This is OSParta!**: You can manually enforce and monitor key security configurations in different parts of an *Ubuntu Linux* OS following verified security practices and using typical software to do it	