Departamento de Informática
Universidad de Oviedo

Source: Bing Chat AI

Laboratory 5. Linux OS Security (Non-Automatable)

DEPARTMENT OF COMPUTER SCIENCE. UNIVERSITY OF OVIEDO

Computer Security | 2023 – 2024 (V4.0 *"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 or linked clone 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 because it is a huge document 😊.

🔍 ***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" (therefore, in Unicode code), so you need to replace them by ASCII-equivalent ones. This is especially problematic with quotes. If the syntax of a command does not work, carefully check your command line for invalid characters copied accidentally. This is applicable from now on, even in future laboratories.***

About *Wazuh*

At the end of the lab, you have the "star" activity of the lab, the deployment of an **XDR/SIEM called *Wazuh***. This is a very powerful free tool **for professional use** with a large number of functions. It is so complete that **we will return to it in labs 6, 8-9 and 11,** and in this one we will only see a part of its functions. Using it comes at a price, however, and that is that you need to increase the resources of the VM that we give you **to 6Gb of RAM and 4 cores for it to work adequately**. To do this, we advise you to do the following:

- If you do not have a machine at home capable of supporting the creation of a VM with these characteristics, we recommend that you **prioritize *Wazuh*'s activity to do it in the laboratory**, where you do have hardware that supports that VM. **You can even start with it without any problems**.

🔍 ***If you also do the exercise L5B4_AUDIT before cloning the machine to create Wazuh's, this XDR will probably be able to capture more information from your system, but it's not strictly necessary for this lab.***

- For this activity **, it is recommended to make a linked clone of the base VM** (much lighter to create and use) and increase the resources only to that clone.
- In this way, you will keep two VMs, but you will only have to start one at a time: **The "normal" VM** (with its snapshots or copies that you want to make and with which you will work in the different labs) and its linked clone**, the VM *"Wazuh"* with** *Wazuh* installed, which does not need any other one to work. **Please note that if you delete the "normal" VM and have made a linked clone, you will lose the "*Wazuh*" VM as well**. If you expect to delete it, it is better to create a complete clone of the VM, even if it takes up more resources.
- **Do not delete the MV *Wazuh* when you create** because we are going to use it in others, please.

- In this lab, we will get it up and running and we will only see part of its features in its interface. We will go back to the same interface later to see what other things mean. But **it is important that you do this *Wazuh* activity first** to get a good understanding of the rest.

Block 1: User-related security

Exercise L5B1_LYNIS1: Current security level evaluation

Description of the activity / Practical application

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

Expected results

This activity is finished when you can evaluate the security of your VM with `lynis`.

Additional information required to do it

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

🔍 ***Lynis is not part of the CIS benchmarks, which have their own audit tool (CIS CAT). It does not even use exactly the same security controls as the CIS, but it is a valid and very quick to use audit tool, which we can use to monitor how much we have improved. It is also a proof that there is not a single source of security controls, and that different sources prioritize different things***

Exercise L5B1_SINGLEUSER: Accounts and Access: Prevent single-user mode

Description of the activity / Practical application

You need to prevent anybody booting on your machine and acquiring `root` privileges by entering single-user (aka maintenance) 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.

Additional information required to do it

One of the most important things you should do regarding account management is to **prevent access to single user mode**.

🔍 ***This mode is used for disaster recovery, when the system detects a problem during boot, or it also lets you enter it by manually selecting it in the bootloader... But when you log in, you're virtually the*** `root` 😊

To prevent single user mode boot, follow *CIS benchmark* task ***1.4.3 Ensure authentication required for single user mode***.

🔍 ***NOTE: The recommended command in the CIS Benchmark,*** `grep -Eq '^root:\$[0-9]' /etc/shadow || echo "root is locked"` ***is not correct with the latest revision of Ubuntu 22.04. They have changed the KDF algorithm to a different one called "yescrypt". This algorithm is no longer referenced with a number in the*** `/etc/shadow` ***file, but with the character "y". Change the regular expression accordingly.***

🛠 Exercise L5B1_BANNER: Warning banners

🥷 Description of the activity / Practical application

You need to tell potential intruders to get off your lawn (like Clint Eastwood in "Gran Torino" 😊)

🎖 Expected results

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

📖 Additional information required to do it

It is a good security practice to warn remote users about the consequences of connecting to a system, in case they are not legitimate users, or about usage restrictions (if they are legitimate users). To put correct warning banners into the different access points of a system, follow *CIS benchmark* task ***1.7 Command Line Warning Banners***. Concretely, implement only these tasks:

- ***1.7.1 Ensure message of the day is configured properly.***
- ***1.7.2 Ensure local login warning banner is configured properly.***
- ***1.7.3 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/

🛠 Exercise L5B1_SECSSH: Additional security for *OpenSSH*

🥷 Description of the activity / Practical application

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

🎖 Expected results

This activity will be finished when you **strengthen the configuration of the ssh service** according to the CIS benchmark specifications.

Additional information required to do it

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 to fix any disaster 😊.

```
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 inside your current SSH configuration file, add a line with it and the recommended value. The set of tasks to implement is the ***5.2 Configure SSH Server*** benchmark block.

Exercise L5B1_PAM: Secure PAM module configuration

Description of the activity / Practical application

You need to **strengthen your password policies** as much as possible

Expected results

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

Additional information required to do it

PAM (*Pluggable Authentication Modules*) is a service that implements authentication system 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 ***5.4 Configure PAM*** and its subtasks.

- ***5.4.1 Ensure password creation requirements are configured.***
- ***5.4.2 Ensure lockout for failed password attempts is configured.***
- ***5.4.3 Ensure password reuse is limited.***
- ***5.4.4 Ensure password hashing algorithm is SHA-512.***
- ***5.4.5 Ensure all current passwords uses the configured hashing algorithm.***

Exercise L5B1_PWDAGE: Password age and inactive accounts

Description of the activity / Practical application

You need to remove **unused accounts and old passwords**

Expected results

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

Additional information required to do it

Password age (the number of days a password has been left unchanged) and inactive accounts may be a vector of security problems.

> 🔍 ***Not only because you have more user accounts to manage, but also because nobody is using them (and hence nobody will detect if someone else is using their account) and/or it is easier to leak a valid password if it is never changed.***

We can control this vector by applying the *CIS benchmark* task ***5.5.1 Set Shadow Password Suite Parameters*** and its subtasks:

- ***5.5.1.1 Ensure password expiration is 365 days or less.***
- ***5.5.1.2 Ensure minimum days between password changes is configured.***
- ***5.5.1.3 Ensure password expiration warning days is 7 or more.***
- ***5.5.1.4 Ensure inactive password lock is 30 days or less.***
- ***5.5.1.5 Ensure all users last password change date is in the past***

Exercise L5B1_LOGONROOT: Restrict root and system accounts logins

Description of the activity / Practical application

You need to make sure neither `root` nor service accounts can do an interactive login

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.

Additional information required to do it

There are a series of measures required to harden `root` and service account logins, or to check that these are correctly set up. We can do that following the *CIS benchmark* task ***5.5.2 Ensure system accounts are secured*** and the task ***6.2.10 Ensure root is the only UID 0 account*****.**

Exercise L5B1_COMMONPWD: Prevent using common passwords

Description of the activity / Practical application

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

Expected results

This activity will be finished when **you prevent common password usage** when changing or creating passwords in the system. You can also answer these questions:

- *Do you think there might be a strong password that it is common?*
- *Do you understand then why this security measure is a complement to the one that requires strong passwords?*

Additional information required to do it

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 cannot be easily done on our system. This activity prevents that our users may create passwords belonging 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.

🔍 ***Yes, this activity is not part of the CIS benchmarks, but it has been added because, in our personal experience, it is a very common problem***

Block 2: Process-related security

Exercise L5B2_COMPILERS: Compilers

Description of the activity / Practical application

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

Expected results

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

Additional information required to do it

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++
```

Exercise L5B2_COREDUMPS: Disable core dumps

Description of the activity / Practical application

You need to **remove core dump files** that can be used to obtain 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. You can answer this question: *Do you know what type of things these files may contain and why they are dangerous?*

Additional information required to do it

Follow the *CIS benchmark* task ***1.5.4 Ensure core dumps are restricted*** to disable this potential source of confidential information

Exercise L5B2_APPARMOR: *AppArmor management*

Description of the activity / Practical application

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

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*.

Additional information required to do it

AppArmor is a **mandatory access control system** (MAC) that limits programs to a set of resources they can use or access.

> 🔍 ***This means that it restricts programs to a set of files, attributes, and capabilities, so they are not able to do serious harm if they are modified to make unwanted operations or are malicious***

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 ***1.6.1 Configure AppArmor***, but more precisely these two.

- ***1.6.1.1 Ensure AppArmor is installed***
- ***1.6.1.3 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 there are disabled profiles 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 like this:

- 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
```

Exercise L5B2_SECAPT: Install apt security software

Description of the activity / Practical application

You need to be sure **not to install corrupted packages**

Expected results

This activity will be finished when the mentioned software is installed, and you are able to **check the integrity** of some packages

Additional information required to do it

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.

🔍 ***This activity is not in the CIS benchmarks, but it is in the Lynis security controls***

Exercise L5B2_PACCT: Process accounting

Description of the activity / Practical application

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

Expected results

This activity will be finished when **this accounting software is installed** and running.

Additional information required to do it

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`

🔍 ***This activity is not in the CIS benchmarks, but it is in the Lynis security controls***

HACK FIRST
HACK HARD
NO MERCY
大和
Cobra Kai

Block 3. Host networking security

Exercise L5B3_NETSTACK: Network stack configuration of an individual server

Description of the activity / Practical application

You need to configure **the network stack of your machine** as secure as possible

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.**

Additional information required to do it

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 Disable unused network protocols and devices***
  - *3.1.1 Ensure system is checked to determine if IPv6 is enabled*
  - *3.1.2 Ensure wireless interfaces are disabled*
- ***3.2 Network Parameters (Host Only)***
  - *3.2.1 Ensure packet redirect sending is disabled*
  - *3.2.2 Ensure IP forwarding is disabled*
- ***3.3 Network Parameters (Host and Router)***
  - *3.3.1 Ensure source routed packets are not accepted*
  - *3.3.2 Ensure ICMP redirects are not accepted*
  - *3.3.3 Ensure secure ICMP redirects are not accepted*
  - *3.3.4 Ensure suspicious packets are logged*
  - *3.3.5 Ensure broadcast ICMP requests are ignored*
  - *3.3.6 Ensure bogus ICMP responses are ignored*

- *3.3.7 Ensure Reverse Path Filtering is enabled*
- *3.3.8 Ensure TCP SYN Cookies is enabled*
- *3.3.9 Ensure IPv6 router advertisements are not accepted*

Exercise L5B3_RAREPROT: Disable uncommon network protocols

Description of the activity / Practical application

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

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.

Additional information required to do it

This exercise 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*

Block 4: Logging and monitoring

Exercise L5B4_AUDIT: Configure audit rules

Description of the activity / Practical application

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

Expected results

This activity will end when you ensure that the security controls of the CIS *benchmark* that configure a **robust audit policy** are in place, and the audit service works once restarted after making all the changes.

Additional information required to do it

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 of the benchmark 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*

🛠 Exercise L5B4_SYSSTAT: Install *sysstat* to see resource consumption

Description of the activity / Practical application

You need to **monitor resource usage** in your Linux OS

Expected results

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

Additional information required to do it

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.

🔍 ***Resource usage control tools are not part of the CIS benchmark, but they have been added in order to get an idea of what is happening if in the course of the labs the MV becomes very slow for no apparent reason***

🛠 Exercise L5B4_GLANCES: Install *glances*

Description of the activity / Practical application

You need to **monitor CPU usage** in your Linux OS

Expected results

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

Additional information required to do it

Glances is another popular package that allows to monitor all resource usage of a system in a single screen, so resource usage 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 the command line.

🔍 ***Resource usage control tools are not part of the CIS benchmark, but they have been added in order to get an idea of what is happening if in the course of the labs the MV becomes very slow for no apparent reason***

Exercise L5B4_LYNIS2: Evaluate the final security level with *Lynis*

Description of the activity / Practical application

You know how to **compare hardening indexes** once hardening has been applied over your virtual machine

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.

Additional information required to do it

Once all the manual security hardening 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.

Exercise L5B4_WAZUH: HIDS implemented with a professional XDR: *Wazuh* as a monitoring tool for files, processes, and users

Description of the activity / Practical application

It consists of **learning how to install a professional *XDR software*** that can monitor your VM so that you have a complete view of what is happening in it, and thus understand part of the functions that an XDR can perform on a machine, and its importance in a real deployment.

Expected results

You understand how the *XDR Wazuh*, among its many features, **can function as a HIDS** and help you keep an eye on anomalies occurring in your MV. You can answer these questions:

- *Now that you know how to deploy an XDR/SIEM and its agents, how do you think companies that rent one from a security company (that monitors their machines for anomalies) work?*
- *What do you think it means that Wazuh reports a successful sudo execution?*
- *And one that hasn't been successful?*
- *What do you think it means when Wazuh detects a software installation that you did not start? Can you think of any legitimate possibility of that happening? And illegitimate? (Okay, this last one is very easy 😊)*

- *If you put Wazuh to monitor important OS files in certain directories and it detects a change in the hash of any of them, does that always mean something bad has happened?*
- *For example, what legitimate and non-legitimate reasons can be for a change in* the `/etc/shadow` or the `/etc/passwd` file?
- *Although in this exercise we see how Wazuh detects suspicious behavior, we are not responding to it in any way. If I tell you that in a SOC they have a response procedure prepared for any of these anomalies, so that when one is detected and confirmed, that procedure is set in motion to respond to it (manual, automatic or mixed), do you now understand more or less how a SOC works in general?*

Additional information required to do it

A *Host-Based Intrusion Detection System* (HIDS) is an **intrusion detection system** capable of monitoring and analyzing **the internal components** of a computer system. This type of software can monitor all or part of the dynamic behavior and status of a computer system, depending on how it is configured.

🔍 ***For example, a HIDS might detect which program accesses which resources and discover that, for example, a word processor has suddenly and inexplicably begun modifying the system's password database, which is indicative of an ongoing infection.***

In addition, a HIDS could examine the state of a system, its stored information, whether in RAM, in the file system, log files, etc., and verify that the content of these is what is expected, that is, that it has not been changed by intruders or unexpectedly. To do this, it uses the creation of **cryptographic signatures** (such as those in **Topic 2** of theory) of the files that we send it to monitor, so that any changes in them are detected.

Compared to network-based intrusion detection systems, HIDS has the advantage of **being able to identify insider attacks**. NIDS examines network traffic, while HIDS examines data originating from operating systems.

It is important to mention that HIDS usually go to great lengths to prevent their object / checksums databases and their reports from being tampered with in any way. If intruders were able to modify any of the objects that the HIDS monitor, they could also modify the data that a HIDS uses.

In *the CIS* Benchmarks there is a section to install a HIDS, because its presence considerably raises the security level of a system. In this case, the CIS has opted for a HIDS that **does not rely on any external component** to function and that works within the same host, called ***AIDE.*** Specifically, its installation is covered in these sections:

- ***1.3 Filesystem Integrity Checking***
  - ***1.3.1 Ensure AIDE is installed (Automated)***
  - ***1.3.2 Ensure filesystem integrity is regularly checked (Automated)***

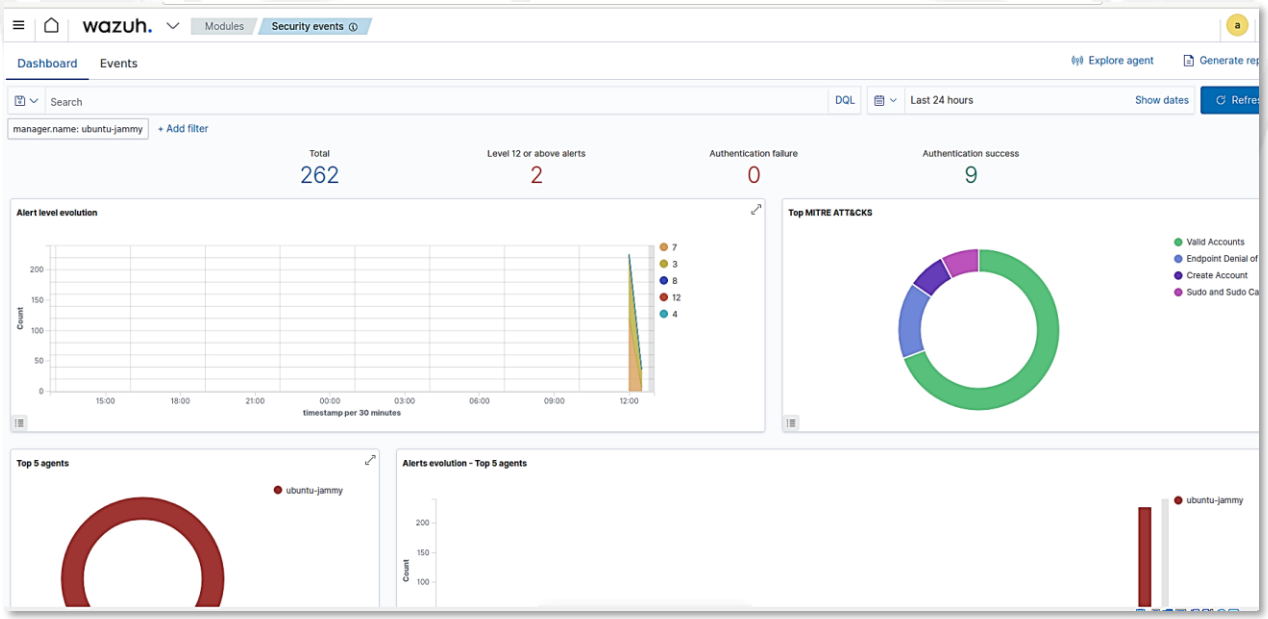
A very similar HIDS to this one, which is sometimes used as a substitute, is ***Tripwire*** (https://linuxconfig.org/intrusion-detection-systems-using-tripwire-on-linux). However, these HIDS, precisely because they do not depend on anything external, have a very limited series of functionalities (basically detecting changes in important files). *Wazuh* far surpasses them in functions, as we will see, as it is an XDR that includes HIDS functions. In the following sections, we will see how to implement it in our infrastructure, which would more than cover the full **1.3** point of the CIS benchmarks.

Wazuh is not part of the CIS benchmarks, but since it is a tool that covers the same needs and many more that we will use in subsequent labs, so we are going to use it instead.

Installing *Wazuh* single-node on a VM using *Docker*

To get *Wazuh up* and running easily, we recommend following these steps:

- **Make the linked clone** of the VM of the course as we indicated at the beginning of the lab, with **6Gb of RAM and 4 cores** (power has a price!). Turn off your regular MV and boot it up to work.
- There are several ways to install *Wazuh*, but the best in our case is ***Wazuh-Docker***, which installs it using the same system that we use in the course to deploy the laboratories. Specific instructions can be found here: https://documentation.wazuh.com/current/deployment-options/docker/wazuh-container.html, and you need to make sure of the following:
  - The type of installation we are going to do is ***Single-node deployment***. It is just as functional as the other one, but with everything in one machine (we do not have the resources to do a distributed installation).
  - In the step of generating the certificates, it gives us two options: **generate them or provide them ourselves**. In this case, it is better **to generate them** with the *software* provided (`generate-indexer-certs.yml`). Make sure you run it once before launching *Wazuh* as the website indicates. It is not necessary to change any of the files they provide.
- Launch the XDR now as indicated in the tutorial, going to the directory where you have unzipped the contents of the *git* and typing `docker-compose up -d`. Please wait about 30 seconds for the entire system to go live. When everything is ready, you should be able to access the XDR by opening the VM browser and going to https://127.0.0.1. The default user account is "`admin`" and the password is "`SecretPassword`". If all goes well, we should see a screen like this:



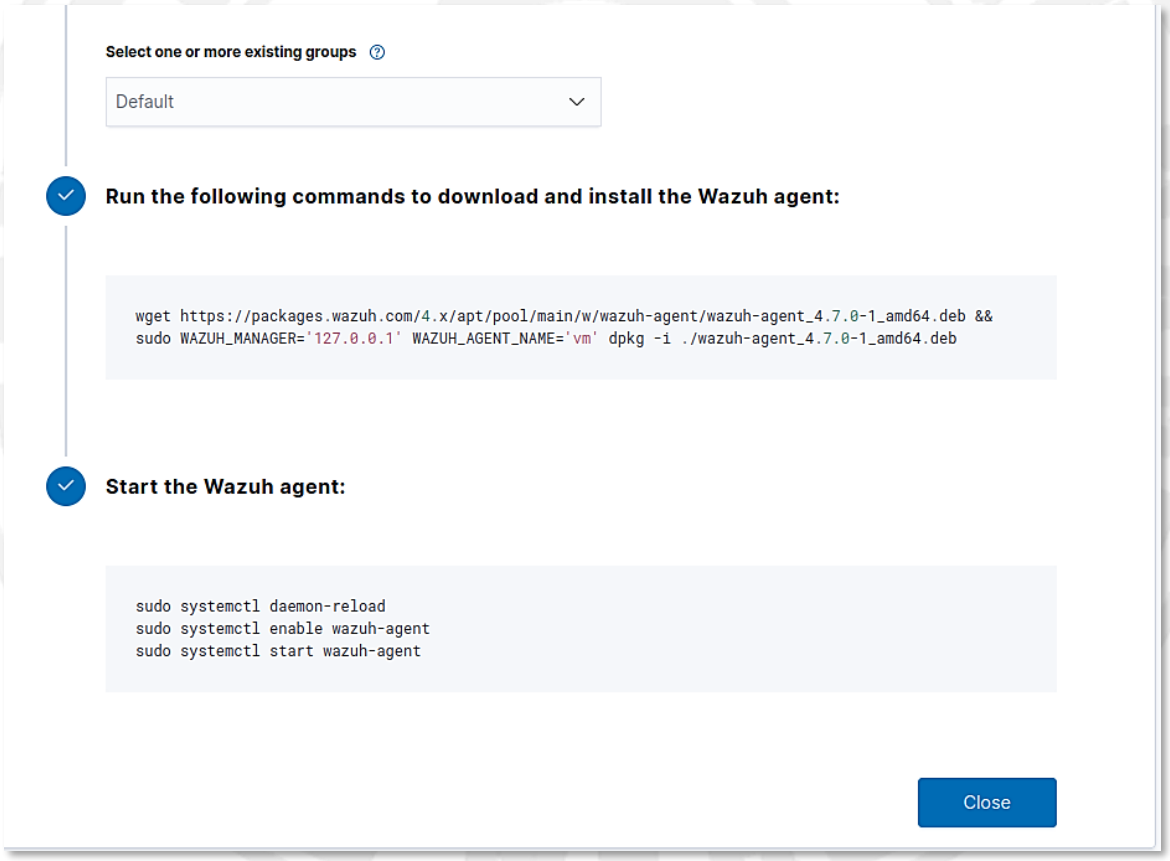
If you plan to deploy this in a real environment, please do not forget to change the admin password!

Wazuh working as a HIDS

Wazuh is a surveillance system right now, but right now **it has no machines to keep an eye on**. In fact, he warns us that **it has no agents**. An agent is a piece of *software* that we deploy on each monitored machine, and that is responsible for:

- **Collecting information** from the system in which it is installed
- Be configured to collect **different types of information** (things like the different *logs* of the actions that occur, file signatures... everything you need)
- **Send it to the XDR** using strong ciphered communications. In fact, If you mess around a little with the GUI you can see the algorithm it uses to send the information. *Do you know what algorithm it uses? Is it symmetric or asymmetric encryption?* (**Labs 3-4**)

The XDR then interprets the information of all the systems it monitors (have its agents deployed) it to detect anomalies. *How is an agent deployed?* No problem, because ***Wazuh* generates them for us**. We click on the message that complains that there are no agents and configure a new one to monitor our own MV, selecting the creation of a 64-bit `.deb` agent for *Ubuntu* with these features:

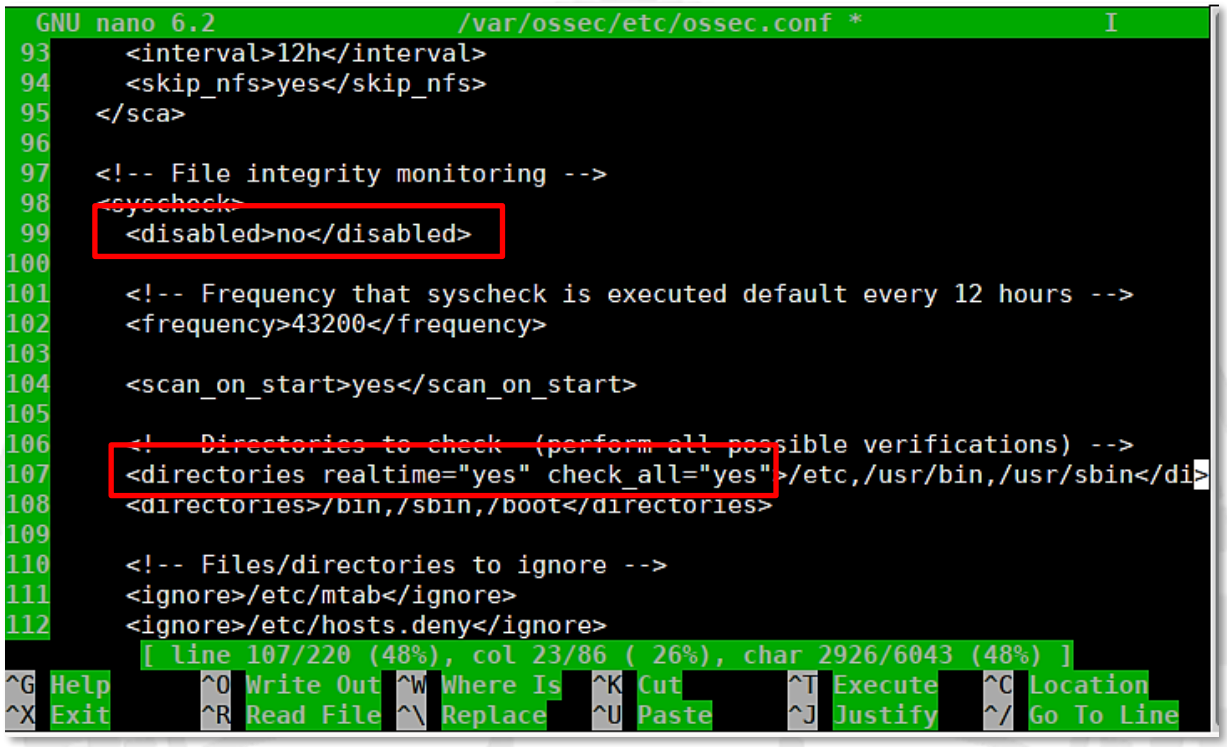


Now, from a terminal in our VM **we copy the commands that *Wazuh*** itself gives us to install and start the agent. With this, *Wazuh* should show that it has **1 active agent** and will start collecting information from the machine running it. Needless to say, monitoring more machines is the same, and we would only have to indicate the IP where the VM in which you run *Wazuh is* when generating the agent, obviously assuming that this IP is reachable from the machine to be monitored.

🔍 ***Do you understand now how tremendously easy it is to monitor an entire machine infrastructure with a single Wazuh? The main problem is that the more monitored machines the more resources you need, but for a small installation it is a very good solution. By the way, in case you are wondering, yes, Wazuh has agents***

for Windows and MAC as well. Also, you may consider than Wazuh requirements are overkill, but if you can monitor multiple machines with a single central installation, this problem is less critical

Wazuh, in order to save resources, **does not come with all its functionalities active**. One of the inactive ones is interesting: monitoring of changes in important files (***File Integrity Monitoring*** or FIM) and their subsequent notification if any are detected. For this, *Wazuh* uses a software that is integrated into its agents called **OSSEC.** By default, OSSEC comes with this functionality disabled, and you must activate it by changing its configuration file (`/var/ossec/etc/ossec.conf`) in the VM like this:



With this configuration we are asking OSSEC to monitor all changes in real time to the `/etc`, `/usr/bin`, and `/usr/sbin` directories, which contains critical OS executable programs. More information:

- https://www.ossec.net/docs/manual/syscheck/index.html
- https://documentation.wazuh.com/current/user-manual/capabilities/file-integrity/how-to-configure-fim.html

🔍 ***We can monitor more directories, and even individual files, by adding more entries to the file. It all depends on what we are interested in and what is in the system, of course. It is just a matter of testing and experimenting!*** 😊

Once this is done, we only have **to restart the agent in the VM** (`sudo systemctl wazuh-agent restart`) and we will be able to detect changes in the files in those directories. Now we have *Wazuh* ready to work as a HIDS, so to finish the exercise we will do the following tests in the VM, then checking in the *Wazuh* GUI (click on *Refresh*) if they are detected:

- `sudo` a command, but **enter the wrong password** so that it fails
- Do a `sudo`, but this time with the **right password**
- Install something with `apt` or update the OS

- Edit the `/etc/passwd` file and change the `root` user shell from `nologin` to `/bin/bash`. *What do you think this does? What could it be used for if, instead of* `root`*, it was another more "innocent" user? Does Wazuh detect it?*

🔍 ***Remember: Keep this VM for future labs!***

✍️ 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 can be brought to lab exams.

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: *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?*	
	You 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?*	
	You understand why being `root` is a strong security problem and why all the measures preventing unauthorized `root` logins exist.	
	You understand why service accounts should not be used for interactive logins. You can answer the following question: *if we suppose that by default no service 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? Which one would you use to monitor CPU usage?*	
	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… 😉)	
	You understand the importance of checking logs and having tools able to detect unusual behaviors through their contents.	
	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)?*	
	You can install and use `wazuh` in a VM and understand the usefulness of XDR to monitor user activities, processes, and machine file usage. 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 various parts of an *Ubuntu Linux* OS following verified security practices and using typical software to do it	