HARDERING

Lynis:

Lynis (https://cisofy.com/lynis) és una eina d'auditoria de seguretat que serveix per informar, després d'haver repassat l'estat del sistema, sobre possibles millores a l'hora de enfortir la seva seguretat (des de múltiples punts de vista). Per instal.lar-lo es pot fer simplement *apt install lynis* o bé, si es vol obtenir la versió més moderna, anar a la seva pàgina de descàrregues (https://cisofy.com/downloads/lynis) i escollir o bé obtenir el paquet tar.gz (que inclou el binari ja compilat) o bé el paquet deb/rpm. En qualsevol cas, per executar el programa només cal (com a root) escriure: *sudo lynis audit system*

NOTA: Per veure més possibilitats d'aquest programa es pot fer *lynis show help*

Això generarà un seguit de tests que aniran comprovant diferents aspectes relacionats amb la seguretat del sistema i es mostrarà si aquests test es passen correctament o cal arreglar alguna cosa. El més interessant és que la manera per arreglar allò que s'ha identificat com a potencialment perillós o poc segur es troba detallada al final de l'informe. Aquest informe (amb els resultats dels tests, els avisos, les suggerències, etc) es guarda a més a l'arxiu "**/var/log/lynis-report.dat**" per poder-lo consultar sempre que es vulgui (també és interessant l'arxiu "**/var/log/lynis.log**", el qual conté detalls més tècnics sobre els tests).

És molt interessant observar com a cada "suggestion" que ofereix Lynis per tal de millorar un aspecte concret de la seguretat del sistema apareix un codi entre claudàtors (com per exemple [KRNL-5830]). Aquest codi ens permet obtenir una informació més concreta sobre com ha arribat Lynis a detectar les possibles millores sobre aquesta suggerència i quins serien els passos específics que s'haurien de realitzar per tal de sol.lucionar-la si fem servir la comanda *lynis show details KRNL-5830*

D'altra banda, la bateria de tests que fa Lynis ve donada per un "perfil". Al paquet de l'Ubuntu només ve el perfil "/etc/lynis/profile.prf" però a partir d'ell es poden crear variacions (és un simple fitxer de text amb diferents directives, com ara "skip-test", per exemple). En qualsevol cas, per utilitzar un perfil diferent s'ha d'indicar amb el paràmetre *--profile nomPerfil.*

EXERCICIS:

1.-a) Executa la comanda *sudo lynis audit system | less -R* a una màquina virtual qualsevol i observa els "warnings" i "suggestions" que et mostra. Fes tot el possible per augmentar el "Hardering Index" per sobre de 70/100. Executa els tests tantes vegades com sigui necessari per arribar finalment a aquesta fita.

b) Suposant que tenim dos resultats de tests fets en moments diferents anomenats "/var/log/lynis-report.dat" i "/var/log/lynis-report-previous.dat" , ¿per a què serviria executar el següent script ? Pista: fixa't en els paràmetres *-I* i *-y* de la comanda *diff*

```
#!/bin/bash
DIFFERENCES=$(diff -I report_datetime /var/log/lynis-report.dat /var/log/lynis-report-previous.dat)
if [[ $? -gt 0 ]]
then
    diff -y /var/log/lynis-report-previous.dat /var/log/lynis-report.dat | grep -v "report_datetime"
fi
```

c) Què explica aquest article https://linux-audit.com/how-to-create-custom-tests-in-lynis ?

OpenSCAP i altres guies:

OpenSCAP (https://www.open-scap.org) és un programa similar en objectius a Lynis però molt més formalitzat i rígid de forma que es puguin realitzar tests estandaritzats a nivell legal i gubernamental independementment de la distribució Linux utilitzada. Els tests (que aquí s'anomenen "Security Policies" o "SP") es basen en arxius de tipus XML (concretament, escrits en el format XCCDF, de "Extensible Configuration Checklist Description Format"), un conjunt bàsic dels quals ve recollit, a Fedora, dins d'un paquet anomenat "scap-security-guide" i, a Ubuntu, en els paquets "ssg-base", "ssg-debderived", "ssg-debian", "ssg-nondebian" i "ssg-applications".

> NOTA: En instal.lar-se, els fitxers XCCDF se solen ubicar dins de la carpeta "/usr/share/xml/scap/ssg/content" i sol venir un fitxer XCCDF diferent (que inclou una bateria de tests) per cada tipus/distribució de sistema possible a auditar (hi ha un per Fedora, hi ha un per Ubuntu, hi ha un per Debian, etc), així com també per diferents programes específics (Chrome, Jboss, etc)

> NOTA: El repositori oficial dels fitxers XCCDF pertanyents al projecte "Scap Security Guide" (els que es descarreguen a través dels paquets indicats als paràgrafs anteriors) és https://github.com/ComplianceAsCode/content

> NOTA: SCAP stands for *Security Content Automation Protocol*. It is an open standard which defines methods for security policy compliance. It comes originally from the US National Institute of Standards and Technology (NIST) to provide a way for US government agencies to audit its systems for regulatory compliance

Aquests tests XCCDF són llegits i implementats per un programa anomenat "oscap-scanner" (tant a Fedora com a Ubuntu) per tal d'auditar així el sistema en qüestió d'una forma automatitzada. Cada test es pot dividir al seu torn en diferents "profiles", també estandaritzats (PCI-DSS, STIG, USGCB, etc), depenent del tipus de política de seguretat que es vulgui implementar a la màquina concreta: cada "profile" sel.lecciona diferents regles dins del test en qüestió i usa diferents valors per aquestes. La idea és que no és el mateix auditar un sistema militar (on, per exemple, els ports USB es considera que haurien d'estar desactivats) que un ordinador d'una oficial. El "profile" civil més important és PCI-DSS i de fet, aquest és l'estàndar "de facto" en els entorns Linux empresarials.

L'utilització del "oscap-scanner" per línia de comandes és una mica farragosa. Afortunadament, existeix una interfície GUI d'aquest que facilita el seu ús anomenada "scap-workbench" (disponible tant a Fedora com a Debian/Ubuntu). Als exercicis farem servir aquesta eina.

> NOTA: És possible escriure tests propis però per això és recomanable instal.lar el paquet "openscap-engine-sce", el qual proporciona l'anomenat "Script Check Engine", una extensió del protocol SCAP que permet escriure test mitjançant Bash, Python o Ruby. A https://www.open-scap.org/security-policies/customization/ podeu trobar més informació

> NOTA: D'altra banda, cal dir que a part dels fitxers XCCDF, els programes SCAP fan ús de fitxers OVAL ("Open Vulnerability and Assessment Language"), els quals serveixen per detectar vulnerabilitats i pegats. Aquests fitxers, a diferència dels XCCDF, cal descarregar-los i distribuir-los en tots els sistemes a auditar de forma freqüent i periòdica (diàriament, per exemple) per tal d'estar assegurats contra les darreres vulnerabilitats detectades. Llocs d'on es poden obtenir són https://www.debian.org/security/oval (Debian), Ubuntu https://people.canonical.com/~ubuntu-security/oval (Ubuntu) o https://github.com/CISecurity/OVALRepo (en general), entre d'altres llistats aquí: http://oval.mitre.org/repository/index.html

EXERCICIS:

1.-a) Instal.la l'"Scap-workbench" a una màquina virtual amb entorn gràfic i, en executar-lo, escull el fitxer XCCDF corresponent al sistema operatiu on l'estiguis fent servir. Seguidament escull, d'entre els "profiles" disponibles a testejar en aquest fitxer XCCDF, l'anomenat "PCI-DSS" i procedeix a realitzar el test. Observa els resultats obtinguts. ¿Quins aspectes et marca com a millorables? ¿Coincideix amb el que et deia Lynis?

aII) Un cop obtinguts els resultats de l'apartat anterior, ¿per a què serveix el botó "Show report" del programa "Scap-workbench"? ¿I el botó "Save results" ->"HTML report"? ¿I el botó "Generate remediation role"->"Ansible"?

b) Vés a https://www.open-scap.org/tools/ i digues, a més de l'eina "OpenSCAP Base" i "SCAP Workbench", per a què serveixen les eines "OpenSCAP Daemon" i "SCAPTimony".

2.-Visita https://dev-sec.io/baselines i digues què ofereix aquest projecte (clica sobre una categoria i fixa't llavors en la llista d'ítems que apareix i també en el botó "Ansible remedation role"). Pots saber més llegint https://dev-sec.io/project/

A Internet existeixen moltes "Hardering Guides" que aconsellen sobre la configuració més adient (a nivell de sistema operatiu i/o d'aplicació servidor concreta) per assegurar la màquina en contra de possibles intents d'intrussió. En general, aquestes guies tracten, entre molts d'altres, temes com:

*Com usar i configurar mòduls PAM (per exemple, "libpam-cracklib" per augmentar la complexitat de les contresenyes, "libpam-google-authenticator per activar el 2FA, etc)

*Com configurar l'accés via SSH per claus i només per determinats usuaris "base" (no per root). De fet, deshabilitar la contrasenya de root sol ser una bona pràctica

*Com confirmar que *sudo* sol.liciti contrasenya i només funcioni per aquests determinats usuaris base i/o per determinats programes

*Com configurar el tallafocs i algun sistema HIDS adientment

*Com instal.lar un servei de detecció d'intrussions (com ara "fail2ban", "psad", "monit", etc)

*Com realitzar còpies de seguretat

*Com desinstal.lar tot aquell software i comptes d'usuari superflus

*Com actualitzar a les darreres versions del software necessari (o bé per evitar-ne les actualitzacions)

*Com establir configuracions particulars pels dimonis utilitzats (IPs i ports on escolten, etc), a més de comprovar la propietat i permisos dels fitxers de configuració, claus i carpetes que aquests gestionen, comprovar que l'usuari que els posa en marxa no tingui shell definida, establir que l'usuari usat per entrar a la base de dades (si n'hi hagués) tingui privilegis mínims sobre aquesta, comprovar el límit d'ús de recursos (CPU, RAM, etc), etc.

*Com encriptar carpetes i/o discos durs

*Com assegurar l'accés a la BIOS/UEFI, al gestor d'arranc i, en general, l'accés físic a la màquina.

Podem trobar un recull d'aquest tipus de llistes de "comprovació" al següents enllaços:

https://github.com/ernw/hardening
https://security.utexas.edu/admin
https://seinecle.github.io/linux-security-tutorials
https://github.com/trimstray/the-practical-linux-hardening-guide
https://www.owasp.org/index.php/Secure_Configuration_Guide
https://linux-audit.com/how-to-secure-linux-systems-auditing-hardening-and-security
https://bettercrypto.org
https://github.com/lfit/itpol/blob/master/linux-workstation-security.md (més avançat)

3.-a) Llegeix aquest article (https://ryaneschinger.com/blog/securing-a-server-with-ansible) i digues tres ítems que menciona per assegurar un sistema Linux, descrivint a més les línies del "play" d'Ansible corresponent.

b) Llegeix aquest article (https://www.elarraydejota.com/como-realizar-actualizaciones-unicamente-de-seguridad-en-distribuciones-debian-gnulinux-y-derivadas) i raona per a què voldríem fer el que allà s'explica

NOTA: A sistemes RedHat les comandes equivalents serien: *dnf updateinfo summary, dnf updateinfo list, dnf updateinfo info FEDORA-23t5 , dnf --security upgrade* o *dnf --advisory=FEDORA-23t5 update*

Rkhunter, ClamAV i Tyton:

Rootkits are software secretly installed by a malicious intruder to allow that user continued access to the server once security is breached. This is an extremely dangerous problem, because even after the entry vector that the user originally used to gain access is fixed, they can continue to enter the server using the rootkit they installed.

One tool that can help you protect your system from these kinds of problems is rkhunter (http://rkhunter.sourceforge.net) . This software checks your system against a database of known rootkits. Additionally, it can check other system files to make sure they are in line with expected properties and values. Un cop instal.lat dels repositoris oficials d'Ubuntu o Fedora, podem fer servir les següents opcions (com a root):

rkhunter --versioncheck : Comprova si la versió del programa està actualitzada. Per fer-ho necessita que l'opció WEB_CMD de /etc/rkhunter.conf tingui el valor del nom d'algun client HTTP instal.lat al sistema, com ara "curl" (sense cometes)

rkhunter --list : Mostra les capacitats del programa

rkhunter --update : Actualitza la base de dades de rootkits d'Internet. Cal que l'opció WEB_CMD estigui definida també

rkhunter –propupd : Fa una primera passada al sistema per tenir-la com a referència respecte els xequeig que es faran a posteriori per tal de comparar i veure si hi ha hagut algun canvi.

rkhunter --check : Realitza el xequeig pròpiament dit. Es pot afegir el paràmetre *--rwo* per indicar que només es volen mostrar els "warnings" i *–sk* per dir-li que faci tot el xequeig de cop sense esperar a pulsar ENTER cada cert temps (comportament per defecte). També es poden indicar els paràmetre *--enable* i/o *–disable* per realitzar explícitament (o bé deixar de realitzar) determinades comprovacions (s'admeten els valors *all* i *none*)

Els resultats dels tests els podrem trobar a "/var/log/rkhunter.log"

Es pot configurar el comportament d'aquest programa a dos arxius diferents i complementaris: a l'arxiu "/etc/rkhunter.conf" (aquí, per exemple, es pot activar l'enviament de correu si es troba alguna alterta mitjançant les directives MAIL-ON-WARNING i MAIL_CMD) i a l'arxiu "/etc/default/rkhunter" (aquí, per exemple, es on pot activar l'actualització periòdica automàtica a través d'Internet la base de dades dels rootkits)

NOTA: Una altra eina similar és Chkrootkit (http://www.chkrootkit.org)

EXERCICIS:

1.-a) Instal.la rkhunter, actualitza la base de dades, genera una primera passada amb *–propupd* i tot seguit realitza l'escaneig. ¿Quins són els resultats?

b) Què explica el següent article: http://web.archive.org/web/20080109214340/http://www.cert.org/tech_tips/intruder_detection_checklist.html ?

2.-a) Descarrega't l'antivirus ClamAV dels repositoris oficials d'Ubuntu o Fedora (paquets "clamav" i "clamav-update"). Seguidament, executa des d'un terminal la comanda *sudo freshclam* per actualitzar les signatures dels virus

NOTA: Aquesta comanda té un arxiu de configuració anomenat "/etc/freshclam.conf" que inclou directives interessants com ara *DatabaseDirectory, Checks* o *DatabaseMirror* però no caldrà modificar-ne cap; podeu trobar més informació a *man freshclam.conf*.

A partir d'aquí, podries executar manualment una comanda com *clamscan -r -i /ruta/carpeta* per realitzar l'escaneig de la carpeta indicada (on -r vol dir de forma recursiva i -i vol dir que només es vol veure a pantalla informació dels arxius infectats) però farem una altra cosa.

El paquet "clamd" (a Fedora) o "clamav-daemon" (a Ubuntu) proporciona un dimoni que estarà permanent encés però només farà escanejos (als fitxers/carpetes prèviament configurats) sota demanda. "Sota demanda" vol dir que no escaneja en temps real ni programa escanejos de forma programada: només escaneja quan és disparat mitjançant la comanda client *clamdscan*, bé localment bé remotament.

b) Instal.la el paquet "clamd" i edita les següents línies del seu fitxer de configuració, anomenat "/etc/clamd.d/scan.conf" per tal de què el servei atengui peticions (locals) d'escaneig pels canals i permisos adients (podeu trobar més informació a *man clamd.conf*):

*Comenta les línies *Example* i *User* ... (per a què quedin així: *#Example* i *#User* ... , respectivament)
*Descomenta totes les línies *#LocalSocketxxx* ... (per a què quedin així: *LocalSocketxxx* ...)

bII) Posa en marxa el servei, així: *sudo systemctl start clamd@scan* (caldrà haver executat al menys una vegada *freshclam* abans per a què tingui una base de dades on treballar)

NOTA: Si tens problemes amb SELinux, executa la comanda *sudo setsebool -P antivirus_can_scan_system 1*
NOTA: Directives útils per limitar l'abast de l'escaneig són *ExcludePath, MaxDirectoryRecursion o CrossFilesystems*

bIII) Per a què no tinguis problemes de permisos en executar *clamdscan* l'usuari en qüestió (l'anomenarem "pepito") ha de pertànyer al grup "virusgroup". Això ho pots aconseguir executant s*udo usermod -a -G virusgroup pepito* i reiniciant sessió. Un cop fet tot això, ja podràs disparar un escaneig (per exemple, a totes les carpetes personals que hi hagi al sistema) simplement executant la comanda *clamdscan /home*

c) ¿Per a què serveix la directiva *VirusEvent* de l'arxiu "scan.conf"?

Tyton (https://github.com/nbulischeck/tyton) és un detector de rootkits més avançat que funciona com a mòdul del kernel i és capaç de detectar diferents tipus d'"encobriments", com ara mòduls maliciosos ocults, enganxaments maliciosos a crides al sistema, o a protocols de xarxa, o al tallafocs Netfilter, o a interrupcions, etc (per més informació tècnica sobre com funciona, podeu llegir https://nbulischeck.github.io/tyton) i enviar-ne la notificació corresponent al Journald. A més proporciona un servei anomenat "tyton-notify" que, un cop encés (*sudo systemctl start tyton-notify*), rastreja aquests missatges generats pel mòdul i enviats al Journald per mostrar-los gràficament a l'entorn dels escriptoris més rellevants (Gnome, Kde, etc) gràcies a la llibreria "libnotify", integrada en aquests.

NOTA: Tyton suporta l'esquema DKMS ("Dynamic Kernel Module Support"), el qual permet actualitzar-se automàticament cada cop que hi ha una actualització del kernel. D'aquesta manera no cal recompilar el mòdul a mà quan passa aquesta actualització.

3.-a) Segueix els passos que indica a la seva web per instal.lar i carregar el mòdul Tyton i inicia el servei de notificació "tyton-notify".

b) Segueix els passos indicats a https://github.com/f0rb1dd3n/Reptile per tal d'instal.lar el rootkit "Reptile" a la màquina. Seguidament, prova les següents possibilitats explicades a https://github.com/f0rb1dd3n/Reptile/wiki/Usage i comprova si funcionen (pensa una manera concreta de fer la comprovació de cada possibilitat) i/o si Tyton és capaç de detectar-les:

*"Give root to unprivileged users"
*"Hide (Reptile) files, directories and kernel module"
*"Hide processes"
*"Hide TCP connections"
*"File content tampering"

NOTA: Un altre rootkit de codi lliure per Linux que pots provar és Diamorphine: https://github.com/m0nad/Diamorphine
Una llista molt completa d'altres rootkits lliures per Linux es pot trobar a https://github.com/milabs/awesome-linux-rootkits

Maldet (LMD):

Malware consists of viruses, trojans, worms and more. To find Malware a scanning program will look over specified folders and/or files. When the program scans, it is looking for signatures. A signature is made from a bit of unique code from malware (that is, a contiguous byte sequence that potentially can match some variant of a malware family). The code is then hashed and placed in a database. The scanning program gets the hash from the database and looks through files to see if the hash exists. If the signature is found then the scanning program can alert the user that a threat has been found.

Linux Malware Detect (LMD, https://github.com/rfxn/linux-malware-detect) is a malware scanner that uses threat data from network edge intrusion detection systems to extract malware that is actively being used in attacks and generates signatures for detection. It uses signatures such as HEX pattern and MD5 file hashes. They can also be extracted from a variety of detection tools including ClamAV. In addition, LMD threat data can also be extracted from user submissions with the checkout feature in LMD from malware resources.

The Maldet installation pack is not available from online repositories, but is distributed as a tarball from the project's web site. So, to install it, execute following commands:

wget http://www.rfxn.com/downloads/maldetect-current.tar.gz
tar -xvf maldetect-current.tar.gz
cd maldetect-1.6.4
sudo ./install sh

After the installation is completed, a daily execution via cron is scheduled by placing a script in "/etc/cron.daily" folder. This helper script will, among other things, clear old temporary data, check for new LMD releases and updates of the signature files, stores quarantine data for not more than 14 days, runs a daily scan of all recent file on the system and also scans the default Apache and web control panels (i.e., CPanel, DirectAdmin, to name a few) default data directories.

The configuration of LMD is handled through "**/usr/local/maldetect/conf.maldet**" file, where you will find the following sections, enclosed inside square brackets: GENERAL OPTIONS, QUARANTINE OPTIONS, SCAN OPTIONS , and MONITORING OPTIONS. Each of these sections contains several variables that indicate how LMD will behave and what features are available.

*Set *email_alert=1* if you want to receive email notifications of malware inspection results. Set *email_subj="Your subject here"* and *email_addr=username@localhost* to configure further.
*Set *quarantine_hits=1* if you want to move infected file to "/usr/local/maldetect/quarantine" folder and alert; if you only want to alert, set it to 0
*Set *quarantine_suspend_user=1* if you want to suspend the account of users with hits (that is whose owned files have been identified as infected).
*Set *scan_user_access=1* if you want regular users can do scans, too (not only root)
*Set *clamav_scan=1* if you want LMD to attempt to detect the presence of ClamAV binary and use it as default scanner engine. This yields an up to four times faster scan performance and superior hex analysis. This option only uses ClamAV as the scanner engine, and LMD signatures are still the basis for detecting threats.

You can run manually a scan by executing this command: *maldet -a /folder/path* (by default the folder's path is "/home") and see the results by executing *maldet -e SCANID* To see a list of all reports use the command *maldet -e list* If you want to quarantine the files that have been found in some report (due to there's the line *quarantine_hits=0* in configuration file), you could do *maldet -q SCANID* Sometimes you might have a false positive leading to a file being quarantined for the wrong reason: to restore such a file, run the following command: *maldet -s {/path/file | SCANID }* To ensure Maldet is up-to-date, run *maldet -u*

There's a service too: *sudo systemctl start maldet (*which needs "inotify-tools" to be installed to function!). If we look at the *ExecStart=* directive (with *systemctl cat maldet*), LMD uses *--monitor* argument to read a file which lists the paths of files or directories whose changes LMD will be monitoring. This file is defined by the default_monitor_mode variable's value, which is assigned in *EnvironmentFile=* file and it's "/usr/local/maldetect/monitor_paths". Each file or directory to monitor should be added to that file, ensuring they are delimited between them with a new line. The file should also end with a new line, even if just one entry is added.

> NOTA: Monitoring defaults to every 30 seconds; if required, this can be modified in the configuration file, with the directive "inotify_sleep".

EXERCICIS:

1.-a) Fes que el fitxer "/usr/local/maldetect/monitor_paths" tingui aquest contingut ...:

/home
/var/www

...i, després d'haver activat l'opció "quarantine_hits" a l'arxiu de configuració, reinicia el servidor maldet

b) We will download some example virus files from EICAR, the European Institute for Computer Anti-Virus Research as standard user to its directory (these files are not viruses as such but should be picked up by AV software). So, do this...:

$ cd $HOME
$ wget http://www.eicar.org/download/eicar.com
$ wget http://www.eicar.org/download/eicar.com.txt
$ wget http://www.eicar.org/download/eicar_com.zip
$ wget http://www.eicar.org/download/eicarcom2.zip

...and wait to see if LMD reports something by running maldet -e lsit and/or seeing if it appears new content inside "/usr/local/maldetect/quarantine" folder.

Aide:

The package manager has a built-in verify function that checks all the managed files in the system for changes: to verify of all files, run the command *rpm -Va* : it compares current size, digest, type, permissions, owner and group of all files installed by any package with the stored metadata value taken from rpm database: any discrepancy is shown. However, this command will also display changes in configuration files and you will need to do some filtering to detect important changes. An additional problem to the method with RPM is that an intelligent attacker will modify rpm itself to hide any changes that might have been done by some kind of rootkit which allows the attacker to mask its intrusion and gain root privilege. To solve this, you should implement a secondary check that can also be run completely independent of the installed system. A nice tool to do this is Aide (https://aide.github.io)

To tell AIDE which attributes of which files should be checked, use the "/etc/aide.conf" configuration file. The first section handles general parameters like the location of the AIDE database file; more relevant for local configurations are the "Custom Rules" and the "Directories and Files" sections. A typical rule looks like the following:

PEPE= p+i+n+u+g+s+b+m+c+md5+sha1

Important options for creating rules are:

Option	Description
p	Check for the file permissions of the selected files or directories.
i	Check for the inode number. Every file name has a unique inode number that should not change.
n	Check for the number of links pointing to the relevant file.
u	Check if the owner of the file has changed.
g	Check if the group of the file has changed.
s	Check if the file size has changed.
b	Check if the block count used by the file has changed.
m	Check if the modification time of the file has changed.
c	Check if the files access time has changed.
md5	Check if the md5 checksum of the file has changed.
sha1	Check if the sha1 (160 Bit) checksum of the file has changed.

Then, a typical monitored subject looks like the following, which it represents the checking for all files in "/sbin" (it could be a regular expression!) with the options defined in PEPE but omiting the /sbin/conf.d/ directory:

/sbin PEPE
!/sbin/conf.d

To check whether the configuration file is valid, run: *sudo aide -D* If everything is ok, then you can initialize the Aide database running the command: *sudo aide -i* : a file called "aide.db.new.gz" will be generated inside the "**/var/lib/aide**" folder.

Finally, copy the generated database to a save location like a DVD-R, a remote server or a flash disk for later use. This step is essential as it avoids compromising your database. It is recommended to use a medium which can be written only once to prevent the database being modified. Then, to start using the database, remove the ".new" substring from the initial database file name: sudo mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz

Since this moment, you can perform the check with the following command: *sudo aide -C* If the output is empty, everything is fine: if AIDE found changes, it displays a summary of changes (to learn about the actual changes, increase the verbose level of the check with the parameter *-V*)

You should remake the baseline AIDE database after package updates or configuration files adjustments. To do this you can execute: *sudo aide -u* To start using this updated database for integrity checks, remove the ".new" substring from the file name.