Linux Advanced Security

Lab Guide

Formatierung

Format	Beschreibung
Note	Wichtiger Hinweis
`command`	Kommandozeilen-Ausgabe oder Befehlszeile

Übersicht

Lab 01: Systeme analysieren/aufräumen und automatische Updates aktivieren

1. Login auf node1 und node2
2. Analysieren installierter Pakete (*über und ohne Repository*)
3. Entfernen unnötiger Pakete
4. Automatische Security-Updates aktivieren

Vorgehensweise

`node1`

Login auf node1 und Auflisten installierter Pakete **ohne Repository**:

```
# dnf list installed | grep commandline
libt3config0.x86_64                  1.0.0-3.37
@@commandline
libt3highlight-utils.x86_64          0.5.0-1.36
@@commandline
libt3highlight2.x86_64               0.5.0-1.36
@@commandline
libt3key-utils.x86_64                0.2.10-4.1
@@commandline
libt3key1.x86_64                     0.2.10-4.1
@@commandline
libt3widget2.x86_64                  1.2.2-1.3
@@commandline
libt3window0.x86_64                  0.4.1-1.1
@@commandline
libtranscript1.x86_64                0.3.3-2.37
@@commandline
tilde.x86_64                         1.1.3-1.4
@@commandline
```

Entfernen der Anwendung:

```
# dnf remove tilde
```

Versuchtes automatisches Entfernen nicht mehr benötigter Bibliotheken:

```
# dnf autoremove
Dependencies resolved.
Nothing to do.
Complete!
```

Hinweis: Das entfernen nicht mehr benötigter Abhängigkeiten funktioniert **nicht**, wenn Pakete ohne Repository installiert werden.

Manuelles Entfernen der nicht mehr benötigten Abhängigkeiten:

```
# dnf remove libt3* libtranscript1
```

Auflisten installierter Pakete:

```
# rpm -qa | less
```

Entfernen nicht zwingend notwendiger Pakete:

```
# dnf remove telnet cowsay figlet
```

Installation von `dnf-automatic` für automatische Security-Updates:

```
# dnf install dnf-automatic
```

Konfigurationsdatei (`/etc/dnf/automatic.conf`) anpassen:

```
...
[commands]
upgrade_type = security
download_updates = yes
apply_updates = yes
...
```

Timer aktivieren:

```
# systemctl enable --now dnf-automatic.timer
```

node2

Login auf node2 und Auflisten installierter Pakete **ohne Repository**:

```
# apt list --installed|fgrep '[installed,local]'
libt3config0/now 1.0.0-2 amd64 [installed,local]
libt3widget2/now 1.2.2-1 amd64 [installed,local]
libt3window0/now 0.4.1-1 amd64 [installed,local]
tilde/now 1.1.3-1 amd64 [installed,local]
```

Entfernen der Anwendung:

```
# apt-get remove tilde
```

Versuchtes automatisches Entfernen nicht mehr benötigter Bibliotheken:

```
# apt-get autoremove
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

Hinweis: Das entfernen nicht mehr benötigter Abhängigkeiten funktioniert **nicht**, wenn Pakete ohne Repository installiert werden.

Manuelles Entfernen der nicht mehr benötigten Abhängigkeiten:

```
# apt-get remove libt3* libtranscript1
```

Auflisten installierter Pakete:

```
# dpkg -l
```

Entfernen nicht zwingend notwendiger Pakete:

```
# apt-get remove --purge -y telnet cowsay figlet
```

Installation von `unattended-upgrades` für automatische Security-Updates:

```
# apt-get install unattended-upgrades
```

Konfigurationsdatei (`/etc/apt/apt.conf.d/20auto-upgrades`) erstellen:

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Unattended-Upgrade "1";
```

Hinweis: Hier werden automatische Upgrades lediglich aktiviert, weitere Einstellungen finden sich in der Datei `/etc/apt/apt.conf.d/50unattended-upgrades`.

Funktionstest durchführen:

```
# unattended-upgrades --dry-run --debug
Starting unattended upgrades script
Allowed origins are: o=Ubuntu,a=focal, o=Ubuntu,a=focal-security,
o=UbuntuESMApps,a=focal-apps-security, o=UbuntuESM,a=focal-infra-security
...
applying set ['linux-tools-common']
/usr/bin/dpkg --status-fd 10 --no-triggers --unpack --auto-deconfigure
/var/cache/apt/archives/linux-tools-common_5.4.0-126.142_all.deb
/usr/bin/dpkg --status-fd 10 --configure --pending
...
All upgrades installed
InstCount=0 DelCount=0 BrokenCount=0
The list of kept packages can't be calculated in dry-run mode.
```

Dienst aktivieren:

```
# systemctl enable --now unattended-upgrades.service
```

Lab 02: Dateisystem-Berechtigungen überprüfen und anpassen

1. Login auf node1
2. Überprüfen von Dateien mit zu großzügigen Berechtigungen
3. Anpassen der Berechtigungen

Vorgehensweise

Login auf node1 und nach zu großzügigen Berechtigungen suchen:

```
# find / -type f -perm 777
/var/www/html/web-passwords

# ll /var/www/html/web-passwords
-rwxrwxrwx. 1 root root 16 Mar  8 15:41 /var/www/html/web-passwords

# cat /var/www/html/web-passwords
sgiertz:fakuchad
```

Eine Passwort-Datenbank wurde mit **öffentlichen** Lese- und Schreibrechten versehen.

Suchen nach Dateien mit **setuid**-Flag:

```
# find / -perm /4000
...
/usr/bin/notarootshell
```

Das Programm `notarootshell` ist auffällig, es hat einen **unüblichen Namen**. Ein Blick in den **Inhalt** könnte für Klarheit sorgen:

```
# cat /usr/bin/notarootshell
echo 'This could have been a binary for gaining root access. Luckily, this is just a training.'
```

Suchen nach Dateien mit **setgid**-Flag:

```
# find / -perm /2000
...
/var/www/html/database_dispatcher.php
```

Das Skript `database_dispatcher.php` wird mit den Berechtigungen der Gruppe apache ausgeführt. Hier könnte ein bösartiges Skript versuchen, Schadcode im Kontext des Webservers auszuführen:

```
# cat /var/www/html/database_dispatcher.php
This could have been a script that retrieves commands via HTTP GET and execute them as the webserver user. Luckily, this is just a training.
```

Suchen nach Dateien mit `sticky`-Bit:

```
# find / -perm /1000
...
/dev/mqueue
/dev/shm
/sys/fs/bpf
/var/tmp
/var/tmp/systemd-private-b65c78a7bb6f44ebb9e6e633041dfc23-chronyd.service-boTRzK/tmp
/var/tmp/systemd-private-b65c78a7bb6f44ebb9e6e633041dfc23-php-fpm.service-22QI8V/tmp
/var/tmp/systemd-private-b65c78a7bb6f44ebb9e6e633041dfc23-httpd.service-CXVAEK/tmp
/tmp
/tmp/systemd-private-b65c78a7bb6f44ebb9e6e633041dfc23-chronyd.service-ftUsbo/tmp
/tmp/systemd-private-b65c78a7bb6f44ebb9e6e633041dfc23-php-fpm.service-7aJVHF/tmp
/tmp/systemd-private-b65c78a7bb6f44ebb9e6e633041dfc23-httpd.service-nYOhup/tmp
```

Es sind verschiedene **temporäre Ordner** und öffentlich Dateien zu sehen. Hier sind keine Auffälligkeiten zu entdecken.

Die zu großzügigen Dateiberechtigungen müssen entfernt werden:

- Die Passwortdatei muss nicht ausführbar sein, nur root sollte sie lesen und schreiben dürfen
- `notarootshell` sollte nicht über **setuid** verfügen
- `database_dispatcher.php` sollte nicht über **setgid** verfügen.

```
# chmod u-x,g-x,o-rwx /var/www/html/web-passwords
# chmod u-s /usr/bin/notarootshell
# chmod g-s /var/www/html/database_dispatcher.php
```

Hinweis: Im Praxis-Alltag sollten die auffälligen Schadskripte `notarootshell` und `database_dispatcher.php` umgehend entfernt werden!

Lab 03: ServerTokens anpassen

1. Login auf node1
2. `mod_security` aktivieren
3. ServerTokens anpassen und überprüfen

Vorgehensweise

Login auf node1 und aktuelle Token auslesen:

```
$ curl -I http://localhost
...
Server: Apache/2.4.37 (rocky) OpenSSL/1.1.1k
```

Hinweis: Auf dem System kommen Apache 2.4.37 und OpenSSL 1.1.1k zum Einsatz. Diese Informationen stehen auch Angreifenden zur Verfügung.

`mod_security` installieren:

```
# dnf install -y mod_security
```

Konfigurationsdatei (`/etc/httpd/conf.d/serversignature.conf`) erstellen, um Versionshinweise zu Verschleiern:

```
<IfModule security2_module>
  SecRuleEngine on
  ServerTokens Full
  SecServerSignature "Pinkepank Server (13.37-foobar)"
</IfModule>
```

Hinweis: Weitere Informationen finden sich in der offiziellen Apache-Dokumentation (https://httpd.apache.org/docs/2.4/mod/core.html#serversignature).

Berechtigungen anpassen und Konfiguration neu einlesen

```
# chown root: /etc/httpd/conf.d/serversignature.conf
# chmod 0644 $_
# systemctl reload httpd
```

ServerToken erneut auslesen:

```
$ curl -I http://localhost
...
Server: Pinkepank Server (13.37-foobar)
```

Hinweis: Die Verionsinformationen wurden verschleiert.

Fingerprinting via nmap dürfte fehlschlagen:

```
$ nmap -sV -p T:80 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2022-09-27 11:30 UTC
...
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000088s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    Pinkepank Server (13.37-foobar)
1 service unrecognized despite returning data. If you know the service/version,
please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?
new-service :
...
```

Lab 04: AIDE verwenden

1. Login auf node1 und node2
2. AIDE installieren und konfigurieren
3. Änderung verursachen

Vorgehensweise

node1

Login auf node1 und Installation von AIDE:

```
# dnf install aide
```

Datenbank initialisieren:

```
# aide --init
...
AIDE initialized database at /var/lib/aide/aide.db.new.gz

Number of entries:      102205

# mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz
```

Überprüfung ausführen:

```
# aide --check
AIDE found NO differences between database and filesystem. Looks okay!!
```

Änderungen vornehmen:

```
# touch /usr/bin/hackertool
# echo "i bims 1 datei" > /usr/lib64/libsus.so.6
```

Überprüfung erneut ausführen:

```
# aide --check
Summary:
  Total number of entries:      102208
  Added entries:                3
  Removed entries:              0
  Changed entries:              0

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

f++++++++++++++++: /root/.bash_history
f++++++++++++++++: /usr/bin/hackertool
f++++++++++++++++: /usr/lib64/libsus.so.6
```

Hinweis: Es wurden **zwei neue** Dateien angelegt.

Die Informationen sind auch im Protokoll zu sehen:

```
# less /var/log/aide/aide.log
```

node2

Login auf node2 und Installation von AIDE:

```
# apt-get install aide
```

Initialisieren der Datenbank - hier kommt gegenüber RPM-basierten Distributionen ein **anderes Kommando** zum Einsatz:

```
# aideinit
Running aide --init...
...
AIDE initialized database at /var/lib/aide/aide.db.new

Number of entries:      195543

# mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db
```

Überprüfung ausführen:

```
# aide --config /etc/aide/aide.conf --check
AIDE found NO differences between database and filesystem. Looks okay!!
```

Änderungen vornehmen:

```
# touch /usr/bin/hackertool
# echo "i bims 1 datei" > /usr/lib64/libsus.so.6
```

Überprüfung erneut ausführen:

```
# aide --config /etc/aide/aide.conf --check
Summary:
  Total number of entries:      102208
  Added entries:                3
  Removed entries:              0
  Changed entries:              0

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

f++++++++++++++++: /root/.bash_history
f++++++++++++++++: /usr/bin/hackertool
f++++++++++++++++: /usr/lib64/libsus.so.6
```

Hinweis: Es wurden **zwei neue** Dateien angelegt.

Lab 05: SELinux entdecken

1. Überprüfen des SELinux-Status
2. Anzeigen der Startkonfiguration
3. Anzeigen verschiedener Informationen

- Laufende Prozesse mit SELinux-Kontext mittels ps
- SELinux-Kontext des aktuellen Verzeichnisses

Vorgehensweise

Login auf node1 und Überprüfen ob SELinux aktiv ist:

```
$ getenforce
Enforcing
```

In der Datei /etc/sysconfig/selinux überprüfen, ob SELinux beim Booten aktiviert wird:

```
SELINUX=enforcing
SELINUXTYPE=targeted
```

Anzeigen laufender Prozesse mit SELinux-Kontext:

```
$ ps -Z
LABEL                                PID TTY          TIME CMD
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 4451 pts/0 00:00:00 sudo
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 4453 pts/0 00:00:00 bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 6961 pts/0 00:00:00 ps
```

Anzeigen des aktuellen Ordners mit Datei-Kontext:

```
$ ll -Zd
drwx------. 4 user user unconfined_u:object_r:user_home_dir_t:s0 143 Jul 25
17:20 .
```

Lab 06: SELinux-Dateikontext überprüfen

1. Konfigurieren eines neuen `DocumentRoot`-Ordners für Apache
2. Überprüfen von Protokollen
3. Anpassen des Dateikontext
4. Überprüfen der Funktionalität

Vorgehensweise

Login auf node1 und Anpassen der Apache-Konfigurationsdatei, um das Startverzeichnis von `/var/www/html` nach `/data` zu verschieben:

```
# sed -i "s/\/var\/www\/html/\/data/g" /etc/httpd/conf.d/vhosts.conf
```

Startseite (`/data/index.html`) anlegen:

```
Hello World
```

Apache-Server neustarten und Zugriff testen:

```
# systemctl restart httpd
$ curl http://localhost
```

Hinweis: Es wird ein **403**-Fehler (*Forbidden*) angezeiegt.

Überprüfen des Apache-Fehlerprotokolls:

```
# tail -f /var/log/httpd/error_log
...
[core:error] [pid 7429:tid 123425733433088] (13)Permission denied: [client
127.0.0.1:52520] AH00035: access to /index.html denied (filesystem path
'/data/index.html') because search permissions are missing on a component of
the path
```

Überprüfen, ob der Fehler mit temporär deaktiviertem SELinux auch auftritt:

```
# setenforce 0
# getenforce
Permissive
$ curl http://localhost
```

Hinweis: Die Webseite wird nun korrekt angezeigt.

SELinux unterbindet den Zugriff also aufgrund einer fehlerhaften Konfiguration.

Erneutes Aktivieren von SELinux:

```
# setenforce 1
# getenforce
Enforcing
```

Überprüfen des Datei-Kontexts:

```
# ls -Zd1 /var/www/html /data
      unconfined_u:object_r:default_t:s0 /data
system_u:object_r:httpd_sys_content_t:s0 /var/www/html

# ls -Z1 /data/index.html
unconfined_u:object_r:default_t:s0 /data/index.html
```

Der Ordner `/data` sowie dessen Inhalt müssen dem Typ `httpd_sys_content_t` zugeordnet sein. Die Änderung kann wie folgt gesetzt und getestet werden:

```
# chcon -t httpd_sys_content_t /data/index.html
$ curl http://localhost
```

Hinweis: Die Webseite wird korrekt angezeigt.

Wenn SELinux Datei-Kontexte das nächste Mal überprüft und zurückgesetzt werden (*z.B. nach der Installation eines Policy-Updates*), geht diese Einstellung jedoch verloren:

```
# restorecon -Rv /data
Relabeled /data/index.html from unconfined_u:object_r:httpd_sys_content_t:s0 to
unconfined_u:object_r:default_t:s0
$ curl http://localhost
```

Hinweis: Die Webseite wird nicht angezeigt.

Damit die Änderung dauerthaft übernommen wird, wird der Dateikontext permanent geändert:

```
# semanage fcontext -a -t httpd_sys_content_t "/data(/.*)?"
# restorecon -Rv /data
Relabeled /data from unconfined_u:object_r:default_t:s0 to
unconfined_u:object_r:httpd_sys_content_t:s0
Relabeled /data/index.html from unconfined_u:object_r:default_t:s0 to
unconfined_u:object_r:httpd_sys_content_t:s0
```

Nun funktioniert die Webseite:

```
$ curl http://localhost
```

Als Vorbereitung für die nächste Aufgabe muss der `DocumentRoot`-Pfad wieder zurückgesetzt werden:

```
# sed -i "s/\/data/\/var\/www\/html/g" /etc/httpd/conf.d/vhosts.conf
# systemctl restart httpd
```

Lab 07: SELinux-Booleans

1. Testen einer Web-Anwendung
2. Anzeigen gesetzter Booleans
3. Troubleshooting der Anwendung
4. Setzen eines Booleans

Vorgehensweise

Login auf node1 sowie Testen einer Web-Anwendung:

```
$ curl http://localhost/app.php
Permission denied (13)
```

Hinweis: Die Webseite kann auch unter der öffentlichen IP-Adresse erreicht werden, um lustige Katzenbilder zu sehen.

Der Zugriff funktioniert nicht - ein Blick in das Protokoll:

```
# audit2why -i /var/log/audit/audit.log
type=AVC msg=audit(1658844386.214:1618): avc:  denied  { name_connect } for
pid=14469 comm="php-fpm" dest=80 scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:http_port_t:s0 tclass=tcp_socket permissive=0

        Was caused by:
        One of the following booleans was set incorrectly.
        Description:
        Allow httpd to can network connect

        Allow access by executing:
        # setsebool -P httpd_can_network_connect 1
...
```

Auslesen aktuell gesetzter SELinux-Booleans:

```
# getsebool -a | less
```

Permanentes Setzen eines Booleans:

```
# setsebool -P httpd_can_network_connect on
```

Überprüfen des Booleans:

```
# getsebool httpd_can_network_connect
httpd_can_network_connect --> on
```

Aufrufen der Webseite:

```
$ curl http://localhost/app.php
External access possible
```

Hinweis: Die Anwendung funktioniert nun.

Lab 08: SELinux-Ports verwalten

1. Konfigurieren eines neuen Netzwerkports für Apache
2. Überprüfen von Protokollen
3. Anpassen der Port-Konfiguration
4. Überprüfen der Funktionalität

Vorgehensweise

Login auf node1 und Anpassen der Apache-Konfiguration (`/etc/httpd/conf/httpd.conf`):

```
Listen 1337
```

Neustarten des Dienstes:

```
# systemctl restart httpd
Job for httpd.service failed because the control process exited with error
code.
See "systemctl status httpd.service" and "journalctl -xe" for details.
```

Anzeigen des Journals:

```
# systemctl status httpd
...
httpd[8729]: (13)Permission denied: AH00072: make_sock: could not bind to
address [::]:1337
httpd[8729]: (13)Permission denied: AH00072: make_sock: could not bind to
address 0.0.0.0:1337
```

Der Prozess darf nicht auf Port **1337** lauschen. Zeit, zu überprüfen, ob das Problem mit temporär deaktiviertem SELinux auf auftritt:

```
# setenforce 0
# getenforce
Permissive
# systemctl restart httpd
# curl http://localhost:1337
```

Hinweis: Der Dienst funktioniert.

SELinux unterdrückt hier aufgrund einer Fehlkonfiguration der Ports.

SELinux kann nun wieder aktiviert werden:

```
# setenforce 1
# getenforce
Enforcing
```

Auflisten der für `httpd` zulässigen Ports:

```
# semanage port -l|grep ^http
http_cache_port_t              tcp      8080, 8118, 8123, 10001-10010
http_cache_port_t              udp      3130
http_port_t                    tcp      80, 81, 443, 488, 8008, 8009, 8443,
9000
```

Der Typ `http_port_t` erlaubt eine Reihe von TCP-Ports für `httpd` - **1337** gehört jedoch nicht dazu. Diese Einstellung kann aber wie folgt vorgenommen werden:

```
# semanage port -a -t http_port_t -p tcp 1337
# semanage port -l|grep ^http
...
http_port_t                    tcp      1337, 80, 81, 443, 488, 8008, 8009,
8443, 9000
# systemctl restart httpd
$ curl http://localhost:1337
```

Hinweis: Der Aufruf funktioniert nun wie gewünscht.

Lab 09: AppArmor entdecken

1. Überprüfen des AppArmor-Status
2. Anzeigen definierter AppArmor-Profile

Vorgehensweise

Login auf node2 und Überprüfen des AppArmor-Status:

```
# aa-enabled
# aa-status
apparmor module is loaded.
44 profiles are loaded.
24 profiles are in enforce mode.
   /usr/bin/man
   ...
20 profiles are in complain mode.
   /usr/bin/irssi
   ...
1 processes have profiles defined.
0 processes are in enforce mode.
0 processes are in complain mode.
1 processes are unconfined but have a profile defined.
   /usr/sbin/haveged (552)
```

Im Fehlerfall muss AppArmor erst noch gestartet werden:

```
# systemctl enable --now apparmor.service
```

Erkunden definierter Profile:

```
# ls /etc/apparmor.d
abstractions
disable
force-complain
local
sbin.dhclient
sbin.klogd
...
```

Lab 10: AppArmor-Profil erstellen

1. Versetzen einer unbekannten Anwendung in den **Enforce**-Modus
2. Sichten von Fehlermeldungen
3. Erweitern des Profils
4. Testen der Anwendung

Vorgehensweise:

Login auf node2 sowie Starten der fragwürdigen Anwendung:

```
# sus
Successfully read file /home/user/.bashrc
Successfully read file /etc/profile
Successfully read file /var/run/crond.pid
```

Hinweis: Die Anwendung versucht verschiedene Dateien auszulesen.

```
# aa-autodep sus
Writing updated profile for /usr/local/bin/sus.
```

Dadurch wird die Datei `/etc/apparmor.d/usr.local.bin.sus` erstellt:

```
# Last Modified: Wed Jul 27 14:23:19 2022
#include <tunables/global>

/usr/local/bin/sus flags=(complain) {
  #include <abstractions/base>

  /usr/bin/env ix,
  /usr/local/bin/sus r,

}
```

Testweises Versetzen des Profils in Enforce-Modus:

```
# aa-enforce sus
Setting /usr/local/bin/sus to enforce mode.

# sus
/usr/bin/env: ‘python3’: Permission denied
```

Hinweis: Die Anwendung funktioniert nicht mehr.

Versetzen des Profils in Complain-Modus:

```
# aa-complain sus
Setting /usr/local/bin/sus to complain mode.

# sus
Successfully read file /home/user/.bashrc
Successfully read file /etc/profile
Successfully read file /var/run/crond.pid
```

Hinweis: Die Anwendung funktioniert wieder - erhält aber Zugriff auf Dateien, der verhindert werden soll.

Interaktives Erweitern des Profils:

```
# aa-logprof
Reading log entries from /var/log/syslog.
Updating AppArmor profiles in /etc/apparmor.d.

Profile:  /usr/local/bin/sus
Execute:  /usr/bin/python3.8
Severity: unknown

(I)nherit / (C)hild / (P)rofile / (N)amed / (U)nconfined / (X) ix On / (D)eny /
Abo(r)t / (F)inish
Complain-mode changes:

= Changed Local Profiles =

The following local profiles were changed. Would you like to save them?

 [1 - /usr/local/bin/sus]
(S)ave Changes / Save Selec(t)ed Profile / [(V)iew Changes] / View Changes b/w
(C)lean profiles / Abo(r)t
S

Writing updated profile for /usr/local/bin/sus.
```

Das Profil (`/etc/apparmor.d/usr.local.bin.sus`) sieht nun wie folgt aus:

```
# Last Modified: Wed Jul 27 14:28:17 2022
#include <tunables/global>

/usr/local/bin/sus {
  #include <abstractions/base>

  /usr/bin/env ix,
  /usr/bin/python3.8 mrix,
  /usr/local/bin/sus r,

}
```

Die Anwendung startet jedoch immer noch nicht, weswegen der letzte Schritt wiederholt werden muss. Alle Zugriffe zur Datei `crond.pid` sollen ignoriert werden:

```
# sus
Unable to read file /home/user/.bashrc
Unable to read file /etc/profile
Unable to read file /var/run/crond.pid

# aa-logprof
Reading log entries from /var/log/syslog.
Updating AppArmor profiles in /etc/apparmor.d.
```

```
Complain-mode changes:

Profile:  /usr/local/bin/sus
Path:     /usr/local/lib/python3.8/dist-packages/
New Mode: owner r
Severity: unknown

 [1 - #include <abstractions/python>]
  2 - #include <abstractions/totem>
  3 - owner /usr/local/lib/python3.8/dist-packages/ r,
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew /
Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish
A

Adding #include <abstractions/python> to profile.

Profile:  /usr/local/bin/sus
Path:     /home/user/.bashrc
New Mode: r
Severity: 4

 [1 - /home/*/.bashrc r,]
  2 - /home/user/.bashrc r,
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew /
Audi(t) / Abo(r)t / (F)inish
A

Adding /home/*/.bashrc r, to profile.

Profile:  /usr/local/bin/sus
Path:     /etc/profile
New Mode: owner r
Severity: 1

  1 - #include <abstractions/bash>
 [2 - owner /etc/profile r,]
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew /
Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish
A

Adding owner /etc/profile r, to profile.

Profile:  /usr/local/bin/sus
Path:     /run/crond.pid
New Mode: owner r
Severity: unknown

 [1 - owner /run/crond.pid r,]
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew /
Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish

I
```

```
Profile:  /usr/local/bin/sus
Path:     /run/crond.pid
New Mode: owner r
Severity: unknown

 [1 - owner /run/crond.pid r,]
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew /
Audi(t) / (O)wner permissions off / Abo(r)t / (F)inish
I

= Changed Local Profiles =

The following local profiles were changed. Would you like to save them?

 [1 - /usr/local/bin/sus]
(S)ave Changes / Save Selec(t)ed Profile / [(V)iew Changes] / View Changes b/w
(C)lean profiles / Abo(r)t
S
Writing updated profile for /usr/local/bin/sus.
```

Die Anwendung kann nun die ersten beiden Dateien auslesen, aber nicht die `crond.pid`-Datei:

```
# sus
Successfully read file /home/user/.bashrc
Successfully read file /etc/profile
Unable to read file /var/run/crond.pid
```

Das finale Profil (`/etc/apparmor.d/usr.local.bin.sus`) sieht wie folgt aus:

```
# Last Modified: Wed Jul 27 15:37:25 2022
#include <tunables/global>

/usr/local/bin/sus {
  #include <abstractions/base>
  #include <abstractions/python>

  /home/*/.bashrc r,
  /usr/bin/env ix,
  /usr/bin/python3.8 mrix,
  /usr/local/bin/sus r,
  owner /etc/profile r,

}
```

Lab 11: AppArmor verwalten

1. Geladene Profile überprüfen
2. Profil entladen
3. Profil im Complain-Modus laden
4. AppArmor stoppen und starten

Vorgehensweise

Login auf node2 sowie Überprüfen der geladenen Profile:

```
# aa-status
apparmor module is loaded.
45 profiles are loaded.
25 profiles are in enforce mode.
   /usr/local/bin/sus
   ...
```

Entfernen des Moduls für das sus-Kommando:

```
# ln -s /etc/apparmor.d/profile.name /etc/apparmor.d/disable/
# apparmor_parser -R /etc/apparmor.d/usr.local.bin.sus

# aa-status
apparmor module is loaded.
44 profiles are loaded.
24 profiles are in enforce mode.
```

Laden eines Moduls in den Complain-Modus:

```
# rm /etc/apparmor.d/disable/usr.local.bin.sus
# apparmor_parser -C /etc/apparmor.d/usr.local.bin.sus

# aa-status
apparmor module is loaded.
45 profiles are loaded.
24 profiles are in enforce mode.
...
21 profiles are in complain mode.
   /usr/local/bin/sus
   ...
```

Stoppen der AppArmor-Dienste:

```
# systemctl stop apparmor.service
# aa-status
apparmor module is loaded.
45 profiles are loaded.
24 profiles are in enforce mode.
...
```

Hinweis: Das Stoppen des Dienstes reicht nicht aus.

Entladen aller geladener AppArmor-Profile:

```
# aa-teardown
Unloading AppArmor profiles

# aa-status
apparmor module is loaded.
0 profiles are loaded.
0 profiles are in enforce mode.
0 profiles are in complain mode.
0 processes have profiles defined.
0 processes are in enforce mode.
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
```

Erneutes Laden aller Profile:

```
# systemctl start apparmor.service

# aa-status
apparmor module is loaded.
45 profiles are loaded.
25 profiles are in enforce mode.
...
```

Lab 12: OpenVAS Host-Audit

1. Definition zweier Ziele
2. Erstellen und Ausführen eines Scans
3. Anzeigen des Reports

Vorgehensweise

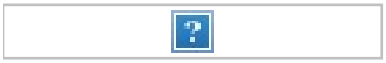
- Anmelden an der OpenVAS-Benutzeroberfläche

Ziele erstellen

- Konfiguration -> Ziele
- Neues Ziel
    - **Name**: `node1`
    - **Hosts**: Manuell -> `192.168.56.20`
    - **Portliste**: `All IANA assigned TCP`
    - Speichern
- Neues Ziel
    - **Name**: `node2`
    - **Hosts**: Manuell -> `192.168.56.30`
    - **Portliste**: `All IANA assigned TCP`
    - Speichern

Scans durchführen

- Scans -> Aufgaben
- Neue Aufgabe
    - **Name**: `(Extern) node1`
    - **Scan-Ziele**: `node1`
    - Speichern
- Neue Aufgabe
    - **Name**: `(Extern) node2`
    - **Scan-Ziele**: `node2`
    - Speichern



Die Scans tauchen anschließend in der Übersicht auf. Mit einem Klick auf **Start** unterhalb **Aktionen** wird der Scan gestartet - der Status ändert sich hierdurch zu **Angefragt**.

Anzeigen der Reports

Nach Abschluss des Scans können die Ergebnisse mit einem Klick auf den Zeitstempel unterhalb von **Letzter Bericht** angezeigt werden:

Lab 13: fail2ban einsetzen

1. SSH-Jail auf `node1` konfigurieren
2. SSH-Verbindungsversuche mit fehlerhaftem Passwort vornehmen
3. Aktivität von `fail2ban` verfolgen
4. IP-Adresse entsperren

Vorgehensweise

Login auf `node1` über `controller`:

```
$ ssh user@controller
controller$ ssh node1
```

Jail für `sshd` unter `/etc/fail2ban/jail.d/10-sshd.conf` konfigurieren:

```
[sshd]
enabled = true
bantime = 1h
maxretry = 3
```

`fail2ban` starten:

```
# systemctl enable --now fail2ban
```

Jail-Status überprüfen:

```
# fail2ban-client status
Status
|- Number of jail:      1
`- Jail list:   sshd
```

Gebannte IP-Adressen überprüfen:

```
# fail2ban-client get sshd banned
[]
```

fail2ban-Log anzeigen:

```
# tail -f /var/log/fail2ban.log
```

Drei Loginversuche über zweiten SSH-Client und falsche Zugangsdaten vornehmen:

```
$ ssh user@<IP-Adresse>
user@<IP-Adresse>'s password:
Permission denied, please try again.
...
```

Im fail2ban-Log sollten die Anmeldungen und die Sperre zu sehen sein:

```
2022-07-21 10:17:11,946 fail2ban.filter         [6889]: INFO    [sshd] Found
10.0.2.2 - 2022-07-21 10:17:11
2022-07-21 10:18:13,445 fail2ban.filter         [6889]: INFO    [sshd] Found
10.0.2.2 - 2022-07-21 10:18:13
2022-07-21 10:18:17,446 fail2ban.filter         [6889]: INFO    [sshd] Found
10.0.2.2 - 2022-07-21 10:18:16
2022-07-21 10:18:18,176 fail2ban.actions        [6889]: NOTICE  [sshd] Ban
10.0.2.2
```

Ein Login ist nun nicht mehr möglich:

```
$ ssh user@<IP-Adresse>
kex_exchange_identification: read: Connection reset by peer
Connection reset by <IP-Adresse> port 2200
```

Sperre überprüfen und aufheben:

```
# fail2ban-client get sshd banned
['10.0.2.2']
# fail2ban-client set sshd unbanip 10.0.2.2
1
# fail2ban-client get sshd banned
[]
```

Logins sind nun wieder möglich:

```
$ ssh user@<IP-Adresse>
user@<IP-Adresse>'s password:
Last failed login: Thu Jul 21 10:18:20 UTC 2022 from 10.0.2.2 on ssh:notty
There were 3 failed login attempts since the last successful login.
Last login: Thu Jul 21 10:10:14 2022 from 192.168.56.10
[user@0-node1 ~]$
```

Lab 14: Anwenden von Dev-Sec

1. Herunterladen der `DevSec Linux Security Baseline`
2. Analysieren des Systems via InSpec
3. Anwenden eines Ansible Playbooks zur automatischen Härtung
4. Erneutes Analysieren des Systems

Vorgehensweise

Herunterladen der Baseline:

```
$ git clone https://github.com/dev-sec/linux-baseline.git
```

Analysieren von `node1`:

```
$ inspec exec linux-baseline -t ssh://user@node1 --sudo -i ~/.ssh/id_rsa
```

Hinweis: Es werden zahlreiche unsichere Konfiugrationen erkannt.

Definieren der benötigten Ansible-Inhalte in der Datei `requirements.yml`:

```
---
collections:
  - name: devsec.hardening
    version: 7.8.0
```

Herunterladen der Inhalte:

```
$ ansible-galaxy collection install -r requirements.yml
```

Erstellen eines Ansible-Playbooks - `hardening.yml`:

```
- name: Harden servers
  hosts: node1
  become: true
  roles:
    - name: devsec.hardening.os_hardening
      sysctl_overwrite:
        # Enable IPv4 forwarding (needed for containers)
        net.ipv4.ip_forward: 1
```

Ausführen des Playbooks:

```
$ ansible-playbook hardening.yml
PLAY [Harden servers] ***********************************************************
...
TASK [devsec.hardening.os_hardening : Configure selinux | selinux-01] *********
ok: [node1]

PLAY RECAP *********************************************************************
node1                      : ok=53   changed=1    unreachable=0    failed=0
skipped=28   rescued=0    ignored=0
```

Erneutes Analysieren von node1:

```
$ inspec exec linux-baseline -t ssh://user@node1 --sudo -i ~/.ssh/id_rsa
```

Hinweis: Es werden bedeutend weniger unsichere Konfiugrationen erkannt.

Lab 15: Erstellen eines InSpec-Profils

1. Erstellen eines InSpec-Profils
   a. Überprüfen, dass ein SSH-Server auf TCP-Port 22 lauscht
   b. Überprüfen, dass das Paket nano installiert ist
2. Checken des Nodes

Vorgehensweise

Erstellen der Ordnerstruktur:

```
$ mkdir -p mycompany-baseline/controls
$ cd mycompany-baseline
```

Erstellen der Datei `inspec.yml`:

```
---
name: mycompany-baseline
title: MyCompany Security Baseline
maintainer: Paula Pinkepank
copyright: MyCompany LTD
copyright_email: ppinkepank@example.com
license: Apache-2.0
summary: Test suite for best practice MyCompany hardening
inspec_version: '>= 4.6.3'
version: 1.0.0
supports:
  - os-family: linux
```

Erstellen der Datei `controls/mycompany_spec.rb`:

```
control 'app-01' do
  impact 1.0
  title 'SSH server running'
  desc 'Ensure that the SSH server is listening'
  describe port(22) do
    it { should be_listening }
  end
end

control 'app-02' do
  impact 1.0
  title 'Editor installed'
  desc 'Ensure that an editor is installed'
  describe package('nano') do
    it { should be_installed }
  end
end
```

```
$ cd ..
$ inspec exec mycompany-baseline -t ssh://user@node1 -i ~/.ssh/id_rsa

Profile:   MyCompany Security Baseline (mycompany-baseline)
Version:   1.0.0
Target:    ssh://root@node1:22
Target ID: 5805abaf-8b62-53e0-97db-519f1b90e551

  ✔  app-01: Web server running
     ✔  Port 80 is expected to be listening
  ✔  app-02: Editor installed
     ✔  System Package nano is expected to be installed


Profile Summary: 2 successful controls, 0 control failures, 0 controls skipped
Test Summary: 2 successful, 0 failures, 0 skipped
```