PSAwise2324Team10Aufgabe11

Security

We decided to use VM3 for this worksheet. It uses Ubuntu 22.04.

NMAP

NMAP is a useful tool for network discovery and security auditing. You can also use it for network inventory, managing service upgrade schedules, and monitoring host or service uptime.

First, make sure it is installed on our VM.

```
sudo apt-get install nmap
```

We can use nmap to either scan a single host or scan ranges. For example, we can use the following command to scan our network comprehensively.

```
# -p1-65535: scan all TCP ports
# -sS: SYN scan (stealthy method for determining open ports without completing
the full TCP handshake)
# -A: enable aggressive scanning (OS detection, version detection, script
scanning, and traceroute)
sudo nmap -sS -A -p1-65535 192.168.0.0/20
```

Consult the official nmap documentation for more information. Nmap is very easy to install and to use, however scanning a large number of hosts is very slow, thus it's better to use in cases where you want to scan a small number of hosts.

Here is the output of the command above for scanning a single host (192.168.1.2)

```
Starting Nmap 7.80 ( https://nmap.org ) at 2024-03-09 14:59 UTC
Nmap scan report for 192.168.1.2
Host is up (0.0017s latency).
Not shown: 65532 closed ports
PORT      STATE SERVICE    VERSION
22/tcp    open  ssh        OpenSSH 9.4p1 Debian 1 (protocol 2.0)
111/tcp   open  rpcbind    2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|_  100000  3,4          111/udp6  rpcbind
10050/tcp open  tcpwrapped
No exact OS matches for host (If you know what OS is running on it, see
https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=3/9%OT=22%CT=1%CU=32587%PV=Y%DS=3%DC=T%G=Y%TM=65EC7988
OS:%P=x86_64-pc-linux-gnu)SEQ(SP=103%GCD=1%ISR=10E%TI=Z%CI=Z%II=I%TS=A)OPS(
OS:O1=M5B4ST11NW7%O2=M5B4ST11NW7%O3=M5B4NNT11NW7%O4=M5B4ST11NW7%O5=M5B4ST11
OS:NW7%O6=M5B4ST11)WIN(W1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%W6=FE88)ECN(
OS:R=Y%DF=Y%T=40%W=FAF0%O=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS
OS:%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=
OS:Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=
OS:R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T
OS:=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=
OS:S)

Network Distance: 3 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 3306/tcp)
HOP RTT     ADDRESS
1   0.56 ms 192.168.10.1
2   1.28 ms 192.168.255.254
3   1.88 ms 192.168.1.2

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 35.62 seconds
```

Summary

Nmap is a very useful but simple tool for the quick scanning of networks. However, it is not specifically created with vulnerability scanning or security evaluation, thus nmap users need to have a high level of knowledge of security in regard to networks to be able to gather useful insights.

OpenVAS

OpenVAS is a vulnerability scanner, that is capable of unauthenticated and authenticated testing, multiple high-level and low-level internet and industrial protocols, performance tuning for large-scale scans and an internal programming language to implement any type of vulnerability test.

OpenVAS requires a minimum of 20GB free disk space and 4GB of RAM. So we first increase both for our VM

```
sudo apt-get install openvas
```

Follow the steps of this tutorial, to run OpenVAS using Greenbone Security Assistant, which gives us a web-based GUI for managing and controlling OpenVAS, within a Docker container:

Installation:

```
sudo apt install ca-certificates curl gnupg

for pkg in docker.io docker-doc docker-compose podman-docker containerd runc;
do sudo apt remove $pkg; done

sudo install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -
o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-
by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt update

sudo apt install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

Setup:

```
sudo usermod -aG docker $USER && su $USER

export DOWNLOAD_DIR=$HOME/greenbone-community-container && mkdir -p
$DOWNLOAD_DIR

cd $DOWNLOAD_DIR && curl -f -L
https://greenbone.github.io/docs/latest/_static/docker-compose-22.4.yml -o
docker-compose.yml
```

Start the Greenbone Containers:

```
docker compose -f $DOWNLOAD_DIR/docker-compose.yml -p greenbone-community-
edition pull

docker compose -f $DOWNLOAD_DIR/docker-compose.yml -p greenbone-community-
edition up -d

# get a continuous stream of the log output of all services
docker compose -f $DOWNLOAD_DIR/docker-compose.yml -p greenbone-community-
edition logs -f
```

Change the admin password (per default it's "admin"):

```
docker compose -f $DOWNLOAD_DIR/docker-compose.yml -p greenbone-community-
edition \
    exec -u gvmd gvmd gvmd --user=admin --new-password='<password>'
```

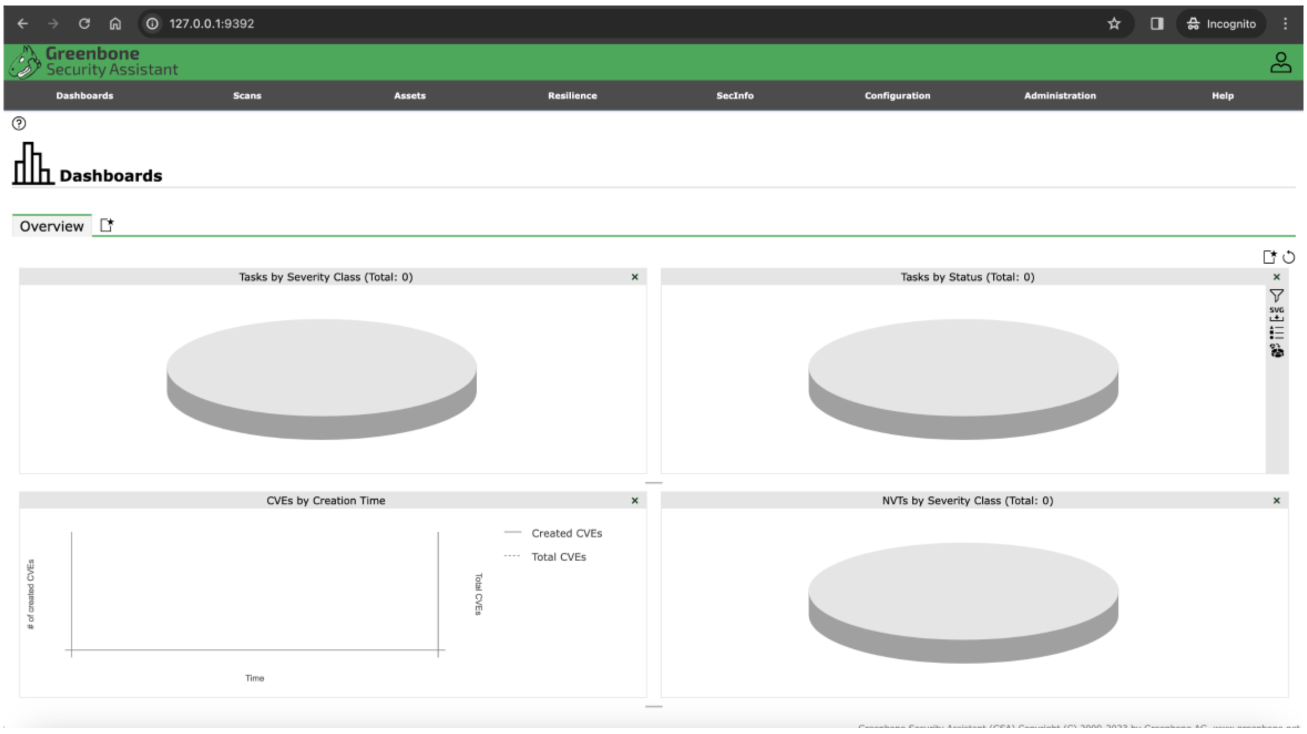
To be able to run the service, please turn the firewall off temporarily using systemctl stop nftables as we were not able to access then OpenVAS dashboard while having an active firewall. We weren't able to figure out in time which port was missing for our whitelist.

Start the Vulnerability Manager:

Run this command on the VM: `xdg-open "http://127.0.0.1:9392" 2>/dev/null >/dev/null &`.

To access the WebUI, we now need to create a SSH tunnel, which allows us to securely forward traffic from a local port on our local machine to a port on the VM (in this case port 9392), so run the following command on your local machine and afterwards you can access the WebUI by opening 127.0.0.1:9392 on your local browser.

```
ssh -L 9392:127.0.0.1:9392 -p 61003 patricia@psa.in.tum.de
```



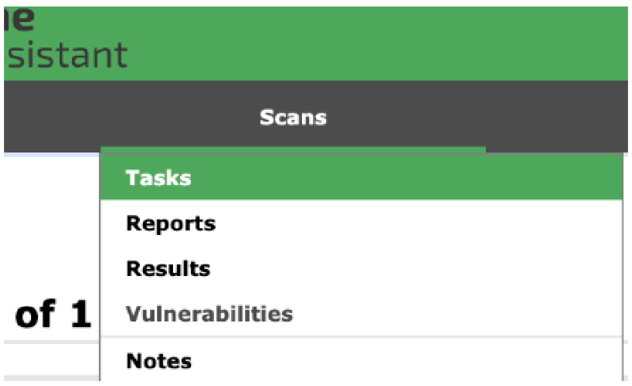
Setup and Start Greenbone Script:

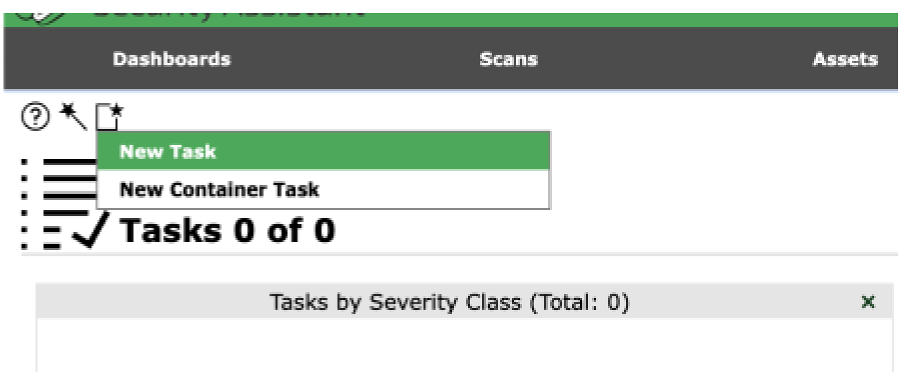
```
curl -f -O https://greenbone.github.io/docs/latest/_static/setup-and-start-
greenbone-community-edition.sh && chmod u+x setup-and-start-greenbone-
community-edition.sh

./setup-and-start-greenbone-community-edition.sh
```

Note: Executing this script may take several minutes up to hours.

Create a new task

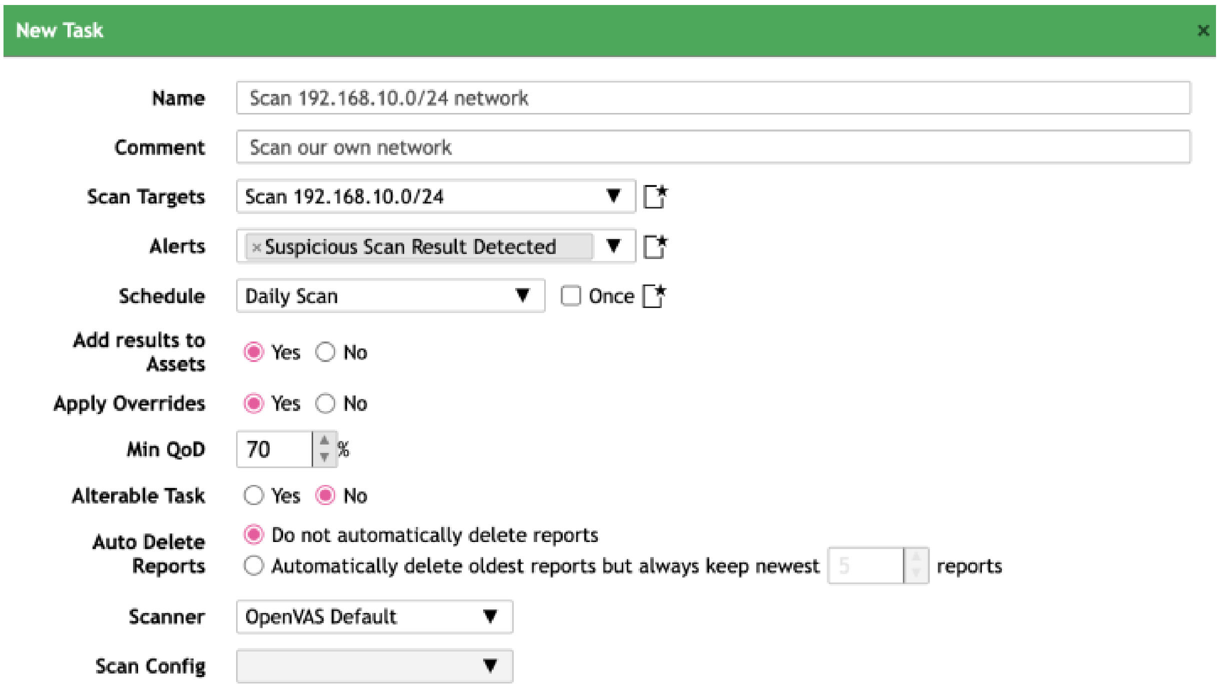
We want to create a new task that scans all hosts in our network for any vulnerabilities and for logging.





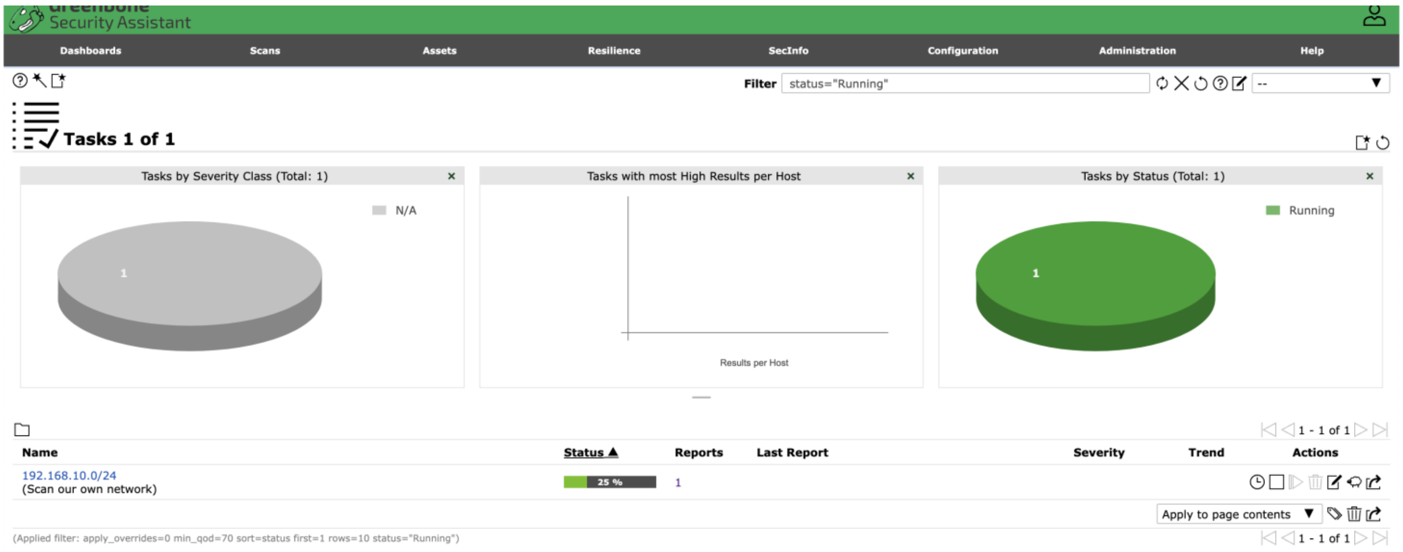
Tasks 0 of 0

Tasks by Severity Class (Total: 0)



Name	Scan 192.168.10.0/24 network
Comment	Scan our own network
Scan Targets	Scan 192.168.10.0/24
Alerts	Suspicious Scan Result Detected
Schedule	Daily Scan ☐ Once
Add results to Assets	[✓ Yes] No
Apply Overrides	[✓ Yes] No
Min QoD	70 %
Alterable Task	Yes [✓ No]
Auto Delete Reports	[✓ Do not automatically delete reports] Automatically delete oldest reports but always keep newest 5 reports
Scanner	OpenVAS Default
Scan Config	

We want the scan to be perform to be performed once daily and we want it to scan the hosts in our subnet. For convenience, we created a file containing a list the IP addresses of our hosts and uploaded the file to the UI, when asked which hosts we want to scan. After we are done with creating the task, we can now view its progress and details on our dashboard. However, scanning all of our hosts takes quite some time so we need to be patient until we can see the full results.



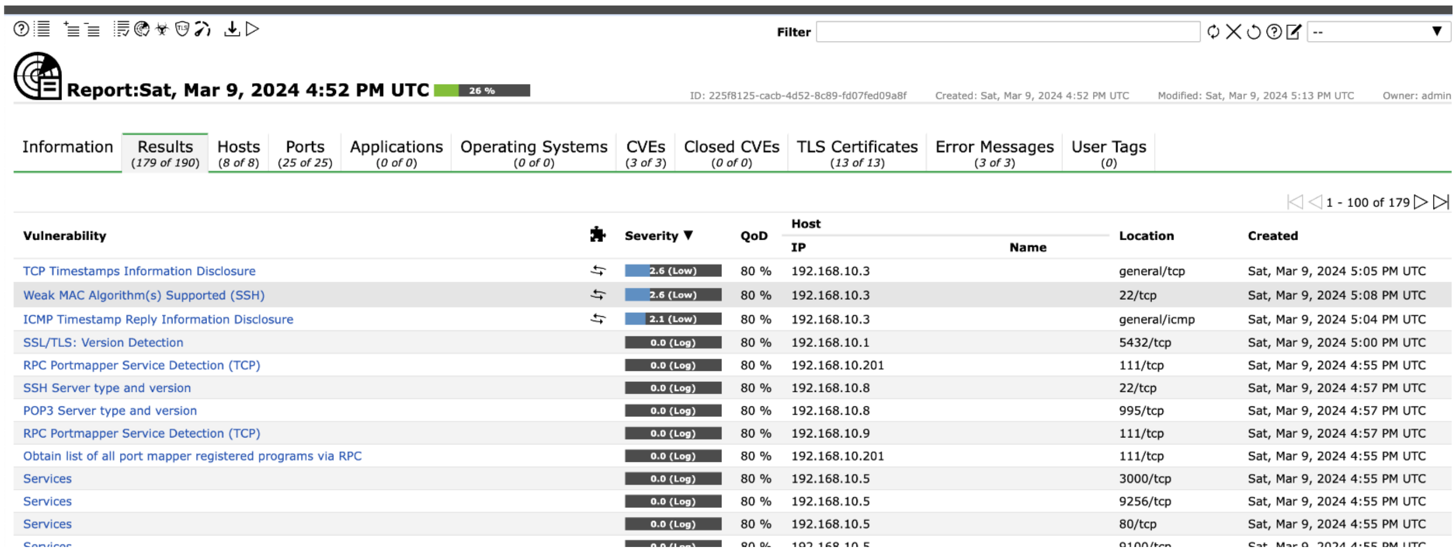
Filter: status="Running"

Tasks 1 of 1

Name	Status ▲	Reports	Last Report	Severity	Trend	Actions
192.168.10.0/24 (Scan our own network)	25 %	1				

(Applied filter: apply_overrides=0 min_qod=70 sort=status first=1 rows=10 status="Running")

We can also filter for the severity level of the results. For transparency, we decided we want to see log messages as well.



Report:Sat, Mar 9, 2024 4:52 PM UTC 26 %

ID: 225f8125-cacb-4d52-8c89-fd07fed09a8f Created: Sat, Mar 9, 2024 4:52 PM UTC Modified: Sat, Mar 9, 2024 5:13 PM UTC Owner: admin

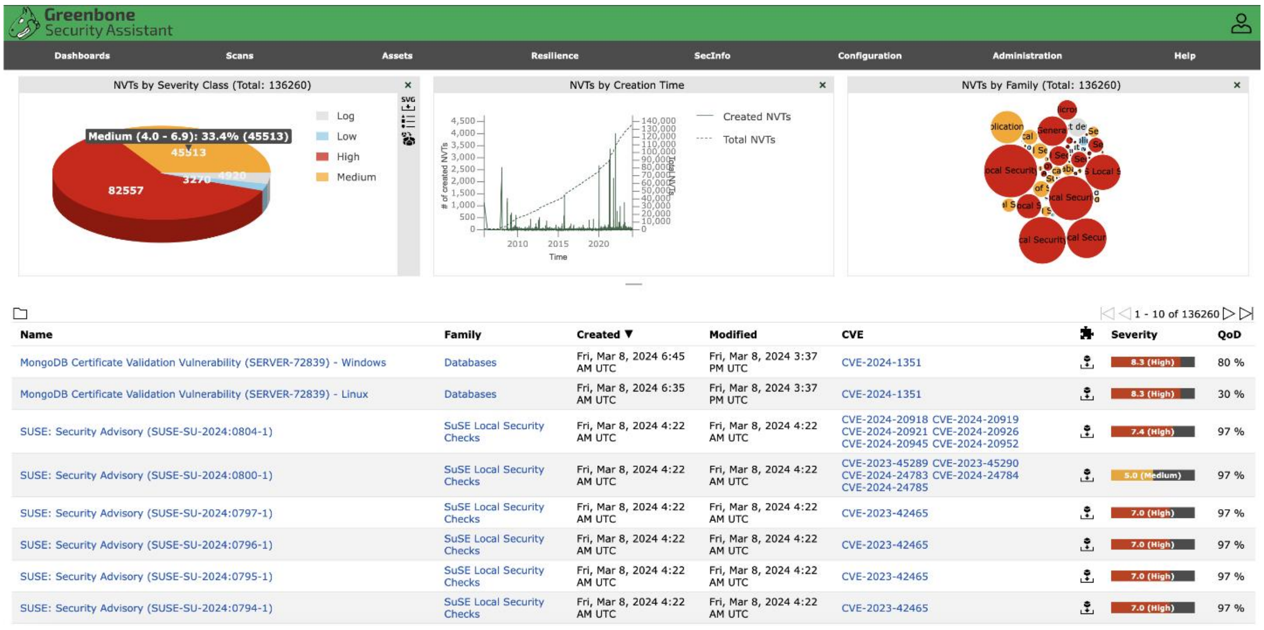
Vulnerability		Severity ▼	QoD	Host		Location	Created
				IP	Name		
TCP Timestamps Information Disclosure	⇆	2.6 (Low)	80 %	192.168.10.3		general/tcp	Sat, Mar 9, 2024 5:05 PM UTC
Weak MAC Algorithm(s) Supported (SSH)	⇆	2.6 (Low)	80 %	192.168.10.3		22/tcp	Sat, Mar 9, 2024 5:08 PM UTC
ICMP Timestamp Reply Information Disclosure	⇆	2.1 (Low)	80 %	192.168.10.3		general/icmp	Sat, Mar 9, 2024 5:04 PM UTC
SSL/TLS: Version Detection		0.0 (Log)	80 %	192.168.10.1		5432/tcp	Sat, Mar 9, 2024 5:00 PM UTC
RPC Portmapper Service Detection (TCP)		0.0 (Log)	80 %	192.168.10.201		111/tcp	Sat, Mar 9, 2024 4:55 PM UTC
SSH Server type and version		0.0 (Log)	80 %	192.168.10.8		22/tcp	Sat, Mar 9, 2024 4:57 PM UTC
POP3 Server type and version		0.0 (Log)	80 %	192.168.10.8		995/tcp	Sat, Mar 9, 2024 4:57 PM UTC
RPC Portmapper Service Detection (TCP)		0.0 (Log)	80 %	192.168.10.9		111/tcp	Sat, Mar 9, 2024 4:57 PM UTC
Obtain list of all port mapper registered programs via RPC		0.0 (Log)	80 %	192.168.10.201		111/tcp	Sat, Mar 9, 2024 4:55 PM UTC
Services		0.0 (Log)	80 %	192.168.10.5		3000/tcp	Sat, Mar 9, 2024 4:55 PM UTC
Services		0.0 (Log)	80 %	192.168.10.5		9256/tcp	Sat, Mar 9, 2024 4:55 PM UTC
Services		0.0 (Log)	80 %	192.168.10.5		80/tcp	Sat, Mar 9, 2024 4:55 PM UTC
Services		0.0 (Log)	80 %	192.168.10.5		9100/tcp	Sat, Mar 9, 2024 4:55 PM UTC

Summary

OpenVAS in combination with Greenbone is a powerful tool for the management of a server cluster, as it offers an easy-to-use web GUI which offers valuable insights into the security level of each host we have scanned.

Creating new tasks does not require a detailed tutorial and the web interface offers filtering options to adjust the dashboard and report overviews to our needs.

OpenVAS also offers a useful overview of newly-found vulnerabilities published on the internet.



1 - 10 of 136260

Name	Family	Created ▼	Modified	CVE	Severity	QoD
MongoDB Certificate Validation Vulnerability (SERVER-72839) - Windows	Databases	Fri, Mar 8, 2024 6:45 AM UTC	Fri, Mar 8, 2024 3:37 PM UTC	CVE-2024-1351	8.3 (High)	80 %
MongoDB Certificate Validation Vulnerability (SERVER-72839) - Linux	Databases	Fri, Mar 8, 2024 6:35 AM UTC	Fri, Mar 8, 2024 3:37 PM UTC	CVE-2024-1351	8.3 (High)	30 %
SUSE: Security Advisory (SUSE-SU-2024:0804-1)	SuSE Local Security Checks	Fri, Mar 8, 2024 4:22 AM UTC	Fri, Mar 8, 2024 4:22 AM UTC	CVE-2024-20918 CVE-2024-20919 CVE-2024-20921 CVE-2024-20926 CVE-2024-20945 CVE-2024-20952	7.4 (High)	97 %
SUSE: Security Advisory (SUSE-SU-2024:0800-1)	SuSE Local Security Checks	Fri, Mar 8, 2024 4:22 AM UTC	Fri, Mar 8, 2024 4:22 AM UTC	CVE-2023-45289 CVE-2023-45290 CVE-2024-24783 CVE-2024-24784 CVE-2024-24785	5.0 (Medium)	97 %
SUSE: Security Advisory (SUSE-SU-2024:0797-1)	SuSE Local Security Checks	Fri, Mar 8, 2024 4:22 AM UTC	Fri, Mar 8, 2024 4:22 AM UTC	CVE-2023-42465	7.0 (High)	97 %
SUSE: Security Advisory (SUSE-SU-2024:0796-1)	SuSE Local Security Checks	Fri, Mar 8, 2024 4:22 AM UTC	Fri, Mar 8, 2024 4:22 AM UTC	CVE-2023-42465	7.0 (High)	97 %
SUSE: Security Advisory (SUSE-SU-2024:0795-1)	SuSE Local Security Checks	Fri, Mar 8, 2024 4:22 AM UTC	Fri, Mar 8, 2024 4:22 AM UTC	CVE-2023-42465	7.0 (High)	97 %
SUSE: Security Advisory (SUSE-SU-2024:0794-1)	SuSE Local Security Checks	Fri, Mar 8, 2024 4:22 AM UTC	Fri, Mar 8, 2024 4:22 AM UTC	CVE-2023-42465	7.0 (High)	97 %

Metasploit

Metasploit is an open-source penetration testing framework, which is used to find and exploit vulnerabilities in computer systems, networks, and applications. It allows us to simulate real-world attacks and test the security posture of our setup.

```
Installation:
# Install dependencies
sudo apt update && sudo apt install -y git autoconf build-essential libpcap-
dev libpq-dev zlib1g-dev libsqlite3-dev

# Add metasploit repository to package sources
curl -fsSL https://apt.metasploit.com/metasploit-framework.gpg.key | sudo gpg
--dearmor | sudo tee /usr/share/keyrings/metasploit.gpg > /dev/null

echo "deb [signed-by=/usr/share/keyrings/metasploit.gpg]
http://downloads.metasploit.com/data/releases/metasploit-framework/apt lucid
main" | sudo tee /etc/apt/sources.list.d/metasploit.list
sudo apt update
sudo apt install metasploit-framework

# Initialize Metasploit database (as non-root user)
msfdb init --component webservice

# Enter Metasploit console
sudo msfconsole
```

Making sure you're in the right directory to run msfconsole can become tedious, so consider using the following Bash alias:

```
echo 'alias msfconsole="pushd $HOME/git/metasploit-framework && ./msfconsole
&& popd"' >> ~/.bash_aliases
```

Using the console, we can load auxiliary modules with different functionalities. We can use these modules to discover hosts, enumerate services, and gather information about potential vulnerabilities.

For example, you can use the scanner/portscan/tcp module to scan for open ports:

```
# clarify which module we want to use
use scanner/portscan/tcp
# set IP range
set RHOSTS 192.168.0.0/24
# show all options
show options
set PORTS 22,23,25,53,80,110,143,443
run
```

Be patient, as the scanning can take some time.

Metasploit can also be used in combination with other tools like nmap to get even more detailed information.

```
# For example
db_nmap -sV -A -p 22,23,25,80,110,143,443 [IP address]
```

For full disclosure: To successfully run the above command we need to be able to access the Metasploit DB, which is based on PostgreSQL. While it initialized and started without any issues and msfdb status shows us an active DB we were not fully able to figure out why we get a "no database connection" error when running the above command. We believe the reason may be because we decided to use VM3, which is already configured as a synchronous backup of our PostgreSQL database, potentially leading to clashes with Metasploit.

Summary

Metasploit is a very powerful tool that, due to the ability to use auxiliary modules, is not only capable of scanning for vulnerabilities but also offers exploits that we can run.

However, because of its diverse range of features, it can be challenging to use.

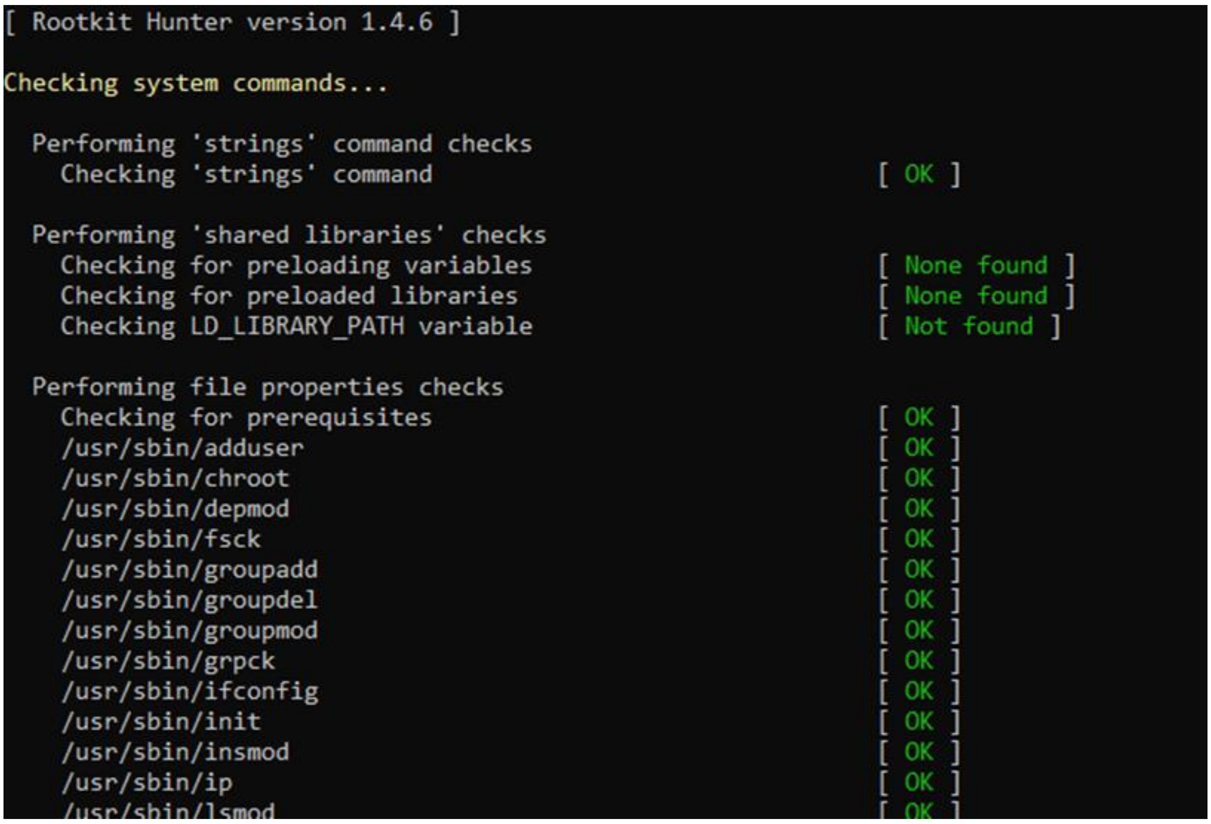
Rootkit Hunter

Rootkit Hunter can be used to look for root kits. To install it, simply run:

```
apt install rkhunter
```

The command rkhunter offers a number of options.

A very useful one is -c, which makes it run checks of the local system and config files respectively. --update is also important, as it is used to keep RootkitHunter's data files up-to-date. You should run it whenever you want to use rkhunter.



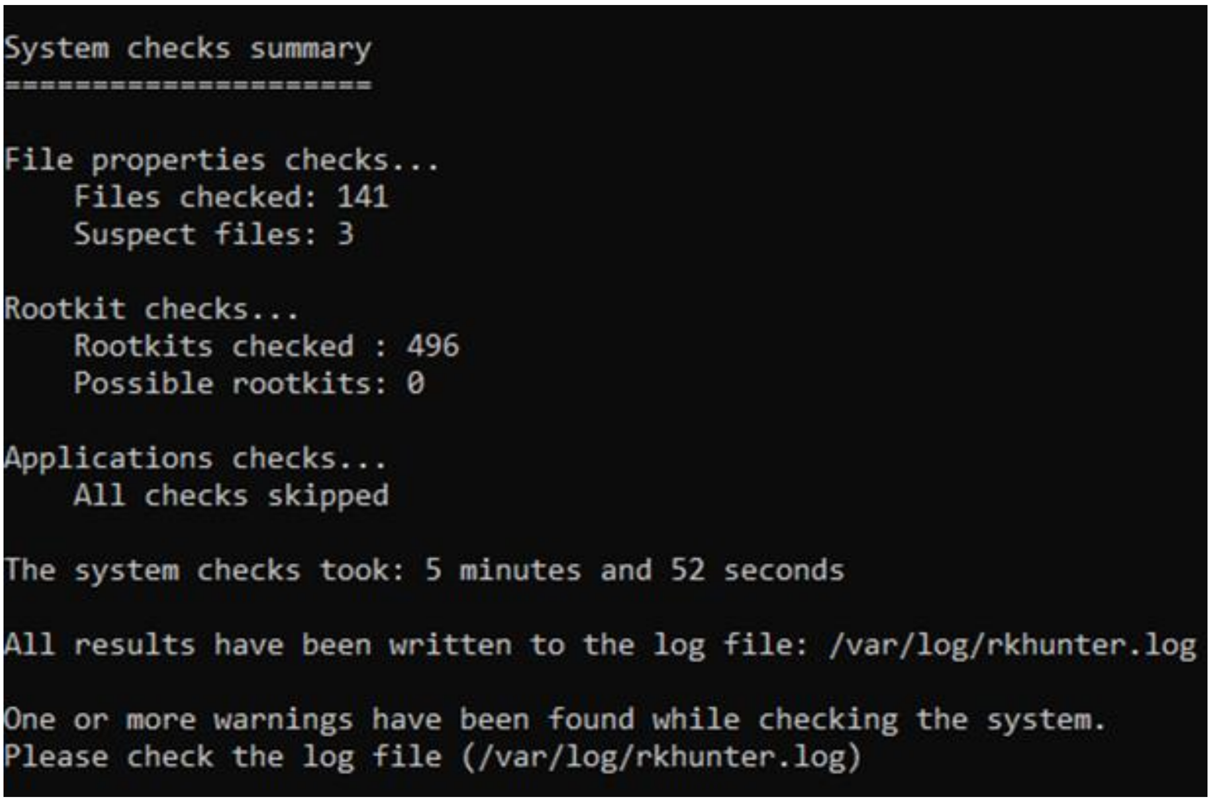
```
[ Rootkit Hunter version 1.4.6 ]

Checking system commands...

  Performing 'strings' command checks
    Checking 'strings' command                               [ OK ]

  Performing 'shared libraries' checks
    Checking for preloading variables                        [ None found ]
    Checking for preloaded libraries                         [ None found ]
    Checking LD_LIBRARY_PATH variable                        [ Not found ]

  Performing file properties checks
    Checking for prerequisites                               [ OK ]
    /usr/sbin/adduser                                        [ OK ]
    /usr/sbin/chroot                                         [ OK ]
    /usr/sbin/depmod                                         [ OK ]
    /usr/sbin/fsck                                           [ OK ]
    /usr/sbin/groupadd                                       [ OK ]
    /usr/sbin/groupdel                                       [ OK ]
    /usr/sbin/groupmod                                       [ OK ]
    /usr/sbin/grpck                                          [ OK ]
    /usr/sbin/ifconfig                                       [ OK ]
    /usr/sbin/init                                           [ OK ]
    /usr/sbin/insmod                                         [ OK ]
    /usr/sbin/ip                                             [ OK ]
    /usr/sbin/lsmod                                          [ OK ]
```

When you run rkhunter -c, several sections of checks will run. You will be prompted to press Enter in between sections, in order for the tool to continue

The sections are, roughly

- system commands
- rootkits
- additional rootkits
- ports/interfaces
- local host

At the end, it will print out a short summary of its findings:



```
System checks summary
=====================

File properties checks...
    Files checked: 141
    Suspect files: 3

Rootkit checks...
    Rootkits checked : 496
    Possible rootkits: 0

Applications checks...
    All checks skipped

The system checks took: 5 minutes and 52 seconds

All results have been written to the log file: /var/log/rkhunter.log

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter.log)
```

We ran sudo rkhunter -c on all of our VMs and no rootkits were detected. One suspicious file was detected on multiple machines - /usr/bin/custom_passwd_ldap - which we wrote ourselves. It covertly replaces the functionality of passwd, so rkhunter was right to find it suspicious, even though we put it there intentionally.

For more information on the individual warnings, we suggest looking at /var/log/rkhunter.log as indicated. It's especially useful in combination with grep. We looked at the warnings and decided that everything seems normal. For example, "SSH root access is allowed" is required for this course.

We found one actual issue using rkhunter, namely on VM5, PermitRootLogin had not actually been set in the sshd_config. This has now been changed.

Exploits

1. Own VMs

Generally, using complex passwords is crucial to deter malicious individuals from brute-forcing them. A strong password should have the following properties (for example taken from Microsoft):

1. At least 12 characters long but 14 or more is better.
2. A combination of uppercase letters, lowercase letters, numbers, and symbols.
3. Not a word that can be found in a dictionary or the name of a person, character, product, or organization.

Another effective security measure involves utilizing unique passwords for each website and application. This practice ensures that if one account is compromised due to a data breach or a compromised website, the security of other accounts remains intact.

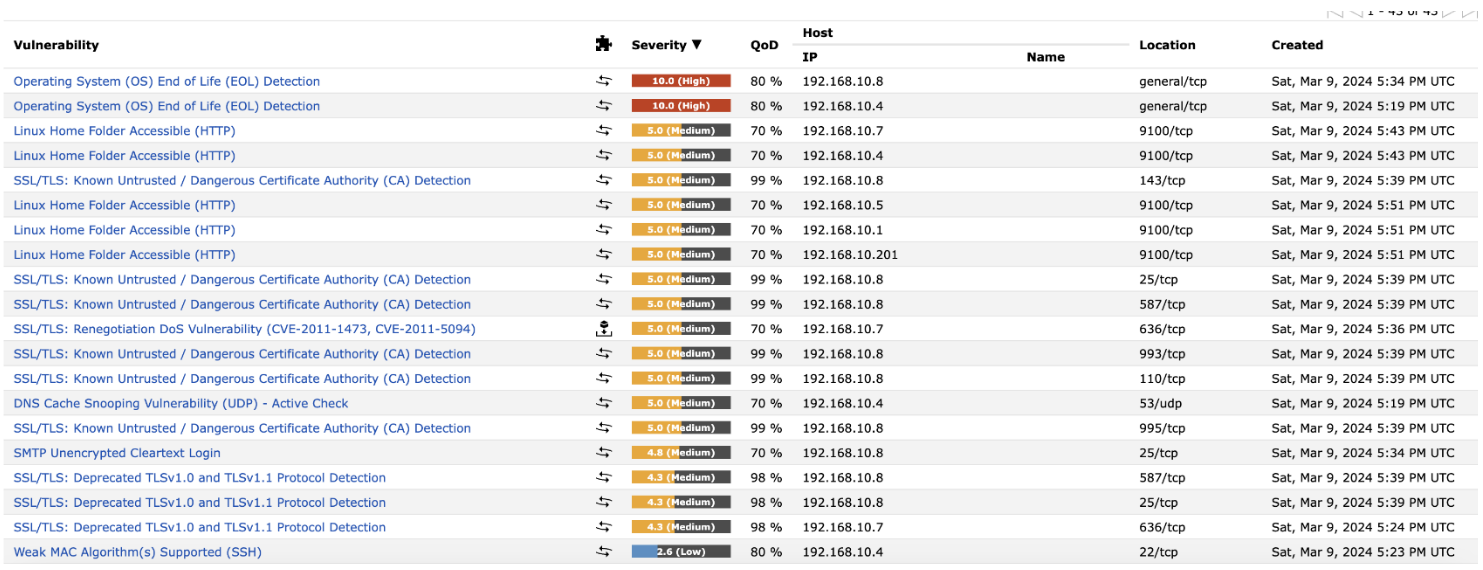
It is also important to use tools like a password manager, who can not only generate complex passwords of variable length and containing diverse letters, numbers and special characters, but can also store them as it would be very difficult for users to learn all of their passwords by heart. However, even password managers can suffer data leaks, thus it would be even more secure to use a self-hosted password manager. For example, the password manager Bitwarden offers a self-hosting option in addition to being able to use Bitwarden's own servers.

In this course, we opted for simple passwords and reused them across various tools and machines. However, it's crucial to emphasize that this practice is highly unsafe in a professional environment and should be avoided at all costs. We could have changed our passwords before the deadline to make them stronger and more resistant to exploitations. Nonetheless, we focused on completing the exercises until just before the deadline and decided against making changes to avoid any complications at that stage.

We have also written some of our passwords into multiple of our articles on this wiki for full transparency of our configurations. While we have since removed our passwords, they can be obviously still found in the page history. It is obvious, that one should never share passwords in the way we did it here but since we have configured our machines as part of a university course and not in a real-life context, we consider this vulnerability negligible as we assume our fellow PSA participants are good-willed and wouldn't act maliciously towards us.

As mentioned in the section above, Rootkit Hunter found one suspicious file on multiple of our machines (/usr/bin/custom_passwd_ldap). However, as we have created this file intentionally, we can disregard this warning.

On the other hand, openVAS reported multiple issues, which we have attached below.



1 - 43 of 43

Vulnerability		Severity ▼	QoD	Host		Location	Created
				IP	Name		
Operating System (OS) End of Life (EOL) Detection	⇆	10.0 (High)	80 %	192.168.10.8		general/tcp	Sat, Mar 9, 2024 5:34 PM UTC
Operating System (OS) End of Life (EOL) Detection	⇆	10.0 (High)	80 %	192.168.10.4		general/tcp	Sat, Mar 9, 2024 5:19 PM UTC
Linux Home Folder Accessible (HTTP)	⇆	5.0 (Medium)	70 %	192.168.10.7		9100/tcp	Sat, Mar 9, 2024 5:43 PM UTC
Linux Home Folder Accessible (HTTP)	⇆	5.0 (Medium)	70 %	192.168.10.4		9100/tcp	Sat, Mar 9, 2024 5:43 PM UTC
SSL/TLS: Known Untrusted / Dangerous Certificate Authority (CA) Detection	⇆	5.0 (Medium)	99 %	192.168.10.8		143/tcp	Sat, Mar 9, 2024 5:39 PM UTC
Linux Home Folder Accessible (HTTP)	⇆	5.0 (Medium)	70 %	192.168.10.5		9100/tcp	Sat, Mar 9, 2024 5:51 PM UTC
Linux Home Folder Accessible (HTTP)	⇆	5.0 (Medium)	70 %	192.168.10.1		9100/tcp	Sat, Mar 9, 2024 5:51 PM UTC
Linux Home Folder Accessible (HTTP)	⇆	5.0 (Medium)	70 %	192.168.10.201		9100/tcp	Sat, Mar 9, 2024 5:51 PM UTC
SSL/TLS: Known Untrusted / Dangerous Certificate Authority (CA) Detection	⇆	5.0 (Medium)	99 %	192.168.10.8		25/tcp	Sat, Mar 9, 2024 5:39 PM UTC
SSL/TLS: Known Untrusted / Dangerous Certificate Authority (CA) Detection	⇆	5.0 (Medium)	99 %	192.168.10.8		587/tcp	Sat, Mar 9, 2024 5:39 PM UTC
SSL/TLS: Renegotiation DoS Vulnerability (CVE-2011-1473, CVE-2011-5094)	⤓	5.0 (Medium)	70 %	192.168.10.7		636/tcp	Sat, Mar 9, 2024 5:36 PM UTC
SSL/TLS: Known Untrusted / Dangerous Certificate Authority (CA) Detection	⇆	5.0 (Medium)	99 %	192.168.10.8		993/tcp	Sat, Mar 9, 2024 5:39 PM UTC
SSL/TLS: Known Untrusted / Dangerous Certificate Authority (CA) Detection	⇆	5.0 (Medium)	99 %	192.168.10.8		110/tcp	Sat, Mar 9, 2024 5:39 PM UTC
DNS Cache Snooping Vulnerability (UDP) - Active Check	⇆	5.0 (Medium)	70 %	192.168.10.4		53/udp	Sat, Mar 9, 2024 5:19 PM UTC
SSL/TLS: Known Untrusted / Dangerous Certificate Authority (CA) Detection	⇆	5.0 (Medium)	99 %	192.168.10.8		995/tcp	Sat, Mar 9, 2024 5:39 PM UTC
SMTP Unencrypted Cleartext Login	⇆	4.8 (Medium)	70 %	192.168.10.8		25/tcp	Sat, Mar 9, 2024 5:34 PM UTC
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection	⇆	4.3 (Medium)	98 %	192.168.10.8		587/tcp	Sat, Mar 9, 2024 5:39 PM UTC
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection	⇆	4.3 (Medium)	98 %	192.168.10.8		25/tcp	Sat, Mar 9, 2024 5:39 PM UTC
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection	⇆	4.3 (Medium)	98 %	192.168.10.7		636/tcp	Sat, Mar 9, 2024 5:24 PM UTC
Weak MAC Algorithm(s) Supported (SSH)	⇆	2.6 (Low)	80 %	192.168.10.4		22/tcp	Sat, Mar 9, 2024 5:23 PM UTC

Regarding the high severity issues:

The warning is caused because end of life (EOL) has been reached for the version of Ubuntu we use (22.04) as a new version has been published. If we managed our machines in a real-life context, we would have updated our OS versions to be resistant to any vulnerabilities found in the earlier version. However, since this course is at its end and we were made aware of this issue shortly before the deadline, we have decided to disregard this issue. Do keep in mind, that it is crucial to use the newest versions of tools and operating systems in real life to be as secure as possible against possible exploitation avenues.

Linux Home Folder Accessible (medium):

The webserver is able to identify files at a home directory. In our case, these files are detected at for example on VM4:

`http://192.168.10.4:9100/.sh_history`

`http://192.168.10.4:9100/.bash_history`

Port 9100 being the port used by Prometheus. To mitigate this issue, we could restrict access to it or remove it completely.

SSL/TLS: Known Untrusted / Dangerous Certificate Authority (CA) Detection (medium):

This issue is caused by us using self-signed certificates. In a real-life scenario we would of course use trusted certificates.

SSL/TLS: Renegotiation DoS Vulnerability (CVE-2011-1473, CVE-2011-5094) (medium):

Our LDAPS service is prone to a denial of service (DoS) vulnerability. OpenVAS tells us that the flaw exists because the remote SSL/TLS service does not properly restrict client-initiated renegotiation within the SSL and TLS protocols. Note: The referenced CVEs are affecting OpenSSL and Mozilla Network Security Services (NSS) but both are in a DISPUTED state with the following rationale: It can also be argued that it is the responsibility of server deployments, not a security library, to prevent or limit renegotiation when it is inappropriate within a specific environment. Both CVEs are still kept in this VT as a reference to the origin of this flaw.

DNS Cache Snooping Vulnerability (UDP) - Active Check (medium):

The DNS server is prone to a cache snooping vulnerability. DNS Snooping may reveal information about the DNS server's owner, such as what vendor, bank, service provider, etc. they use. OpenVAS proposes to disable recursion, to not allow public access to DNS Servers doing recursion and to leave recursion enabled if the DNS Server stays on a corporate network that cannot be reached by untrusted clients.

SMTP Unencrypted Cleartext Login (medium):

Our VM8 is running a SMTP server that allows cleartext logins over unencrypted connections. We should enable SMTPS or enforce the connection via the 'STARTTLS' command.

SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection (medium):

It was possible to detect the usage of the deprecated TLSv1.0 and/or TLSv1.1 protocol on this system. An attacker might be able to use the known cryptographic flaws to eavesdrop the connection between clients and the service to get access to sensitive data transferred within the secured connection. This vulnerability highlights the importance of keeping your tools up-to-date.

Weak MAC Algorithm(s) Supported (SSH) (low):

The remote SSH server supports the following weak client-to-server MAC algorithm(s):

umac-64-etm@openssh.com
umac-64@openssh.com

We can disable the weak MAC algorithms to mitigate this issue.

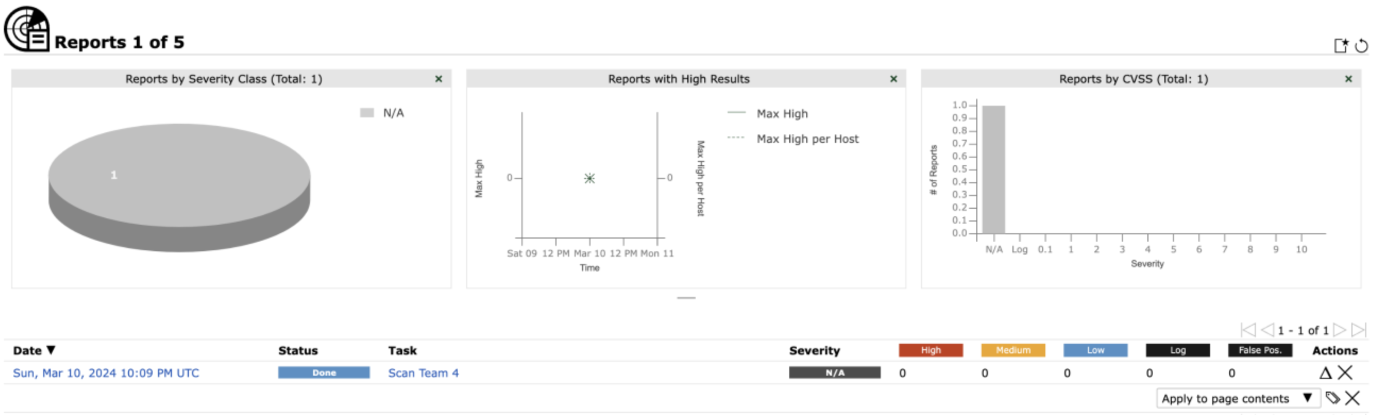
TCP Timestamps Information Disclosure (low):

The remote host implements TCP timestamps and therefore allows to compute the uptime. To disable TCP timestamps on linux add the line 'net.ipv4.tcp_timestamps = 0' to /etc/sysctl.conf. Execute 'sysctl -p' to apply the settings at runtime.

2. Other VMs

To test the security level of other VMs we have randomly decided to scan the hosts of Team 4 using OpenVAS. First, we have used nmap to get a list of all IP addresses in their range that are up. We then used this list to create a new task in the Greenbone dashboard.

```
nmap -sP 192.168.4.0/24

Results in these hosts to be up:
192.168.4.1
192.168.4.2
192.168.4.10
192.168.4.11
```


Reports by Severity Class (Total: 1)
N/A
1
Reports with High Results
Max High
Max High per Host
Sat 09 12 PM Mar 10 12 PM Mon 11
Time
Reports by CVSS (Total: 1)
# of Reports
Severity

Reports 1 of 5

1 - 1 of 1

Date ▼	Status	Task	Severity	High	Medium	Low	Log	False Pos.	Actions
Sun, Mar 10, 2024 10:09 PM UTC	Done	Scan Team 4	N/A	0	0	0	0	0	

Apply to page contents

However, OpenVAS reports no issues found.