Bachelor’s thesis

Information and Communications Technology

2020

Henri Koskinen

SECURING LINUX SYSTEMS IN AN ADVANCED IOT ENVIRONMENT

TURKU AMK
TURKU UNIVERSITY OF
APPLIED SCIENCES

BACHELOR’S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information and Communications Technology

2020 | 31 pages

Henri Koskinen

SECURING LINUX SYSTEMS IN AN ADVANCED IOT ENVIRONMENT

Internet of things (IoT) is an umbrella term for everyday objects with computing power and network abilities. These kinds of items include but are not limited to security cameras, televisions, washing machines, refridgerators, sensors and microcontrollers. Over 7 billion IoT devices are connected to the Internet and the number of the devices increases every year. There have been incidents of undocumented backdoors and buggy software in the devices, allowing an attacker to gain remote access to the system. There is no guarantee that the devices are secure before testing the device security.

The goal of this thesis was to secure devices that uses Linux operating system by using the recommendations given by the NIST and NCSC-FI. Policies were implemented to enforce the sufficient security of the systems. Data of the devices were encrypted by using cryptographic algorithms. As a result, the devices were secured. This thesis serves as a guide to improving system security.

KEYWORDS:

Internet of Things, encryption, scripting

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tieto- ja viestintätekniikka

2020 | 31 sivua

Henri Koskinen

LINUX-JÄRJESTELMIEN SUOJAAMINEN IOT-YMPÄRISTÖSSÄ

Esineiden Internet (IoT) on käsite, jolla tarkoitetaan laitteita jotka lähettävät tai vastaanottavat tietoa verkon yli. Käsitteellä ei viitata perinteisiin tietokoneisiin, vaan nimensä mukaisesti esineisiin, jotka sisältävät verkko-ominaisuuksia. Valvontakamerat, televisiot, pesukoneet, jääkaapit, sensorit ja mikrokontrollerit ovat osa esineiden Internetiä. Arvioiden mukaan yli 7 miljardia IoT-laitetta on kytketty verkkoon ja määrä kasvaa jatkuvasti. Takeita IoT-laitteiden turvallisuudelle ei ole ennen tietoturvan testaamista.

Opinnäytetyössä suojattiin laitteet, jotka käyttävät Linux-käyttöjärjestelmää. Käyttäjille ja laitteille määrättiin turvallisuusvaatimukset, joita noudattamalla varmistettiin riittävän tietoturvan toteutuminen. Laitteiston, kuten suorittimen tietoturvaa parannettiin paikkaamalla tunnetut tietoturva-aukot. Laitteiden sisältämät tiedot suojattiin kryptologisia salausmenetelmiä hyödyntäen.

Lopputuloksena oli tietoturvallinen Linux-järjestelmä. Opinnäytetyö toimii oppaana laitteiden tietoturvan parantamiseen.

ASIASANAT:

Esineiden Internet, tietoturva, salaus

CONTENT

FIGURES

LIST OF ABBREVIATIONS

Abbreviation	Explanation of abbreviation
2FA	Two Factor Authentication
AES	Advanced Encryption Standard
BIOS	Basic Input/Output System
DDoS	Distributed Denial of Service
ETSI	European Telecommunications Standards Institute
FIPS	Federal Information Processing Standard
IoT	Internet of Things
IP	Internet Protocol
LAN	Local Area Network
LUKS	Linux Unified Key Setup
NCSC-FI	National Cyber Security Centre Finland
NFS	Network File System
NIST	National Institute of Technology
OTP	One-time Pad
PAM	Pluggable Authentication Module
PIR	Passive Infrared
SHA	Secure Hash Algorithm
USB	Universal Serial Bus
UUID	Universally Unique Identifier

1 INTRODUCTION

Internet is no doubt one of the greatest inventions in human history and it has changed the ways that information and services are provided. The social media has changed the way that people interact but it also provides an excellent platform for social engineering and tracking of the users online. According to Cisco Annual Internet Report (2018-2023) there will be approximately 25 billion devices connected to the Internet by the year 2021 and nearly 30 billion devices connected by the year 2023. The report goes on and states that by 2023, half of the devices connected to the Internet will be Internet of Things (IoT) devices.[1] The term "IoT" is commonly used to describe objects that have computing power and are capable of sending and receiving information over the network. What devices are considered as IoT objects remains debatable but examples include security cameras, televisions, washing machines, refrigerators, sensors and microcontrollers.

Security of the IoT devices varies from none to excellent and the security standardization process is relatively new and still in progress. The Finnish Transport and Communications Agency Traficom started to issue cyber security labels for IoT products it considers safe on the November of 2019 making Finland the first European country to certify IoT products.[2] The certificate is based on European Telecommunications Standards Institute (ETSI) EN 303 645 standard which is still in draft status.[3]

According to the Finnish Ministry of the Interior "most cybercrime is not reported to the police and most offences that the police investigate remain unsolved, or are only partly solved."[4] It is safe to say that in this context partly solved means remains unsolved. Internet knows no borders and what is considered as a crime in Finland is not necessarily a crime in the country where the cyber attack originated from, making the co-operation between the authorities challenging. The reason why the most cyber attacks are not reported is that they are not detected which is to blame for bad configurations and lack of logging. Devices with default configurations and vulnerable software are connected to the Internet all the time making them easy targets for criminals.

The most widespread attack against IoT devices is a malware called Mirai which was discovered in 2016. Mirai scans devices across the Internet searching for open ports used in telnet connections. When open telnet port is found, Mirai tries to login to the device using a list of default usernames and passwords. Infected devices are used in distributed denial of service (DDoS) attacks.[5]

Devices connected to Internet are under constant threat because the users of the Internet are able to see each other and new security vulnerabilities are discovered every day. Instead of being part of the problem and blindly connecting devices to the Internet it should be encouraged that people play with the settings and become aware of the threats they are exposed to. The biggest misconception about security is that it exists by default. Security as a concept is full of paradoxes because the actions aimed to improve security may as well make the system less secure if the features are not configured properly. Exposing the device to the half of the world’s population is not something to be considered safe either. Being offline does not mean that the systems are any safer but there are less potential threats. Security may be seen as balancing between the usability of the system and restrictions of the users.

The goal of this thesis is to improve overall system security and to make people think about the ways they could improve their own security with simple actions and realize that it takes some time and planning to achieve that. Implementing security features can be fun and educational at the same time. A laptop computer and Raspberry Pi devices were used in this thesis. Laptop computer uses Debian 10 Buster as an operating system and Raspberry Pi devices use Raspbian 10 Buster.

The structure of this thesis is divided into eight chapters. This chapter is an introduction chapter. Chapter 2 introduces standards and best practices used in this thesis. Chapter 3 describes the documentation policy. Chapter 4 describes the access control policy. Chapter 5 describes the user authentication policy. Chapter 6 describes the mobile device policy. Chapter 7 describes the physical safety measurements of the devices and Chapter 8 is a conclusion.

To protect against spoofing attacks, hard-coded information such as serial numbers were replaced using multiple X, Y or Z letters corresponding the actual amount of characters present in the serial number.

2 STANDARDS AND BEST PRACTICES

There is no universal set of rules how to enforce security and the need for it vary from person to person and organization to organization. It is up to the individuals themselves to implement security policies that fit their need. Security policies are documented set of rules that define how to secure the information systems and facilities in use. The main purpose of the policies is to describe the processes to protect the data and network infrastructure. Government organizations and subcontract organizations associated with the government organizations use very strict security policies. The more sensitive the data is, the stricter security policies are followed. Policies used in this thesis are based on the standards and security publications from the following organizations:

The **International Organization for Standardization (ISO)** is a non-governmental organization that brings together experts worldwide to develop and publish standards.[6]

National Institute of Standards and Technology (NIST) is part of the United States Department of Commerce and one of its responsibilities is to develop information security standards and guidelines.[7] NIST is responsible for the development of the **Federal Information Processing Standard (FIPS)** publications which are security guidelines for the US federal agencies.[8]

National Cyber Security Centre Finland (NCSC-FI) is part of the Finnish Transport and Communications Agency Traficom and is considered the main authority responsible for the communication regulations and security of communication systems in Finland. It also provides cyber security guides and information about security threats.[9]

3 DOCUMENTATION POLICY

Documentation policy states what kind of information is documented and on what purpose the information is documented for. Documented information is highly sensitive and will compromise the security of all systems in wrong hands. Administrators and the head of the organization must be the only ones with the access to the documented information. Documentation policy should be treated as it is never complete and reviewed on the monthly basis. The information that is not available for the documentation must be marked as not available. The date when the documentation was done and the name of the person or people who did the documentation must be clearly stated and the document signed, making somebody responsible. Most importantly documentation simplifies the management process and helps to identify the possible threats.

The following information was documented:

- Manufacturer name, the vendor name, product name and model number of the device for inventory, management and troubleshooting purpose.
- Serial number of the device in case of theft and for inventory and management purpose.
- Hardware components for inventory and troubleshooting purpose.
- An Universally Unique Identifier (UUID) number of the device for management purpose.
- BIOS version of the device for troubleshooting and management purpose.
- Firmware version of the device for troubleshooting and management purpose.
- Kernel version of the device for troubleshooting and management purpose.
- Operating system name and version number for management and troubleshooting purpose.
- Installed software and their version numbers for troubleshooting and management purpose.
- The administrator (root) password of the device for management purpose.
- Usernames and usergroups of the device for access control and management purpose.
- Names of the people controlling the user accounts for management purpose.
- Network information of the device for networking and management purpose.

- Network topology for networking and management purpose.

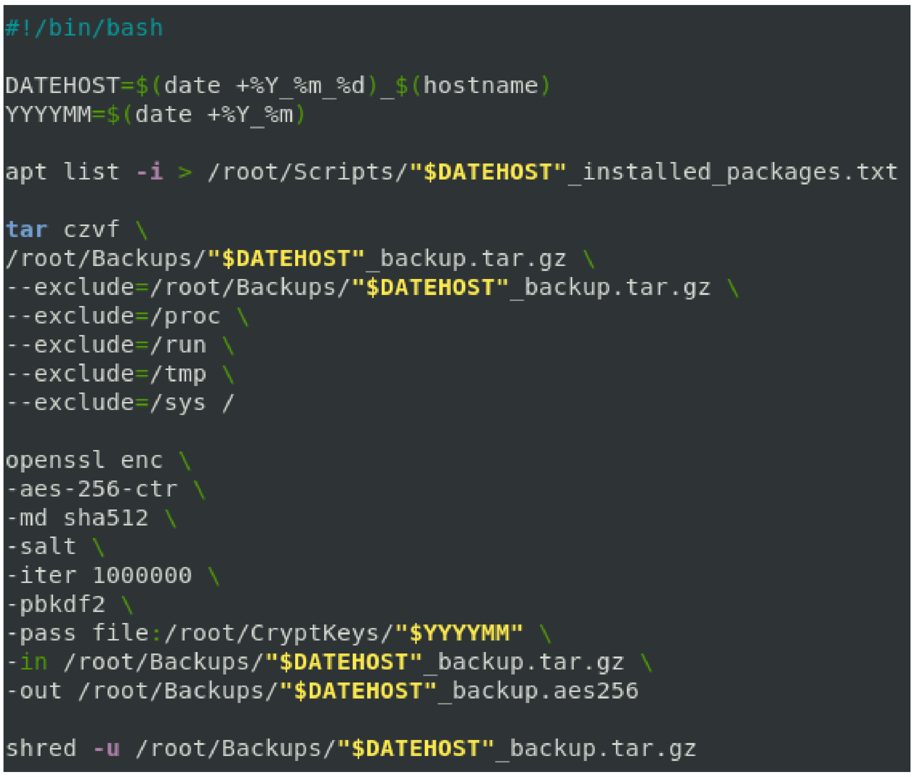
3.1 Threat identification

It is common that devices ship with outdated firmware because the firmware is implemented on the date of manufacturing. Firmware updates are the most critical updates because firmware controls the hardware. Updates should be installed as soon as they come available because they often fix bugs and vulnerabilities, improve hardware performance and add new features. A laptop computer used in this thesis has an Intel processor vulnerable to speculative execution attacks[10] and needed a BIOS update to mitigate the vulnerabilities.

Devices with physically open ports allow connecting devices infected with malicious code or worse. Lockable port blockers should be used to block USB, Firewire, RJ11, RJ45, a microphone, headphone, HDMI, Displayport, eSATA and VGA ports not in use. When the device is turned off or when the keyboard, mouse or other peripherals are not needed, all ports should be locked. Copies of the port blocker keys must be kept safe and people who have access to keys must be documented.

3.2 Automating tasks using cron

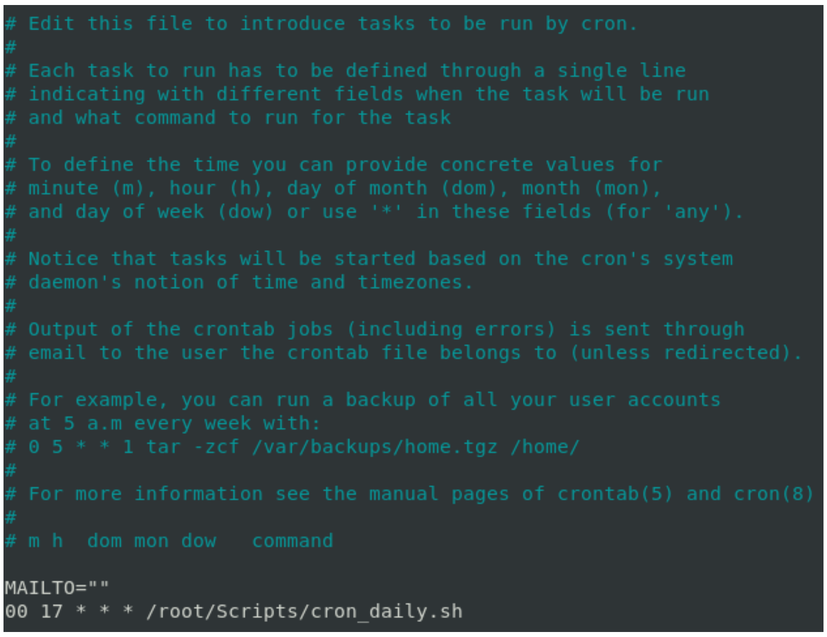
Linux has a program cron which allows the scheduled execution of scripts and commands.[11] Scripts that are executed on a daily basis such as backups and listings of installed programs should use cron. Daily backups are needed to avoid losing as little data as possible when the hard drive fails or the system is compromised. The listing of installed programs helps troubleshooting and finding possible software vulnerabilities. Backup was compressed to a tar archive and encrypted using a monthly changing encryption key located in */root/CryptKeys/* directory. The name of the key file must be in format YYYY_MM where Y is the year and M is the month as the script checks the date to know which key file to use. For example, in June, the key file must be in format 2020_06. The key file contains only one line which is the key used for encryption. Encryption was done using the NIST approved algorithms. The bash script shown in Figure 1 was created to automate the process of creating a daily list of installed packages and system backup.



```
#!/bin/bash

DATEHOST=$(date +%Y_%m_%d)_$(hostname)
YYYYMM=$(date +%Y_%m)

apt list -i > /root/Scripts/"$DATEHOST"_installed_packages.txt

tar czvf \
/root/Backups/"$DATEHOST"_backup.tar.gz \
--exclude=/root/Backups/"$DATEHOST"_backup.tar.gz \
--exclude=/proc \
--exclude=/run \
--exclude=/tmp \
--exclude=/sys /

openssl enc \
-aes-256-ctr \
-md sha512 \
-salt \
-iter 1000000 \
-pbkdf2 \
-pass file:/root/CryptKeys/"$YYYYMM" \
-in /root/Backups/"$DATEHOST"_backup.tar.gz \
-out /root/Backups/"$DATEHOST"_backup.aes256

shred -u /root/Backups/"$DATEHOST"_backup.tar.gz
```

Figure 1. Bash script for the daily listing of the installed programs and backups.

Cron was configured using the `crontab -e` command to execute the bash script *cron_daily.sh* daily at 17:00. The option for sending e-mail when the script is executed was disabled. Figure 2 shows the crontab configuration file with the default comments.

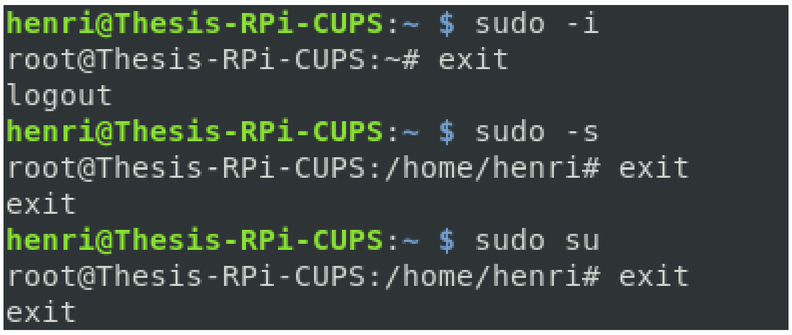


```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command

MAILTO=""
00 17 * * * /root/Scripts/cron_daily.sh
```

Figure 2. The crontab configuration to run scheduled script.

4 ACCESS CONTROL POLICY

Administrator account in Linux is called root account. Privilege escalation to root user is the ultimate goal for any person with malicious intentions and imposes the greatest risk for system security because the root user has full control of the system. A person with root privileges may infect the device with a backdoor or a keylogger and delete the log information, leaving no traces of malicious activity. Raspbian includes a program sudo by default which allows users belonging to the group sudo to execute administrative commands.[12] The problem with sudo is that the default configuration file allows users to execute any administrative command without restrictions. Some commands allow straight privilege escalation to the root account as shown in Figure 3.



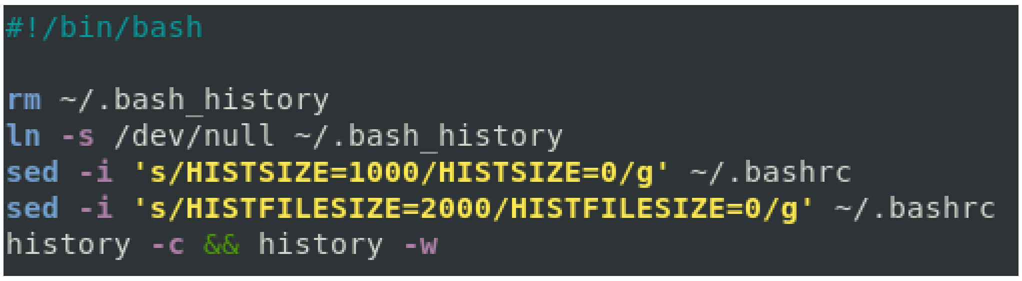
```
henri@Thesis-RPi-CUPS:~ $ sudo -i
root@Thesis-RPi-CUPS:~# exit
logout
henri@Thesis-RPi-CUPS:~ $ sudo -s
root@Thesis-RPi-CUPS:/home/henri# exit
exit
henri@Thesis-RPi-CUPS:~ $ sudo su
root@Thesis-RPi-CUPS:/home/henri# exit
exit
```

Figure 3. Examples of privilege escalation to root using sudo.

The fact that sudo allows normal user too much power combined with the long history of security vulnerabilities[13] gives enough reasons to remove the program. Program sudo was removed using the `apt-get purge sudo` command.

When the user tries to execute the command `sudo`, it may be an indication of malicious activity or a hacked user account. Linux allows the use of aliases which are mainly used to shorten long commands but they may also be used to respond to certain keywords the user executes. Alias has very limited functions and it may only be used to substitute the first keyword the user types with bash commands.[14] Aliases may be directly added to the *~/.bashrc* configuration file but to keep everything tidy, a file *~/.bash_aliases* were used instead. The tilde symbol in the file path represents the home folder of the user and the dot symbol indicates that the file is hidden. It does not matter which file is used for the aliases, as the file *~/.bashrc* checks if the file *~/.bash_aliases* exists and executes any aliases from it. As the program sudo was removed and the user tries to execute the `sudo` command, the bash shell informs the user that the command sudo is not found. Instead of giving that information to the user, an alias was created that logs the user out of the system without warning by killing all the processes the user is running. People using the devices must be made aware of this because killing all the processes unexpectedly means that all the unsaved data is lost. Alias was implemented using the `echo 'alias sudo="pkill -KILL -u $(whoami)"' >> ~/.bash_aliases` command.

Linux saves a list of previously used commands in a *~/.bash_history* file. The user may view the previously typed commands using the up arrow in the keyboard. As the bash history may come handy occasionally, at the same time a person with malicious intentions may use that information to find important files in the system or even passwords the user has typed. The configuration of the bash history is done by modifying the file *~/.bashrc*. Default configuration for Debian and Raspbian saves the last 2,000 commands the user has used. Full bash history may be viewed by opening the *~/.bash_history* file with a text editor or using the `history` command. Bash history was cleared by deleting the *~/.bash_history* file and then creating a symbolic link between */dev/null* and *~/.bash_history*. Symbolic link to */dev/null* disposes of all the information that bash history creates. The command `sed` was used to change the number of the saved commands to zero because without doing so, the history file seems to continue saving commands. The history file still had all the previous commands so it was cleared and saved. A bash script shown in Figure 4 was created to automate the process.



```
#!/bin/bash

rm ~/.bash_history
ln -s /dev/null ~/.bash_history
sed -i 's/HISTSIZE=1000/HISTSIZE=0/g' ~/.bashrc
sed -i 's/HISTFILESIZE=2000/HISTFILESIZE=0/g' ~/.bashrc
history -c && history -w
```

Figure 4. Bash script to clear bash history and history file.

Raspbian has a default user pi with a default password raspberry.[15] The use of default usernames and passwords is strictly forbidden so the user pi was deleted using the `deluser pi` command.

5 USER AUTHENTICATION POLICY

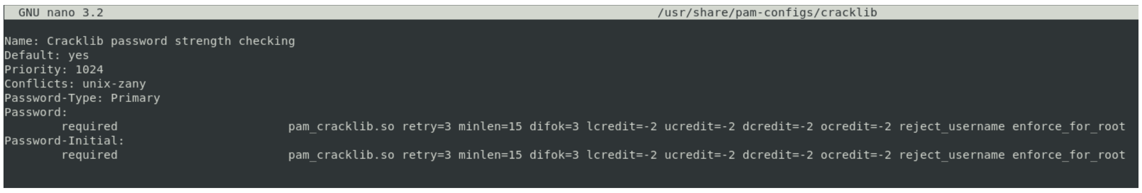
Using just an username and a password for authentication is considered a weak authentication method because the information is guessable and vulnerable to dictionary and brute force attacks.[16] Each character in the password adds more complexity to it, making the password guessing harder. A long password can resist brute force attacks where an attacker tries to guess the password by using every possible character combination. Cracking a strong password using brute force attack is considered computationally infeasible as it takes centuries if not millennia to crack the password.

Dictionary attacks are attacks where an attacker attempts to login by using a list of dictionary words or breached passwords. This is the reason why the use dictionary words and breached passwords is not accepted. Using the words backwards in a password is not going to help because a dictionary attack tries all the words in reverse order as well. Rainbow table attack is a variation of dictionary attack which uses hashes generated from the words instead of plaintext.

By following the NIST[16] and the NCSC-FI[17] recommendations, a strong password should at least meet the following complexity requirements:

- The password needs to be at least 15 characters in length.
- The password does not contain dictionary words.
- The password does not contain any personal information such as a real name, address, phone number or username.
- The password does not contain any geometric, repetitive or sequential pattern on the keyboard such as wasd, qwerty, zxcvb, aaaaa or 123456789.
- The password needs to include numbers, special characters, upper case letters and lower case letters.

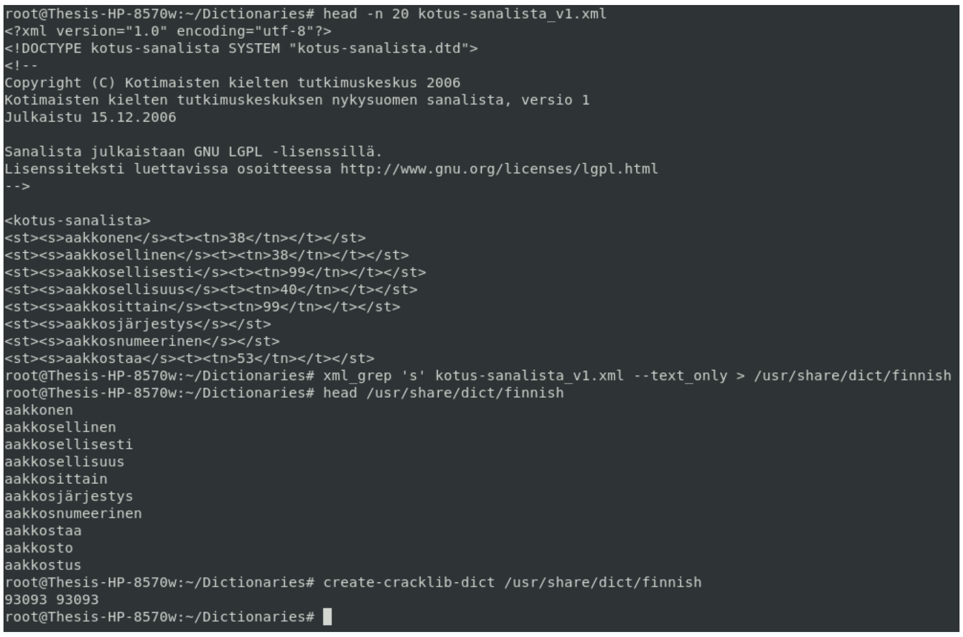
Linux uses pluggable authentication modules (PAM) to enforce additional authentication methods. PAM module cracklib[18] was installed using the `apt-get install libpam-cracklib` command. The cracklib module allows the modification of the password complexity requirements. Global configuration files for PAM are located in the */etc/pam.d/* directory but the module configuration is done by modifying the configuration file in the */usr/share/pam-configs/* directory and using the command `pam-auth-update` to update the global configuration files. The configuration of the cracklib module was done by editing the */usr/share/pam-configs/cracklib* file. Figure 5 shows the modified configuration file that enforces all users including root the use minimum 15 characters in a password that must contain at least two lowercase letters, two uppercase letters, two numbers and two special characters.



```
  GNU nano 3.2                                        /usr/share/pam-configs/cracklib

Name: Cracklib password strength checking
Default: yes
Priority: 1024
Conflicts: unix-zany
Password-Type: Primary
Password:
        required                        pam_cracklib.so retry=3 minlen=15 difok=3 lcredit=-2 ucredit=-2 dcredit=-2 ocredit=-2 reject_username enforce_for_root
Password-Initial:
        required                        pam_cracklib.so retry=3 minlen=15 difok=3 lcredit=-2 ucredit=-2 dcredit=-2 ocredit=-2 reject_username enforce_for_root
```

Figure 5. /usr/share/pam-configs/cracklib configuration file.

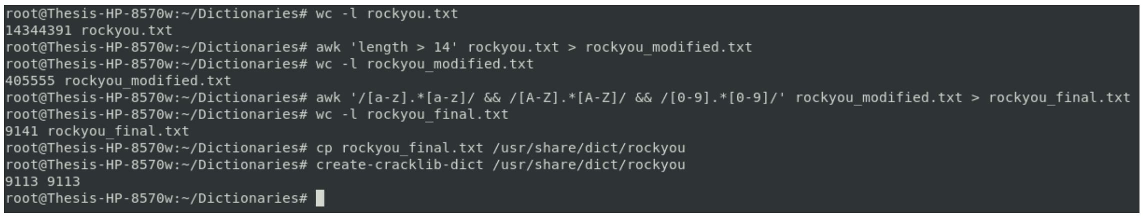
Cracklib module includes an American English dictionary and additional dictionaries can be added as needed. Cracklib uses dictionaries during the password phase to check if the password matches the words in the dictionaries. If the matching word is found, cracklib disallows the use of the password. Institute for the Languages of Finland offers free Finnish dictionary containing over 90,000 Finnish words.[19] The list is in XML format and needs to be converted into a text format in the way that there is a single word per line. Words needed were inside the XML tags so the program xml-twig-tools[20] was installed using the `apt-get install xml-twig-tools` command. The program xml-twig-tools includes a tool xml_grep which allows the extraction of text inside the XML tags and conversion to a text format. The process is shown in Figure 6.



```
root@Thesis-HP-8570w:~/Dictionaries# head -n 20 kotus-sanalista_v1.xml
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE kotus-sanalista SYSTEM "kotus-sanalista.dtd">
<!--
Copyright (C) Kotimaisten kielten tutkimuskeskus 2006
Kotimaisten kielten tutkimuskeskuksen nykysuomen sanalista, versio 1
Julkaistu 15.12.2006

Sanalista julkaistaan GNU LGPL -lisenssillä.
Lisenssiteksti luettavissa osoitteessa http://www.gnu.org/licenses/lgpl.html
-->

<kotus-sanalista>
<st><s>aakkonen</s><t><tn>38</tn></t></st>
<st><s>aakkosellinen</s><t><tn>38</tn></t></st>
<st><s>aakkosellisesti</s><t><tn>99</tn></t></st>
<st><s>aakkosellisuus</s><t><tn>40</tn></t></st>
<st><s>aakkosittain</s><t><tn>99</tn></t></st>
<st><s>aakkosjärjestys</s></st>
<st><s>aakkosnumeerinen</s></st>
<st><s>aakkostaa</s><t><tn>53</tn></t></st>
root@Thesis-HP-8570w:~/Dictionaries# xml_grep 's' kotus-sanalista_v1.xml --text_only > /usr/share/dict/finnish
root@Thesis-HP-8570w:~/Dictionaries# head /usr/share/dict/finnish
aakkonen
aakkosellinen
aakkosellisesti
aakkosellisuus
aakkosittain
aakkosjärjestys
aakkosnumeerinen
aakkostaa
aakkosto
aakkostus
root@Thesis-HP-8570w:~/Dictionaries# create-cracklib-dict /usr/share/dict/finnish
93093 93093
root@Thesis-HP-8570w:~/Dictionaries# █
```

Figure 6. Finnish dictionary.

A dictionary feature may also be used to ban the use of specific words such as breached passwords by creating a dictionary file out of a password list. The file rockyou.txt[21] was used as the basis of a filelist as the file is 133 megabytes in size and contains 14,344,391 lines of breached passwords and usernames. The number of lines were reduced to 405,555 by removing all lines containing less than 15 characters which is the minimum length of the allowed password. The password list still included passwords that did not meet the complexity requirements of the password policy so the program awk[22] were used to find words that contained at least two lower case letters, two upper case letters and two numbers. Using awk to find lines that include special symbols may cause incorrect results depending on the localization settings and the encoding of the password list file so the password list was left with a few unneeded words because it is better to have all the needed words with a few extras than missing words. The number of lines were reduced to 9,141 and the final size of the modified rockyou.txt was 226 kilobytes. Figure 7 shows the process of reducing the password list size and creating a dictionary from it.

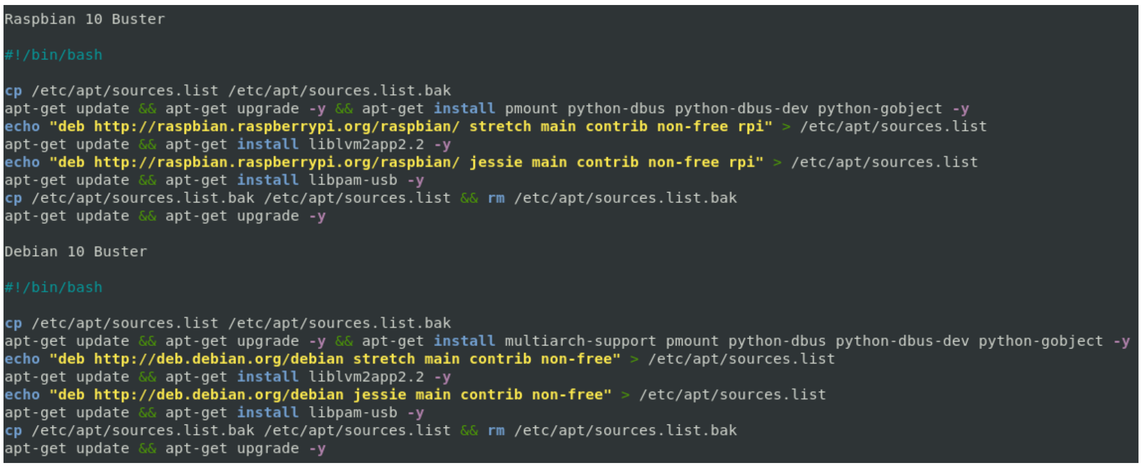


```
root@Thesis-HP-8570w:~/Dictionaries# wc -l rockyou.txt
14344391 rockyou.txt
root@Thesis-HP-8570w:~/Dictionaries# awk 'length > 14' rockyou.txt > rockyou_modified.txt
root@Thesis-HP-8570w:~/Dictionaries# wc -l rockyou_modified.txt
405555 rockyou_modified.txt
root@Thesis-HP-8570w:~/Dictionaries# awk '/[a-z].*[a-z]/ && /[A-Z].*[A-Z]/ && /[0-9].*[0-9]/' rockyou_modified.txt > rockyou_final.txt
root@Thesis-HP-8570w:~/Dictionaries# wc -l rockyou_final.txt
9141 rockyou_final.txt
root@Thesis-HP-8570w:~/Dictionaries# cp rockyou_final.txt /usr/share/dict/rockyou
root@Thesis-HP-8570w:~/Dictionaries# create-cracklib-dict /usr/share/dict/rockyou
9113 9113
root@Thesis-HP-8570w:~/Dictionaries# █
```

Figure 7. Reducing the password list size using awk.

All users including the root are enforced to use two-factor authentication (2FA). 2FA means that the person must authenticate using two of the three following options: something they know, something they have or something they are. The username and a password is considered as something that is known, USB flash drives and smartcards are considered as something that the user has and fingerprint and other biometric features are considered as something that the user is.[16]

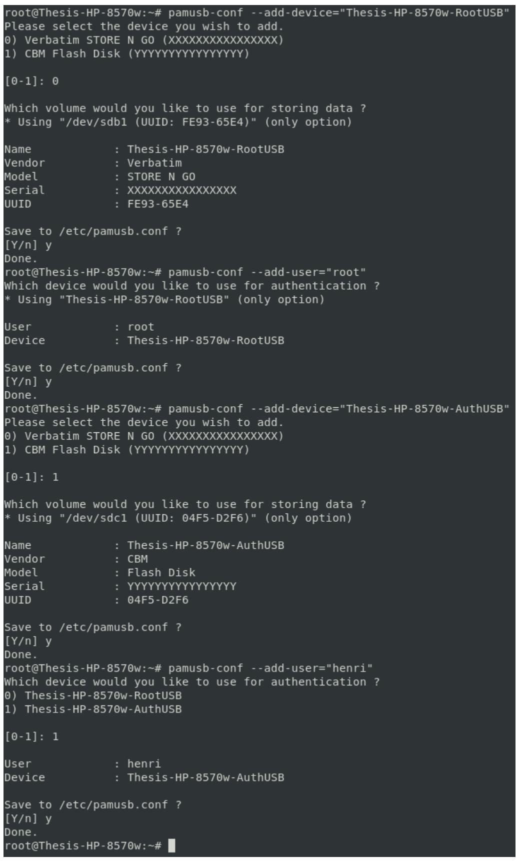
The implementation of the USB authentication needs a PAM module libpam-usb which is not available in the current Buster package repository. The latest version of the package is available at the Jessie package repository.[23] Dependencies of the libpam-usb were installed such as the latest available version of each program was used. Editing package repositories manually is time-consuming and because the installation process is identical between the devices using the same operating system, the bash scripts shown in Figure 8 were created to automate the installation process.



```
Raspbian 10 Buster

#!/bin/bash

cp /etc/apt/sources.list /etc/apt/sources.list.bak
apt-get update && apt-get upgrade -y && apt-get install pmount python-dbus python-dbus-dev python-gobject -y
echo "deb http://raspbian.raspberrypi.org/raspbian/ stretch main contrib non-free rpi" > /etc/apt/sources.list
apt-get update && apt-get install liblvm2app2.2 -y
echo "deb http://raspbian.raspberrypi.org/raspbian/ jessie main contrib non-free rpi" > /etc/apt/sources.list
apt-get update && apt-get install libpam-usb -y
cp /etc/apt/sources.list.bak /etc/apt/sources.list && rm /etc/apt/sources.list.bak
apt-get update && apt-get upgrade -y

Debian 10 Buster

#!/bin/bash

cp /etc/apt/sources.list /etc/apt/sources.list.bak
apt-get update && apt-get upgrade -y && apt-get install multiarch-support pmount python-dbus python-dbus-dev python-gobject -y
echo "deb http://deb.debian.org/debian stretch main contrib non-free" > /etc/apt/sources.list
apt-get update && apt-get install liblvm2app2.2 -y
echo "deb http://deb.debian.org/debian jessie main contrib non-free" > /etc/apt/sources.list
apt-get update && apt-get install libpam-usb -y
cp /etc/apt/sources.list.bak /etc/apt/sources.list && rm /etc/apt/sources.list.bak
apt-get update && apt-get upgrade -y
```

Figure 8. Bash scripts for the installation of PAM module libpam-usb.

Each user needs their own entry in the */etc/pamusb.conf* file to be able to login to the system.[24] Entries were added using the pamusb-conf tool as shown in Figure 9.

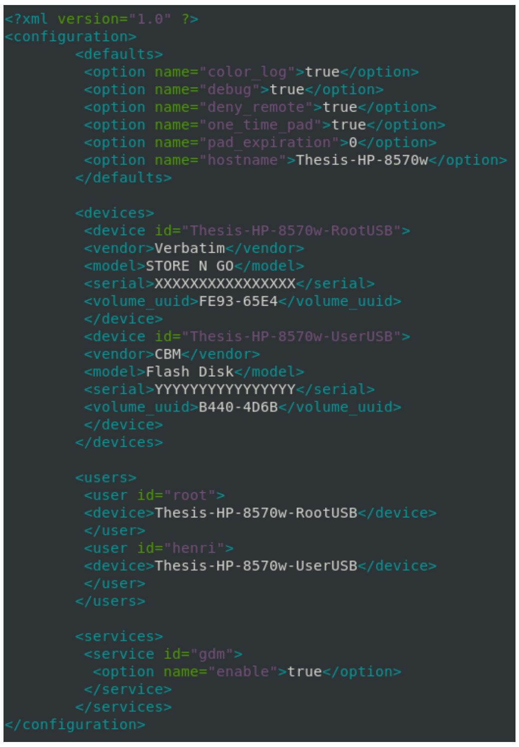


```
root@Thesis-HP-8570w:~# pamusb-conf --add-device="Thesis-HP-8570w-RootUSB"
Please select the device you wish to add.
0) Verbatim STORE N GO (XXXXXXXXXXXXXXXX)
1) CBM Flash Disk (YYYYYYYYYYYYYYYY)

[0-1]: 0

Which volume would you like to use for storing data ?
* Using "/dev/sdb1 (UUID: FE93-65E4)" (only option)

Name            : Thesis-HP-8570w-RootUSB
Vendor          : Verbatim
Model           : STORE N GO
Serial          : XXXXXXXXXXXXXXXX
UUID            : FE93-65E4

Save to /etc/pamusb.conf ?
[Y/n] y
Done.
root@Thesis-HP-8570w:~# pamusb-conf --add-user="root"
Which device would you like to use for authentication ?
* Using "Thesis-HP-8570w-RootUSB" (only option)

User            : root
Device          : Thesis-HP-8570w-RootUSB

Save to /etc/pamusb.conf ?
[Y/n] y
Done.
root@Thesis-HP-8570w:~# pamusb-conf --add-device="Thesis-HP-8570w-AuthUSB"
Please select the device you wish to add.
0) Verbatim STORE N GO (XXXXXXXXXXXXXXXX)
1) CBM Flash Disk (YYYYYYYYYYYYYYYY)

[0-1]: 1

Which volume would you like to use for storing data ?
* Using "/dev/sdc1 (UUID: 04F5-D2F6)" (only option)

Name            : Thesis-HP-8570w-AuthUSB
Vendor          : CBM
Model           : Flash Disk
Serial          : YYYYYYYYYYYYYYYY
UUID            : 04F5-D2F6

Save to /etc/pamusb.conf ?
[Y/n] y
Done.
root@Thesis-HP-8570w:~# pamusb-conf --add-user="henri"
Which device would you like to use for authentication ?
0) Thesis-HP-8570w-RootUSB
1) Thesis-HP-8570w-AuthUSB

[0-1]: 1

User            : henri
Device          : Thesis-HP-8570w-AuthUSB

Save to /etc/pamusb.conf ?
[Y/n] y
Done.
root@Thesis-HP-8570w:~# █
```

Figure 9. Devices and users added to USB authentication.

The pamusb module uses the vendor name, product name, serial number, an universally unique identifier (UUID) and one-time pad (OTP) for authentication. OTP is considered as the only cryptographic algorithm to provide perfect security and it can not be cracked. This is because OTP is only used once before it is destroyed or in this case, expired. OTP encrypts each character of the plaintext with a random character that is only used once. This way any message matching the length of the original message may be considered equally correct or incorrect. For example the word “WEDNESDAY” has a length of nine characters as does the words “EDUCATION”, “ASTRONAUT” and “LEGENDARY”. Without knowing the matching OTP it is impossible to prove that the decrypted plaintext is correct or incorrect and that is the reason why OTP is considered providing perfect security.[25] Matching OTP tokens are saved in the *~/.pamusb* directory and in the USB flash drive under the directory *.pamusb*.

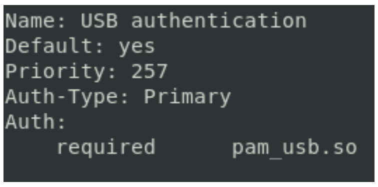
This method allows the use of only one USB flash drive per user, using multiple USB flash drives per user locks the user out of the system. The problem will likely persist to exist in the future because the current status of the project is unmaintained.[26] Additional configurations such as setting the OTP expiration after every login and making USB authentication available for laptop using GNOME were done editing the file */etc/pamusb.conf* as shown in Figure 10.



```
<?xml version="1.0" ?>
<configuration>
        <defaults>
         <option name="color_log">true</option>
         <option name="debug">true</option>
         <option name="deny_remote">true</option>
         <option name="one_time_pad">true</option>
         <option name="pad_expiration">0</option>
         <option name="hostname">Thesis-HP-8570w</option>
        </defaults>

        <devices>
         <device id="Thesis-HP-8570w-RootUSB">
         <vendor>Verbatim</vendor>
         <model>STORE N GO</model>
         <serial>XXXXXXXXXXXXXXXX</serial>
         <volume_uuid>FE93-65E4</volume_uuid>
         </device>
         <device id="Thesis-HP-8570w-UserUSB">
         <vendor>CBM</vendor>
         <model>Flash Disk</model>
         <serial>YYYYYYYYYYYYYYYY</serial>
         <volume_uuid>B440-4D6B</volume_uuid>
         </device>
        </devices>

        <users>
         <user id="root">
         <device>Thesis-HP-8570w-RootUSB</device>
         </user>
         <user id="henri">
         <device>Thesis-HP-8570w-UserUSB</device>
         </user>
        </users>

        <services>
         <service id="gdm">
          <option name="enable">true</option>
         </service>
        </services>
</configuration>
```

Figure 10. /etc/pamusb.conf configurations.

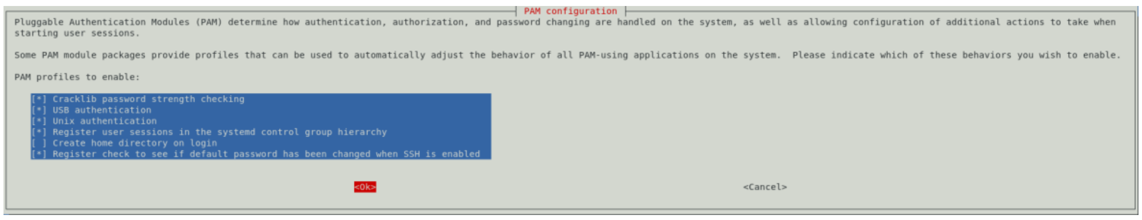
USB authentication was enforced by editing the file */usr/share/pam-configs/usb* to include setting *required* instead of the default setting *sufficient* as shown in Figure 11.



```
Name: USB authentication
Default: yes
Priority: 257
Auth-Type: Primary
Auth:
        required        pam_usb.so
```

Figure 11. /usr/share/pam-configs/usb configuration.

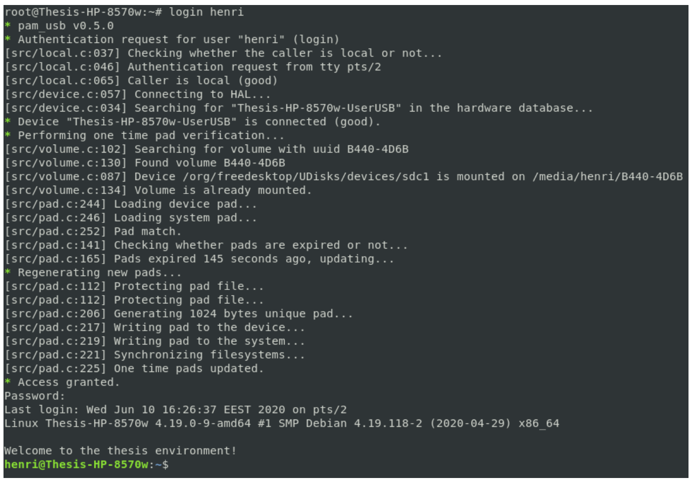
The command `pam-auth-update` was used to update the global configuration settings. Options "Cracklib password strength checking" and "USB authentication" were selected as shown in Figure 12.



```
PAM configuration
Pluggable Authentication Modules (PAM) determine how authentication, authorization, and password changing are handled on the system, as well as allowing configuration of additional actions to take when
starting user sessions.

Some PAM module packages provide profiles that can be used to automatically adjust the behavior of all PAM-using applications on the system.  Please indicate which of these behaviors you wish to enable.

PAM profiles to enable:

    [*] Cracklib password strength checking
    [*] USB authentication
    [*] Unix authentication
    [*] Register user sessions in the systemd control group hierarchy
    [ ] Create home directory on login
    [*] Register check to see if default password has been changed when SSH is enabled


                        <Ok>                                        <Cancel>
```

Figure 12. pam-auth-update configuration file.

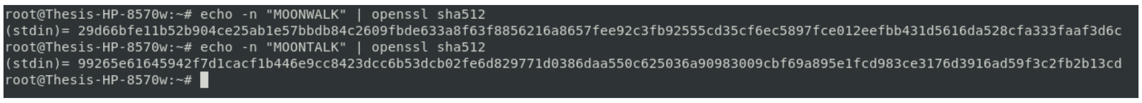
The combination of using the USB flash drive with username and a password is now required for successful login. When the user logs in to the system, PAM module checks if the correct USB flash drive is present and if the OTP in the USB flash drive matches the one saved in the system. If the match is found, user may login using the password as shown in Figure 13.



```
root@Thesis-HP-8570w:~# login henri
* pam_usb v0.5.0
* Authentication request for user "henri" (login)
[src/local.c:037] Checking whether the caller is local or not...
[src/local.c:046] Authentication request from tty pts/2
[src/local.c:065] Caller is local (good)
[src/device.c:057] Connecting to HAL...
[src/device.c:034] Searching for "Thesis-HP-8570w-UserUSB" in the hardware database...
* Device "Thesis-HP-8570w-UserUSB" is connected (good).
* Performing one time pad verification...
[src/volume.c:102] Searching for volume with uuid B440-4D6B
[src/volume.c:130] Found volume B440-4D6B
[src/volume.c:087] Device /org/freedesktop/UDisks/devices/sdc1 is mounted on /media/henri/B440-4D6B
[src/volume.c:134] Volume is already mounted.
[src/pad.c:244] Loading device pad...
[src/pad.c:246] Loading system pad...
[src/pad.c:252] Pad match.
[src/pad.c:141] Checking whether pads are expired or not...
[src/pad.c:165] Pads expired 145 seconds ago, updating...
* Regenerating new pads...
[src/pad.c:112] Protecting pad file...
[src/pad.c:112] Protecting pad file...
[src/pad.c:206] Generating 1024 bytes unique pad...
[src/pad.c:217] Writing pad to the device...
[src/pad.c:219] Writing pad to the system...
[src/pad.c:221] Synchronizing filesystems...
[src/pad.c:225] One time pads updated.
* Access granted.
Password:
Last login: Wed Jun 10 16:26:37 EEST 2020 on pts/2
Linux Thesis-HP-8570w 4.19.0-9-amd64 #1 SMP Debian 4.19.118-2 (2020-04-29) x86_64

Welcome to the thesis environment!
henri@Thesis-HP-8570w:~$
```

Figure 13. Login authentication.

Linux stores passwords in the */etc/shadow* file in a salted SHA-512 format and only root user has access to the file. SHA-512 is a one-way hash function that always produces the same result from the same given source. Seemingly the same looking plaintexts MOONWALK and MOONTALK would produce two completely different hashes that are not even remotely close to each other despite the fact the plaintexts differ only by one character as shown in Figure 14.

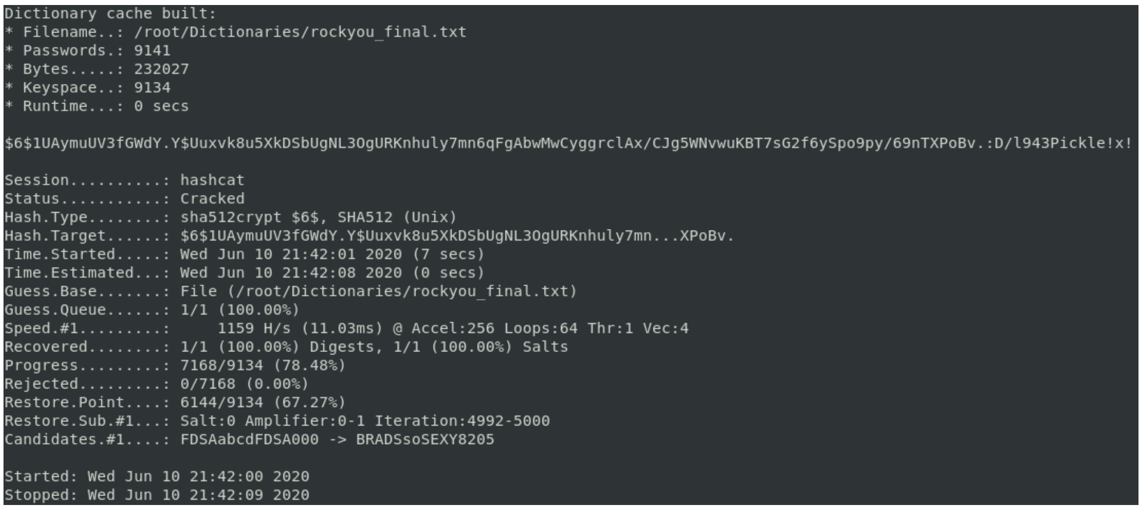


```
root@Thesis-HP-8570w:~# echo -n "MOONWALK" | openssl sha512
(stdin)= 29d66bfe11b52b904ce25ab1e57bbdb84c2609fbde633a8f63f8856216a8657fee92c3fb92555cd35cf6ec5897fce012eefbb431d5616da528cfa333faaf3d6c
root@Thesis-HP-8570w:~# echo -n "MOONTALK" | openssl sha512
(stdin)= 99265e61645942f7d1cacf1b446e9cc8423dcc6b53dcb02fe6d829771d0386daa550c625036a90983009cbf69a895e1fcd983ce3176d3916ad59f3c2fb2b13cd
root@Thesis-HP-8570w:~# 
```

Figure 14. SHA-512 hash comparison.

Hashes are mainly used to verify that the data has not been altered and is not suitable for password storage without adding salt to hash. Salt is a random data that is added to the hash function making hashes harder to crack. Adding salt to the hash makes the end result always different and the hash may not be cracked without knowing the salt that was used in the process. If a person acquires the hashed password, an offline attack may be performed. A person needs to find a matching hash value by generating hash values from the existing word lists, using rainbow tables or by using the computational brute force. When two hash values match, the correct password has been found. Using brute force with all possible combinations of 12 characters was most that the laptop used in this thesis was able to compute. Using 13 characters caused integer overflow and the program was not able to continue. Estimated time given generating the matching hash value from a word containing 12 characters was roughly 19,724 years and there were 3,813,661,310,945,771,520 or approximately 3.8 quintillion possible combinations that were possible for the password. Using brute force to crack long passwords is not computable feasible yet but that could and probably will change in the future when the quantum computers are reality.

To demonstrate the importance of not using dictionary words or breached passwords as a password in any system, a program hashcat was installed using the `apt-get install hashcat` command. Hashcat[27] is a password recovery tool which may be used to test password security by trying to crack the password. The password for user henri was changed to a breached password *D/l943Pickle!x!* and the password hash were copied from the */etc/shadow* file to a *~/hash.txt* file. The command `hashcat --force -m 1800 -a 0 hash.txt ~/Dictionaries/rockyou_final.txt` was used to crack the password hash and it took seven seconds to do that as shown in Figure 15.



```
Dictionary cache built:
* Filename..: /root/Dictionaries/rockyou_final.txt
* Passwords.: 9141
* Bytes.....: 232027
* Keyspace..: 9134
* Runtime...: 0 secs

$6$1UAymuUV3fGWdY.Y$Uuxvk8u5XkDSbUgNL3OgURKnhuly7mn6qFgAbwMwCyggrclAx/CJg5WNvwuKBT7sG2f6ySpo9py/69nTXPoBv.:D/l943Pickle!x!

Session..........: hashcat
Status...........: Cracked
Hash.Type........: sha512crypt $6$, SHA512 (Unix)
Hash.Target......: $6$1UAymuUV3fGWdY.Y$Uuxvk8u5XkDSbUgNL3OgURKnhuly7mn...XPoBv.
Time.Started.....: Wed Jun 10 21:42:01 2020 (7 secs)
Time.Estimated...: Wed Jun 10 21:42:08 2020 (0 secs)
Guess.Base.......: File (/root/Dictionaries/rockyou_final.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     1159 H/s (11.03ms) @ Accel:256 Loops:64 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 7168/9134 (78.48%)
Rejected.........: 0/7168 (0.00%)
Restore.Point....: 6144/9134 (67.27%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:4992-5000
Candidates.#1....: FDSAabcdFDSA000 -> BRADSsoSEXY8205

Started: Wed Jun 10 21:42:00 2020
Stopped: Wed Jun 10 21:42:09 2020
```

Figure 15. Hashcat result for cracked SHA-512 password hash.

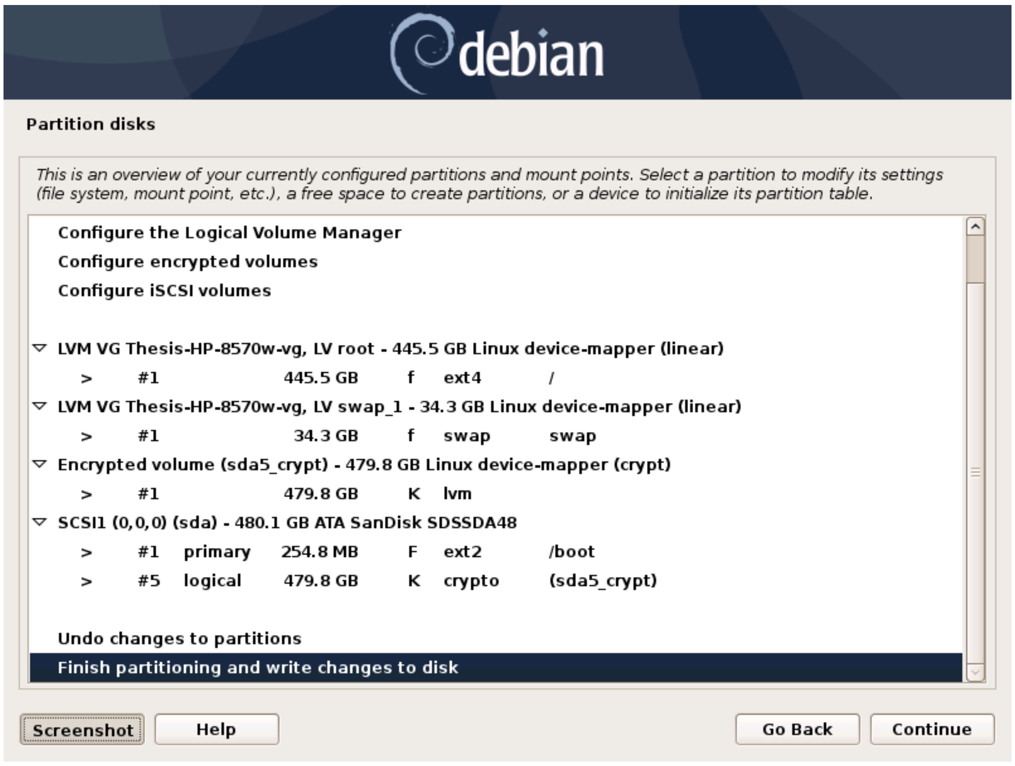
Usernames are as important to keep safe as the passwords, because without knowing the username it is impossible to log in to the system even if a person has gained the password. The username must be unique, meaning it has never been used in the Internet message boards or any other websites. By selecting a unique username, a person with malicious intentions is not able to track the user to other web services and gain more information such as e-mail address, social media account or a phone number which all may be used for social engineering.

The username is not allowed to contain the real name of the user in any form. A user Henri Koskinen can not have hkoskinen, henrik, henrkos or anything similar to an username because guessing the username becomes too easy. An e-mail address with the same username used in the system is not allowed either. Files that are published are not allowed to contain any metadata indicating the username. This thesis uses the user henri just to prove that all the work was really done by myself, it should never be used in the actual system.

6 MOBILE DEVICE POLICY

Examples of mobile devices include laptop computers, phones, tablets, smart watches, voice recorders, cameras and USB flash drives. Mobile devices are high risk devices because they can be used for espionage and for infecting devices with malware. Seventy-five percent of the population in Finland uses Internet with a mobile phone[28] which means that the possibility of an employee having a smart phone with Internet connectivity is high. Even without Internet connection the smart phones may be used to record audio or take pictures and videos. The use of any other mobile device except laptop computers and USB devices that are documented is strictly forbidden in the areas where computers and network devices are located. The use of mobile devices in restricted areas must be controlled with security camera monitoring.

The risk of losing a laptop computer is much higher than the risk of losing a desktop computer. Strong encryption prevents access to the data of the lost laptop. Debian uses the Linux Unified Key Setup (LUKS) for encryption and the implementation is performed during the installation process of the operating system. Figure 16 shows the partition table of the encrypted hard drive.

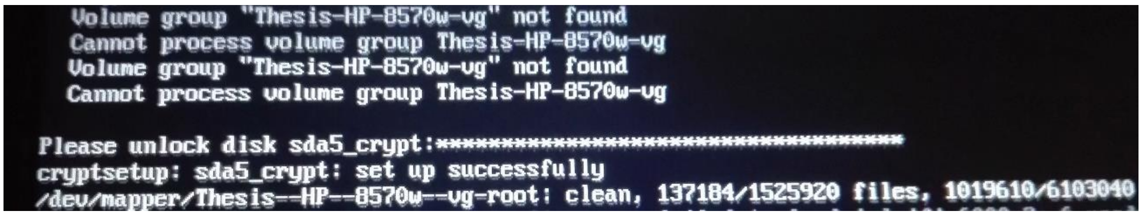


This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.

```
   Configure the Logical Volume Manager
   Configure encrypted volumes
   Configure iSCSI volumes

▽  LVM VG Thesis-HP-8570w-vg, LV root - 445.5 GB Linux device-mapper (linear)
      >     #1                445.5 GB     f    ext4         /
▽  LVM VG Thesis-HP-8570w-vg, LV swap_1 - 34.3 GB Linux device-mapper (linear)
      >     #1                 34.3 GB     f    swap         swap
▽  Encrypted volume (sda5_crypt) - 479.8 GB Linux device-mapper (crypt)
      >     #1                479.8 GB     K    lvm
▽  SCSI1 (0,0,0) (sda) - 480.1 GB ATA SanDisk SDSSDA48
      >     #1    primary     254.8 MB     F    ext2         /boot
      >     #5    logical     479.8 GB     K    crypto       (sda5_crypt)

   Undo changes to partitions
   Finish partitioning and write changes to disk
```

Figure 16. Encrypted partition table for Debian 10 Buster.

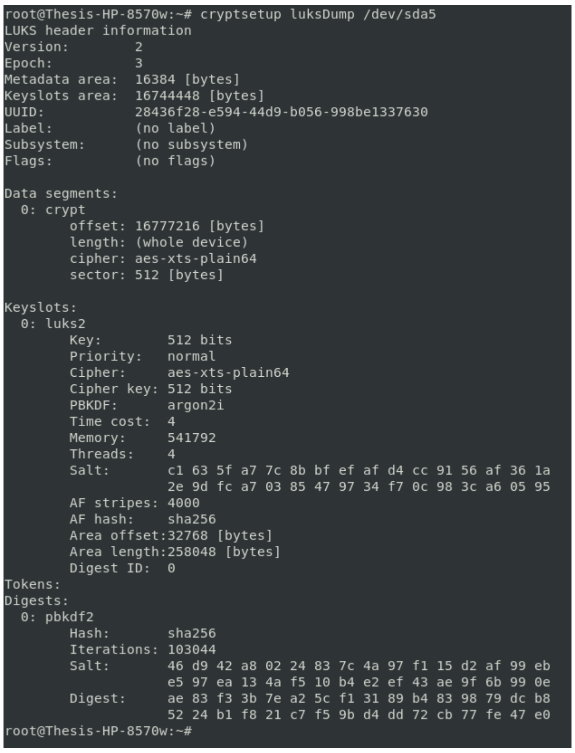
Encryption needs a passphrase which is used to access the encrypted hard drive. The passphrase should be at least 20 characters long but the length is not restricted in any way. During the startup sequence the user is prompted to enter the passphrase in order to gain access to the system. The same principles should be followed selecting the password for encryption than selecting the password for a user. Without knowing the passphrase the user is not able to access the hard drive. Passphrase prompt is shown in Figure 17.



```
    Volume group "Thesis-HP-8570w-vg" not found
    Cannot process volume group Thesis-HP-8570w-vg
    Volume group "Thesis-HP-8570w-vg" not found
    Cannot process volume group Thesis-HP-8570w-vg

  Please unlock disk sda5_crypt:*************************************
  cryptsetup: sda5_crypt: set up successfully
  /dev/mapper/Thesis--HP--8570w--vg-root: clean, 137184/1525920 files, 1019610/6103040
```

Figure 17. Startup prompt asking for a passphrase.

By default, LUKS uses AES-xts-plain64 for encryption and SHA256 for hashing. Algorithms are NIST approved so the settings were left as they were but it is possible to change the algorithms using the `cryptsetup-reencrypt` command. Encryption information of the hard drive is shown in Figure 18.



```
root@Thesis-HP-8570w:~# cryptsetup luksDump /dev/sda5
LUKS header information
Version:        2
Epoch:          3
Metadata area:  16384 [bytes]
Keyslots area:  16744448 [bytes]
UUID:           28436f28-e594-44d9-b056-998be1337630
Label:          (no label)
Subsystem:      (no subsystem)
Flags:          (no flags)

Data segments:
  0: crypt
        offset: 16777216 [bytes]
        length: (whole device)
        cipher: aes-xts-plain64
        sector: 512 [bytes]

Keyslots:
  0: luks2
        Key:        512 bits
        Priority:   normal
        Cipher:     aes-xts-plain64
        Cipher key: 512 bits
        PBKDF:      argon2i
        Time cost:  4
        Memory:     541792
        Threads:    4
        Salt:       c1 63 5f a7 7c 8b bf ef af d4 cc 91 56 af 36 1a
                    2e 9d fc a7 03 85 47 97 34 f7 0c 98 3c a6 05 95
        AF stripes: 4000
        AF hash:    sha256
        Area offset:32768 [bytes]
        Area length:258048 [bytes]
        Digest ID:  0
Tokens:
Digests:
  0: pbkdf2
        Hash:       sha256
        Iterations: 103044
        Salt:       46 d9 42 a8 02 24 83 7c 4a 97 f1 15 d2 af 99 eb
                    e5 97 ea 13 4a f5 10 b4 e2 ef 43 ae 9f 6b 99 0e
        Digest:     ae 83 f3 3b 7e a2 5c f1 31 89 b4 83 98 79 dc b8
                    52 24 b1 f8 21 c7 f5 9b d4 dd 72 cb 77 fe 47 e0
root@Thesis-HP-8570w:~#
```

Figure 18. Hard drive encryption information.

7 DEVICE SECURITY

Raspberry Pi is the size of a credit card measuring 85.60mm x 56mm x 21mm.[29] Small size makes it a perfect target for theft. A lockable safe boxes were used to keep Raspberry Pi devices physically safe. Turning off the device is not always possible and they may be not always be supervised by a person, so an automated safety procedures were implemented. A prototype of a safe box is shown in Figure 19.

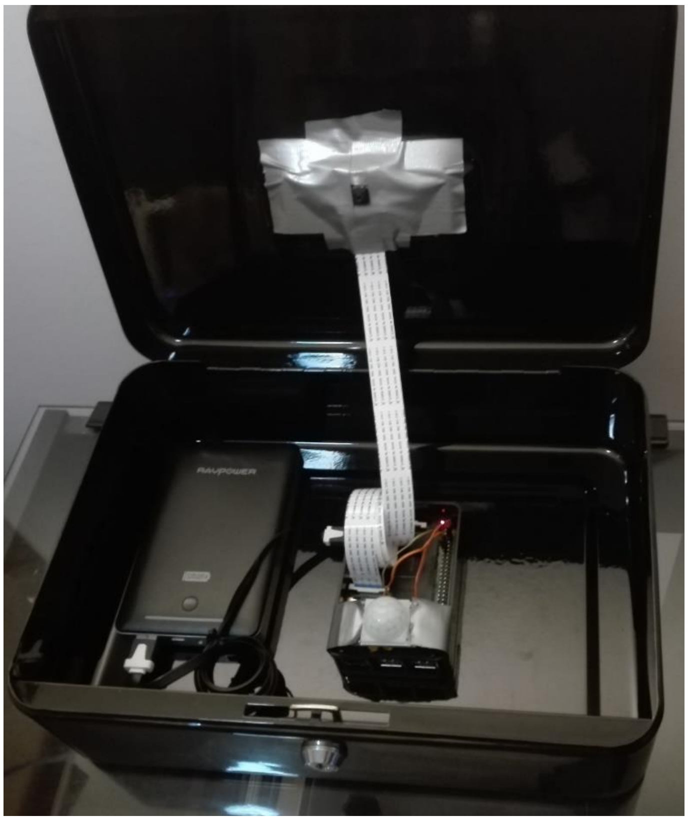
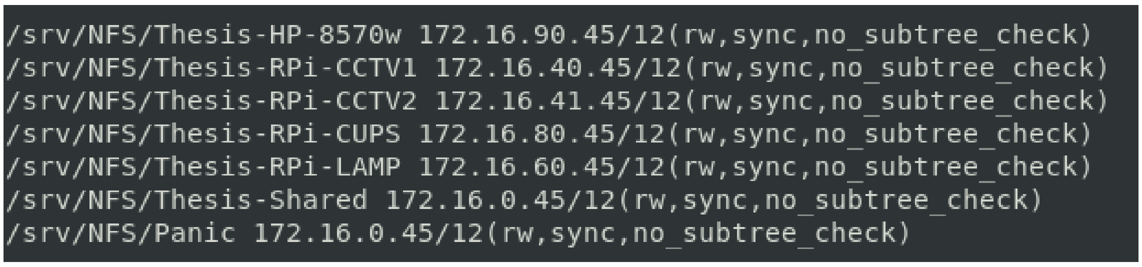


Figure 19. A prototype of a safe box.

Raspberry Pi is connected to the power bank which gives enough power for 24 hours until the device must be turned off and use another power source. An official Raspberry Pi camera module and a passive infrared (PIR) sensor D-Sun RCW-0506 are connected to the Raspberry Pi. When the intruder breaks the lock and opens the box, there is a change to capture a close-up footage of the intruder. The metal box blocks all wireless signals so the only option for network connectivity is to drill a hole in the bottom of the box for the ethernet cable. The hole should be big enough for the cable but small enough for the ethernet connector. Physical lockable port blockers should be used when available because they prevent infecting the device with malicious code or worse. A safe box is not meant to replace real security systems or alarms, its main purpose is to gain enough time for the devices to copy the most important files to a safe location.

A network file system (NFS) allows users to share data over the network. A Raspberry Pi device with a hostname of Thesis-RPi-NFS was used as a NFS server. NFS server[30] was installed on Raspberry Pi using the `apt-get install nfs-kernel-server` command and the configuration was done by editing the */etc/exports* file. The configuration file specifies the mount point of the shared directory and an IP address of the device that is allowed to access the shared directory as shown in Figure 20.



```
/srv/NFS/Thesis-HP-8570w 172.16.90.45/12(rw,sync,no_subtree_check)
/srv/NFS/Thesis-RPi-CCTV1 172.16.40.45/12(rw,sync,no_subtree_check)
/srv/NFS/Thesis-RPi-CCTV2 172.16.41.45/12(rw,sync,no_subtree_check)
/srv/NFS/Thesis-RPi-CUPS 172.16.80.45/12(rw,sync,no_subtree_check)
/srv/NFS/Thesis-RPi-LAMP 172.16.60.45/12(rw,sync,no_subtree_check)
/srv/NFS/Thesis-Shared 172.16.0.45/12(rw,sync,no_subtree_check)
/srv/NFS/Panic 172.16.0.45/12(rw,sync,no_subtree_check)
```

Figure 20. NFS server configuration file /etc/exports.

NFS server setting changes were made effective using the `exportfs -rav` command. Client devices need to mount the NFS shares in order to access them. The program nfs-common[31] was used for this purpose and it was installed using the `apt-get install nfs-common` command. A directory */srv/NFS* was decided to be used as a mount point for the NFS shares in all devices. Directories must exist before the mounting so they were created as shown in Figure 20 above. A shared directory */srv/NFS/Panic* was mounted using the `mount -t nfs 172.16.70.55:/srv/NFS/Panic /srv/NFS/Panic` command.

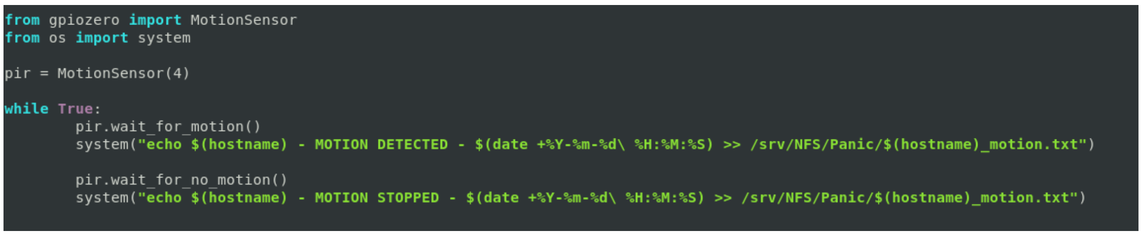
The national target police response time in Finland is ten minutes and the target time is achieved on 80 percent of the cases.[32] Ten minutes is a long time and the actual security systems may not always work. Two Raspberry Pis were used as a motion detecting security cameras leaving the devices in the safe boxes enough time to react to the intruder. Security cameras were given hostnames Thesis-RPi-CCTV1 and Thesis-RPi-CCTV2 and placed in the opposite directions of the room.

The local area network (LAN) consists of five Raspberry Pis and one laptop computer. Three different python scripts were created, one for the CCTV1 and CCTV2, one for the NFS server and one for all the remaining devices.

The process has the following steps:

- Thesis-RPi-CCTV1 or Thesis-RPi-CCTV2 detects motion and creates a text file in the */srv/NFS/Panic* directory.
- Other devices in the network acknowledge the incident by adding a log entry.
- Thesis-RPi-NFS starts to play the sound of a barking dog. Because there are over 700,000 dogs in Finland and approximately two dog attacks per day, the threat of a dog attack is real.[33, 34]
- Another CCTV detects motion also and creates a text file in the */srv/NFS/Panic* directory.
- Other devices in the network acknowledge the incident by adding a log entry.
- NFS server acknowledges that Thesis-RPi-CCTV1 and Thesis-RPi-CCTV2 both detected motion.
- NFS server creates a file which informs the devices to start the panic mode.
- Devices start transferring the backup files to the NFS server. After the transfer is done, files are removed from the local device and the device is turned off.
- After the time limit of five minutes, the NFS server is turned off.

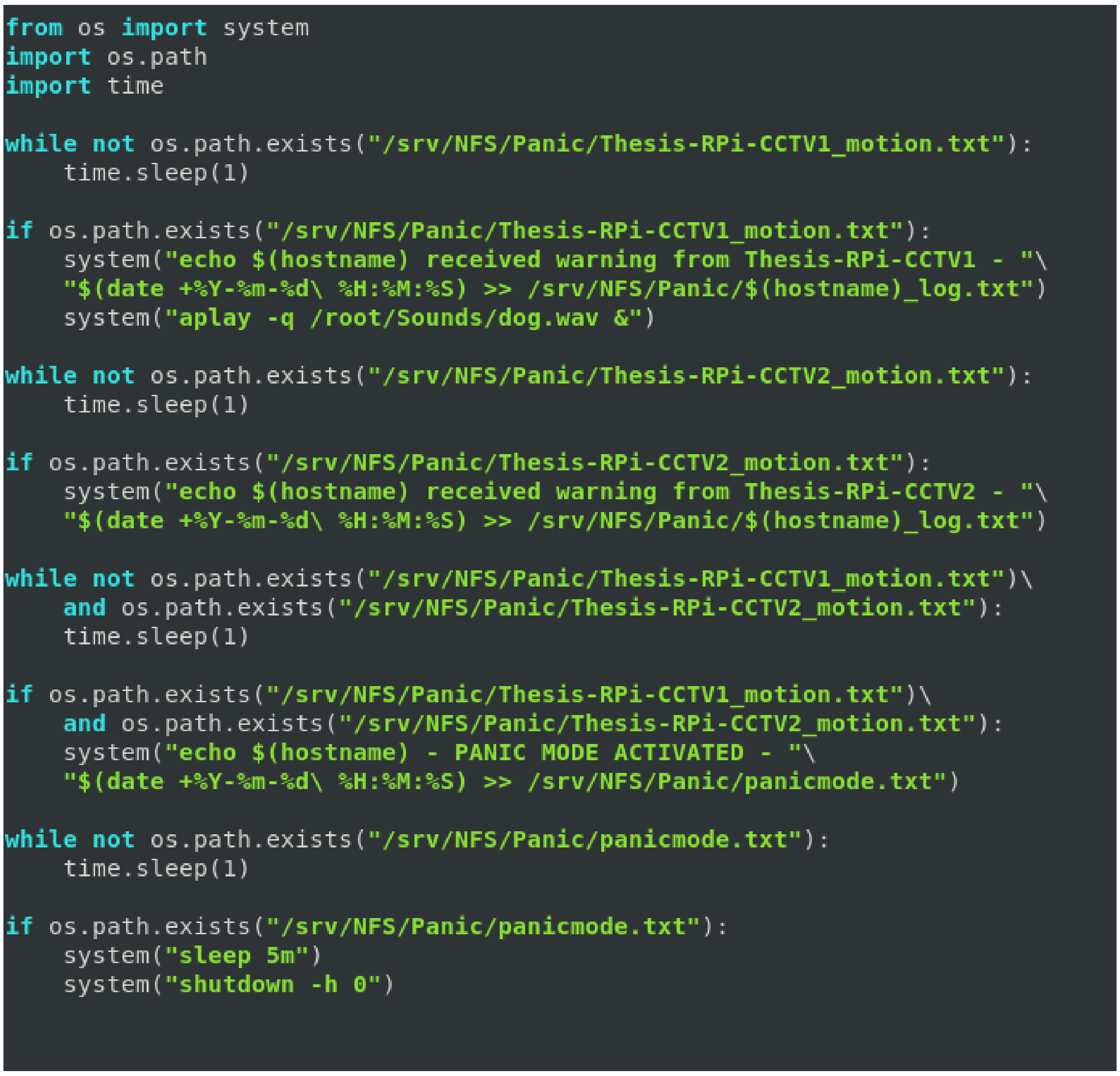
Motion detection part of the script for the CCTV1 and CCTV2 is shown in Figure 21.



```
from gpiozero import MotionSensor
from os import system

pir = MotionSensor(4)

while True:
        pir.wait_for_motion()
        system("echo $(hostname) - MOTION DETECTED - $(date +%Y-%m-%d\ %H:%M:%S) >> /srv/NFS/Panic/$(hostname)_motion.txt")

        pir.wait_for_no_motion()
        system("echo $(hostname) - MOTION STOPPED - $(date +%Y-%m-%d\ %H:%M:%S) >> /srv/NFS/Panic/$(hostname)_motion.txt")
```

Figure 21. Motion detection script snippet.

The script for the Thesis-RPi-NFS server is the most critical of them all because it saves all the files. The python script for the NFS server is shown in its complete form in Figure 22.



```
from os import system
import os.path
import time

while not os.path.exists("/srv/NFS/Panic/Thesis-RPi-CCTV1_motion.txt"):
    time.sleep(1)

if os.path.exists("/srv/NFS/Panic/Thesis-RPi-CCTV1_motion.txt"):
    system("echo $(hostname) received warning from Thesis-RPi-CCTV1 - "\
    "$(date +%Y-%m-%d\ %H:%M:%S) >> /srv/NFS/Panic/$(hostname)_log.txt")
    system("aplay -q /root/Sounds/dog.wav &")

while not os.path.exists("/srv/NFS/Panic/Thesis-RPi-CCTV2_motion.txt"):
    time.sleep(1)

if os.path.exists("/srv/NFS/Panic/Thesis-RPi-CCTV2_motion.txt"):
    system("echo $(hostname) received warning from Thesis-RPi-CCTV2 - "\
    "$(date +%Y-%m-%d\ %H:%M:%S) >> /srv/NFS/Panic/$(hostname)_log.txt")

while not os.path.exists("/srv/NFS/Panic/Thesis-RPi-CCTV1_motion.txt")\
    and os.path.exists("/srv/NFS/Panic/Thesis-RPi-CCTV2_motion.txt"):
    time.sleep(1)

if os.path.exists("/srv/NFS/Panic/Thesis-RPi-CCTV1_motion.txt")\
    and os.path.exists("/srv/NFS/Panic/Thesis-RPi-CCTV2_motion.txt"):
    system("echo $(hostname) - PANIC MODE ACTIVATED - "\
    "$(date +%Y-%m-%d\ %H:%M:%S) >> /srv/NFS/Panic/panicmode.txt")

while not os.path.exists("/srv/NFS/Panic/panicmode.txt"):
    time.sleep(1)

if os.path.exists("/srv/NFS/Panic/panicmode.txt"):
    system("sleep 5m")
    system("shutdown -h 0")
```

Figure 22. A security script for a NFS server.

The third script for the other devices may be created using the clues from the past scripts created in this thesis. To avoid repetition, the third script is left to readers' imagination. These scripts are not perfect but they give the basic idea what can be done.

8 CONCLUSION

The field of cyber security is facing constant challenges as new technologies emerge and manufacturers are thinking more about the usability of the product than the security aspects. The products are easy to use but the default settings are not secure enough. Quantum computers are closer to become reality and every major organization is preparing for that change. The quantum computers will make most of the currently used cryptographic algorithms obsolete because they are based on an assumption that not enough computational power in a sensible time period is available. Hackers and crackers keep on finding the new ways to attack the systems so every organization should get the attitude that someone will eventually break in to the systems. The best way to prevent attacks is to make breaking in to systems as time-consuming as possible.

The goal of this thesis was to secure devices using Linux operating system. The approach that was used to achieve security was to make logging in to the system as hard as possible. The biggest problem with security is that the default configs are nonsense. Raspberry Pi turned out to be an excellent IoT-device which is capable of even demanding tasks.

Computers are getting smaller and based on a scale what computers used to be in the sixties, it would not be surprising if they are no bigger than hair twenty years from now and include quantum properties. When people talk about security, they mean the sense of security because there will never be absolute security. Even offline systems may be compromised no matter how much effort and money is spent preventing that from happening. Security automation should be used as much as possible to prevent user errors and minimize user interaction.

REFERENCES

[1] Cisco | Cisco Annual Internet Report (2018-2023) White Paper. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html [May 25, 2020]

[2] Traficom | Finland becomes the first European country to certify safe smart devices - new Cybersecurity label helps consumers buy safer products. Available: https://www.traficom.fi/en/news/finland-becomes-first-european-country-certify-safe-smart-devices-new-cybersecurity-label [May 29, 2020]

[3] European Telecommunications Standards Institute | Final draft ETSI EN 303 645 V2.1.0 (2020-04) Cyber Security for Consumer Internet of Things: Baseline requirements. Available: https://www.etsi.org/deliver/etsi_en/303600_303699/303645/02.01.00_30/en_303645v020100v.pdf [May 29, 2020]

[4] Ministry of the Interior (Finland) | Cybercrime. Available: https://intermin.fi/en/police/cybercrime [May 22, 2020]

[5] NCCoE | Is my DVR being used in a cyber attack? Available: https://www.nccoe.nist.gov/news/my-dvr-being-used-cyber-attack [May 28, 2020]

[6] ISO | About us. Available: https://www.iso.org/about-us.html [June 7, 2020]

[7] NIST | About NIST. Available: https://www.nist.gov/about-nist [May 27, 2020]

[8] NIST | Compliance FAQs: Federal Information Processing Standard (FIPS). Available: https://www.nist.gov/standardsgov/compliance-faqs-federal-information-processing-standards-fips [May 27, 2020]

[9] NCSC-FI | Homepage. Available: https://www.kyberturvallisuuskeskus.fi/en [May 28, 2020]

[10] Graz University of Technology | Meltdown and Spectre. Available: https://meltdownattack.com [June 7, 2020]

[11] Debian | Details of package cron in buster. Available: https://packages.debian.org/buster/cron [June 7, 2020]

[12] Sudo | Sudo Manual. Available: https://www.sudo.ws/man/1.8.3/sudo.man.html [May 25, 2020]

[13] CVE | Search results for sudo. Available: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=sudo [May 27, 2020]

[14] GNU | Aliases (Bash Reference Manual). Available: https://www.gnu.org/software/bash/manual/html_node/Aliases.html [June 10, 2020]

[15] Raspberry Pi Documentation | Linux users. Available: https://www.raspberrypi.org/documentation/linux/usage/users.md [June 10, 2020]

[16] NIST Special Publication 800-63B - Digital Identity Guidelines - Authentication and Lifecycle Management. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63b.pdf [May 28, 2020]

[17] NCSC-FI | Passwords – Advice on using and managing passwords. Available: https://www.kyberturvallisuuskeskus.fi/sites/default/files/media/file/Salasanat_haltuun.pdf [May 28, 2020]

[18] Debian | Details of package libpam-cracklib in buster. Available: https://packages.debian.org/buster/libpam-cracklib [June 2, 2020]

[19] Institute for the Languages of Finland | Modern Finnish wordlist. Available: http://kaino.kotus.fi/sanat/nykysuomi/ [May 30, 2020]

[20] Debian | Details of package xml-twig-tools in buster. Available: https://packages.debian.org/buster/xml-twig-tools [May 30, 2020]

[21] GitLab | Wordlists packaking for Kali Linux. Available: https://gitlab.com/kalilinux/packages/wordlists [May 28, 2020]

[22] Debian | Details of package gawk in buster. Available: https://packages.debian.org/buster/gawk [June 2, 2020]

[23] Debian | Details of package libpam-usb in jessie. Available: https://packages.debian.org/jessie/libpam-usb [May 30, 2020]

[24] PAM_USB | Quickstart. Available: http://www.pamusb.org/doc/quickstart.html [May 30, 2020]

[25] Crypto Museum | One-Time Pad. Available: https://www.cryptomuseum.com/crypto/otp/index.htm [June 5, 2020]

[26] GitHub | aluzzardi/pam_usb: [UNMAINTAINED] Hardware authentication for Linux using ordinary USB Flash Drives. Available: https://github.com/aluzzardi/pam_usb [May 30, 2020]

[27] Debian | Details of package hashcat in buster. Available: https://packages.debian.org/buster/hashcat [June 2, 2020]

[28] Statistics Finland | Official Statistics of Finland (OSF): Use of information and communications technology by individuals. Available: https://tilastokeskus.fi/til/sutivi/2018/sutivi_2018_2018-12-04_tie_001_en.html [May 22, 2020]

[29] Raspberry Pi Foundation | FAQs. Available: https://www.raspberrypi.org/documentation/faqs/ [June 14, 2020]

[30] Debian | Details of package nfs-kernel-server in buster. Available: https://packages.debian.org/buster/nfs-kernel-server [June 14, 2020]

[31] Debian | Details of package nfs-common in buster. Available: https://packages.debian.org/buster/nfs-common [June 14, 2020]

[32] Yle | Police response times up to an hour slower in rural areas. Available: https://yle.fi/uutiset/osasto/news/police_response_times_up_to_an_hour_slower_in_rural_areas/11108792 [June 14, 2020]

[33] Statistics Finland | Official Statistics of Finland (OSF): Households' consumption 2016. Available: https://www.stat.fi/til/ktutk/2016/ktutk_2016_2020-04-20_tie_001_en.html [June 14, 2020]

[34] Yle | Dog attacks increasing in Finland, police and insurance firms confirm. Available: https://yle.fi/uutiset/osasto/news/dog_attacks_increasing_in_finland_police_and_insurance_firms_confirm/10361439 [June 14, 2020]