KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Communication Systems

Second cycle, 30 credits

Ethical Hacking of a Smart IoT Camera

A Penetration Test on D-Link DCS 8515-LH Smart Camera

CHUNYU ZHUANG

Stockholm, Sweden, 2023

Ethical Hacking of a Smart IoT Camera

A Penetration Test on D-Link DCS 8515-LH Smart Camera

CHUNYU ZHUANG

Master's Programme, Communication Systems, 120 credits
Date: March 29, 2023

Supervisor: Carlos Barreto
Examiner: Carlo Fischione
School of Electrical Engineering and Computer Science
Swedish title: Etisk hackning av en smart IoT-Kamera
Swedish subtitle: Ett Penetrationstest på D-Link DCS 8515-LH Smart Kamera

Abstract

The trending usage of IoT devices raises serious security concerns. IoT devices have complete access to users' network environments. In the eyes of hackers, the value of IoT devices is exceptionally high. From minor disturbances to major crimes, all could happen in no time with compromised IoT devices. As the IoT devices collects sensitive data, properly protect users' privacy is also a crucial aspect for IoT devices. Thus, IoT devices need to be secure enough against modern cyber-attacks.

In this work, a smart camera DCS-8515LH from D-Link is under penetration tests. Threat modeling is first performed as an analysis of the IoT system following by a dozen cyber attacks targeting this smart camera. The penetration tests provide valuable information that can reveal the smart camera's vulnerability and weakness, such as security misconfiguration, vulnerability to DoS attacks. The smart camera is discovered to be vulnerable to DoS attacks and exploits on the zero-configuration protocol. Several weaknesses which violate the users' privacy exist in the mobile application and Android storage system. This work evaluated all the vulnerabilities and weaknesses discovered from a security aspect. This report exposes attacks that are effective on the smart camera and also serves as a fundamental basis for future penetration tests on this smart camera.

Keywords

Cybersecurity, IoT security, IoT camera, IoT devices, Penetration testing, Ethical hacking, Threat modeling

Sammanfattning

I detta arbete är en smart kamera DCS-8515LH från D-Link under penetrationstester. Hotmodellering utförs först som en analys av IoT-systemet följt av ett dussin cyberattacker riktade mot denna smarta kamera.
Penetrationstesterna ger värdefull information som kan avslöja den smarta kamerans sårbarhet och svaghet, såsom säkerhetsfelkonfiguration, sårbarhet för Dos-attacker. Den smarta kameran har upptäckts vara sårbar för DoS-attacker och utnyttjande av nollkonfigurationsprotokollet. Flera svagheter som kränker användarnas integritet finns i mobilapplikationen och Android-lagringssystemet. Detta arbete utvärderade alla sårbarheter och svagheter som upptäckts ur en säkerhetsaspekt. Den här rapporten avslöjar attacker som är effektiva på den smarta kameran och fungerar också som en grundläggande bas för framtida penetrationstester på denna smarta kamera.

Nyckelord

Cybersäkerhet, IoT säkerhet, IoT Kamera, Penetrationstestning, Etisk hacking, Hotmodellering

Acknowledgments

I would like to take this acknowledgement to thank my supervisor Dr. Carlos Barreto for the great help and support on this thesis, and Professor Carlo Fischione for his flexibility during the execution of this work.

Stockholm, Sweden, March 2023
Chunyu Zhuang

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The rise of technology and Internet of Things (IoT) has brought a new level of convenience to our daily lives. IoTs has been adapted to various industries such as healthcare, manufacturing, retail and energy. An interesting prediction that the population of IoT will be reaching 125 billion in the next ten years [1] is likely to come true if there's no breakdown in the technology ecosystems. The development of internet protocols, wireless communication technologies and embedded system devices improves the communication, processing and sensing capabilities of the smart devices [2]. However, the widespread adoption of IoT devices also raises concerns about privacy and security, as connected devices transmit sensitive data which can lead to breaches or cyber attacks.

As many conveniences as the IoTs can provide, the security concerns in IoTs remain an unsolved challenge and are highly likely to be a critical battlefield between developers and hackers (ethical or not). The security concerns of IoT devices are two-fold: the robustness of the device against attacks to take control, and preservation of privacy against attacks that assessing sensitive information. IoTs collect and transmit countless data from the users, which makes them highly valuable to vicious attackers. The vulnerabilities will lead to severe consequences such as financial losses and reputation damage, followed by loss of private data, loss of control over multiple devices, loss of money or other values, and most importantly, safety incidents. According to [3], only 2% of the data traffic flowing through the internet is encrypted, leaving a tremendous amount of data unprotected. Smart cameras are particularly of great importance since it reveals live footage of actual locations and people's daily privacy. An example of cyber attacks on live footage is that 73011 unsecured cameras in 256 countries was once revealed on a public

available website [4].

Due to the above concerns about the robustness of IoT devices, this thesis conducts penetration testing on the smart camera mydlink DCS-8515LH. The author performs ethical hacking on the IoT device, analyzes the security level from the exploiting results, evaluates the resilience level of the camera, and provides constructive insights to increase security levels and better protect users' privacy.

From the author's investigation, no public available security evaluation report is available about D-Link DCS-8515LH smart camera. This work combines both classic and newly developed methods to penetration test the smart camera. The main contributions are three-fold. First, this work use threat modeling to present the structure and traffic maps of the smart camera. Second, this work performed penetration tests from an overall perspective, including network inspecting, decompilation of Android APK, traffic analysis, zero-conf protocol exploitation, system storage investigation, RTSP exploit, identification and authentication failure, MiTM attack, WiFi direct attack, DoS attack, Firmware reverse engineering. Third, this work evaluate the results from the penetration tests and analysis the security level and configuration. At the same time, provide constructive suggestions on security improvement.

1.1 Background

An IoT device has the ability to transfer data and communicate over a network without human interception. One of the first IoT devices Coca Colawas a vending machine at Carnegie Mellon University which was the first ARPANET-connected appliance, able to reporting inventory and sense whether the newly loaded drinks were cold enough [5]. After almost 40 years, it is expected that in 2022, the IoT devices in the world will reach 16.4 billion while taking 61.9% of total devices [6]. Moreover, the leading companies are investing massively in the IoT to satisfy the market's high demands. For example, IBM announced a 3 billion business unit focusing on the development of solutions for the IoT [7].

While IoT devices are increasing in popularity, security and privacy issues remain challenging for developers. Usually security measures are not taken promptly compared to the security vulnerabilities of IoT devices. Due to the number of IoT devices increasing in the foreseeable future, the vulnerabilities increase naturally [8]. For instance, a software security vulnerability present in TRENDnet's IP-connected cameras was exploited in 2021, and hackers posted links to the private live feeds of nearly 700 of the cameras [9]. Furthermore, in

2014, over 73000 video cameras were found to be streaming their surveillance footage on the web [10]. The scope of caused damage will expand not only to homes but also to offices and enterprises. Malicious attackers can easily take advantage of network and radio frequency to gain unauthorized access causing severe consequences such as violating users' privacy and retrieve sensitive data. Ethical hacking and penetration testing are two effective methods to access the vulnerabilities and evaluate the IoT devices security levels.

1.2 Problem

The work aims to address the issue of the security evaluation on the smart IoT camera and test the robustness of the smart camera mydlink DCS-8515LH. The security camera claims to "make your home safer" using motion detecting and live-streaming on the phone app. Nevertheless, will it really make the home safer or provide potential security concerns? The thesis will investigate possible attacks on the smart camera and perform ethical hacking on the device. Then, shall the attacks perform successfully, the consequences will also be investigated. Last but not least, I will carry out an overall evaluation of the smart camera's security level and exploitable weakness.

1.3 Purpose

The purpose of the thesis includes several aspects. First of all, the author discover the vulnerabilities in the smart camera and evaluate the severity with a security analysis. Second, the evaluation leads to some constructive suggestions about the camera security issues, which will benefit all the users and developers. Last but not least, the work will further expand the author's skills and comprehension of IoT ethical hacking subject.

1.4 Goals

The goal of the thesis is to evaluate the security level of the mydlink HD Pan & Tilt Wi-Fi Camera DCS-8515LH. While this has been divided into the following sub-goals:

1. To discover the mydlink android application, the wireless communication, and the firmware on the camera have any vulnerabilities.

2. There are potential vulnerabilities, exploit the vulnerabilities discovered.
3. To evaluate the damage the exploited vulnerabilities could cause. If there is no vulnerability found, analyze the protection configured within the scope.
4. Provide constructive suggestions in order to improve the overall security level of the tested camera.

1.5 Research methodology

The ethical hacking process includes exploiting the vulnerabilities through attacks such as authentication bypass, Man-in-the-middle (MiTM) attacks. Then, hypothetically the ethical hacker will take advantage of the exploited vulnerabilities and get access or even root access to the system. In this work, we take several methods and techniques into consideration. First, a literature review will be performed to develop an in-depth insight into the research field in the task. Second, PTES [11] and STRIDE [12] will be used to identify potential threats in the system. The methods include but are not limited to threat modeling, ethical attacking, vulnerability analysis, and suggestion of mitigation.

1.6 Delimitations

This work will mainly perform penetration tests on the smart camera, the firmware of the camera, the Android mobile application, and the wireless communication from the camera. Hardware, IoS application, radio frequency, cloud server and Bluetooth are outside this work's scope due to the time limitation.

1.7 Structure of the thesis

Chapter 2 presents related background information on IoT device penetration testing. Chapter 3 details the methodology applied in this work. Chapter 4 provides an overview of the smart camera system and describes the threat modeling of the smart camera. Chapter 5 is the penetration tests done on the target. Chapter 6 discusses the overall work done and results gained. Chapter 7 concludes this work and state the possible future extension of this work.

Chapter 2

Background

This section will provide the necessary background knowledge needed to understand the thesis, prior related work that authors find insights about this subject, and the motivation of this work.

2.1 IoT security in the modern era

Underlying the undeniable convenience from IoT devices, there are security concerns. Developers and manufacturers tend to ignore or look down on security considerations in order to save time or lower costs. The negligence of security considerations enables the exposure of information, ranging from unprotected video streams to recordings and passwords of the IoTs [13]. These cyber incidents will endanger not only users' privacy but also the reputation of the publishing companies. It is also recommended for users to check on the security level before purchasing an IoT device [14]. The unprotected IoT devices owned by users needs cyber operations to prevent unforeseen attacks from happening.

2.1.1 Smart WiFi camera security

Modern smart cameras appear as different components, such as monitor devices, home surveillance cameras, and even a small part of home automation systems. These systems can leak out private information or even allow attackers to spy on the camera set scenarios. Moreover, the compromised cameras leads to more serious consequences such as break-in, robbery without the recordings of suspects.

2.1.2 D-Link cameras vulnerabilities found and fixed

A serious issue was found in the D-Link DCS-2132L cloud camera, the unencrypted transmission of the video stream. The transmission was unencrypted in the camera to the cloud and cloud to the client-side app. This exposure provides great possibilities for intruders to perform man-in-the-middle (MiTM) attacks and spy on the victims' video streams. The intruders use the TCP connection data stream on the server port (2018) to view the HTTP requests, which could be replayed and reconstructed by the intruders to intercept audio and video streams from the cameras [15].

2.2 Identify and evaluate the Vulnerabilities in smart cameras

Common vulnerability management includes four stages: Vulnerability identification, vulnerability analysis, vulnerability response planning, and vulnerability controlling [16]. The first two stages are taken into consideration in this thesis. The first stage uses vulnerability databases as a guideline, combining vulnerability scanners to identify affected components. During this stage, the system's assets are specified to guide the analysis further [17]. The second stage is evaluation, prioritising efforts to exploit and analyse. The Common Vulnerability Scoring System (CVSS) is a public industry standard for evaluating the severity of security vulnerabilities. The use of CVSS allows prioritising vulnerabilities on the score calculated by several metrics on impact and possibilities [18].

2.3 Related work

This section will introduce some of the prior research on smart camera security and penetration testing cases on which the author finds value and inspiration.

2.3.1 Vulnerabilities in camera firmware exploit

In 2013, Craig Heffner successfully exploited 0-day vulnerabilities by remotely gained administrative and even root access to more than 50 users, and professional surveillance camera models in various scenarios [19]. The attack was initiated by analysing and extracting firmware source codes with

dissemblance, emulation and debugging. The vulnerabilities that lie within the device system were identified and then exploited.

2.3.2 Exposed vulnerabilities in the implementation

Yogeesh et al. [20] identified unencrypted data transmission during traffic capturing using Wireshark. All the data on the camera under test are sent without encryption, revealing the WiFi credentials in clear text. The credentials were obtained in the captured packets, which would further lead to severe consequences to the users' home network. In the mentioned paper, the researchers also tried to brute force the URLs for RTSP protocol to stream videos, through which the users' email and stored password were spotted as well. The poor configuration and implementation lowered the security level of the camera itself and, consequentially, the network.

2.3.3 Bypassing authentication with URL

Attacks on cameras from the company Swann are proved to be successful on broken authentication attacks in [21]. The attacker in this work discovers a URL path to steal the live stream from Swann surveillance systems via telnet with a root password in the firmware. Through RTSP services in ports 554 and 6001 the attacker bypass the authentication mechanism and successfully get a view of the stream. This unauthenticated URL usage could expose the live stream footage to attackers and even be exposed on the internet.

2.3.4 Denial of service attacks via ARP poisoning

In [22], the researchers performed an ARP poisoning attack by sending poisoned ARP packets to the cameras. The devices fail to detect with cache. All camera systems' video streaming in consideration is interrupted and blocked. IP cameras poor resilience to MiTM attacks left possibilities for malicious intruders to utilize the ignorance of checking the ARP cache on the network before gaining the trust of the new entries. Denial of service (DoS) attacks could keep the users from accessing the smart cameras and cause potentially serious impacts such as break in and robbery in certain scenarios, especially for home security cameras.

2.3.5 Exploitation targeting the RTSP-server

A vulnerability in code execution was discovered on the LIVE555 RTSP server library by Cisco Talos [23]. A crafted packet exploits the vulnerability, leading to a buffer overflow, resulting in code execution. The LIVE555 Media libraries are used widely on streaming protocols like RTSP, and the open port supports the client's communication through the RTSP tunnel through HTTP. Then the attacker could successfully pull char arrays with some request string function. Afterwards, the break function in the array was found to be vulnerable and caused the code to go back to the loop resulting in an overflow.

2.3.6 DoS attacks WiFi direct

WiFi Direct is a service used by lots of IoT devices that can establish a secure and stable network. [24] addresses the attack impacts on Android-based devices where the attacker could block the connection between the devices that are legitimate and the network environment. The attacks in this work brought serious impacts on the experimental setup when the users were forced to quit, the target device lost connection, or the server dropped all the connections indiscriminately. Later on, this research was investigated by [25], where the researcher proposed using Hotspot to avoid such vulnerabilities. Hotspot security is shown to be better than WiFi P2P when the pairing device needs a passphrase to build a WPA2 connection privately. Moreover, the WiFi P2P options could be switched off in the configuration systems, while in android devices, it also applies to Hotspot.

2.3.7 Deauthentication attacks

The deauthentication attack usually occurs in four states, as stated in [26].

1. Unauthenticated and Unassociated
2. Authenticated but Unassociated
3. Authenticated and Associated
4. Authenticated and Associated and 802.1x Authenticated

Starting from the first state, when joining a network, the client begins scanning all the channels for one target AP and the AP will be discovered broadcasting in a specific channel. The client and the AP authenticate each other through messages exchange, while during the second stage, the client and the AP

cooperate to stage three when they start to exchange data packets. In the 802.11 protocol, the authentication messages supported by 802.11 exchange between them to authenticate. After successful authentication, the client and the AP move to the final step. An attacker can then use tools like aireplay-ng to deauthenticate through data packets with spoofed MAC addresses of the victim.

2.3.8 Automated penetration testing on Smart home

Penetration testing manually is a work that requires time and research. To mitigate the human fault in security testing and save some time, tools for common vulnerabilities are combined into a script in Python 3.6 [27]. The tool can evaluate the security level on a scoring method based on the vulnerability found in the target. This framework was tested on several popular IoT devices and has proven beneficial to people needing more technical knowledge on cyber security.

2.4 Summary

The prior researches on IoT device security has shown many vulnerabilities and weaknesses in similar smart camera systems. The listed works managed to find vulnerabilities and obtain critical information like video streams or credentials. With more works published in journals and conferences, smart camera security is deeply investigated. In such manner, the vendors can continuously improve the security implementation of their products. The related work also enlightens this work to perform relevant and selective attacks. With enough basic knowledge of a smart camera, this work then steps to the next phase.

Chapter 3

Methodology

This chapter describes the chosen methodology for this thesis. The author did not find any methodology available specific for IoT cameras. Therefore this work follows a general methodology known as Penetration Testing Execution Standard (PTES) [11]. The main section of this methodology consists of 7 sections, which are listed below.

1. Pre-engagement Interactions
2. Intelligence Gathering
3. Threat Modeling
4. Vulnerability Analysis
5. Exploitation
6. Post Exploitation
7. Reporting

After navigating to the top-level methodologies recommended by numerous websites, the author chose PTES for the following reasons. First of all, the selected methodology shows very clear content and structure of how this work should be done. Second, when compared to other methodologies like The Open Source Security Testing Methodology Manual (OSSTMM) [28], PTES better suits the thesis for that it is designed to simulate real-world attack scenarios and the approach is well-structured for all aspects of the target system. Last but not least, the scope of PTES covers the methods previously considered before starting the project and is also fulfilled ethically.

3.1 Pre-engagement interactions and intelligence gathering

This section includes all the necessary pre-engagement steps on the target IoT device. A thorough overview of the whole pre-engagement interaction will decide the scope of the project. Particularly, the author needs to understand the risks and scope of penetration testing. The planning and desirable outcomes are decided during this phase. In this work, the author applies grey-box testing since the attacker knows the internal structure with the published source code of the camera and escalates user privileges [29] which means the author has prior knowledge of the username, password, location, full user control of the camera, and a rooted phone. The next step is to collect useful intelligence from all available sources. Intelligence information gathering was essential during wartime. Now it is even more critical in today's cyber war. Some of the intelligence might be helpful in discovering vulnerabilities or weaknesses in the following steps. A thread might give a limited amount of information during this phase. However, in the following analysis, the relevant information in the thread is critical in certain circumstances. Chapter 2 and Section 4.1 conduct the pre-engagement interactions and intelligence gathering step. Chapter 4 presents the thread modeling part of the methodology. Section 5.3, Section 5.1, the introductions and methods parts in Chapter 5 introduces the vulnerability analysis. The exploitation and post-exploitation subsections in the Chapter 5 correspond to exploitation and post exploitation. Chapter 6 and Chapter 7 cover the post exploitation as well and the reporting step.

3.2 Threat modeling and vulnerability analysis

Threat modeling is defined as a process that can be used to analyze potential attacks or threats and can also be supported by threat libraries, or attack taxonomies [30]. STRIDE model by Microsoft [31] is used in this work. The STRIDE model stands for Spoofing (Authenticity), Tampering (Integrity), Repudiation (Non-repudiability), Information Disclosure (Confidentiality), Denial of Service (Availability), Elevation of Privilege (Authorization) [32]. In this threat modeling method, there are certain principles to break down the whole modeling process [33].

1. Identify Assets

2. Create and Architecture Overview
3. Decompose the Application
4. Identify the Threats
5. Document the Threats
6. Rate the Threats

The deliverable from this threat model is detailed documentation about the assets, relations between the assets, the architecture of the test object, and a list of potential existing threats rated. With the threat model set up, testers can start analyzing the actions to make the most value out of the model and are almost ready to start penetration testing. With the help of the tools available on the internet, testers like vulnerability scanners make it a lot easier to perform discovery and inventory within the scope. Tools like [34] could easily scan the device application and provide a detailed and comprehensive report on the discovered vulnerabilities. Besides the automated vulnerability scanner, the manual method is also applied. The Open Web Application Security Project (OWASP) [35] top 10 list is a practical standard to identify vulnerabilities. OWASP top 10 Vulnerabilities in 2021 [36] include:

1. Broken Access Control
2. Cryptographic Failures
3. Injection
4. Insecure Design
5. Security Misconfiguration
6. Vulnerable and Outdated Components
7. Identification and Authentication Failures
8. Software and Data Integrity Failures
9. Security Logging and Monitoring Failures
10. Server-Side Request Forgery

Successfully identifying the vulnerabilities helps testers to proactively initial ethical attacking, discover security flaws and better protect the assets and data. With these works done, the tester can extract practical analysis from the report and get start on the penetration testing phase.

3.3 Exploitation

The exploitation phase is based on the previously decided scope, mapped assets and communications, and detected vulnerabilities. The ethical hacker will initiate attacks on the target device using various tools and describe how the vulnerabilities are exploited in the later reports. The effect caused by the attack will also be demonstrated and analyzed. In this thesis, attacks targeting wireless communications, mobile applications, and software are initiated, while hardware testing will not be conducted.

3.4 Post exploitation and reporting

After finishing the exploitation phase, the tester should archive the methods to gain access to the IoT devices and retrieve valuable information from the system. An overall evaluation of the impact caused by the successful exploitation should also be provided to show how the system's security level could be improved. The evaluation will be in the form of sections in the thesis after each attempted attack. Ethical hackers operate without malicious intent but report vulnerabilities to improve the security measures of organizations [37]. In this thesis, the author reports the founded threats to the corresponding supervisor. The report sent to the supervisor should consist of navigation to the exploit vulnerability and information obtained through the vulnerability, as well as the consequences caused. After the penetration testing, the tester should clean up the environment and restore settings to the factory settings to prevent future unauthorized access to the system [38].

Chapter 4

Threat Modeling

This chapter describes the threat model for DCS-8515LH smart camera. Including the device system, Android mobile application, and wireless network communications, the threat model process identify all the assets and the data flow in between device and servers. As the methodology states, this chapter covers the intelligence gathering step and the threat modeling process.

4.1 Overview of the IoT system

This section shows all the assets in the smart camera system and mobile application.

4.1.1 Assets inventory

An acknowledgement of how the assets are configured will greatly reduce the workload for vulnerability analysis based on CVE and CWE. And, knowing the vulnerability will certainly affect how the vulnerability analysis will be performed. However, since the hardware exploitation is not in the scope of this thesis, only a brief mentioning with no further details will be presented on hardware. A brief list of assets is listed down below.

4.1.1.1 Hardware

Mydlink DCS-8515LH Camera has a microSD card slot under the camera where a microSD card can be inserted. A microphone is also located below the camera for interactions. And a status LED at the bottom front of the device, solid green representing the camera has successfully connected to a Wi-Fi network as the setup is completed. Flashing green means the live view of

the camera is activated. Solid red show the device is booting up and flashing red indicates the firmware is updating. And the flash orange color represents Bluetooth pairing or offline. In the back of the camera, a speaker is located through which it can provide audio playback for the two-way audio feature. And at the bottom of the camera, there is a reset button which could reset the camera to factory default settings and a microUSB power port to supply power to the device. Additionally, a sticker of device information and a QR code of the device is attached at the bottom of the camera as well.

4.1.1.2 Firmware

Firmware consists of the system inside the camera, which enables control features of the camera. The current available version of the firmware is 1.07.01, which the author manages to find online. The firmware of the camera is a .bin file which could be transferred into a microSD card and plotted in the camera slot for an update. However, the .bin serves only as a backup firmware installation solution and cannot be opened manually per se.

4.1.1.3 Mobile application

D-Link company designed a mobile application for some modules of the IoT cameras while replace the web portal of the camera. The camera in the thesis is one of the products. The app enable users to view and manage the camera or other smart home devices as long as they have internet access. It also allows users to set up automation rules through interaction and scheduling which allows user to create scenes when at home, sleep or away. And of course users can see the live view on the camera and control it with the Amazon Alexa and Google Assistant. In the app, user can also manually take snapshots and video clips saved to the mobile device.

4.1.1.4 Communications

The communications of the smart camera are mainly through wireless connections and Bluetooth. The device supports 802.11n/g/b wireless with WPA/WPA2 encryption and Bluetooth Low Energy (BLE)4.0. The network protocols include IPv6 ARP/ICMP, IPv4 ARP/ICMP, TCP/IP, UDP, DHCP Client, HTTPS and Bonjour.

4.1.2 Functionality of the smart camera

The mydlink Pan & Tilt Wi-Fi Camera is one of the new model from the manufacturer. It possess quite a few powerful functions which will comes in very handy. See the list below.

1. App recording and snapshot. The application can record live footage and take a snapshot of the live stream, and store in local storage.

2. SD card recording. The camera has a Micro-SD card slot which accepts a Micro-SD plugin for live footage storage.

3. Auto motion tracking is the automatic pan/tilt following an object as it moves.

4. Panorama view position selection allows the user to tap anywhere on the camera's field of view to quickly rotate to the desired position. This camera uses a high-resolution megapixel sensor with a motorized pan and tilt function for a clear view of the entire space.

5. Sound & motion detection that allows users to receive alert notifications and record video clips. With the help of a motion detector, sound detector and built-in night vision, the camera could detect any disturbance or changes in monitored space, even in complete darkness.

6. Two-way communication. The user could play audio or collect audio in the monitored environment. With the built-in microphone and speaker, the user can talk to anyone in the monitored space while viewing them on the mobile application.

7. Events automation settings. The mobile application allows the user to set certain automation schemes on the camera accordingly when the user is at home, sleep or away.

8. Voice control. Amazon Alexa and Google assistant are embedded in the system, which enables the user to control the camera with voice.

9. Cloud recording. Mydlink offers a subscription to a cloud recording function where the live stream record could be stored in the cloud.

The table 4.1 shows a summarized assets inventory of the system.

ID	Asset	Description
1	Smart Camera	The D-Link DCS-8515LH smart camera provides monitoring functions with sound and motion sensors. The speaker and microphone also enable two-way communication between the user and the monitor area. The user can see the live stream from the mobile application mydlink. The current OS system for the camera is Linux 3.2-3.16.
2	Hardware	Hardware components include light sensors, IR LEDs, camera lens, microphone, microSD card slot, status LED, speaker, reset button and micro-USB power port.
3	Firmware	The firmware is injected into the device through a microSD card slot. The current version of the firmware is V1.07.01 (downloaded from the TW support page, no resource on the EU support page).
4	Mobile Applications	D-Link cameras are used to support the web portal. However, DCS-8515LH model only supports the mydlink app. Functions in the mobile app also include taking snapshots and video recording. The current application version is 2.9.4, build 271. Older D-Link applications, ”mydlink Lite” and ”mydlink+”, are incompatible with DCS-8515LH.
5	Wireless Communications	The network requirements for the camera is 802.11n/g wireless network encrypted with WPA/WPA2. The connectivity also includes Bluetooth Low Energy (BLE)4.0. The network protocols include IPv6, IPv4, ARP, TCP/IP, UDP, ICMP, HTTPS, DHCP Client, and Bonjour.
6	Cloud Storage	D-Link provides a premium cloud storage subscription for users to save video recordings in the cloud server. The cloud server automatically saves the backup recording of events.

Table 4.1: Summary of discovered assets.

4.1.3 Communication protocols in use

This section presents the protocol discovered in wireless communication, detailed communication protocols and descriptions are shown in the table 4.2. A clear view of communication protocols is of great help to find vulnerability accordingly and perform exploits on these protocols.

ID	Communication Protocols	Description
1	TCP	TCP protocol establishes a connection between the smart camera and the servers on the transport layer.
2	TLSv1.2	TLS protocol is applied for communicating and authenticating between the client and the server. TLS 1.2 also includes an SHA-256 hash function, which is more secure than the previous versions.
3	ICMP	ICMP protocol is used to report errors or failed operations between the smart camera and the mobile application. ICMPv6 is used in this communication.
4	HTTP	HTTP functions as a switcher to upgrade TCP protocol to WebSocket protocol and communication with the relay server to exchange data with peers.
5	UDP	UDP protocol is used for voice and video transfer between the mobile application and the smart camera.
6	WebSocket	WebSocket protocol is used for the two-way communication between the smart camera and the server, where the client communicates with the server and gets a response from automatic responses.
7	STUN	STUN protocol is a standardized set of methods [39] for traversal of NAT gateways in real-time voice and video communications.
8	MDNS	MDNS protocol is a zero-configuration protocol for smaller networks. It uses multicast to address the participants in the network.
9	DNS	DNS protocol translates domain names into IP addresses when communicating with mydlink servers.

Table 4.2: Communication protocols in use.

4.1.4 Open ports on the device

This section reports all the open ports discover in the device for later tests, as shown in 4.3. The open ports are discovered through Nmap scanning, these information is a cue for ports exploitation on the system.

ID	Open port	Description
1	Port 554/TCP	Port 554 runs RTSP service, which supports video and audio streaming.
2	Port 7000/TCP	Port 7000 runs afs3-fileserver service. It is an AFS distributed file system supported by OpenAFS dedicated to server communication that uses the ”next available” port.
3	Port 8080/TCP	Port 8080 runs ssl/http-proxy service that examines Web traffic and filers untrusted activities in the connection.
4	Port 8081/TCP	Port 8081 runs ssl/blackice-icecap service, which is part of an admin software for the firewall system.
5	Port 8088/TCP	Port 8088 runs radan-http service. This service cooperates with the general use of HTTP proxies.
6	Port 5353/UDP	Port 5353 runs mdns/zeroconf service. It serves as a DNS-based service discovery. (Only open when setting up the device)

Table 4.3: Open ports and running service on the ports.

4.1.5 Data flow diagram

This section describes the data flow diagram of the testing environment. Figure 4.1 shows general data flow. The only way to for user to communicate with the smart camera is through mobile phone. The mobile phone transmits data to the router in WiFi scenario and to the server in cellular network. During the user authentication process, the transmitting protocols uses HTTPs. While in other requests and responses such as triggered events, the mobile phone communicates with the router through HTTPS and HTTP protocols. When streaming the live footage of the camera, the communication protocols are HTTPS and HTTP.

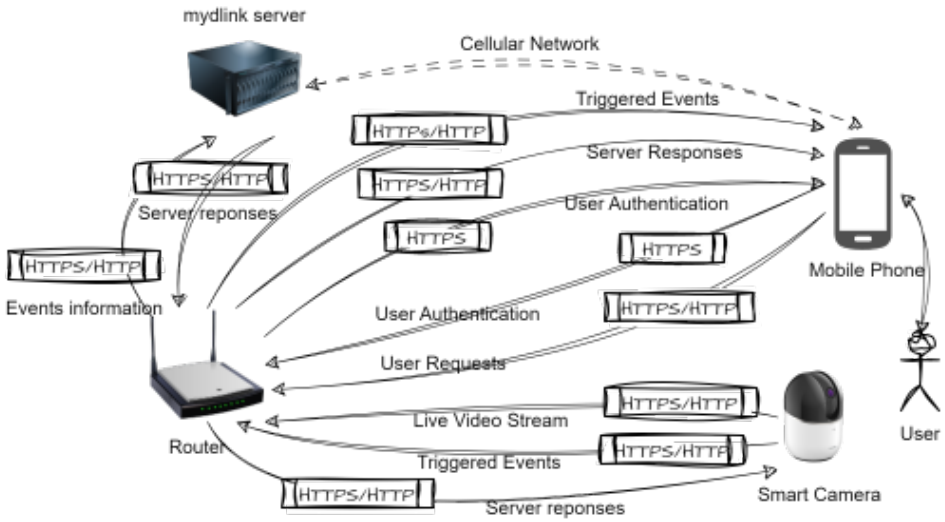

Figure 4.1: Data flow diagram of the smart camera

4.2 Entry points

This section lists all the entry points of the smart camera architecture. Entry points are usually system or network that could be used by attacker to gain unauthorized access to the target environment [40]. The goal of identifying the entry points is to provide convenience for later attempts in exploiting and gaining access to the system. Table 4.4 lists all the possible entry points of the system.

4.3 Experimental setup

The experiments in this work takes place at author's home network. The experimental setup includes a Huawei WS7100-20 AX3 WiFi 6 dual-core router, a Huawei MateBook 14 2020 AMD laptop, a bootable kali USB flash drive Appendix A, a rooted Xiaomi Redmi note 11 smartphone B and the D-Link DCS-8515LH smart camera. The attackers are simulated in the smart phone, laptop, router and in the communication between smart camera and smart phone, smartphone and router, router and laptop, laptop and smartphone, router and server. Figure 4.2 shows visually of how the experimental network is setup and where the attacks are initiated from.

ID	Entry point	Description
1	Smart Camera	The smart camera connects to the local network and pairs with the mobile phone.
2	Firmware	The firmware is available for download from the support page. The user can update the firmware from the micro-USB card slot manually.
3	Wireless communications	The smart camera communicate with the server and mobile application through the local network.
4	Mobile Phone	The mobile phone stores the user's data from the mobile application and communicate with the server and smart camera through wireless connections.
5	Mobile application	The mobile application can be downloaded open-source. The app functions as a control panel for the smart camera.

Table 4.4: Possible entry points

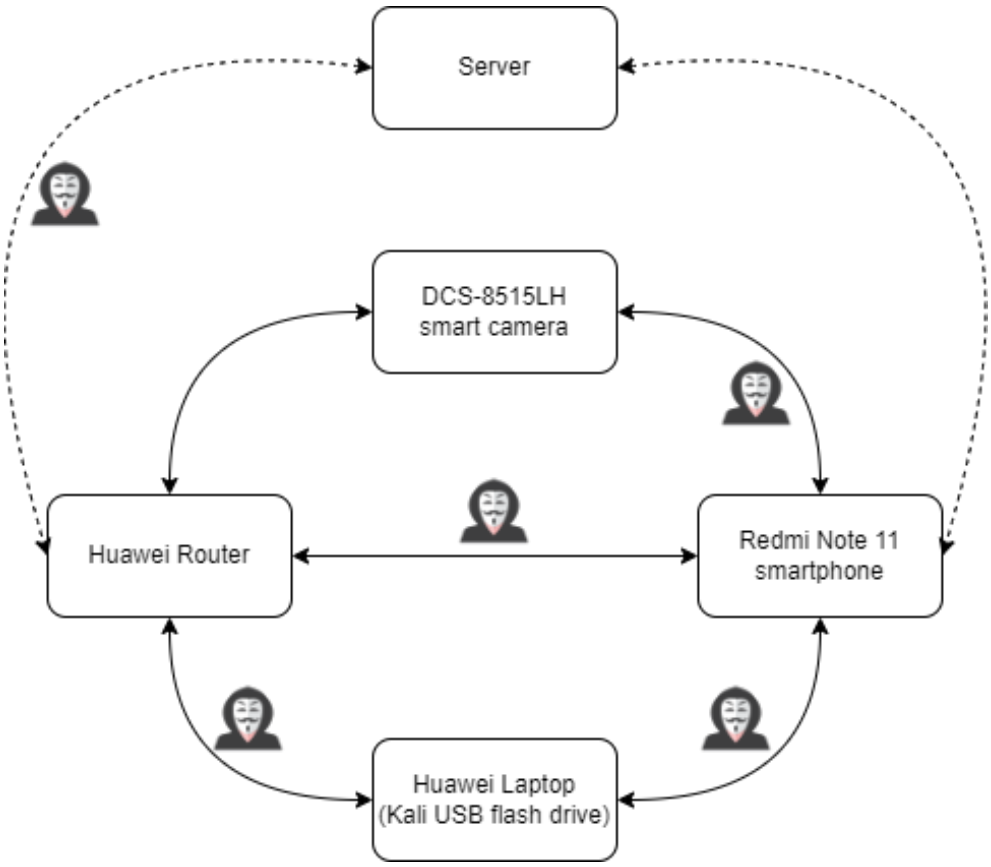

Figure 4.2: Experimental setup and attackers hidden places

4.4 Identified attacks

The STRIDE model [12] is applied in this section to identify threats in the smart camera. Identifying attacks helps to locate vulnerabilities and weakness in the smart camera, which can be used to proactively address vulnerabilities. The identified attacks in this work is shown in Table 4.5.

ID	Threat	Impacts on the system
S	Spoofing	Modification on logs, spoofing on the communication nodes in the system.
T	Tampering	Modification on system storage, modification on the transmitted data packets.
R	Repudiation	Deafen user's actions or disable entirely. Denying on user's request to interact.
I	Information disclosure	Sensitive data exposure. The leak of critical information in the system.
D	Denial of Service	Cripple the system. The system was forced to be unavailable for usage.
E	Elevation of privilege	Gain of the higher privilege of the system.

Table 4.5: STRIDE model threats of the smart camera

4.5 Selected attacks

A significant number of attacks should be tested in a wholesome penetration test. However, some attacks are more suitable and more effective in practice. Therefore, selecting the attacks accordingly is beneficial to accelerate this work and gain the most from the system.

Cryptographic failures is previously known as sensitive data exposure. Although the camera does not provide direct access to the system, this threat is still selected because of the usage of android applications. The Android phone is easier to root compared to the Apple phone, therefore providing a chance to dig into the storage system for sensitive data.

SQL injections could obtain data stored in the database by injecting codes. This attack is selected since the mobile application store data in an SQL database. The database could be cracked without authentication when the data storage format is easy to presume. SQL injection could be used when the user requests login or sends other sensitive information requests.

Insecure design is a newcomer on this list, indicating the attack is getting more effective over time. This attack could be used against the mobile application in case of an insecure design exists. This attack is selected so the attackers could then violate the trust boundary and generate a message with sensitive information. The mobile application and phone storage system are the main target of this attack.

Security misconfiguration is a general issue on security settings which provide attacks easy access to the data. The author selects this attack because the configuration on login, database and some certain features could involve the ignorance of security control. Exploiting through this attack can lead to a lower privilege escalation or, eventually, a higher privilege escalation.

Vulnerable and outdated components is common on apps relying on third-party frameworks [41]. Any outdated framework is not recommended since it might contain open vulnerabilities. Any outdated component could lead to severe damage to the system, therefore selected in this work. This attack becomes handy if any CVE or exploited information can be located on the internet. For example, an overall scanning of the system and traffic is helpful in identifying any components that fit this attack.

Identification and authentication failures are also known as broken authentication. This attack has dropped in popularity on the list but remains influential in cyber attacks. This attack utilizes the nature of login authentication. Gaining knowledge of the sensitive data needed when logging in, the authentication could be impost by an attacker and get control of the system. Attacks like brute force and modification on the URL could be tested here.

Software and data integrity failures is also selected since the firmware is constantly updating, which means there is a possible weakness in the smart camera firmware that needs to be mitigated. If the firmware compromises under attack, the attacker could then get control of the smart camera, which disturbs some of the functions of the camera. Firmware reverse engineering could test the integrity of the firmware in this case.

Secure logging and monitoring failures means the system could not adequately monitor the device's security actions, resulting in ignorance of attacks. This is selected as an ongoing test as all the attacks should be properly prevented by the system. This failure significantly lowers the security level of the smart camera, leaving the target protected but not monitored.

The last one on the list is server-side request forgery. This attack could test out the security level on the server. If the server processes the request without verifying, the attacker could pull information from the server. This attack is

selected but only partially conducted since the server is outside the scope of this work for legal reasons.

Another attack that used to be on the list was the DoS attack. Although this attack is not selected on the list, this attack is widely used and proven effective to the IoT system. Thus, a DoS attack is also conducted in this work.

Owing to the analysis and the compatibility of OWASP's top 10 list and the smart camera, the specific selected attacks are listed below.

1. Brute force attack
2. SQL injection attack
3. Passive reconnaissance
4. Network protocol inspecting
5. Zero-configuration protocol exploit
6. Firmware hacking
7. Mobile application hacking
8. Streaming protocol exploit
9. MiTM attack
10. Denial of Service attack
11. Identification and authentication failure
12. Storage analysis

Chapter 5

Penetration Testing

This chapter contains penetration tests performed on the target devices and a thorough analysis of the attack's outcome. Based on the previous explanation, the actions taken, reasons, results, and analysis will be provided according to the methodology. This chapter reports the vulnerability analysis, exploitation and post exploitation in the methodology. Section 5.1 and Section 5.3 reflect to the intelligence gathering and vulnerability analysis in the methodology. The method subsection reflects the vulnerability analysis, the results reflects the exploitation and the discussion subsection reflects the post-exploitation. The penetration testing is performed in the experimental setup mentioned in 4.3.

5.1 Network inspecting

The very first step of penetration testing or ethical hacking is intelligence gathering. This step is called reconnaissance, which use scanning tools or vulnerability scanner to retrieve relevant information about the device. Network scanning usually means using a computer network to gather information in the local system or remote computing systems. This procedure is used for security assessment, system maintenance, and also for performing attacks by hackers [42]. Network scanning has two parts: Port scanning and vulnerability scanning.

5.1.1 Port Scanning

Port scanning is performed with a method of sending data packets to the scanning target's ports. According to the data packets sent and received, the scanning tool will collect information like port number, network services,

operation systems, etc, therefore getting an overview of the whole target system.

5.1.2 Vulnerability Scanning

Vulnerability scanning refers to the discovery of known vulnerabilities in the target system. The result gives possible weaknesses or vulnerabilities in the current system, which provides convenience and cues for the attackers to perform further investigation.

5.1.3 Method

Nmap comes in very handy in this part because it is an open-source tool for network exploration and security auditing designed to rapidly scan large networks [43]. Nmap is a very powerful tool which is used by countless researchers and attackers for gathering information on the target computer system. Nmap has the capacity of scanning the operating system, port states, running services and communication protocols which are basically everything that an attacker needs to know about the target. Worth mentioning one of the most potent scannings is script scanning (NSE). NSE can perform tasks from as essential as service version detection to vulnerability exploitation. NSE possess great complexity but is surprisingly easy to use which is the reason why it is part of the aggressive scan option.

5.1.4 Exploitation

The scanning results display opening ports and running services on the device, which could potentially reveal vulnerability corresponding to the system. The Nmap scanning results are shown in Figure 5.1.

As stated above, Nmap provides a very functional service called script scanning. When targeting to RTSP protocol, it's defined as a port rule called script RTSP-URL-brute. The script attempts to discover all the valid RTSP URLs by sending DESCRIBE request for each URL in the dictionary. Then it compares the response based on which Nmap determines the validity of the URL [44]. Due to the excessive amount of URLs discovered, a short example of the discovered URLs is shown in Figure 5.2.

```
Nmap scan report for 192.168.3.101
Host is up (0.0083s latency).
Not shown: 995 closed tcp ports (reset)
PORT     STATE SERVICE             VERSION
554/tcp  open  rtsp
7000/tcp open  afs3-fileserver?
8080/tcp open  ssl/http-proxy?
8081/tcp open  ssl/blackice-icecap?
8088/tcp open  radan-http
```

Figure 5.1: Open ports discovered on the smart camera from Nmap scanning

5.1.5 Post-exploitation

The port scanning provides quite valuable results about the open port in the intelligent camera system. In total, five opened ports and the running services are detected, port 554 (RTSP), 7000 (afs3-fileserver), 8080 (ssl/httpproxy), 8088 (radan-http). Port 554 runs the rtsp service, which is a streaming protocol for real-time streaming service. Interestingly, RTSP uses not only port 554 TCP but also 5004 UDP and 5005 UDP. Port 554 is used to accept incoming RTSP client connections and to deliver the data packets to the clients on the stream. Port 5004 is used to send data packets to the client which enables the user to see the stream by using RTSP. Port 5005 is for receiving data packets loss information from the clients and synchronising the clients' stream. Certain vulnerabilities are already known for a few years in RTSP, such as buffer overflow in Hikvision RTSP request header handling (CVE-2014-4879) [45]. These CVE are pretty outdated and tested to be fail in the smart camera. However, up-to-date CVEs were found as well. Further implementation with the reported CVE in the system should be conducted in later sections. Port 7000 runs afs3-fileserver service. AFS stands for Andrew File system, which is a distributed file system that uses trusted servers to present a homogeneous file namespace to the clients' workstations [46]. AFS implements access control lists for the users that allow limited file system access, therefore, have a high-security level. Port 8080 runs an http-proxy service that allows the clients to communicate on the internet through a proxy server. Port 8081 runs ssl/blackice-icecap service; it's a piece of administration software for a firewall system. All the discovered ports have reported CVE. Further actions on these ports should be conducted in the later sections. The script scanning results are

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap --script rtsp-url-brute -p 554 192.168.3.101
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-17 14:42 EST
Nmap scan report for 192.168.3.101
Host is up (0.0090s latency).

PORT    STATE SERVICE
554/tcp open  rtsp
| rtsp-url-brute:
|   other responses:
|     454:
|       rtsp://192.168.3.101/
|       rtsp://192.168.3.101/0
|       rtsp://192.168.3.101/0/video1
|       rtsp://192.168.3.101/1
|       rtsp://192.168.3.101/1.AMP
|       rtsp://192.168.3.101/1/1:1/main
|       rtsp://192.168.3.101/1/cif
|       rtsp://192.168.3.101/1/stream1
```

Figure 5.2: Script scanning feedback RTSP URLs example

surprisingly loaded. Dozens of RTSP URLs are discovered. Among all the URLs, a particular URL like rtsp://192.168.3.101/1/stream1 indicates some functions within the URL. The excessive results hint that the RTSP protocol are overall secure in general. Section 5.5 utilises these results and exploits the protocol.

5.2 Decompilation of android application

The increasing use of mobile terminals has drawn attention to the need for tools and techniques to help security analysis of binary code [47]. Reverse engineering is a method to understand the device or system through deductive reasoning [48]. An Android Application Package(APK) file is used to install application on the Android device. Decompiling APK is a method derived from reverse engineering, which is famous for the investigation of applications. With the knowledge of how the application is compiled, the author will have a better understanding of the application construction and how the application functions during usage.
D-Link has designed a specific application, namely ”mydlink” [49] for its IoT devices. With the help of the mydlink app, users can access and control their mydlink enabled IoT devices from anywhere. The app also provided extra

cloud service for some of the models in a secure and private way. Android is built with java adn the APIs are also designed in Java. The Java platform tends to store more high-level information in its execution file, making it easy to recover the source code [50].

5.2.1 Method

The newest version of the APK file should be the first option for considerations since most users applications are automatically updated. The APK file could be directly opened in the Kali Linux system. However, the dex file within the application needs further actions to view the content. In the android application analysis, methods could be categorized into the following: Static analysis, binary reversing, and dynamic analysis. The static analysis uses a tool called apktool [51], and quark-engine [52] to scan the source code. In this case, a tool called dex2jar-2.0 [53] can convert the file format from dex into the jar file. Then the jar file could be viewed in file explorer or IntelliJ IDEA, or the tool jadx [54] could give a more specific view of the APK file. To take this step further, the java file in the jar file could expose a potential vulnerability in the application. For example, the name of the javascript could indicate the function or usage of the codes, therefore providing an opportunity to modify the codes or insert malicious scripts in the application. When the modified APK file is installed by an unaware user, the user's information or sensitive data could be leaked, causing severe damage. But the more functional it is, the more vulnerabilities it brings, so the author hereby decompiles the android application APK and takes a look around for possible insecure features.

5.2.2 Exploitation

The author downloaded the newest version of the APK files. The APK file is named mydlink 2.8.1.apk; with convenience from kali Linux, it could be directly extracted to files visible. Two dex files were located after extraction, one called ”class.dex” and the other one ”class2.dex”. To further take a look into the dex file, a tool named dex2jar-2.0 [53] is used to convert the files to jar files. The installation of dex2jar was made easy by Kali Linux. After converting to a jar file, the file inside is readable with IntelliJ IDEA, it could also be viewed just in the file system, but IntelliJ IDEA is more user-friendly. Class.dex converted results are provided in the Figure 5.3.

Both converted files shown in Figure 5.4, Figure 5.5, contain a class that is randomly confused with alphabet letters from author's inspection. However,

```
┌──(kali㉿kali)-[~/Downloads/mydlink_2.8.1_Apkpure.apk_FILES]
└─$ d2j-dex2jar classes.dex
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
dex2jar classes.dex → ./classes-dex2jar.jar
java.lang.StringIndexOutOfBoundsException: begin 1, end -1, length 2
        at java.base/java.lang.String.checkBoundsBeginEnd(String.java:3319)
        at java.base/java.lang.String.substring(String.java:1874)
```

Figure 5.3: Converted dex file to jar file with dex2jar

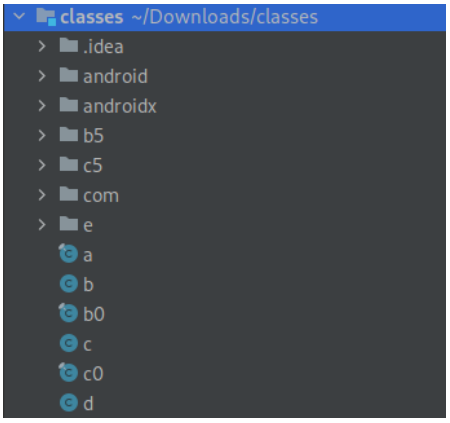

Figure 5.4: Content of converted classes file

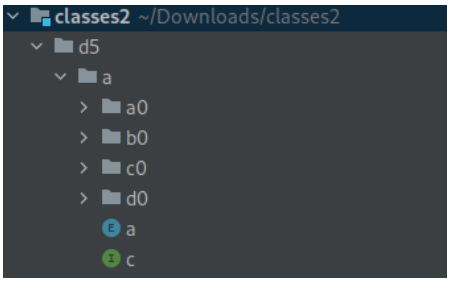

Figure 5.5: Content of converted classes2 file

some of the folders are in clear text, which is investigated with search function in IntelliJ IDEA. However, the files within the clear-text folders are either android native libraries or not relevant to the core functions of the application and does not really affect the overall usage of the users.

In general, the classes and code files are made confused to better protect the APK file. This measure also added obstacles for attackers to plant malicious functions in the application. Since the android application is overall easier to download and install than IoS apps, preventing forge in the application is crucial to prevent innocent users from getting attacks from unknown sources.

Gladly, with the completeness of the security testing method, various testing tools for APK files are available on the internet, which greatly aided the analysis of the security level of the application. A website called Immuniweb [34] provides security tests, mobile application privacy checks and OWASP Mobile Top 10 tests. Therefore, the author uploaded the APK file and got a detailed summary of the APK file.

The summary report gives the author a few insights into the possible vulnerabilities. The privacy policy was not found in the application, which could either be a misconfiguration or a possible weakness.

Figure 5.6: Three of the High Risk OWASP Mobile Top 10 found

```
┌──(kali㉿kali)-[~/Downloads/platform-tools]
└─$ ./adb shell pm path com.dlink.mydlinkunified
package:/data/app/~~sgtugE-u4qx5TgBWtclbbg==/com.dlink.mydlinkunified-YSSstUy--qsOzMDkUPompA==/base.apk
package:/data/app/~~sgtugE-u4qx5TgBWtclbbg==/com.dlink.mydlinkunified-YSSstUy--qsOzMDkUPompA==/split_config.arm64_v8a.apk
package:/data/app/~~sgtugE-u4qx5TgBWtclbbg==/com.dlink.mydlinkunified-YSSstUy--qsOzMDkUPompA==/split_config.en.apk
package:/data/app/~~sgtugE-u4qx5TgBWtclbbg==/com.dlink.mydlinkunified-YSSstUy--qsOzMDkUPompA==/split_config.xxhdpi.apk
```

Figure 5.7: Locate the apk file in the system

Interestingly, three high-risk OWASP Mobile Top 10 alerts were discovered, shown in Figure 5.6. This mobile application uses an unencrypted SQLite database [55], which can be easily accessed by a malicious attacker with physical access to the mobile device or an evil-intentioned application to achieve root access to the device. Also, the application stores information in clear text, even for sensitive information! The company makes a minor security flaw or is deadly confident in its SQL database. Additionally, using the input in raw SQL queries will result in a local SQL injection vulnerability in the mobile application, which could jeopardize private data kept in the database and the compromise of sensitive information stored in the database. Last but not least, the mobile application was discovered to use HTTP protocol to send or receive data, which does not have any encryption of the transmitted data, and can be easily intercepted when the attacker is located in the same network or has the access to the data channel of the victim. However, SQL injection is tested to be fail since the server and the user authenticate through a trusted certificate authority(CA) and data like passwords are not available in the requests intercepted. The insecure HTTP is tested in section 5.4.

The APK file could also be pulled out from the mobile phone with the command ”adb pull”. Therefore the author carried out another test on the APK file pulled from the mobile phone. As shown in Figure 5.7 and Figure 5.8. The base APK file is extracted and analysed here since the base APK contains the most crucial information and function of the file. The apktool is used to extract the package where the author discovered the AndroidManifest.xml in Figure 5.9 which is a file that contains activity and permissions that the application need when launching. The activity could be prone to certain vulners, among other features. More detailed information on where the flaws lies within will be investigated and presented in the following sections. Static analysis is performed with apktool and quark-engine, where the whole APK

```
┌──(kali㉿kali)-[~/Downloads/platform-tools]
└─$ ./adb pull /data/app/~~sgtugE-u4qx5TgBWtclbbg==/com.dlink.mydlinkunified-YSSstUy--qsOzMDkUPompA==/base.apk
/data/app/~~sgtugE-u4qx5TgBWtclbbg==/com.dlink.mydlinkunified... 1 file pulled, 0 skipped. 11.8 MB/s (7889758 bytes in 0.638s)
```

Figure 5.8: Apk file pulled from the mobile phone

```
┌──(kali㉿kali)-[~/Downloads/platform-tools/base]
└─$ cat AndroidManifest.xml
<?xml version="1.0" encoding="utf-8" standalone="no"?><manifest xmlns:android="http://schemas.android.com/apk/res/android" and
roid:compileSdkVersion="31" android:compileSdkVersionCodename="12" package="com.dlink.mydlinkunified" platformBuildVersionCode
="31" platformBuildVersionName="12">
    <queries>
        <package android:name="com.dlink.mydlink" />
        <package android:name="com.dlink.mydlinkbaby" />
        <package android:name="com.amazon.dee.app" />
        <package android:name="com.google.android.apps.maps" />
    </queries>
    <uses-feature android:name="android.hardware.telephony" android:required="false" />
    <uses-feature android:name="android.hardware.bluetooth_le" android:required="false" />
```

Figure 5.9: Exposed AndroidManifest.xml

file is extracted and scanned. However, the quark-engine is not doing well with the automated static analysis of the file, as the Figure 5.10 shows. The API level of the application is too high for quark to do such a scan. Then, we proceed to binary reversing of the APK file; the dex2jar extraction is already performed in the above results. Worth mentioning that the APK tool is also capable of such decompile action.

To better visualize the decompiled file. The tool jadx [54] is used since it provides a GUI to make the file more readable. The search function in the jadx is very convenient in this case, the search results for ”key” is shown in Figure 5.11 Another finding is when searching for password, in the HttpUrl class the format of the UserToken is described shown in Figure 5.12.

5.2.3 Post-exploitation

Since the IDE can not create separate files on the OS, the jar file contains obfuscated class files. The obfuscated file have identifiers renamed to make them harder to understand by human readers. This precaution significantly increases the difficulty of understanding the application and function blocks, therefore increasing the difficulty for attackers to modify or insert malicious code in the application.

However, with the help of apktool and jadx, the author successfully discovered the AndroidManifest.xml file and located some of the critical data formats in the APK file. This exposure of information gives the attacker an opportunity to rewrite the java file and upload it to a public open-source platform, then initiate an unforeseen attack.

Moreover, undoubtedly, the best measure to prevent these kinds of attacks is to avoid installing Android applications from an unknown source or

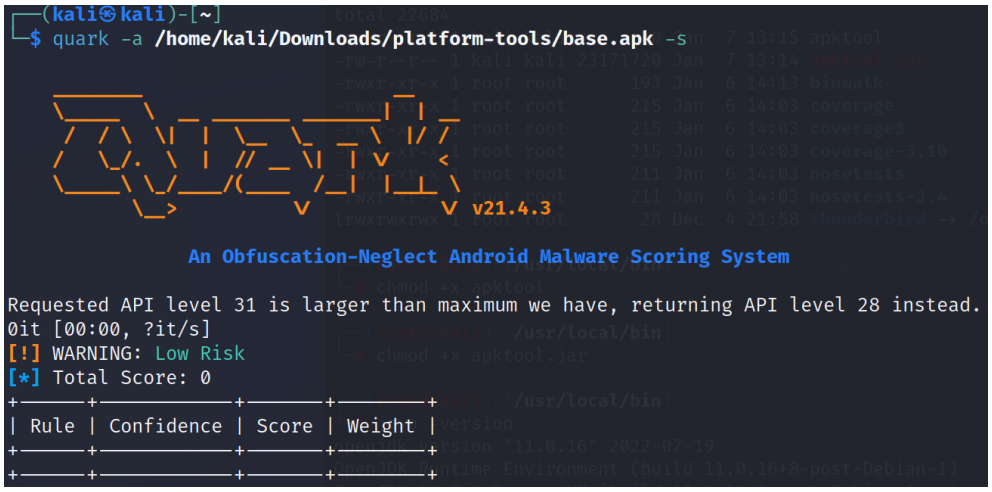

Figure 5.10: API level too high for quark scan

Figure 5.11: Search results for ”key”

unofficial source since the APK file could contain vulnerability or malicious code injected by unknown attackers. However, this is out of the scope of this work.

5.3 Analysis of incoming/outgoing traffic from the smart camera with ARP poisoning

Traffic analysis refers to monitoring and analysing the data packets transferring in or out of the target system. This method is widely used to identify security issues. What's cumbersome is that the traffic can not be monitored directly from Wireshark [56], both in windows or Kali Linux. The only few lines of traffic that could be observed are MDNS protocol traffic. But such could hardly satisfy the standard for traffic analysis. However, the author managed to find two ways to observe the traffic in Wireshark after doing loads of research. Traffic analysis is one kind of passive attack. When the attacker executes passive attacks like traffic analysis, no actively malicious actions need to be

Figure 5.12: Query format for getUserTokenByUserPass

taken. The rapid communication of data could expose potential vulnerabilities on the target device. When the traffic is being monitored, information coming in/out from the system could leak sensitive data to certain vulnerabilities. In [57], a traffic analysis attack is proposed to infer what web page the user visited on the mobile phone only by the network traffic generated by the phone.

5.3.1 Method

An HTTP server can helps to keep a log of all the incoming request to the server. With an HTTP server, any PC in the local network can access or communicate with the phone and hereby to see any ongoing traffic with the camera. With the help of an app called PCAPdroid [58], the traffic could be viewed in wireshark in real-time. The HTTP server mode in PCAPdroid launches an HTTP server which can be accessed by any PC on the local network to download the PCAP file containing the captured traffic which is a default mode of PCAPdroid. This mode is PCAPdroid's default operation because it doesn't require any further setup or a particular operating system. This mode can be used to assess the PCAP on traffic in real time after the capture has begun using a curl command, as shown in Figure 5.13. However, this is not directly captured on the camera, more precisely to be capturing traffic through the mydlink application. The PCAPdroid provides a function for targeting the traffic on one specific traffic. In this case, the option is selected to the mydlink app. See the Figure 5.14 for details.

Another way is to use ettercap [59] to perform ARP poisoning on the router and the smart camera. Ettercap can perform one kind of MiTM attack, which direct the data packets back to the network, thus diverting and switching the data packets streams for analysis. This attack is also called ARP poisoning, where the attacker sends out fake ARP data packets to the network to redirect data flow to a specific IP address to the attacker. With the help of ettercap, the traffic can then be viewed on Wireshark. The captured traffic .pcap file could be analysed manually however, using a analysing tool provides more thoroughly and detailed analysis. Therefore, a tool called A-packets [60] is used for analysing the traffic file.

5.3.2 Exploitation

To provide more information on this traffic analysis section, the author has tried almost all the methods on the internet to view the traffic on Wireshark for further work. Tools like PCAP remote do not work on Android 11+ systems, and the SSH server capture method does not cooperate well with the application since it requires a VPN server connection while the mydlink app can detect the wireless server has been changed and therefore refuse to connect to the camera.

Traffic captured by Wireshark through PCAPdroid is shown in Figure 5.14, most traffic goes through the IP address 10.215.173.1, which is the IP address of the virtual interface created by PCAPdroid.

The second way of capturing the traffic was later on discovered by the author, and it turned out to be more effective. First, ettercap needs to be enabled for ARP poisoning, with the command in Figure 5.15. ”-T” is for text only without any graphical content, ”-S” is used for not using SSL in this command, ”-i” is to set the interface to the selected wlan0 in this case, ”-M” indicates it is a kind of MiTM attack. Then the method of this attack is defined by ”arp:remote”, and after that, we specify the IP address of the router, last it will be the IP address of the smart camera. The Figure 5.15, represents the ARP poisoning has been started and undergoing. Then the traffic could be captured by Wireshark on Kali Linux machine as shown in Figure 5.17. A UDP traffic captured example is also demonstrated in Figure 5.16.

The analysis of the .pcap file displays all the connections including some of the connections that might contain vulnerabilities. As shown in Figure 5.18

5.3.3 Post-exploitation

The first method gives a very comprehensive topology of how the smart camera and mobile phone communicate with the Internet. However, the first method has got one minor flaw. The PCAPdroid uses a proxy 10.215.173.1 to communicate with the outside world, which is not really the mobile phone IP address itself.

```
┌──(kali㉿kali)-[~]
└─$ curl -NLs http://192.168.3.103:8080 | wireshark -k -i -
** (wireshark:29477) 12:56:18.703374 [Capture MESSAGE] -- Capture Start ...
** (wireshark:29477) 12:56:22.543466 [Capture MESSAGE] -- Capture started
** (wireshark:29477) 12:56:22.543535 [Capture MESSAGE] -- File: "/tmp/wireshark_-3IUEX1.pcapng"
```

Figure 5.13: Start capture traffic on the camera

So the emphasis lies on the second method. The second method uses ettercap to perform ARP poisoning for traffic capture. ARP poisoning is very efficient and lies within the heart of ettercap methodology. Upon uploading the .pcap file to A-packets website for analysis purposes, the author should state that this is a public source which means it is openly available for all visiters of the website. Therefore, the uploaded file needs to be deleted afterwards to protect the privacy of the experimental setup.

The report feeds us some very valuable information, such as the HTTP communication and Ethernet services. It also provided a clear network map, although due to some disturbance by the author's iphone, the map is not very ideal, therefore, not showing here. Interesting data traffic captured by Wireshark is using the HTTP protocol, shown in Figure 5.19, which is known as not very safe. Also, some other interesting protocols used in the traffic are captured and shown in Figure 5.20.

Session Traversal Utilities for NAT (STUN) is a standardized set of methods for the traversal of NAT gateways for real-time video and voice, etc. [39]. STUN is also a part of the signaling server that performs handshaking before the conversation between two parties is established. The STUN server identifies the unique IP address for the user in order to establish a P2P connection and exchange video. The traffic analysis with ARP poisoning provides a lot of helpful information that could be used in later penetration testing. Therefore, proven to be a vital step in this work.

5.4 Zero-configuration network exploitation

Zero-configuration networking also refers to "zeroconf" which is a set of protocols that automatically creates a network based on TCP/IP when interconnecting [61]. These kinds of networking do not require the user's interaction with the network. These technologies commonly assume the users in the system have already agreed to participate in the network. This mechanism is vulnerable to spoofing attacks by the system in the same broadcast domain [61]. To utilize this mechanism vulnerability, exploitation on the mDNS protocol is conducted in this section.

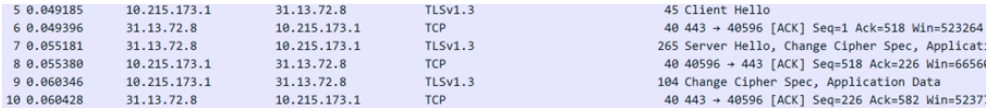

Figure 5.14: Traffic captured by wireshark through PCAPdroid

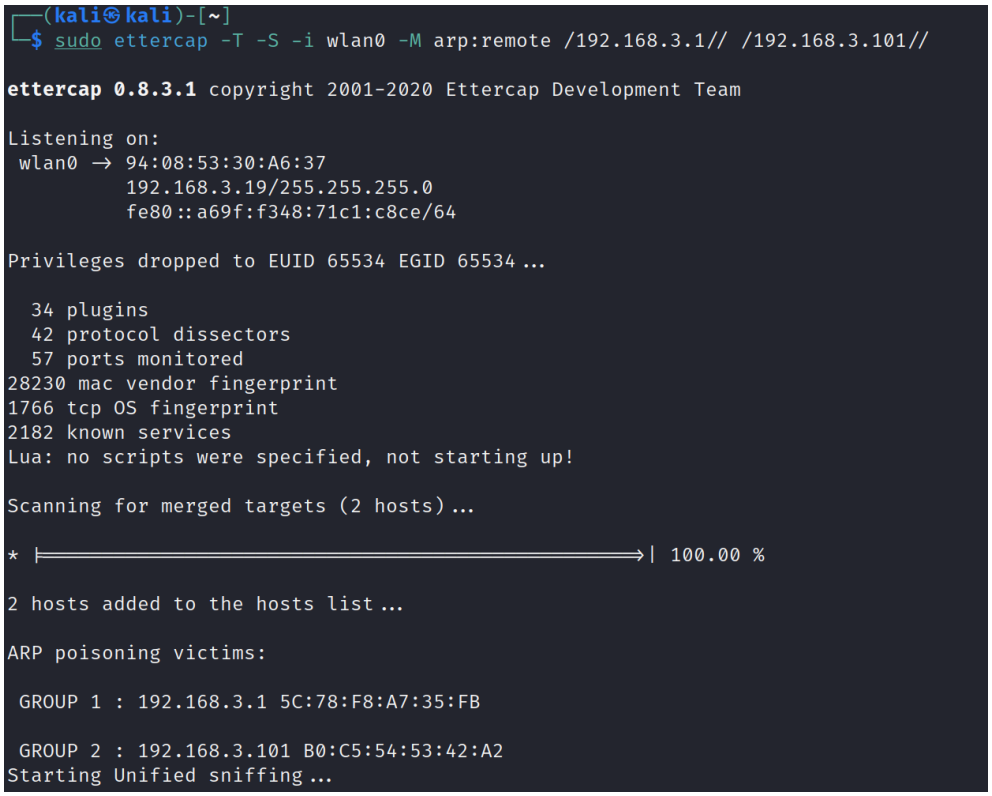

Figure 5.15: Ettercap command and ARP poisoning undergoing

The MDNS protocol is used in the broadcast domain that is the same as other devices, which indicates that the device using mDNS protocol share the same data link layer with the rest of the system. This introduces a vulnerability since the participants in the same network cannot be verified and allow an untrusted party to connect. The MDNS protocol is used when a Linux/Ubuntu system is booted. The device usually sends out a query with the header of xxx.local. Therefore, when the connection changes or alters, the device will start a probe phase and start announcing in the network. The device starts querying the local network and then announces its registered address with mDNS responses. Albeit, the probing phase becomes easy to attack with access to the local network.

5.4.1 Method

A python script called Pholus [62] will be used for this attack. Pholus sends out DNS queries if 15 conflicts happen in less than 10 seconds. The device will then have to wait for 5 secs before initiating another attempt [63].

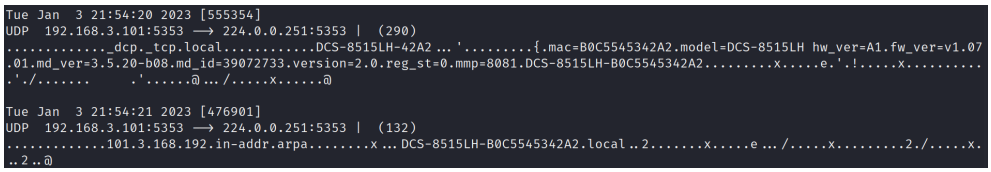

Figure 5.16: Ettercap UDP traffic capture example

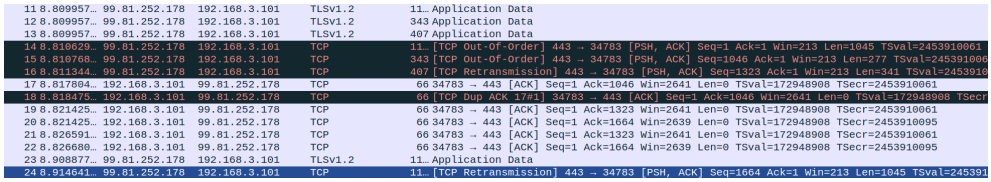

Figure 5.17: An example of wireshark captured traffic

5.4.2 Exploitation

This penetration test starts with scanning for possible victims in the network corresponding to the earlier traffic analysis.

The pholus.py script then discovered the smart camera in the network, as shown in Figure 5.22. After we identified our target, the attack could be started by specifying the interface, attack type and running time of this attack, shown in Figure 5.23. With the attack initiated and running, now it's time to take a look on the live stream view of the smart camera and verify the consequences. The results are shown in Figure 5.24 and 5.25. The pholus.py running successfully stopped the smart camera from accessing the network.

5.4.3 Post-exploitation

The smart camera is not secure against zero-configuration network exploitation attack when the mobile application popup display "video stream lost". It can be concluded that the video stream is completely blocked by the attack since the smart camera is unavailable for usage. These kinds of attacks are adequate for the connected IoT in the local network. Moreover, this forementioned attack could further lead to a more severe MiTM attack. With a MiTM poisoner, the attackers could further use an mDNS poisoning attack to escalate higher privilege.

Nowadays, the attacks on zeroconf are hard to mitigate since the need for mDNS service is still needed in lots of IoT devices. However, a unique identifier for the host could be helpful in preventing these attacks from happening.

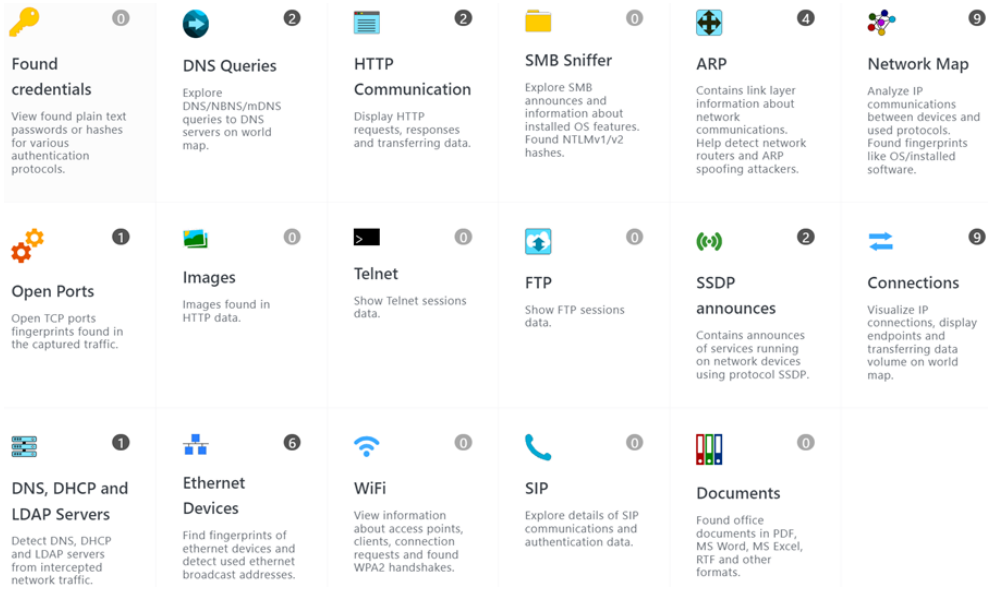

Figure 5.18: A-packets analysis result overview

```
GET /relay HTTP/1.1
Host:  mp-eu-sas-1.auto.mydlink.com
Connection:  Upgrade
Md-Model:  DCS-8515LH
Sec-Websocket-Key:  V9v7bTJOAP1LaRRLrZMSsA==
Sec-Websocket-Protocol:  stream-proxy
Sec-Websocket-Version:  13
Upgrade:  websocket

HTTP/1.1 101 Switching Protocols
Connection: Upgrade
Sec-Websocket-Accept: r6b9RRcLAhQfxjsNM1jeUDBaxQo=
Sec-Websocket-Protocol: stream-proxy
Upgrade: websocket
```

Figure 5.19: Traffic captured using HTTP protocol

5.5 Video streaming protocol exploit

The RTSP protocol is an application-level protocol used on streaming media servers [64]. The client of the server could issue many useful commands like OPTIONS, DESCRIBE, PLAY, PAUSE, TEARDOWN, etc. The protocol is not discovered directly from traffic capture but spotted from the port scanning with Nmap. Therefore the exploitation targeting RTSP port and protocol is exploited in this section.

The RTSP protocol works on the application level of the network communication stack where it provides a framework to establish and control the media stream connection for video or audio. When the server receives an RTSP request from the client, the server then responds to the client with a media description. Then the connection is set up through the set phase and starts

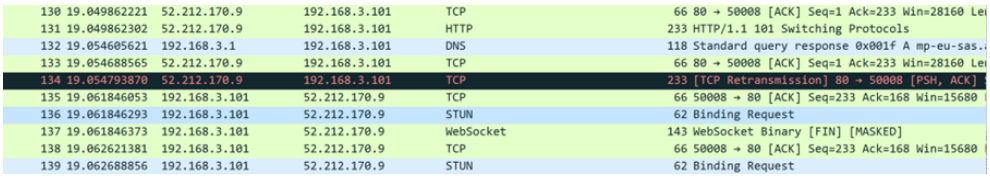

Figure 5.20: Interesting protocols that could contain vulners

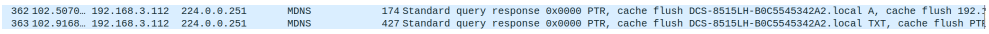

Figure 5.21: Discover mDNS service in the captured traffic

streaming. The protocol is easily compatible with HTTP since they use the same concepts. While the RTSP port is left open in this smart camera, this could lead to several attacks targeting the camera. For example, a tool called cameradar is developed [65] to detect and launch dictionary attacks to get the stream routes, even possibly display the streaming live view. The leaking of the live stream could be a severe violation of the privacy of the users.

5.5.1 Method

Since no traffic is captured with Wireshark, the RTSP port is open from the earlier investigation. This section's attacks mainly target the port and the camera system instead of the traffic flow. The author will try to connect to the device through the RTSP protocol and send a request to retrieve information from the system. Also, metasploit provides several attacks targeting the RTSP port. From the Nmap script scanning results, lots of RTSP URLs are discovered that might be played in VLS media player [66].

5.5.2 Exploitation

Before starting the attacks, a "health check" for RTSP is performed with the curl command, as shown in Figure 5.26. The RTSP server is working fine. Then, it is known that the RTSP protocol also supports telnet connection by the client. We connect to the device using telnet service. The telnet connection shown in 5.27 appears to be normal. However, when setting up traffic listening on the device to, test if the telnet connection could really generate traffic from the smart camera with a ping command. The local traffic listener did not capture anything at all, as shown in Figure 5.28. So it's proven that the telnet connection can not pull any request to the server. Then the DESCRIBE request is sent to the device through telnet, as shown in Figure 5.29 The

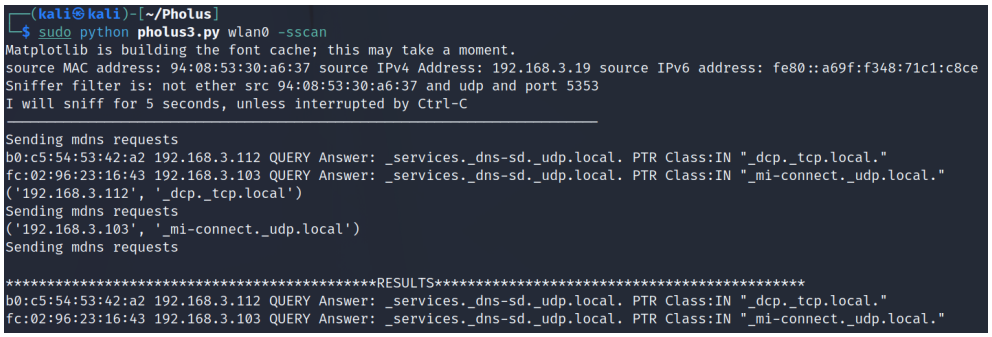

Figure 5.22: Discover the smart camera in the local network

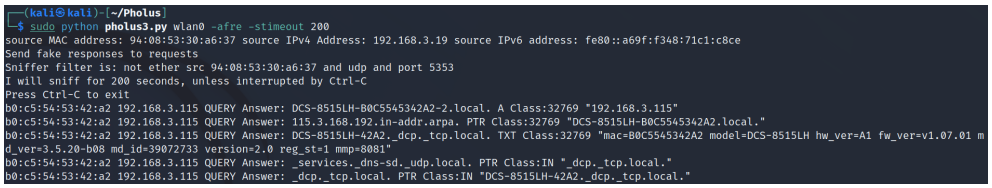

Figure 5.23: Probing phase abuse of the device

exploitation through telnet is stuck, and then we proceed to the metasploit to see any available exploitation options. However, there is only one exploit targeting the surveillance camera from Hikvision and MACOS; therefore it's not applicable in this work. That leaves one thing to be finished, the URLs from script scanning. From earlier nmap script scanning a URL with login credentials is identified in the output, shown in Figure 5.30. The VLC media player [66] is a great tool for using network streaming RTSP URLs, and after trying all the URLs discovered, the VLC gives us no response in Figure 5.31.

5.5.3 Post-exploitation

This section reports several attacks attempted on the RTSP port. During the author's research on the internet and books, the devices with the vulnerability

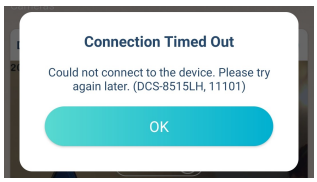

Figure 5.24: Connection Timed Out

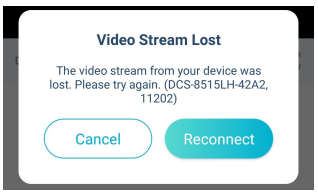

Figure 5.25: Video Stream Lost

Figure 5.26: Health check for rtsp server

Figure 5.27: Telnet connected successfully to the camera

of the RTSP port usually give out more information when sending a request to the server. Moreover, from the responses, the RTSP port is left open securely by the developers for a reason. And the telnet connection appears to be relatively safe since no exploitation succeeded, and even not much information could retrieve from the server. The DESCRIBE request was sent without feedback, most likely for the client is not authorized to get any response from the server, not even an error or denying response. Moreover, the streaming URLs also need to be played through the mobile application corresponding to the unauthorized feedback earlier when scanning port 554. Interestingly, from later research on this URL, for models DCS-XXXXLH series smart camera, using this link for 3rd party apps or network video recorder(NVR) devices is no longer supported. However, earlier models like DCS-936L do support RTSP streaming. So all earlier models are likely to have this vulnerability.

Therefore, the RTSP port appears to be secure and safe against the exploitation applied in this section.

```
┌──(kali㉿kali)-[~]
└─$ sudo tcpdump ip proto \\icmp -i wlan0
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on wlan0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
```

Figure 5.28: No traffic through telnet connection

```
┌──(kali㉿kali)-[~]
└─$ telnet 192.168.3.112 554
Trying 192.168.3.112 ...
Connected to 192.168.3.112.
Escape character is '^]'.
DESCRIBE rtsp://192.168.3.112:554 RTSP/1.0\r\nSeq: 2\r\nAuthorization: Basic YWRtaW4=\r\n\r\n
DESCRIBE rtsp://192.168.3.112:554 RTSP/1.0\r\nCSeq: 2\r\n\r\n
DESCRIBE rtsp://192.168.3.112:554 RTSP/1.0\r\nCSeq: 2\r\nAuthorization: Basic YWRtaW4=\r\n\r\n
```

Figure 5.29: No response from the DESCRIBE request

5.6 A deep look into the android storage system

Data storage needs necessary security protection in case of the leakage of sensitive data. When not encrypted or saved properly, personal information could be exposed to malicious attackers and cause severe damage. With the above reasons stated, let us take a deep look in the Android storage system.
The mobile terminal stores very critical and sensitive data of the user. Leakage or loss of this information could lead to serious impacts and accidents. Therefore, the data in the storage system should be stored safely and encrypted. With limited access of the system, a normal user could only view or modify certain levels of files. Since normal users do not have root access, an analysis of a non-root storage system is definitely within the scope. However, what happens if attackers gain the root privilege? Will the storage system still be able to defend the last fence of information security? To investigate in this topic, the root privilege is gained on the experimental RedmiNote 11 mobile phone.

5.6.1 Method

A rooted Redmi Note 11 mobile phone is analyzed for this test. With the help of the Adb tool, a versatile command-line tool that allows the user's computer to connect to the mobile phone, a connection is built between two computing systems. The mydlink application data are stored in /data/data/com.dlink.mydlinkunified path. In the file location, the main targets

```
rtsp://192.168.3.115/user=admin_password=tlJwpbo6_channel=1_stream=0.sdp?real_stream
```

Figure 5.30: RTSP URL with credentials

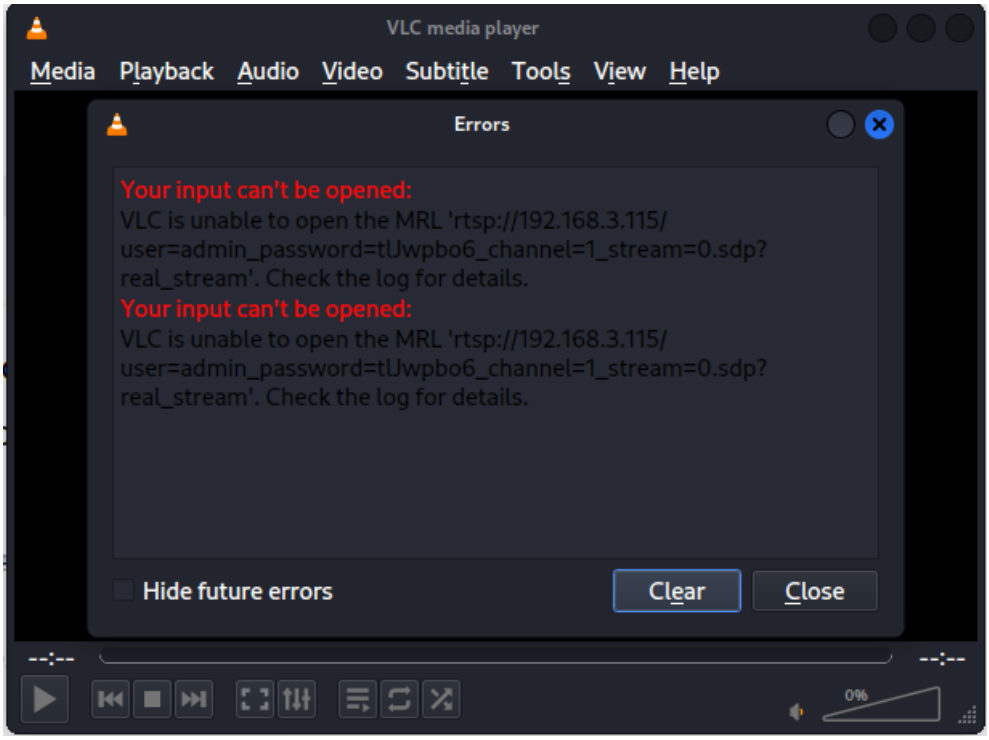

Figure 5.31: No playable url for VLC

are files names database, preferences,. etc. Especially folders that could contain sensitive information potentially. With the later discovery tool DB Browser for SQLite, a clear table view could be delivered and detect exposed sensitive data.

5.6.2 Exploitation

This section uses a rooted Redmi Note 11 phone and a tool called Adb [67]. It is a programming tool used for debugging Android-based devices. The daemon on the phone communicates with the server [67] on the host PC through a USB connection which could give access to the client to view data stored in the system. In this case, with the help of a rooted phone and Adb shell command, access to all the stored data is enabled. Without the root privilege, common commands like -ls could not be used. After digging into the storage system, a path called /data/data/com.dlink.mydlinkunified is found

Figure 5.32: Visiting storage system with root access

Figure 5.33: Email address, username and location exposed in clear text

which is storing content about the application. A databases directory is found which contains myDB database file. And according to what the vulnerability report gave us earlier in the chapter, it is found that some unencrypted user information is discovered.

Mydb is a simple db wrapper for mysql [68]. And correspondingly, the mysql database uses an unencrypted method. From the figure above, the user's username, registered email address and general location are exposed in clear text. More interestingly, the state of the camera is also shown since the camera has event settings which could enable the motion detection function. The author found a tool called DB Browser for SQLite [69], which provides a more user-friendly API to read the database. The exposed information is more secure than the inspected in the plain text in Kali Linux.

In the notification-router table, all the event history is stored with the name of the event, as shown in 5.34. The leakage of this information gives attacker a hold of what happened in the smart camera, such as if the device is online, and whether the camera is activated. The sensitivity of this data is crucial since the device is a smart camera for home, an information leakage like this could potentially cause a home break in or an intrusion.

Exposure of sensitive data of the user is identified as insecure data storage. Although, more sensitive data like passwords are likely to be encrypted and saved in the database.

5.6.3 Post-exploitation

The results indicate a particular security flaw in the storage system. Sensitive data like device state, username, email address and geographical location is leaked in this step. While more critical data like the password stays safe in this

Table: notification_router

	id	event_id	event_name	is_online	goto_id	goto_name
	Filter	Filter	Filter	Filter	Filter	Filter
1	793	1	Device Disconnected	1	0	Home
2	794	1	Device Disconnected	0	7	DeviceSetting
3	795	2	Device Reset to Factory Status	1	1	Timeline
4	796	2	Device Reset to Factory Status	0	1	Timeline
5	797	3	Hub Disconnected	1	0	Home
6	798	3	Hub Disconnected	0	7	DeviceSetting
7	799	4	Hub Reset to Factory Default	1	1	Timeline
8	800	4	Hub Reset to Factory Default	0	1	Timeline
9	801	5	Device back online	1	0	Home

Figure 5.34: Events found in the myDB database

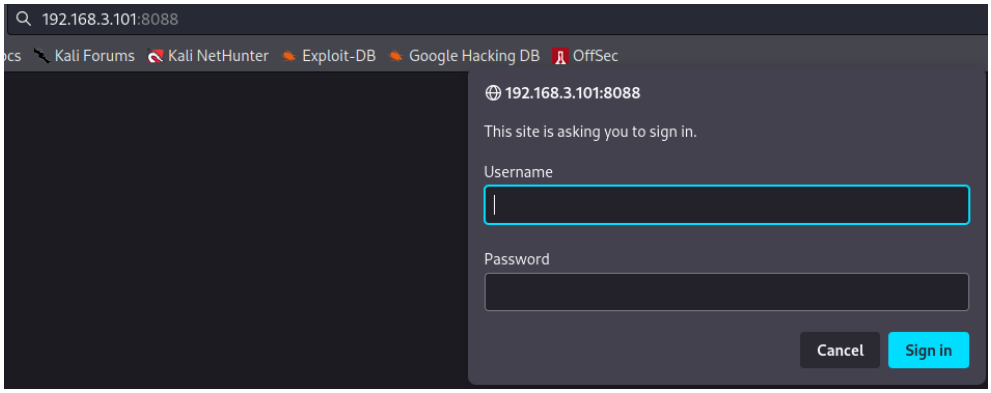

Figure 5.35: Port 8088 has login function

penetration test. One reason is that the password is encrypted in the mydb file and stored safely. This penetration test is performed on a rooted mobile phone, considered an insurance step after the system has yielded. Since the Adb tool requires root privilege to dive into the system, for normal usage, it could be done through some malware or known vulnerabilities on the Android phone. The accessed path is only accessible with root privilege. The rooting process of Xiaomi Redmi Note 11 smart phone is presented in Appendix. B.

All in all, the storage system has security flaws for leaking the users' information. However, this leakage of information requires the android phone to be rooted and a USB or TCP connection. However, as long as the users keep the phone secure and do not make dangerous attempts on unknown URLs or emails, it is considered to be rather safe.

5.7 Identification and authentication failures with HTTP

Broken authentication has been included in OWASP Top 10 since 2017. In 2022 the name was changed to identification and authentication failures which ranks 7th in the top 10 list. This attack usually refers to when an unauthorized attacker acquires legal user credentials or other sensitive information. In this work, the smart camera was earlier discovered to be using HTTP and STUN protocols when communicating. Therefore the authentication attack could be helpful against these two protocols.
Identification and authentication failure has been critical exploitation method for attackers in recent years. The misconfiguration of communicating protocols leaves traceable fingerprints when inputting usernames and passwords to log in. A part of this vulnerability is session management. A typical attack against session management is to hijack the session ID and pretend to be the legitimate user. Another one is hijacking the session by rewriting the session ID URL and navigating into the stream. This is also how the ”Zoombombing” attack happened in 2020, causing zoom calls and public zoom chat to end forcefully [70]. This attack usually leads the unauthorized user into the network session and thus cause a leak of private and sensitive data.

5.7.1 Method

In this attack, according to an earlier discovered HTTP port 8088 URL, burpsuite, ncat [71] and metasploit [72] are used to exploit the session. The URL is earlier discovered in the android storage penetration testing. Burpsuite can intercept the traffic and add credentials according to the intercepted credential format. Also, ncat is used to check the session credential format and HTTP authentication method. Last but not least, a brute force attack will be conducted when burpsuite and ncat fail to exploit the HTTP URL.

5.7.2 Exploitation

First, the author uses burpsuite to intercept the login request for HTTP, the intercepted request is shown in the Figure 5.36. We use ncat to send HTTP requests and try to log in as an unauthorised user, the shown in Figure 5.37. The connection request from ncat display and unauthorised message with WWW-authenticate content. No further information is leaked when trying

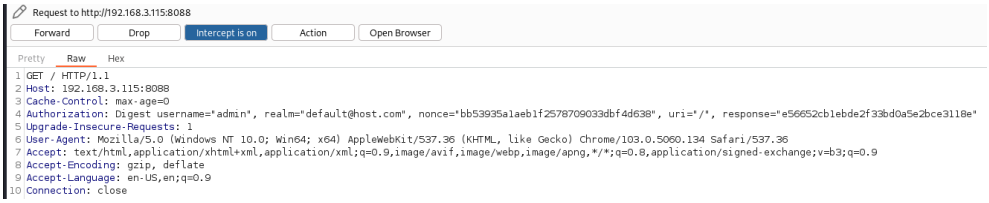

Figure 5.36: Burpsuite intercept HTTP login request

Figure 5.37: unauthorised response from port 8088

to connect. From the two results here, we discover that only the username is displayed in the request. The HTTP use an authentication method with digest and realm.

Given the results of the previous unsuccessful connection, a brute force attack against the HTTP login page, as shown in Figure 5.35. A username list and a password list are downloaded from github. The author also added a few possible login credentials in both lists accordingly. However, given the computing ability and time limit, the brute force attack through metasploit remains no result which can be seen from Figure 5.38.

5.7.3 Post-exploitation

This section conducted attacks on the HTTP URL discovered in the last section. The HTTP connection is able to authorise only legitimate users and not leak more information about the system. Leaving a login port like this is secure to brute force attack in this work, while further attempts will not be conducted due to the delimitation of this work. The HTTP protocol here uses an authentication framework when the user sends a request, the framework of HTTP authentication is shown in Figure 5.39 When the attacker sends out invalid credentials, the server responds with 401 unauthorised message and refuses the connection. To further explain the connection, a mechanism of digest access authentication is applied in the protocol. It uses MD5 hashes with nonce value to prevent replay attacks [73]. The digest access authentication verifies both communicating parties to a shared secret, which is why the

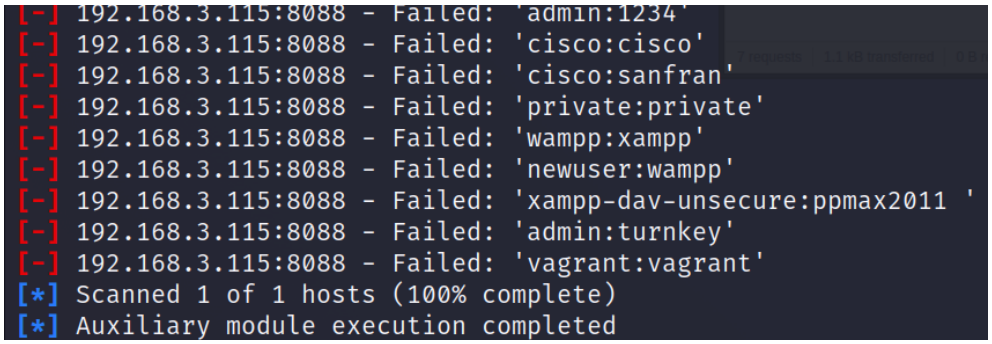

Figure 5.38: Brute force attack with 8088 fails

password does not need to exist in clear text, which prevents the most obvious weakness [74].

One more comment on this attack type: the mobile application's login function has a protection function when encountering too many failed login attempts. After 10 incorrect user email or password attempts, the login function is automatically frozen for 3 minutes, shown in Figure 5.40. This mechanism reasonably protects against identification failure.

5.8 MiTM attack at the client side

The man-in-the-middle attack is a typical cyber attack in which the attackers secretly intercept or monitor the communication between the target and other parties. The trick in this attack is to let both sides believe that they are communicating with each other without knowing the attacker is in between of the data transmission. This attack could also be considered as a lack of verification attack between each party. However, it varies when the target has a specific verification approach for the server to prevent the traffic from being intercepted.

The MiTM attacks include IP spoofing, DNS spoofing, SSL hijacking and eavesdropping, etc. As early as World War II, MiTM attacks are orchestrated by British Intelligence to forge messages to Nazi forces with the intent to demoralise them. In recent years, MiTM attacks remained an efficient method in the range of cyber attacks. A password reset MiTM (PRMiTM) is presented in [75]. The Google and Facebook users in the experiments found that the passwords reset process is vulnerable to PRMiTM attacks. In [76], with the help of SSL strip, SSL encryption alone fails to protect the confidentiality of the communication channel. It can be concluded that MiTM attack has

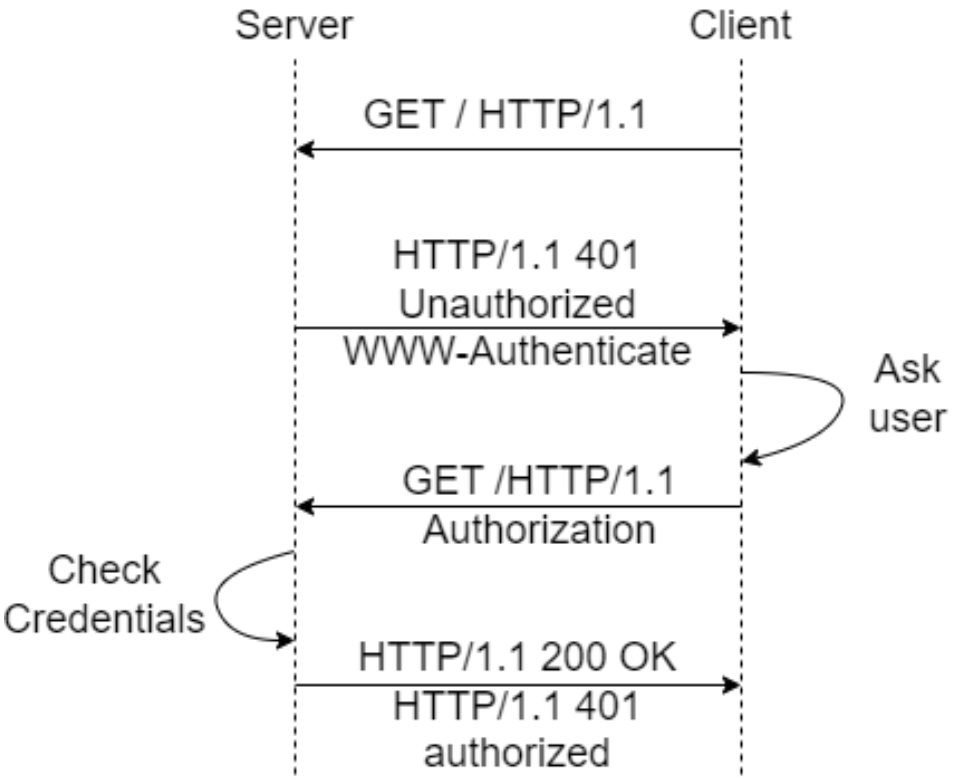

Figure 5.39: HTTP authentication framework

excellent value and a successful rate to be considered in this work.

5.8.1 Method

A common MiTM attack is initiated by installing a data packet sniffer that intercepts network traffic. The attacker could retrieve the user's sensitive data when the users attempt to log in or process any procedure that requires credentials. Nowadays, this is typically done through an insecure website. In this work, a proxy server is configured with Burp Suite [77]. The WiFi proxy on the mobile application is manually configured to redirect to the listening proxy. A CA certificate from Burp Suite needs to be installed in the mobile phone system in order for the traffic to be redirected. Then the credentials the users submitted during this step should be able to be harvested by Burp Suite.

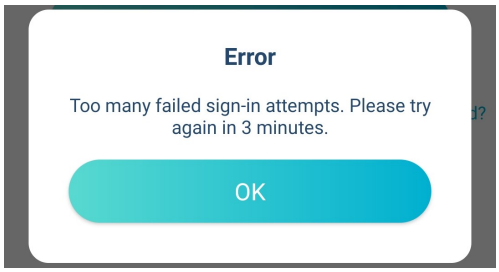

Figure 5.40: 3 minutes waiting time when too many failed sign-in attempts are detected

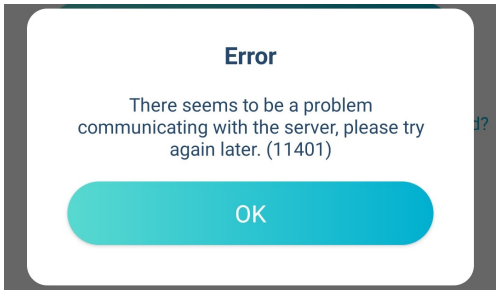

Figure 5.41: Login error when directing to listening proxy

5.8.2 Exploitation

The MiTM attacks with burp suite were initiated and did not reach expectation since the mydlink app detected an unverified server and therefore displayed an error message in Figure 5.41. Since mydlink detects that the communication with the server somehow could be altered, the login function is disabled. Two attempts on both the Android and IoS systems achieved unsatisfying results.

5.8.2.1 Denial of Login and Registration in Android System

The application displays an error message in the Android system when attempting to log in. While registering a new account, the application shows a more specific error that the certificate of the server was different from the expected signing certificate. Without the possibility of accessing the next page of the application, the MiTM attack is stuck in this step, as shown in Figure 5.41, 5.42.

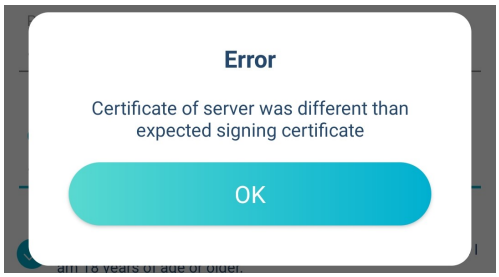

Figure 5.42: Fail to create account when directing to listening proxy in Android

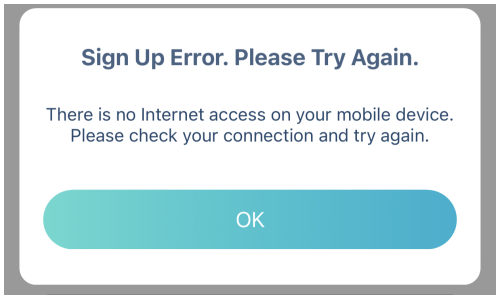

Figure 5.43: Can not log in when directing to listening proxy

5.8.2.2 Denial of Login and Registration in IoS system

Since the MiTM middle attack is not very successful in the Android system, IoS user comes to consideration. Like the Android system, the application displays an error notification after the WiFi proxy is configured and the profile is installed and trusted. When trying to log in, the sign-in button is not accessible because it would trigger a ”No Internet Connection” notification, as shown in Figure 5.43. Furthermore, when trying to create a new account, the signup error notification also indicates no internet access on the mobile device, shown in Figure 5.44.

During the log-out session in both systems, no traffic or packet could be intercepted since the logging-out process does not need to verify the trusted server with a CA certificate.

5.8.3 Post-exploitation

The smart camera is secure against initiated MiTM attack in this work. The attacker is able to configure a proxy and install a CA certificate on both mobile systems, regardless of the possibility of this step in reality. Both systems' mydlink applications detect this modification on the server side since other

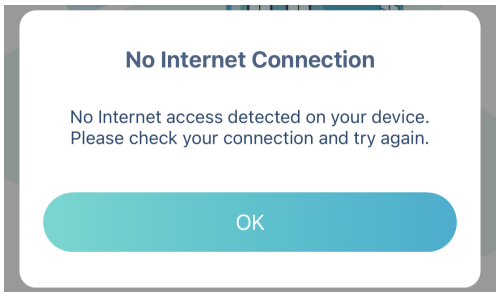

Figure 5.44: Fail to create account when directing to listening proxy in iOS

applications or websites work just fine in the same situation. According to the message from the Android application, the certificate of the server was different from the expected signing certificate. This shows that the application's communication with the server utilizes a verification function, which will verify the server's certificate when sending data packets with critical credentials. The application could use public key-based authentication or certain PKI to ensure the identity of the server. Therefore, prevent MiTM from the fake proxy server from happening. This actually makes more sense since the previous D-Link cameras had a web portal, while the newer generation smart cameras are almost all based on only mobile applications. The promotion of mobile applications increases the security level of the camera greatly. In conclusion, the mydlink application is resilient to MiTM attack in this work.

5.9 Attacks on WiFi direct

WiFi direct is a standard used widely nowadays. It permits IoT devices to connect to the mobile phone without going through a wireless access point (AP). Still, it is probably connecting to the local network as well afterwards. In this case, the mobile phone acts as the ”group owner”. The smart camera could access the local network by connecting to the mobile terminal. In this section, the author will try to take advantage of and trick the victim into connecting to a prone WiFi access point, then intercept the traffic and capture sensitive information from the smart camera.

Wi-Fi direct usually establishes a connection between the device and the ”group owner” through four phases, as shown in the Figure 5.45. So the smart camera sends out a broadcast message trying to find a MAC address, and then the device and the pairing target start a unicast service to the device on the pairing. In this case, the mobile terminal would be the role of the ”group

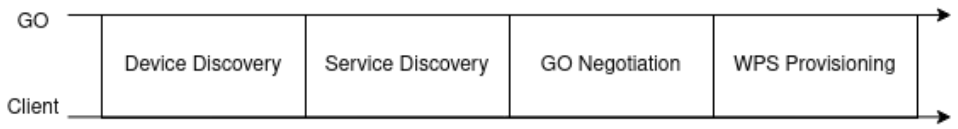

Figure 5.45: Main phases of WiFI direct connection

owner". Afterwards, the Wifi direct use a hand of the Wi-Fi Protected Setup (WPS) to connect the device and the mobile phone in a secure way. The smart camera provides a QR code scanning method to connect to the mobile phone and another way of Pin entry for connection. The pin entry mode is done with a PIN code at the bottom of the smart camera and manually input to the index box on the application.

5.9.1 Method

From the above explanation, the device uses a pin code entry mode when pairing, and this kind of WiFi direct could be vulnerable to brute force attack. Therefore, a brute force attack using reaver [78] is exploited to test the resilience of the smart camera. Reaver is a preinstalled tool on Kali Linux that performs brute force attacks on the AP WPS. Once the pin code is discovered, the AP settings could be consequentially changed unaware. Another attack which is usually for Push-Button Configuration (PBC), is called "EvilDirect Hijacking Attacks". This attack usually interacts with the PBC connection process while the attacker tries to intercept the request for a connection and trick the target device into connecting to a prone system. This attack uses airecrack-ng suite and will also be investigated in this section.

5.9.2 Exploitation

The attack is started during the device's pairing process with the perquisites of a reaver attack undergoing. In this step, the wlan0 BSSID is specified in the command, but nothing was intercepted. The process was running for 100 seconds and more when the author restarted the pairing process for three times. Still, no useful results from reaver, as shown in Figure 5.46.

Then the second attack EvilDirect Hijacking Attack comes to play. First, the wlan0 needs to be set on monitor mode, as shown in Figure 5.47. Then airbase-ng is started targeting the smart camera, which basically forges an AP and encourages the victim to connect to it. The BSSID is randomly written just to satisfy the address. This attack remains failed with no responses after

```
┌──(kali㉿kali)-[~]
└─$ sudo reaver -i wlan0 -b 5C:78:F8:A7:36:08 -vv

Reaver v1.6.6 WiFi Protected Setup Attack Tool
Copyright (c) 2011, Tactical Network Solutions, Craig Heffner <cheffner@tacnetsol.com>

[+] Waiting for beacon from 5C:78:F8:A7:36:08
[+] Switching wlan0 to channel 1
[+] Switching wlan0 to channel 2
[+] Switching wlan0 to channel 3
[+] Switching wlan0 to channel 4
```

Figure 5.46: No beacon received by reaver

```
┌──(kali㉿kali)-[~]
└─$ sudo airmon-ng start wlan0

Found 2 processes that could cause trouble.
Kill them using 'airmon-ng check kill' before putting
the card in monitor mode, they will interfere by changing channels
and sometimes putting the interface back in managed mode

    PID Name
   1507 NetworkManager
   1544 wpa_supplicant

PHY     Interface       Driver          Chipset

phy0    wlan0           rtw_8822ce      Realtek Semiconductor Co., Ltd. RTL8822CE 802.11ac PCIe Wireless Networ
k Adapter
                (monitor mode enabled)
```

Figure 5.47: wlan0 monitor mode is enabled

initiation. The smart camera did not step into the trap of fake AP access points either.

5.9.3 Post-exploitation

The smart camera is secure against attacks on WiFi direct in this work. The two attacks performed in this section both failed without any response. The WiFi direct function of the smart camera successfully bypasses these two attacks. This could be the newer technology of near-field communication in use or the smart camera pairs with the mobile terminal in some other ways. The reason is also the pre-selection of the pairing smart camera model, which implies that they might communicate in a certain frequency for connection in this pairing process. The security measure on pairing is well deployed since most users will choose a QR code for connection when they have access to the QR code instead of the pin code.

However, the insecure elements of WPS remain. The mitigation for this could be enabling MAC address filtering to make sure there is no phishing hook in the network and disabling the WPS function.

```
┌──(kali㉿kali)-[~]
└─$ sudo airbase-ng -c 6 -e Z-man -a BB:BB:BB:BB:BB:BB wlan0
ioctl(SIOCSIWMODE) failed: Device or resource busy
21:48:37  Created tap interface at0
21:48:37  Trying to set MTU on at0 to 1500

ti_set_mac failed: Cannot assign requested address
You most probably want to set the MAC of your TAP interface.
ifconfig <iface> hw ether BB:BB:BB:BB:BB:BB


21:48:37  Access Point with BSSID BB:BB:BB:BB:BB:BB started.
```

Figure 5.48: Airbase did not lure the victim to the forged AP

5.10 Denial of Service attack

Denial of Service (DoS) attack aims to drain the wireless or system resource on the target device, which forces the service to be shut down or not able to connect regularly. Typical consequences include super slow content loading, denial of visit requests, can not connect to service, etc. One of the most famous DoS attack happened in 2017, which was targeted at Google service. The scale runs up to 2.54 Tbps [79]. This attack could test the robustness of the wireless connection of the smart camera, which is essential for a home security camera. In this section, the author successfully deauthenticate the client from accessing the camera live view and other actions on the smart camera.

Deauthentication attack is one of the most potent Dos attacks. The attacker sends a deauthentication form to a wireless access point with a spoofed MAC address for the target device [80]. The attack ideally would result in a disconnection between the victim and wireless access point (AP), thus deauthenticate the user from the target. As stated in [81], a majority of DoS attacks takes advantage of the unencrypted management and control frames since all the encryption method of 802.11 standard merely encrypted the frames of the data. A tool called Aireplay-ng [82]is introduced for the deauthentication attack. The main capability of aireplay-ng is to generate traffic for later on aircrack-ng needs for cracking the WEP and WPA-PSK keys [83].

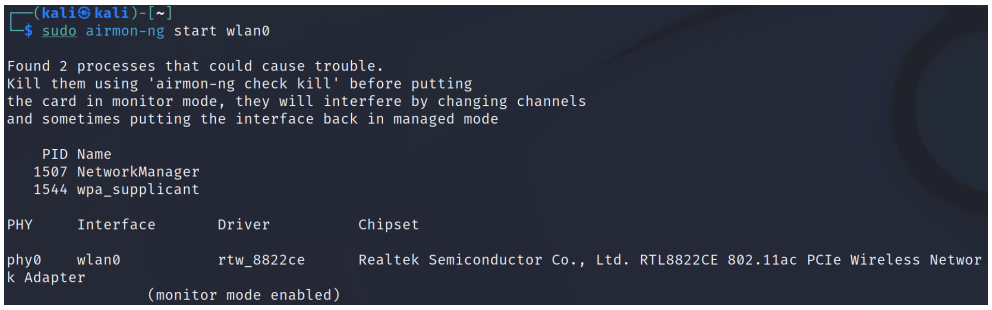

Figure 5.49: wlan0 monitor mode enabled

5.10.1 Method

The method of the DoS attack is present as follows, first of all, all the communication in the wireless network needs to be terminated and start again to ensure the attack environment. Moreover, the wireless adapter monitor mode needs to be enabled. Then, wireless transfer from the wireless AP and the BSSID of the router needs to be captured. Last but not least, use Aireplay-ng to send deauthentication frames to the smart camera's MAC address to deauthenticate the user from accessing the camera's service.

5.10.2 Exploitation

All the wireless communication in the wireless environment is checked and killed, then restarted for this experimental setup with the command "sudo airmon-ng check kill". The iwconfig command could be useful to check the communication state in the experimental network for existing wireless communication. Then the WiFi adapter is restarted with the command "sudo airmon-ng start wlan0" and switched to monitor mode, as shown in Figure 5.49. The sudo airedump-ng command is applied on wlan0 to discover the router's MAC address (BSSID) and channel for the router and ESSID as well. We need to switch the WiFi adapter channel to the target router with the command "sudo iwconfig wlan0 channel 1" in this test. To this end, all the preparation for the attack is set up. We can finally start the deauthentication attack.

The -0 represents deauthentication attack. One could also use –deauth for the same use, and the second 0 decides the running time of the attack, where 0 represents infinite running time. Furthermore, from the Figure 5.51, it can be seen that the deauthentication is undergoing, and we can check if the mobile application can connect to the smart camera now. The deauthentication attack

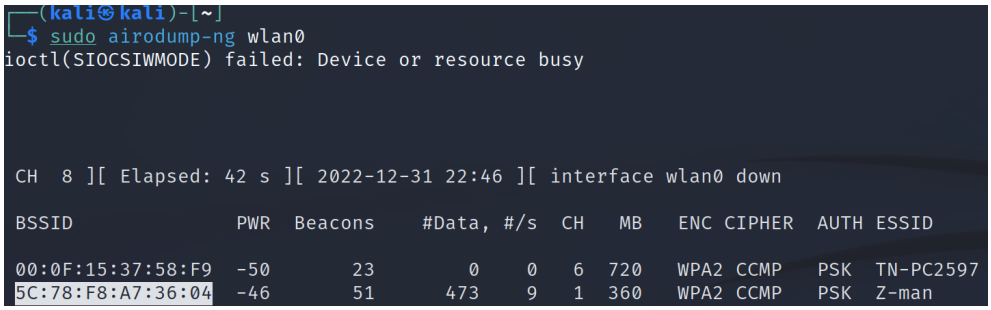

Figure 5.50: discover the target's BSSID and channel

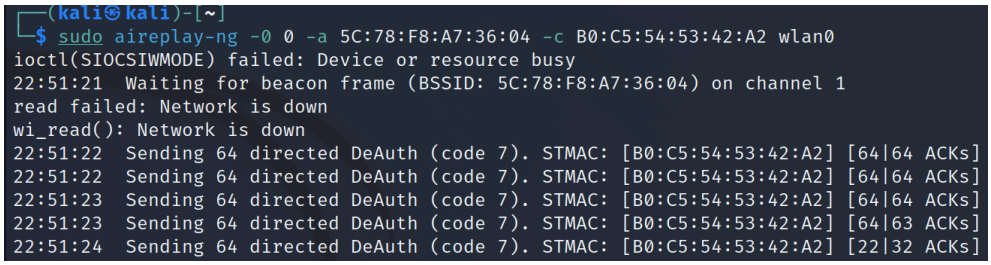

Figure 5.51: Deauthentication attack running

appears to be successful given the fact that the user could really access the service in any kind as it's shown in Figure 5.52 and Figure 5.53. During the time that deauthentication attack is undergoing, the smart camera is inaccessible through the mobile application. After loading for an annoyingly long time, a pop-out window displays a notification that the application can not connect to the device, meaning the connection timed out. When the author tries to access the camera after closing the pops out, the device appears to be offline in the user interface. This attack is considered to be successful due to this unavailability.

5.10.3 Post-exploitation

The smart camera is shown not secure against DoS attack in this work. This attack needs certain prerequisite on the attackers machine, which are not very hard to satisfy. At this point, the author understands why this attack is useful against big corporations and service providers (SP). As aforementioned, this kind of attack could be prevented by a two-way key exchange protocol with the token exchange. The solution in [26] applies cryptographic functions and the generation of unique tokens for a more secure protocol to prevent such deauthentication attacks targeting 802.11 wireless communications. Note that

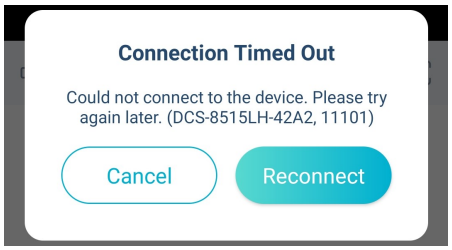

Figure 5.52: Connection timeout pops out

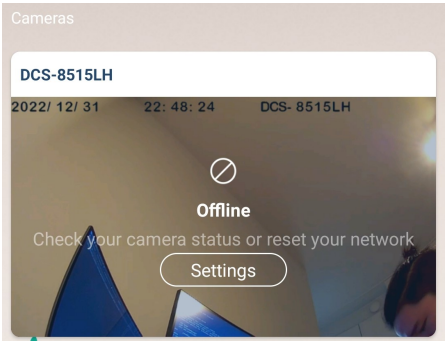

Figure 5.53: Device Offline

when the attack is initiated in a virtual Kali Linux machine, the testers might need a plug-in wireless adapter to enable the WLAN function. During the check and kill process, it could shut down all the internet connections in the machine. To restart the network service on the experiment virtual machine, simply use the ”sudo service Network Manager restart” command.

5.11 Firmware reverse engineering

Firmware is a software inserted directly into the device where it is explicitly designed to cope with the hardware. Usually, the vendor provides automatic updates or sources of the latest updated firmware for the users to install on the device. At the same time, a firmware attack is when an attacker injects malicious code into the firmware and injects the ”newly updated” firmware into the target device. In this section, the author will use reverse engineering to test the firmware security of the smart camera.
The unauthorised update of the firmware could potentially alter the default settings of the device, which leaves a hidden backdoor for the attacker to dive into the system. Without serious restriction or updating, the attacker could invade the local network with the compromised device. The advantage of this is that it is tough to be detected and not easy to remove. The compromise of one device is commonly not noticable. However, when attacks on a larger scale happen, usually, there is a backdoor hidden in the firmware.

5.11.1 Method

Typical methods of the attack need to open the case of the device and identify the memory component, then use a UART cable to download the firmware

Type	Firmware
Description	Firmware: DCS-8515LH A2 FW v1.07.01
Download	DCS-8515LH A2_Release Note_for_FW_V1.07.01.pdf 20210909-DCS-8515LH_v1.07.01.bin
Last modified	2022/04/14

Figure 5.54: Firmware successfully found and downloaded

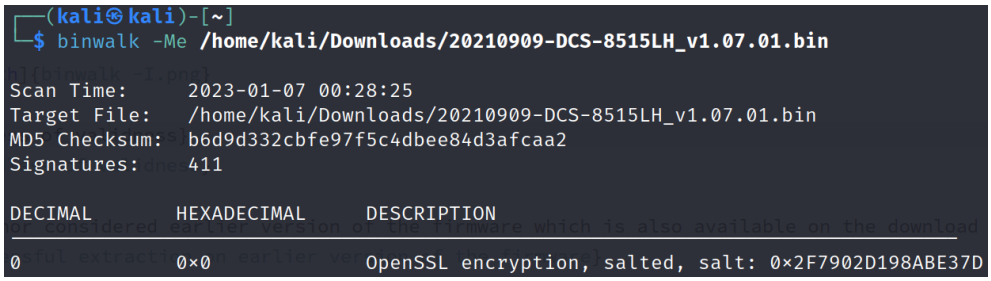

Figure 5.55: The binwalk extracted step appears to be failure

from the device. However, the hardware operations are out of the scope of the thesis. Therefore the author found published firmware on the support website of the D-Link corporate. A tool called binwalk [84] is used in this attack to analyse the firmware found.

5.11.2 Exploitation

The author first uses the format of Google Dork ”title=DCS-8515LH” ”intext=Firmware Download” to search for DCS-8515LH firmware on the internet. Then after clicking around more search on the index box, the author managed to find the firmware on the support page of D-Link TW as the Figure 5.54 shown. This version of the firmware is confirmed to be the latest version and the current version on the target smart camera. Then, let us proceed to the binwalk and try to analyze the firmware by seeing what is inside first. So this extraction does not succeed, the author uses the ”binwalk -I” command to check to see invalid results. The invalid output is indeed inaccurate since the .bin file is significantly smaller than the image size displayed. Upon this unsuccessful attempt, the author considered an earlier version of the firmware, which is also available on the download page. The result is shown in Figure 5.57.

Two of the above attempts did not give desired extracted firmware.

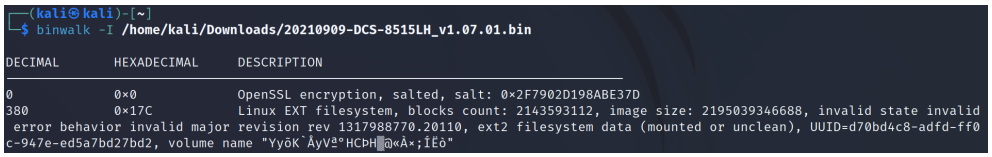

Figure 5.56: Display all results regardless of validness

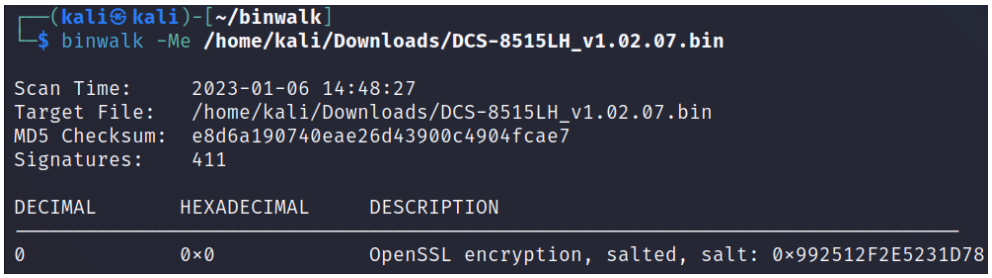

Figure 5.57: Unsuccessful extraction on earlier version of the firmware

5.11.3 Post-exploitation

The smart camera is shown to be secure against the inspection in this attack. Reverse engineering is mainly carried out for a few purposes, except for curiosity about the device. It could also help software or hardware developers to do security checks that no one could understand or get access to this internals. The last possibility is a security researcher when doing a security audit on a piece of hardware to examine the safety and security level of the device.

Here the author did not find any further detailed information on the firmware. However, an MD5 checksum and salt hash function are discovered. The MD5 checksum is applied to the source code to check the integrity of the data. Moreover, the salt hash function is used on the MD5 checksum for extra strong encryption of the firmware integrity.

To prevent these kinds of attacks, users could be more alarmed and cautious about an unknown device firmware update.

Chapter 6

Discussion

Among the 11 penetration tests reported in this work, the DoS attacks are proven to be effective in general. Exploiting the zero-configuration protocol results in a DoS. Using airmon-ng suite, deauthentication attack blocks the video stream for the users. Although DoS attack is known as the most common attack and easiest to implement on IoT devices, it should raise security considerations on IoT devices to prevent such attacks. The DoS attacks are again proven in this work to be adequate for the smart camera. These attacks result in the lost of the video stream, which is the most crucial function of this smart camera. By disabling this capability, the attacker could start further criminal activities since the environmental surveillance is gone for good. When any serious incident happens in the monitor area, the users are still nowhere to be informed as they should be.

The rest of the attacks are potentially useful for further ethical hacking on this smart camera. The network inspection discovers massive URLs for the rtsp protocol and the OS for the device system. This work did not successfully exploit any open ports on the smart camera, and the URL value remains a mystery use.

Decomplication of the Android application exposes some important files like AndroidManifest.xml, which could be beneficial in initiating future attacks. Moreover, according to the jadx tool, the puzzled classes and another file in clear text could be analyzed and searched. Injecting malicious code into the application and uploading it to some open-source application sites can trick unaware users into installing it.

The ARP poisoning on the traffic discovered vulnerable protocols like HTTP, and slightly outdated protocols like TLSv1.2. This protocol is vulnerable to more advanced attacks and needs further investigation.

However, the other protocols found, except for mDNS protocol, are authenticated between the server and the client.

The RTSP protocol is tested and proven to be properly secured on the smart camera. Based on the no-response requests sent in this work, the RTSP protocol is healthy and securely working.

In the Android storage system, the application is supposed to store nothing but the log files. However, some sensitive information data of the user, like email and location, are stored in clear text. Luckily, the password is not stored in the system. Another important file is discovered, the system event. This file indicates whether the camera is functioning well and the connection status, thus, should have been protected accordingly.

Identification and authentication failures are new names for broken authentication since 2022. An HTTP URL with a login function is tested with burp suite and brute force attack. Although burp suite could not insert anything or replace anything in the request, the brute force attack is expected to succeed with enough attempts since the login has no restriction on the number of attempts.

The MiTM attack failed in this work. The mobile application and the server communicate with a signed certificate; therefore, refuse the connection from the proxy server. This shows proper protection against MiTM attacks on the client side.

Attacks on the WiFi direct function forged an AP to trick the camera into connecting, however this attack did not affect the smart camera. It can be concluded that the smart camera use more advanced near-field communication technologies with the phone when in the same local network.

The firmware is available on the TW support page. The firmware is encrypted with OpenSSL and hashed with MD5 and salt functions. Extraction or inspection is not performing well, thus indicating that the firmware is appropriately encrypted and secured.

One thing that should be mentioned is that the camera is somehow in a bug when turning on. The camera used to rotate to full left, then full right when plugging in the power cable. However, after a metasploit attack on mDNS protocol, the camera rotated to full left and stuck with the gear stuck sound for several seconds.

Chapter 7

Conclusions and Future work

This chapter draws conclusions of this work and states possible future work accordingly.

7.1 Conclusions

All in all, the smart camera is resilient to most of the attacks performed in this work. The security implement shows precautions and strong defence against cyber attacks. Although, it is vulnerable to mDNS protocol and DoS attacks. Some of the insights about the security level on the camera include but are not limited to: Improving the storage system security level. Using zeroconf protocol with authentication on the environment and performing DoS Cloud-based security or setting up a response plan when encountering such an attack.

The results meet the goal of this work and alert developers of the smart camera and the mobile application to improve the security level in some aspects. The selected attacks in this work are moderately effective since the camera is out of function under some attacks. More attacks based on this work should open up more security problems and weaknesses in the system. More specifically, attacks on inserting malicious code in the mobile application are doable, and more penetration tests on testing the communication protocols are also reasonable.

The usage of IoT devices has been blowing up in recent years with more advanced technology. However, it is worth noting to keep the security implementation along as well. For example, zero-configuration protocols need to be implemented with corresponding security configurations. Furthermore, the DoS attacks remain a security problem for the smart camera, just like most IoT devices. Also, updating the firmware and the mobile application

is necessary to prevent attacks on the current version.

7.2 Limitations

The most obvious limitation of this work is time constraints. The author has much passion for ethical hacking and penetration testing. However, due to the limited time of the thesis, the number of pentests produced is limited.

Other limitations include hardware hacking, cloud hacking, Bluetooth communication testing, radio hacking, and IoS application.

7.2.1 Delimitation in this work

7.2.1.1 Hardware

The author's previous knowledge is mostly based on ethical hacking of software, wireless communication and databases. Hardware hacking, however, is outside of the author's knowledge base. Therefore, hardware hacking is out of consideration.

7.2.1.2 Cloud hacking

Cloud hacking is beneficial for cloud exploitation by offensive professionals. However, the cloud server for the camera is managed by D-Link corporation, which is unethical to attack or perform any kind of penetration testing on. Therefore, the cloud hacking test is outside this work's scope.

7.2.1.3 Bluetooth communication

Bluetooth hacking includes Bluetooth jacking, where one Bluetooth device hijacks and sends spam advertising or other messages to another device if they are within 10-30 feet [85]. Although these kinds of attacks need specific Bluetooth devices, it is out of reach of this work.

7.2.1.4 Radio hacking

Radio hacking is related to hardware hacking, where the attackers either jam or replay the signals. For example, jamming deafens any receivers within reach, which will have a crucial impact on radio wave systems. However, this requires hardware support and is beyond this work's scope.

7.2.1.5 IoS application

IoS application is significantly more encrypted than the Android application, with the encrypted installation packets and non-reversible features of the IPA. In addition, the penetration tests mostly require an IoS Jailbroken device which is also rather difficult to acquire. Thus, tests on the IoS application will not be carried out in this work.

7.3 Future Work

Owing to the limited time and author's knowledge, the penetration tests in this work are limited. There are more attacks that could be potentially useful on the smart camera. Alternatively, in another perspective, further investigation could be carried out based on the tested attacks.

The open ports on the system are proven secure the attacks in this work. However, more attacks could be generated in this direction. A telnet connection is an interesting way to communicate with the system. While the requests tested in this work are not fruitful, more requests could be tested until finding favourable requests. Also, the cloud server is untested since it is against the ethics of this work, but it's still a very important part of the whole networking architecture. With permission from the company, the cloud server should also be tested. The HTTP login URL is exploitable since it has no restriction on attempts.

Security testing is considered a long-term project since no system is perfect to modern attacks. While the mobile application and firmware are updating within time, security checks should also be conducted on every updated version.

Last but not least, with the ability to eliminate the limitations of this work, all the limited scope of this work should also be tested devotedly.

7.4 Sustainability and Ethics

All the equipment used to set up the experimental environment could be reused for other penetration tests. The smart camera and the mobile phone are returned to the NSE lab for future usage of other students. This work is done in the author's home network, which is set up as the experimental environment. All the personal information in the system is from the author, therefore not violating anyone's privacy.

The scope is strictly limited to the smart camera, mobile application and the Redmi Note 11. Noted that any testing on the server or cloud server should be permitted by the vendor. Any illegal scanning or tampering on the server could cause legal matters, so as an ethical hacker, keep the hands in the pocket.

The DCS8515-LH smart camera and Redmi Note 11 mobile phone are provided by NSE lab, and the author has gained permission before rooting the mobile phone.

References

[1] Darren Anstee, “Rise of the internet of things (iot),” 2019. [Online]. Available: https://www.techradar.com/news/rise-of-the-internet-of-things-iot [Page 1.]

[2] K. Patel, S. Patel, P. Scholar, and C. Salazar, “Internet of things-iot: Definition, characteristics, architecture, enabling technologies, application & future challenges,” pp. 6122–6131, 2016. [Page 1.]

[3] Sharath Acharya, “Why iot security is important for today’s networks?” 2022. [Online]. Available: https://sectrio.com/why-iot-security-is-important-for-todays-networks/ [Page 1.]

[4] unknown, “Hacking wifi cameras dangers & precautions,” unknown. [Online]. Available: https://www.cctvsg.net/hacking-wifi-cameras-dangers-and-precautions/ [Page 2.]

[5] Alexander S. Gillis, “What is the internet of things (iot)?” 2021. [Online]. Available: https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT [Page 2.]

[6] Josh Howarth, “80+ amazing iot statistics (2022-2030),” 2022. [Online]. Available: https://explodingtopics.com/blog/iot-stats [Page 2.]

[7] Paul Miller, “Ibm bets 3 billion dollars on internet of things opportunity,” 2015. [Online]. Available: https://www.forbes.com/sites/paulmiller/2015/03/31/ibm-bets-3-billion-on-internet-of-things-opportunity/?sh=187557e8a485 [Page 2.]

[8] H.-H. Kim and J. Yoo, “Analysis of security vulnerabilities for iot devices.” *Journal of Information Processing Systems*, vol. 18, no. 4, 2022. [Page 2.]

[9] J. Bugeja, D. Jönsson, and A. Jacobsson, "An investigation of vulnerabilities in smart connected cameras," in *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2018. doi: 10.1109/PERCOMW.2018.8480184 pp. 537–542. [Page 2.]

[10] J. Bugeja, A. Jacobsson, and P. Davidsson, "On privacy and security challenges in smart connected homes," in *2016 European Intelligence and Security Informatics Conference (EISIC)*, 2016. doi: 10.1109/EISIC.2016.044 pp. 172–175. [Page 3.]

[11] The PTES team, "The penetration testing execution standard," 2014. [Online]. Available: http://www.pentest-standard.org/index.php/Main_Page [Pages 4 and 11.]

[12] Microsoft, "The stride threat model," 2009. [Online]. Available: https://learn.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)?redirectedfrom=MSDN [Pages 4 and 23.]

[13] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying iot security: An exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2702–2733, 2019. doi: 10.1109/COMST.2019.2910750 [Page 5.]

[14] Australian Cyber Security Centre, "Before purchasing an iot device," 2020. [Online]. Available: https://www.cyber.gov.au/acsc/view-all-content/guidance/purchasing-iot-device [Page 5.]

[15] Miloš Čermák, Milan Fránik, "D-link camera vulnerability allows attackers to tap into the video stream," 2019. [Online]. Available: https://www.welivesecurity.com/2019/05/02/d-link-camera-vulnerability-video-stream/ [Page 6.]

[16] Sam Humphries, "4 stages of vulnerability management: A process for risk mitigation," Sam Humphries. [Online]. Available: https://www.exabeam.com/information-security/vulnerability-management/ [Page 6.]

[17] ——, "4 stages of vulnerability management: A process for risk mitigation," 2022. [Online]. Available: https://www.exabeam.com/information-security/vulnerability-management/ [Page 6.]

[18] First Organization, "Common vulnerability scoring system version 3.1: Specification document," 2019. [Online]. Available: https://www.first.org/cvss/specification-document [Page 6.]

[19] C. Heffner, "Exploiting surveillance cameras like a hollywood hacker. 2013. url:https://media.blackhat.com/us-13," *US-13-Heffner-Exploiting-Network-Surveillance-Cameras-Like-A- Hollywood-Hacker-WP. pdf*. [Page 6.]

[20] Y. Seralathan, T. T. Oh, S. Jadhav, J. Myers, J. P. Jeong, Y. H. Kim, and J. N. Kim, "Iot security vulnerability: A case study of a web camera," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*, 2018. doi: 10.23919/ICACT.2018.8323686 pp. 172–177. [Page 7.]

[21] J. Valente, K. Koneru, and A. Cardenas, "Privacy and security in internet-connected cameras," in *2019 IEEE International Congress on Internet of Things (ICIOT)*, 2019. doi: 10.1109/ICIOT.2019.00037 pp. 173–180. [Page 7.]

[22] T. Doughty, N. Israr, and U. Adeel, "Vulnerability analysis of ip cameras using arp poisoning," in *8th International Conference on Soft Computing, Artificial Intelligence and Applications (SAI 2019)*, 2019, pp. 163–172. [Page 7.]

[23] Lilith of Cisco Talos, "Rtspserver code execution vulnerability," 2018. [Online]. Available: https://github.com/r3dxpl0it/RTSPServer-Code-Execution-Vulnerability [Page 8.]

[24] A. Hadiks, Y. Chen, F. Li, and B. Liu, "A study of stealthy denial-of-service attacks in wi-fi direct device-to-device networks," in *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*, 2014. doi: 10.1109/CCNC.2014.6994425 pp. 507–508. [Page 8.]

[25] A. M. López, F. A. Mendoza, P. A. Cabarcos, and D. D. Sánchez, "Wi-fi direct: Lessons learned," in *2016 Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2016. doi: 10.1109/MedHocNet.2016.7528493 pp. 1–8. [Page 8.]

[26] A. Arora, "Preventing wireless deauthentication attacks over 802.11 networks," 2019. [Online]. Available: https://arxiv.org/abs/1901.07301 [Pages 8 and 60.]

[27] R. Akhilesh, O. Bills, N. Chilamkurti, and M. J. M. Chowdhury, "Automated penetration testing framework for smart-home-based iot devices," *Future Internet*, vol. 14, no. 10, 2022. doi: 10.3390/fi14100276. [Online]. Available: https://www.mdpi.com/1999-5903/14/10/276 [Page 9.]

[28] Pete Herzog, "Osstmm.3," 2010. [Online]. Available: https://www.isecom.org/research.html#content5-9d [Page 11.]

[29] STF, "Gray box testing," 2020. [Online]. Available: https://softwaretestingfundamentals.com/gray-box-testing/ [Page 12.]

[30] A. V. Uzunov and E. B. Fernandez, "An extensible pattern-based library and taxonomy of security threats for distributed systems," *Computer Standards & Interfaces*, vol. 36, no. 4, pp. 734–747, 2014. [Page 12.]

[31] Microsoft, "Microsoft threat modeling tool threats," 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats [Page 12.]

[32] jegeib, brittmsantos, DavidCBerry13, msmbaldwin, barbkess, "Microsoft threat modeling tool threats," 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats [Page 12.]

[33] J.D. Meier, Alex Mackman, Michael Dunner, Srinath Vasireddy, Ray Escamilla and Anandha Murukan, "Improving web application security: Threats and countermeasures," 2003. [Online]. Available: https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ff648644(v=pandp.10) [Page 12.]

[34] ImmuniWeb, "Immuniweb ai for application security," 2022. [Online]. Available: https://www.immuniweb.com/ [Pages 13 and 32.]

[35] M. Bach-Nutman, "Understanding the top 10 owasp vulnerabilities," 2020. [Online]. Available: https://arxiv.org/abs/2012.09960 [Page 13.]

[36] Andrew van der Stock, Brian Glas, Neil Smithline, Torsten Gigler, "Owasp top ten," 2022. [Online]. Available: https://owasp.org/www-project-top-ten/ [Page 13.]

[37] V. Chandrika, "Ethical hacking: Types of ethical hackers," *International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)*, vol. 11, no. 1. [Page 14.]

[38] T. C. team, “A complete guide to the phases of penetration testing,” 2022. [Online]. Available: https://cipher.com/blog/a-complete-guide-to-the-phases-of-penetration-testing/ [Page 14.]

[39] J. Rosenberg, J. Weinberger, C. Huitema, R. Mahy, “Stun - simple traversal of user datagram protocol (udp) through network address translators (nats),” 2003. [Online]. Available: https://www.rfc-editor.org/rfc/rfc3489 [Pages 19 and 38.]

[40] AJ Kumar, “Discovering entry points,” 2016. [Online]. Available: https://resources.infosecinstitute.com/topic/discovering-entry-points/ [Page 21.]

[41] Chiradeep BasuMallick, “Owasp top 10 vulnerabilities in 2022,” 2022. [Online]. Available: https://www.spiceworks.com/it-security/vulnerability-management/articles/owasp-top-ten-vulnerabilities/ [Page 24.]

[42] T. Contributors, “Network scanning,” 2015. [Online]. Available: https://www.techopedia.com/definition/16124/network-scanning [Page 27.]

[43] G. F. Lyon, *Nmap network scanning: The official Nmap project guide to network discovery and security scanning*. Insecure. Com LLC (US), 2008. [Page 28.]

[44] P. Karlsson, “Script rtsp-url-brute,” 2022. [Online]. Available: https://nmap.org/nsedoc/scripts/rtsp-url-brute.html [Page 28.]

[45] “CVE-2014-4879,” Available from MITRE, CVE-ID CVE-2014-4879., 2014. [Online]. Available: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-4879 [Page 29.]

[46] John Sopko, “Andrew file system,” 2014. [Online]. Available: https://cs.unc.edu/resources/andrew-file-system-afs/ [Page 29.]

[47] C. Cifuentes, T. Waddington, and M. Van Emmerik, “Computer security analysis through decompilation and high-level debugging,” 02 2001. doi: 10.1109/WCRE.2001.957846. ISBN 0-7695-1303-4 pp. 375 – 380. [Page 30.]

[48] SPI DYNAMICS, “Decompilation and application security,” 2003. [Online]. Available: https://www.blackhat.com/presentations/bh-usa-03/bh-us-03-spett.pdf [Page 30.]

[49] mydlink, “mydlink,” 2022. [Online]. Available: https://se.mydlink.com/content/productfamily [Page 30.]

[50] Ben Lutkevich, “reverse-engineering,” 2021. [Online]. Available: https://www.techtarget.com/searchsoftwarequality/definition/reverse-engineering [Page 31.]

[51] Connor Tumbleson - Current Maintainer, Ryszard Wiśniewski - Original Creator, “Apktool,” 2022. [Online]. Available: https://ibotpeaches.github.io/Apktool/ [Page 31.]

[52] KunYuChen, haeter525, Surendrajat, Shaun Dang, “Quark-engine,” 2022. [Online]. Available: https://github.com/quark-engine [Page 31.]

[53] pxb1988, “dex2jar,” 2021. [Online]. Available: https://github.com/pxb1988/dex2jar [Page 31.]

[54] skylot, “jadx,” 2022. [Online]. Available: https://github.com/skylot/jadx [Pages 31 and 34.]

[55] SQLite, “What is sqlite,” 2022. [Online]. Available: https://www.sqlite.org/index.html [Page 33.]

[56] Wireshark, “Wireshark,” 2022. [Online]. Available: https://www.wireshark.org/ [Page 35.]

[57] F. Kausar, S. Aljumah, S. Alzaydi, and R. Alroba, “Traffic analysis attack for identifying users’ online activities,” *IT Professional*, vol. 21, no. 2, pp. 50–57, 2019. doi: 10.1109/MITP.2018.2876988 [Page 36.]

[58] emanuele f, “Pcapdroid,” 2022. [Online]. Available: https://github.com/emanuele-f/PCAPdroid [Page 36.]

[59] Alberto Ornaghi, Marco Valleri,.etc.., “Ettercap-project,” 2022. [Online]. Available: https://www.ettercap-project.org/index.html [Page 36.]

[60] A-Packets developers, “A-packets,” 2022. [Online]. Available: https://apackets.com/ [Page 36.]

[61] F. Siddiqui, S. Zeadally, T. Kacem, and S. Fowler, “Zero configuration networking: Implementation, performance, and security,” *Computers & electrical engineering*, vol. 38, no. 5, pp. 1129–1145, 2012. [Page 38.]

[62] Anotnios Atlasis, “Pholus,” 2022. [Online]. Available: https://github.com/aatlasis/Pholus [Page 39.]

[63] F. Chantzis, I. Stais, P. Calderon, E. Deirmentzoglou, and B. Woods, *Practical IoT Hacking: The Definitive Guide to Attacking the Internet of Things*. No Starch Press, 2021. ISBN 9781718500914. [Online]. Available: https://books.google.se/books?id=GbYEEAAAQBAJ [Page 39.]

[64] H. Schulzrinne, A. Rao, R. Lanphier, “Real time streaming protocol (rtsp),” 1998. [Online]. Available: https://www.rfc-editor.org/rfc/rfc2326.html [Page 41.]

[65] Ullaakut, “Cameradar,” 2022. [Online]. Available: https://github.com/Ullaakut/cameradar [Page 42.]

[66] VideoLAN organization, “Vlc media player for debian gnu/linux,” 2022. [Online]. Available: https://www.videolan.org/vlc/download-debian.html [Pages 42 and 43.]

[67] Android developers, “Android debug bridge (adb),” 2023. [Online]. Available: https://developer.android.com/studio/command-line/adb [Page 46.]

[68] Y. King, “mydb 0.1.2,” 2022. [Online]. Available: https://pypi.org/project/mydb/ [Page 47.]

[69] Github Contributors, “Db browser for sqlite,” 2022. [Online]. Available: https://sqlitebrowser.org/ [Page 47.]

[70] Taylor Lorenz, “‘zoombombing’: When video conferences go wrong,” 2020. [Online]. Available: https://www.nytimes.com/2020/03/20/style/zoombombing-zoom-trolling.html [Page 49.]

[71] Anatoly Techtonik, Mike Frysinger, “Netcat 1.10,” 2018. [Online]. Available: https://sourceforge.net/p/nc110/git/ci/master/tree/ [Page 49.]

[72] rapid7, “Metasploit-framework,” 2022. [Online]. Available: https://github.com/rapid7/metasploit-framework [Page 49.]

[73] R. Shekh-Yusef, Ed., D. Ahrens, S. Bremer, “Http digest access authentication,” 2015. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7616 [Page 50.]

[74] P. J. Franks, P. Hallam-Baker, L. C. Stewart, J. L. Hostetler, S. Lawrence, P. J. Leach, and A. Luotonen, "HTTP Authentication: Basic and Digest Access Authentication," RFC 2617, Jun. 1999. [Online]. Available: https://www.rfc-editor.org/info/rfc2617 [Page 51.]

[75] N. Gelernter, S. Kalma, B. Magnezi, and H. Porcilan, "The password reset mitm attack," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 251–267. [Page 51.]

[76] A. R. Chordiya, S. Majumder, and A. Y. Javaid, "Man-in-the-middle (mitm) attack based hijacking of http traffic using open source tools," in *2018 IEEE International Conference on Electro/Information Technology (EIT)*, 2018. doi: 10.1109/EIT.2018.8500144 pp. 0438–0443. [Page 51.]

[77] PortSwigger, "Burp suite editions," 2022. [Online]. Available: https://portswigger.net/burp [Page 52.]

[78] Craig Heffner, "Reaver," 2022. [Online]. Available: https://www.kali.org/tools/reaver/ [Page 56.]

[79] Sumeet Wadhwani, "How google foiled the largest ddos attack by chinese hackers," 2020. [Online]. Available: https://www.spiceworks.com/it-security/network-security/news/how-google-foiled-the-largest-ddos-attack-by-chinese-hackers/ [Page 58.]

[80] John Bellardo, Stefan Savage, "802.11 denial-of-service attacks:real vulnerabilities and practical solutions," 2003. [Online]. Available: http://users.csc.calpoly.edu/~bellardo/pubs/usenix-sec03-80211dos-html/aio.html [Page 58.]

[81] M. Agarwal, S. Biswas, and S. Nandi, "Detection of de-authentication dos attacks in wi-fi networks: A machine learning approach," in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, 2015. doi: 10.1109/SMC.2015.55 pp. 246–251. [Page 58.]

[82] MISTERX, "Aircrack-ng 1.7," 2022. [Online]. Available: https://aircrack-ng.blogspot.com/2022/05/aircrack-ng-17.html [Page 58.]

[83] Aireplay-ng, "Aireplay-ng," 2022. [Online]. Available: https://www.aircrack-ng.org/doku.php?id=aireplay-ng [Page 58.]

[84] devttys0 and others from ReFirm Labs, "binwalk," 2022. [Online]. Available: https://github.com/ReFirmLabs/binwalk [Page 62.]

[85] Shannon Flynn, “Bluejacking: How bluetooth can be used to hack your devices,” 2022. [Online]. Available: https://www.makeuseof.com/bluejacking-hack-your-devices/ [Page 67.]

[86] X. Developers/MIUI, “Apply for permissions to unlock mi devices,” 2022. [Online]. Available: https://en.miui.com/unlock/download_en.html [Page 81.]

[87] topjohnwu, “Magisk,” 2022. [Online]. Available: https://github.com/topjohnwu/Magisk [Page 81.]

[88] A. O. S. Project, “Flashing devices,” 2022. [Online]. Available: https://source.android.com/docs/setup/build/running [Page 81.]

Appendix A

Installing USB bootable kali machine

Previously, the author mostly used the Kali machine on a virtual machine, which is the Oracle VM VirtualBox. According to the official page of Kali Linux, live boot Kali provides direct access to hardware, an unaltered host system and a customized Kali kernel. Compared to limit direct access to hardware and the system requirements, the bootable kali has undeniable advantages over virtual machines. Owning a USB bootable kali also allows the user to bring the USB drive along and does no damage to the host system. Therefore as part of the setup of the experiment environment, a bootable Kali is set up.

A USB drive with a storage system that is larger than the Kali image, Etcher or Rufus burning tool and an official ISO image are needed for this step. First, plug in the USB and verify that the functionality of the device is running normally. From lots of failed experiences, a USB from a reliable vendor is recommended since some of the USB devices could be having unknown problems when burning the image. Secondly, use etcher or rufus to flash the ISO file to the USB. After the flash process is done, the bootable Kali is ready for booting. Last but not least, to be able to save settings and files in the bootable Kali machine, the user needs to select the Live USB Encrypted Persistence option to create persistence in the Kali machine.

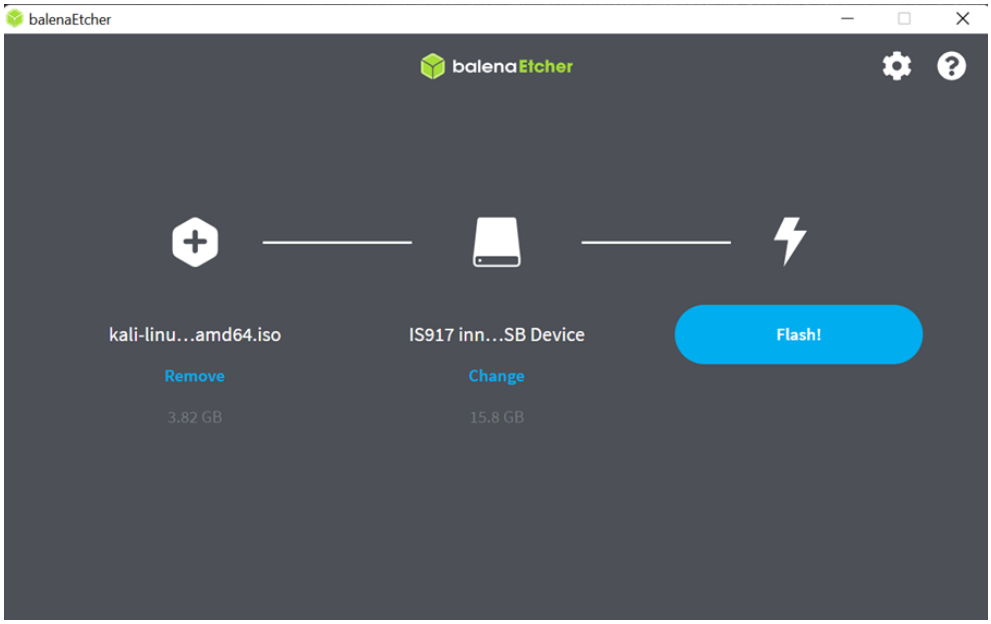

Figure A.1: Flash ISO file into the USB

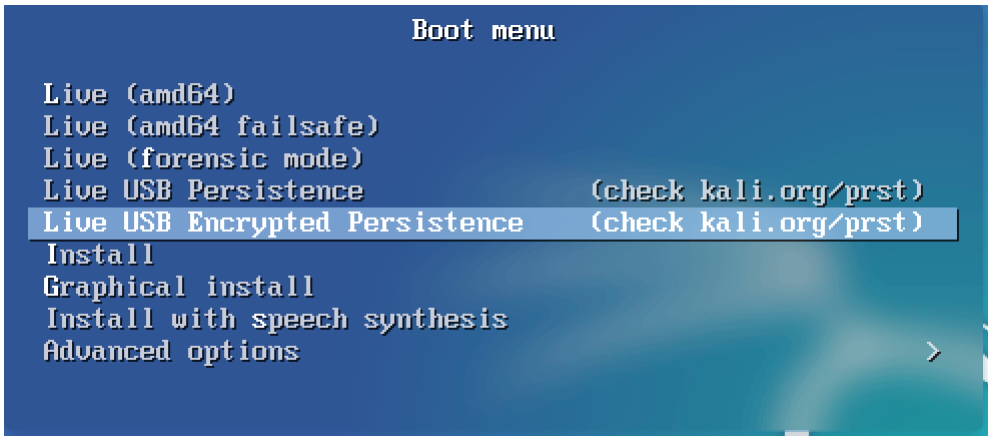

Figure A.2: Use bootable Kali with persistence

Appendix B

Gaining root access on android phone

This is an additional chapter for a prerequisite for the following attacks, which include the reasons, the methods and the results for rooting Redmi Note 11 phone.

Root access is always essential when viewing internal storage and granting access to specific applications and proxies, which is why the author decided to put this step in the report. Not only because the author spends a lot of time and effort on it but also because it is very critical for penetration testing.

The booting process starts with unlocking the phone by enabling the developer options in the system and unlocking it with the official Xiaomi Unlock tool [86]. After unlocking the phone, a tool called Magisk [87] can automatically patch the downloaded boot.img file in the application. Then the fastboot tool is utilized to boot and reboot the system [88]. The cmd power shell action for gaining root access is shown in Figure B.1. This step is basic, however it is fundamental for the upcoming exploitation.

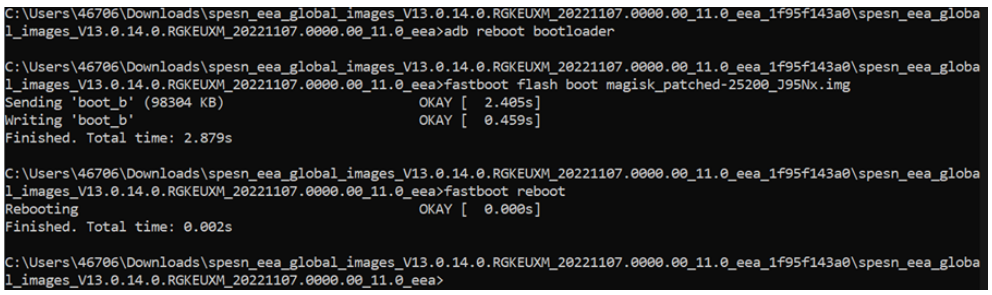

Figure B.1: Gaining root access of the phone

TRITA-EECS-EX-2023:109