KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Computer Science and Engineering

Second cycle, 30 credits

Is your camera spying on you?

A security evaluation of a smart camera

PONTUS KARLBERG

Stockholm, Sweden, 2023

Is your camera spying on you?

A security evaluation of a smart camera

PONTUS KARLBERG

Degree Programme in Computer Science and Engineering
Date: May 19, 2023

Supervisor: Pontus Johnson
Examiner: Roberto Guanciale
School of Electrical Engineering and Computer Science
Swedish title: Spionerar din kamera på dig?
Swedish subtitle: En säkerhetsutvärdering av en smart kamera

Abstract

There are billions of Internet of Things (IoT) devices deployed world wide. Despite this there have been numerous studies showing that these devices very often have major security vulnerabilities. Ensuring the security of IoT devices is important, but especially so for devices that handle sensitive data. The goal of this thesis was to evaluate the security of one such device, a smart camera. It is not possible to ensure the security of every device out there, but analyzing the security of as many devices as possible gives valuable insight into what kinds of vulnerabilities affect different kinds of devices. This can both be used to patch current devices and as a guideline for what parts of the system needs to be secured during development of new devices. The security was evaluated by first building a threat model and then identifying threats against it. Penetration tests were then created to test for the existence of these threats. The thesis found that the evaluated camera has many unnecessary and/or insecure services installed which weakens the security of the device without providing much value. The device was also found to not receive timely updates and the tools installed are old and contain multiple vulnerabilities.

Keywords

IoT, Security, Hacking

Sammanfattning

Det finns flera miljarder Internet of Things (IoT) enheter ute i världen. Trots det så har ett flertal studier visat att dessa enheter väldigt ofta har stora säkerhetsbrister. Det är viktigt att se till att IoT-enheter är säkra, men desto mer när enheten hanterar känslig information. Målet med denna avhandling var att utvärdera säkerheten av en sådan enhet, en smart kamera. Det är inte möjligt att se till att alla enheter är säkra, men genom att analysera säkerheten av så många enheter som möjligt skapas en värdefull insyn i vilka typer av sårbarheter som påverkar olika typer av enheter. Detta kan både användas för att uppdatera befintliga enheter och som en vägledning för vilka delar av systemet som behöver säkras vid tillverkningen av nya enheter. Säkerheten utvärderades genom att först genomföra hotmodellering och sedan identifiera hot mot den. Penetrationstester genomfördes sedan för att undersöka existensen av dessa hot. Avhandlingen fann att den undersökta kameran har många onödiga och/eller osäkrade tjänster installerade som försvagar enhetens säkerhet utan att tillföra betydande värde. Det upptäcktes även att enheten inte får uppdateringar i tid och att installerade verktyg är gamla och har flera sårbarheter.

Nyckelord

IoT, Säkerhet, Dataintrång

Acknowledgments

I want to thank my supervisor Pontus Johnson who has helped guide my work throughout this project. I also want to thank my examiner Roberto Guanciale and opponent Sebastian Williams for their feedback on the report. At times it has been very tough working on this project, so most of all I want to thank my wonderful family who has supported me every step of the way through it. Without them I might not have had the willpower to see it through to the end.

Stockholm, May 2023
Pontus Karlberg

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

DFD	Data Flow Diagram
DoS	Denial of Service
DREAD	Damage potential, Reproducibility, Exploitability, Affected users and Discoverability
GPU	graphics processing unit
HTTPS	Hypertext Transfer Protocol Secure
IoT	Internet of things
LAN	Local Area Network
md	memory display
MQTT	MQ Telemetry Transport
SSID	service set identifier
STRIDE	Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege
STUN	Session Traversal Utilities for NAT
TCP	Transmission Control Protocol
TTL	Transistor–transistor logic
UART	Universal asynchronous receiver-transmitter
USB	Universal Serial Bus
Wi-Fi	Wireless Fidelity

Chapter 1

Introduction

The prevalence of Internet of things (IoT) devices has significantly increased, with approximately 10 billion devices connected worldwide in 2021 [1]. These devices are commonly deployed in households and serve various functions. For instance, some can monitor and regulate house temperature [2], some can automate window blinds [3], and some can detect entries and exits through the front door [4]. Among these devices, smart cameras are a popular type of IoT device used to enhance security in homes and businesses [5]. However, like many IoT devices, smart cameras are prone to vulnerabilities [6–12].

There is a need for further investigation and mitigation of vulnerabilities in smart cameras. This is crucial to safeguard users and gain a better understanding of vulnerabilities within smart cameras and the broader spectrum of IoT devices. In this thesis, the security of the HELO View Indoor Security Camera is analyzed, which has not been previously assessed. The objective is to answer the research question: Is the smart camera HELO View Indoor Security Camera secure against cyber attacks?

1.1 Delimitations

This thesis is conducted as part of a Software-based Master's degree program, and thus modifications to the camera's hardware are not included in this project. Instead, the focus is on software security and only easily accessible hardware components will be examined.

The project adheres to Swedish law, and therefore, no attacks against cloud servers will be considered.

Due to time constraints, not all identified threats can be thoroughly explored. Prioritization of threats will be based on their characteristics.

1.2 Structure of the Thesis

The Background chapter provides an introduction to concepts that are relevant to understanding the project. The Methodology chapter outlines the research methodology employed throughout the project to answer the research question. The Camera Selection chapter explains the process of selecting the camera for examination. The system under consideration chapter provides a comprehensive description of the functionality of the analyzed camera. The Related work chapter presents previous studies and identifies vulnerabilities discovered in related work, serving as inspiration for threat identification. The Threat modeling chapter presents the identified threat vectors, potential threats, and prioritization of threats for further investigation. The Penetration tests chapter examines the selected threats, with each threat having its own specialized Method, Results, and Discussions sections. The Results presented in this chapter serve as the primary outcomes of the project. An additional Results chapter is omitted as it provides minimal added value. Each threat is also discussed within this chapter. The Discussion chapter addresses broader issues concerning the project, while the Conclusions and Future Work chapter presents the final conclusions and discusses future work.

Chapter 2

Background

This chapter provides background information that is useful for understanding this report. It should be noted that while limited information is presented here, additional background information is provided throughout the report when relevant. The chapters on Methodology and Related work could be considered background information but have been separated for a more comprehensive reading experience.

2.1 UART

Universal asynchronous receiver-transmitter (UART) is a serial protocol that facilitates communication between two hardware devices. It involves the transmission and reception of data via two ports, typically labeled TX (transmit) and RX (receive). The TX port of one device is connected to the RX port of the other device. UART is commonly used in embedded devices for debugging purposes, allowing developers to obtain valuable data and insights from the device [13]. It can be configured as a serial console, with the boot loader writing debugging information [14].

2.2 Tuya

Tuya is a company that offers a cloud-based platform for developers of IoT devices. Tuya provides software development kits (SDKs) that enable IoT devices to register with their cloud. This registration allows devices to integrate with Tuya's app, Tuya Smart [15], or with the developer's own app using Tuya's app SDKs. For devices that support multimedia streaming, such as smart cameras, the SDKs handle the streaming between the device and the app, as well as providing a cloud-based storage solution. As of September 30, 2022, the Tuya platform had over 647,000 registered developers serving more than 8,400 customers. They also claim that Tuya-powered smart devices are available in approximately 120,000 stores [14].

Chapter 3

Methodology

3.1 Methodology overview

This chapter introduces the research methodologies used in this thesis. The threat modeling methodology is based on the methods presented in the books *The Security Development Lifecycle* [16] and *Writing Secure Code* [17]. The method consists of three major parts. The first part is gathering as much information about the camera as possible. The second part is to use the information to formalize a threat model. The goal of this part is to create a list of threats that might affect the camera. The third part is to create penetration tests for a subset of the identified tests in order to establish if the camera is vulnerable to them. Any vulnerabilities found or the process to rule out their existence will contribute to answering the research question.

3.2 Information gathering

Information is gathered about the camera in many different ways. The network traffic traveling to and from the camera as well as the mobile app is monitored using Wireshark [18]. The system is also scanned for open ports using the network utility tool Nmap [19]. An SD-Card is inserted in order to figure out what kinds of files the camera writes during normal operations. FCC-documentation is also consulted as well as disassembling the camera in order to find common debug ports. Documentation found to be relevant for the camera is also consulted.

3.3 Threat modeling

The information gathered in the previous part is used to decompose the system into a Data Flow Diagram (DFD).

Previous works are consulted to get a grasp of vulnerabilities that exist in other IoT devices, including other cameras. Potential threat vectors are then identified by taking inspiration from these works as well as brainstorming for new vulnerabilities. STRIDE (acronym for Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege) is used to ensure that relevant threats are found for each part of the DFD.

Since this thesis is limited by time it isn't feasible to explore every potential threat that has been identified. Instead, each threat is rated and ranked using DREAD (acronym for Damage potential, Reproducibility, Exploitability, Affected users and Discoverability). This ranking is then used to prioritize which potential threats to explore further.

3.3.1 DFD

A DFD is a diagram that can be used to model a system. A diagram is built with different kinds of elements. The exact meaning of each element can vary depending on the context behind the creation of the diagram. Table 3.1 defines the elements that are used in this thesis.

3.3.2 STRIDE

STRIDE is a method developed and used by Microsoft to develop secure software [16]. STRIDE can help guide the discovery of possible vulnerabilities in a system. It maps a set of threat categories for each element type in the system DFD. Each pair of a threat category and element in the DFD then represents a relevant attack vector which could have vulnerabilities. A general guideline is therefore to find at least one possible vulnerability for each category-element pair. This way one can be reasonably sure that there aren't any big and obvious areas that hasn't been explored for vulnerabilities.

The threat categories are Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege. A description of each category is presented in Table 3.2. Table 3.3 shows all threat categories that are relevant for each element type. Note the special marking for the Repudiation category of Data stores. Repudiation is relevant for data stores if they contain

Table 3.1: DFD elements

Shape	Type	Description
Circle	Process	A system that can be interacted with and that handles inputs and outputs
Rectangle	External entity	An external system or person that interacts with some part of the system.
Parallel lines	Data store	Persistent storage
Arrowed Line	Data flow	Represents some type of communication between two elements. The arrow indicates the direction and can be double-ended to represent communication in both directions
Dotted line	Trust boundary	A boundary that differentiates areas of trust. An example is the boundary between a Local Area Network (LAN) and the rest of the internet.

some kind of logging data which could be modified by an attacker.

3.3.3 DREAD

DREAD is a method used to evaluate threats [17]. Each threat is evaluated by five criteria. A number between one and ten is assigned to each criteria and an average is calculated to get the final risk score for the threat. The criteria are Damage potential, Reproducibility, Exploitability, Affected users and Discoverability. Table 3.4 contains descriptions of each criteria. DREAD has historically been used at Microsoft [17], but its use was later discouraged due to the use of numbers, which they claim can lead to the analysis being very subjective [16]. The risks calculated in this thesis are however only used as a heuristic to figure out which potential vulnerabilities are the most worthwhile to investigate, since time is a limiting factor. For this task, the ease of applying and of ordering the results when using DREAD outweigh the drawback.

Table 3.2: STRIDE categories

Category	Description
Spoofing	Spoofing threats allow an attacker to pose as someone or something else.
Tampering	Tampering threats involve malicious modification of data, either at rest or in transit.
Repudiation	A repudiation threat is when an attacker can perform an action and then deny that it has happened. Examples are a lack of logging or a threat where log files can be deleted by an attacker
Information disclosure	A threat in which information is exposed to an unauthorized party.
Denial of Service	Denial of Service threats degrades performance or denies access to valid users.
Elevation of Privilege	A threat in which an attacker gains increased privileges than authorized.

Table 3.3: Mapping between STRIDE categories and DFD elements

Element type	S	T	R	I	D	E
Process	X	X	X	X	X	X
External entity	X		X			
Data store		X	+	X	X	
Data flow		X		X	X	

Table 3.4: DREAD criteria

Criteria	**Description**
Damage potential	How great the damage of an attack would be.
Reproducibility	How easy it is to get a potential attack to work.
Exploitability	How much effort and expertise is required from the attacker.
Affected users	How many users would be affected by an attack.
Discoverability	Commonly assumed to be 10 for every threat.

3.4 Penetration testing

The potential threats identified in the previous part now needs to be tested in order to determine if the system is vulnerable to them. This is done by creating penetration tests[20]. Each threat is tested using a method unique to the threat. The method used will be based on previous works from others as well as previous experience from the author. The results of the test are then presented and discussed.

Chapter 4

Camera Selection

Pricerunner [5] was used to get a list of potential Smart Cameras to investigate. The list was sorted by popularity among consumers. Each camera was then compared based on perceived effect of compromise, perceived attack surface, track record of the manufacturer, previous studies and cost. The *HELO View Indoor Security Camera* was chosen since it was the camera with the highest rank that fulfilled all criteria adequately:

- Effect of compromise: It is an indoor camera aimed at households. It was judged that such a camera was likely to be placed in view of something the user wishes to protect, for example an entrance, a pet or even a child. The camera can be rotated remotely, can see in the dark and record and play audio.
- Attack surface: The camera is connected to Wireless Fidelity (Wi-Fi) and has an active connection to the Cloud. It also stores video locally on an SD-Card. It is controlled through an app with which it also has integration with Google Assistant and Alexa.
- Manufacturer: The company selling the camera is called Strong [21] and has a focus on TV peripherals and only has a small collection of cameras and smart plugs. It was judged that this might mean their IoT products were less secure due to the company's unfamiliarity with the domain. No prior security studies were found for any of their products.
- Previous studies: No previous studies were found for the chosen camera. There were also no registered CVEs found for it. This increases the value of this study since no prior public work has been done to find vulnerabilities for this camera.

- Cost: The camera is affordable. At the time of purchase it was 399 SEK.

Chapter 5

The system under consideration

The camera is linked to the app using a QR-code. The data of the QR-code is a JSON string which contains the SSID and Password of the Wi-Fi network the camera should connect to as well as a token valid for 10 minutes used for linking purposes[22]. There is also a button on the camera which puts it back into pairing mode.

TCP port 23 and 6668 are open on the camera. Port 23 hosts a BusyBox telnetd service[23]. Port 6668 is used to enable Tuya devices to communicate with the app directly over the LAN[24][25]. We will call this the Tuya Service. The camera periodically sends broadcasts towards port 6667 which contain additional information about the service. In order to communicate with the Tuya service a Local Key is needed, which is an AES key generated during the linking process[26].

During boot the camera establishes a persistent TLS connection to port 8883 of the domain m2.tuyaeu.com. This is an encrypted connection of MQ Telemetry Transport (MQTT)[27]. The camera also occasionally makes TLS connections to port 443 of the domain a2.tuyaeu.com. These are Hypertext Transfer Protocol Secure (HTTPS) connections[28]. The servers running at subdomains of tuyaeu.com are part of what we will call the Tuya Cloud, or the Cloud. The connections to the Tuya Cloud are used to synchronize the state of the camera with the Cloud and the app. The app also uses similar connections to the Cloud. Almost all interactions with the camera by the app are facilitated through the Cloud. A notable exception is the video stream which is sent using UDP directly to the app. The direct connection is established through the Session Traversal Utilities for NAT (STUN)[29] protocol.

There is a slot for an SD-Card in the camera. If an SD-Card is inserted video will be locally saved to it. The saved video is then accessible through the app. The camera also saves log files to the SD-Card.

Pins labeled GND, TX and RX can be found inside the camera. These provide a serial console through UART. The console provides a detailed boot log for the camera. After the camera has booted it also prompts for user credentials. The boot process can also be interrupted to enter U-Boot[30].

Chapter 6

Related work

The selected camera has not been previously studied, as per the criteria for its selection. However, there is ample existing research on the security of other cameras and IoT devices in general. OWASP provides a taxonomy of the 10 most common types of attacks on IoT devices, which serves as a comprehensive reference for potential vulnerabilities to consider [31].

Multiple studies have identified common vulnerabilities in IoT devices, including cameras. Weak standard credentials have been found to be a recurring problem [6, 7, 9–11]. Encryption issues in network requests have also been highlighted as a common vulnerability [6, 8–12]. Some studies have revealed instances where unencrypted recordings and log files are saved on removable media [6, 12]. The use of outdated software versions that can be exploited has been documented as well [9, 32]. Additionally, several studies have identified the exposure of the UART interface, which can be leveraged by adversaries [6, 32–35].

Apart from security studies, there is a significant interest among consumers in controlling their IoT devices locally without relying on the cloud. An example is the TuyaAPI tool, available on GitHub, which allows for local control of Tuya devices on the LAN by reverse-engineering the Tuya service [26]. This makes the Tuya service an interesting attack surface that can be further explored.

The findings mentioned here, along with less common vulnerabilities, are utilized when identifying threats in Section 7.2.

Chapter 7

Threat modeling

7.1 DFD

The information gathered through network monitoring was used to create the DFD presented in Figure 7.1. Table 7.1 maps the threat types to each element in the DFD. Processes, External entities and the Data Store have been given a name and an ID. Data Flows between these elements will be denoted by connecting their IDs with -> or <->. If there are more than one Data Flow between two elements the protocol used can also be used to differentiate them. For example: the MQTT connection between the camera and the Cloud is denoted by MQTT 1<->5.

Note that Data Flow 1<->4 represents an internal flow within the camera which can't be accessed. Also note that while Tuya Cloud is modeled as a Process, there will not be any attacks considered against it. Tuya Cloud is an external system we do not have permission to run penetration tests against, so no attacks will be targeted against it.

7.2 Identified threats

This section describes all identified threats that may be relevant to the system. They were selected according to the process described in Section 3.3.

7.2.1 Insecure or missing encryption of network requests

Network communication is inherently insecure. There are many potential threat actors between any two parties when communicating over the internet.

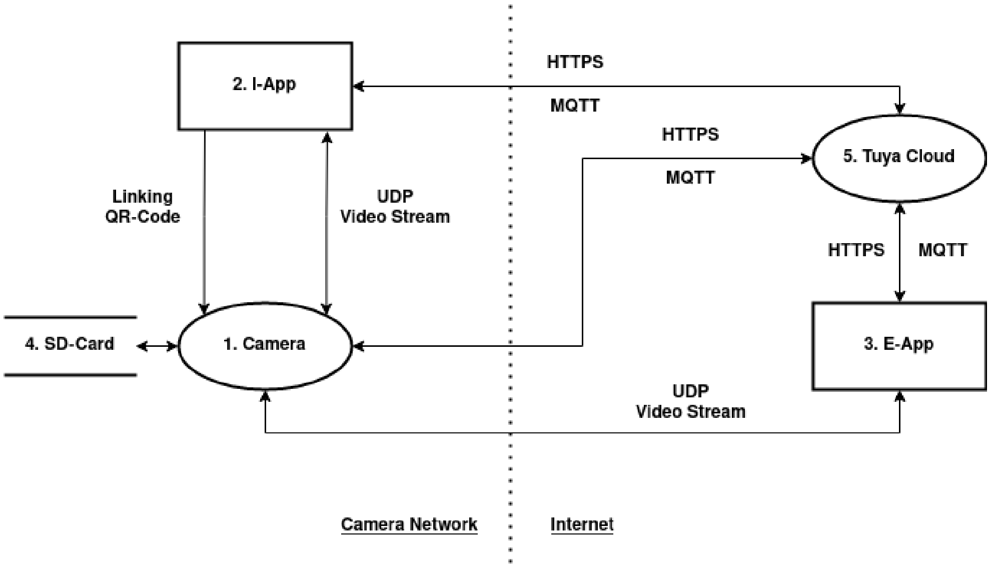

Figure 7.1: DFD of camera

Table 7.1: Table of relevant STRIDE threat types per DFD element

Element type	Threat types	Element
Process	STRIDE	1. Camera
External entity	SR	2. Internal App (I-App) 3. External App (E-App)
Data store	TRID	4. SD-Card
Data flow	TID	HTTPS 1<->5, 2<->5, 3<->5 MQTT 1<->5, 2<->5, 3<->5 Video Stream 1<->2, 1<->3 QR-Link 2->1

Encryption can be used to prevent unauthorized parties from eavesdropping or modifying information in transit. Despite this there have been many cases where other cameras have used insecure or missing encryption for network requests[6, 8–12]. This threat is also the seventh threat on *OWASP IoT Top 10*: "Insecure Data Transfer and Storage" [31].

The lack of, or usage of insecure, encryption is a considerable threat. That said, in the case of the camera under consideration all connections to the Cloud are encrypted using TLS1.2. It also doesn't accept self-signed certificates and offers secure suite of ciphers. The protocol used for the video stream is not known, though the communication is most likely facilitated through Tuya's SDKs[36–40] and seems to use encryption[41].

This threat is related to two STRIDE categories:

- Tampering: Unencrypted network requests could be modified by any party that is located between the sender and receiver.
- Information Disclosure: Any party between the sender and receiver can read the contents of an unencrypted network request.

This threat corresponds to Threat ID 1.1.1 through 1.1.8.

7.2.2 Behaviour derivation from encrypted traffic

While encryption of network requests hides the content being transmitted, it doesn't hide to whom, how much or how often it is. There have been previous studies were this kind of data is used to infer what kinds of and when different kinds of activities are performed by a user or system [42–45]. This kind of analysis could possibly also be performed on the camera under consideration during this project. Every data flow which transmits network data are relevant. This threat is related to one STRIDE category:

- Information Disclosure: Information about activities that are performed could be leaked to a passive network observer.

This threat corresponds to Threat ID 1.2.1 through 1.2.8

7.2.3 Disturbing network traffic

Devices that connect to servers over the internet are reliant on a functioning network connection. It might be possible for an attacker to disrupt the traffic flowing between the camera and the server or phone app to disrupt the usage of the normal functions of the camera. In study done by Loi *et al.* [11] they examined 20 different IoT devices and found, among many other things, that most of them were susceptible to some kind of Denial of Service (DoS) attack. This threat is related to one STRIDE category:

- Denial of Service: Disrupting the traffic flow between the camera and the server or phone app would lead to disruptions in the service for legitimate users.

This threat corresponds to Threat ID 1.3.1 through 1.3.8

7.2.4 Linking devices through QR-Code might be insecure

A QR-code is a standard way to encode information as an image. They can contain arbitrary data. The QR-Code used for linking is generated by the app and is then scanned by the camera. The camera uses the information encoded in the QR-code to complete the linking. The data encoded in the QR-Code could potentially be altered or recorded, leading to some kind of attack on the camera.

This threat was uncovered through brainstorming. The QR-code used for linking the camera only contains limited information, so the threat doesn't seem very likely.

This threat is related to two STRIDE categories:

- Tampering: If the QR-Code contains sensitive configuration information it could be vulnerable to tampering where an adversary is able to control some part of the configuration.
- Information Disclosure: The QR-Code could contain sensitive information that is useful to an adversary.

This threat has ID 1.4.1

7.2.5 Insecure or missing encryption of stored recordings on SD-Card

The camera saves recordings to the connected SD-Card. There are previous cases where these haven't been encrypted[6][12]. An attacker with physical access to the camera could then copy all recordings. They could also delete all or parts of the recordings, for example to hide their own presence. They could also potentially change the recordings to show something different to what the camera has recorded.

The recordings found for the examined camera do seem to be encrypted. This assessment is based on file analysis with Binwalk[46] as well as the fact that media storage of Tuya devices can be handled by their SDK[36, 47, 48].

This threat is related to four STRIDE categories:

- Tampering: Unencrypted recordings could be modified by an adversary
- Repudiation: An adversary could alter the unencrypted recordings in such a way where any action of theirs that has been recorded could be removed.

- Information Disclosure: An adversary could look at the recordings
- Denial of Service: An adversary could delete the recordings which would hinder legitimate users from utilizing the playback feature of the camera.

This threat has ID 2.1.1

7.2.6 Encrypted recordings on SD-Card can be tampered with

The recordings on the SD-Card might be encrypted. It might still be possible to tamper with the data. For example, an attacker might be able to delete or corrupt all or some of the recordings. This would be a Denial of Service attack. This threat was uncovered through brainstorming. This threat is related to three STRIDE categories:

- Tampering: Encrypted recordings could potentially be modified by an adversary. They could also be deleted.
- Repudiation: An adversary could potentially alter the encrypted recordings in such a way where any action of theirs that has been recorded could be removed. They could also be deleted.
- Denial of Service: An adversary could delete the recordings which would hinder legitimate users from utilizing the playback feature of the camera.

This threat has ID 2.1.2

7.2.7 Sensitive log files are stored on SD-Card

The camera might store other files on the SD-Card than the recordings. These files might contain sensitive information that an attacker could use to gather additional information in order to compromise the camera. In a study performed by Alharbi and Aspinall [12] the authors found several cases where smart cameras used removable storage to insecurely store IDs, passwords, serial numbers and cryptographic keys. Any log file written to the SD-Card should therefore be examined for sensitive information.

In the case for this camera we are also interested in finding information that could aid with the penetration tests for the other threats, which makes this an interesting threat to explore.

This threat is related to one STRIDE categories:

- Information Disclosure: An adversary could read these sensitive logs.

This threat has ID 2.1.3

7.2.8 Get access to user credentials and connect with app

It might be possible for an adversary to obtain credentials for the app and use these to connect as a legitimate user. There are many different possibilities for how this could be possible. There have been studies done on other cameras where the app stored the user credentials unencrypted in the app's local storage[8, 10]. This would require an adversary to have read access to these restricted directories, which most likely requires a rooted device. Other possibilities involve finding the credentials through missing or insecure encryption of network connections or through log files available to an adversary, both of which are possible vulnerabilities already identified.

Finding these credentials are dependent on the findings of Section 7.2.1, Section 7.2.7 and Section 7.2.12. Finding the credentials inside the local storage of the app is also not very significant since an adversary already needs to very high privileges on the device where the app is installed to be able to access them.

This threat is related to one STRIDE category:

- Spoofing: An adversary with these credentials can connect to the app and camera like a legitimate user and perform all normal user actions.

This threat has ID 3.1.1

7.2.9 Telnet service is accessible through weak standard credentials

The most common vulnerability according to *OWASP IoT Top 10* is the usage of "Weak, Guessable, or Hardcoded Passwords" [31]. There are many cases where this had been the case for smart cameras as well [6, 7, 9–11]. The Telnet service running on the camera might have this vulnerability. This would let an attacker gain code execution on the camera and potentially perform a full system takeover. This threat is related to six STRIDE categories:

- Spoofing: An adversary could gain access to the weak credentials and pretend to be a legitimate user by connecting to the Telnet service.
- Tampering: Access to the Telnet service could let an adversary modify important files on the camera.
- Repudiation: An adversary with access to the Telnet service could delete log files and other proof of their activities from the camera.
- Information Disclosure: An adversary with access to the Telnet service could read sensitive files from the camera.
- Denial of Service: An adversary with access to the Telnet service could delete files and stop services to break the camera.
- Elevation of Privilege: An adversary getting access to the Telnet service would gain additional privileges than before they had access. They could potentially gain full system privileges of the camera.

This threat has ID 4.1.1

7.2.10 Local control of camera through Tuya service

The service running on Transmission Control Protocol (TCP) port 6668 is common on Tuya devices and is used to get data from and control their devices locally[24]. Communication with the service requires access to an AES key, commonly called the local key, created when a device is linked with a Tuya compatible app. Owners of other Tuya devices have investigated how to get the AES key for their own devices. Some approaches require creating a Tuya development account and linking the camera directly to that. Another approach utilises an old version of Tuya's app Smart Life which stores an unencrypted copy of the key in app storage[26].

With the key it should be possible to communicate with the camera to read and configure settings. It might also be possible to gain further access depending on what functionality is implemented.

The investigated camera has the Tuya service activated. It would be interesting to acquire the local key and to access the Tuya service.

This threat is related to six STRIDE categories:

- Spoofing: An adversary using the Tuya service would be pretending to be a legitimate user or a tool that is utilizing the functionalities of the service.

- Tampering: An adversary could potentially use the Tuya service to modify settings of the camera.
- Repudiation: An adversary could potentially disable the camera before performing an action recorded by it.
- Information Disclosure: An adversary could extract sensitive information using the Tuya service.
- Denial of Service: An adversary could potentially disable the camera using the Tuya service.
- Elevation of Privilege: An adversary gaining access to the Tuya service would be receiving privileges they did not previously possess.

This threat has ID 4.1.2

7.2.11 Old and vulnerable packages are used

The fifth most common vulnerability according to *OWASP IoT Top 10* is ”Use of Insecure or Outdated Components” [31]. The services running on the camera or the software it uses might be old and outdated and contain known vulnerabilities. An attacker might be able to exploit one of these to gain access to the camera. In the study performed by Tekeoglu and Tosun [9] the authors identified that BusyBox was used and utilized the official documentation to further exploit the system. Patrick Wardle & Colby Moore [32] identified an old version of OpenSSL[49] and were able to perform a client-side heartbleed[50] attack to gain the private keys and certificates used by their camera.

In the case of this study, by identifying the version of packages that are exposed to the network it might be possible to gain further access through known vulnerabilities. This threat is related to six STRIDE categories:

- Spoofing: Insecure services could let an adversary gain access as someone they are not.
- Tampering: Insecure services might let an adversary modify configuration data.
- Repudiation: Insecure services might let an adversary remove log files or recordings.

- Information Disclosure: Insecure services might leak information to an adversary.
- Denial of Service: Insecure services might let an adversary delete important files or disable services in order to break the camera.
- Elevation of Privilege: Insecure services might let an adversary gain code execution on the camera.

The threat has ID 4.1.3

7.2.12 Sensitive data is logged on the UART interface

The UART interface can log information during boot. Borg and Francke [34] used information printed by UART to figure out the system architecture and where the firmware was installed. The UART interface might also print other valuable information, for example IDs, passwords, serial numbers or cryptographic keys.

In the case for this camera we are also interested in finding information that could aid with the penetration tests for the other threats, which makes this an interesting threat to explore.

This threat is related to one STRIDE categories:

- Information Disclosure: An adversary could read these sensitive logs.

The threat has ID 4.1.4

7.2.13 Gain access to the firmware through the UART interface

The UART interface has been shown to sometimes allow an adversary to interrupt the boot sequence and enter the bootloader[32, 33]. From the bootloader it is sometimes possible to read data from the device. The bootloader might let an adversary alter boot parameters or read memory from the device. If an adversary is able to read memory from the device they could download the firmware installed [35]. After extracting the firmware it might also be possible to modify and re-flash it.

The camera has an enabled UART interface and pressing the letter Q within 3 seconds of boot interrupts the boot sequence and presents a shell in the bootloader U-boot[30]. Exploring the bootloader in order to extract the firmware seems like an interesting threat.

This threat is related to six STRIDE categories:

- Spoofing: Credentials might be found in the firmware letting an adversary authenticate as a valid user.
- Tampering: An adversary might be able to flash a modified version of the firmware
- Repudiation: An adversary might be able to flash a modified version of the firmware such that the camera doesn't recognize the actions performed by the adversary.
- Information Disclosure: Gaining access to the installed firmware would leak sensitive information.
- Denial of Service: An adversary might be able to flash a modified version of the firmware which breaks the camera.
- Elevation of Privilege: An adversary might be able to flash a modified version of the firmware giving full access to the camera, or Credentials might be found in the firmware letting an adversary gain new privileges, or bugs in the firmware can be found which lets an adversary gain increased privileges.

The threat has ID 4.1.5

7.2.14 Gain code execution through the UART interface

The UART interface sometimes gives direct access to a root shell[6]. Otherwise access to the bootloader might let an adversary modify how the system boots in order to gain code execution. One common method is to modify the bootargs in the bootloader in order to start a shell upon boot[32, 33]. The boot log that is displayed might also contain credentials or other information that would let an adversary gain access through other services or bugs.

In the case of this study, after the boot log has been displayed the camera displays a login prompt on the UART interface. The boot log contains much interesting information, but of note for this threat is that there are debug lines which indicates that the camera is checking for an update file on the SD-card. This might be interesting to explore, especially if we gain access to the firmware.

This threat is related to six STRIDE categories:

- Spoofing: If an adversary gains code execution on the camera they are able to pretend to be the current user.
- Tampering: An adversary that can run code on the camera may modify important files.
- Repudiation: An adversary that can run code on the camera may remove log or video files from the camera.
- Information Disclosure: An adversary that can run code on the camera might learn more information about installed and running applications.
- Denial of Service: An adversary that can run code on the camera may break important functionality.
- Elevation of Privilege: An adversary that has gained code execution on the camera will have gotten many new privileges privileges.

The threat has ID 4.1.6

7.3 Risk analysis

Each threat was evaluated using DREAD. The Threat Traceability Matrix presented in Figure 7.2 presents the rating for each threat. Threats with high ratings were given priority when picking which penetration tests to perform. Note that the ratings are specifically based on the system under consideration, not the general case. All information about the system that has been gathered affects the rating. For example, since the network analysis strongly suggests that the connections are properly encrypted, the R and E rating for those potential threats are lower than one might otherwise expect. The value of finding information which might help locate further vulnerabilities is also valued as part of this study, so the D ratings for information disclosure threats are higher.

Threat ID	Threat	STRIDE	D	R	E	A	D	Rating
1	**Data flow**	**TID**						
1.1	Insecure or missing encryption	TID						
1.1.1	(2<->5, I-App and Cloud) MQTT:	TID	7	3	6	8	10	6.8
1.1.2	(2<->5, I-App and Cloud) HTTPS	TID	7	3	6	8	10	6.8
1.1.3	(3<->5, E-App and Cloud) MQTT	TID	7	3	6	8	10	6.8
1.1.4	(3<->5, E-App and Cloud) HTTPS	TID	7	3	6	8	10	6.8
1.1.5	(1<->5, Camera and Cloud) MQTT	TID	7	3	6	8	10	6.8
1.1.6	(1<->5, Camera and Cloud) HTTPS	TID	7	3	6	8	10	6.8
1.1.7	(1<->2, Camera and I-App) Video Stream	TID	7	3	4	2	10	5.2
1.1.8	(1<->2, Camera and E-App) Video Stream	TID	7	3	6	8	10	6.8
1.2	Behavior derivation from encrypted traffic	I						
1.2.1	(2<->5, I-App and Cloud) MQTT:	I	4	2	3	4	10	4.6
1.2.2	(2<->5, I-App and Cloud) HTTPS	I	4	2	3	4	10	4.6
1.2.3	(3<->5, E-App and Cloud) MQTT	I	4	2	3	4	10	4.6
1.2.4	(3<->5, E-App and Cloud) HTTPS	I	4	2	3	4	10	4.6
1.2.5	(1<->5, Camera and Cloud) MQTT	I	4	2	3	4	10	4.6
1.2.6	(1<->5, Camera and Cloud) HTTPS	I	4	2	3	4	10	4.6
1.2.7	(1<->2, Camera and I-App) Video Stream	I	4	2	1	2	10	3.8
1.2.8	(1<->2, Camera and E-App) Video Stream	I	4	2	3	4	10	4.6
1.3	Disturbing network traffic	D						
1.3.1	(2<->5, I-App and Cloud) MQTT:	D	2	3	4	8	10	5.4
1.3.2	(2<->5, I-App and Cloud) HTTPS	D	2	3	4	8	10	5.4
1.3.3	(3<->5, E-App and Cloud) MQTT	D	2	3	4	8	10	5.4
1.3.4	(3<->5, E-App and Cloud) HTTPS	D	2	3	4	8	10	5.4
1.3.5	(1<->5, Camera and Cloud) MQTT	D	2	3	4	8	10	5.4
1.3.6	(1<->5, Camera and Cloud) HTTPS	D	2	3	4	8	10	5.4
1.3.7	(1<->2, Camera and I-App) Video Stream	D	2	3	2	2	10	3.8
1.3.8	(1<->2, Camera and E-App) Video Stream	D	2	3	4	8	10	5.4
1.4	Insecure linking of devices through QR-Code	TI						
1.4.1	(2->1) Linking devices through QR-Code might be insecure	TI	8	3	2	2	10	5
2	**Data Store**	**T(R)ID**						
2.1	(4. SD-Card):	TRID						
2.1.1	Insecure or missing encryption of stored recordings	TRID	8	5	4	2	10	5.8
2.1.2	Encrypted recordings can be tampered with	TRD	4	5	3	2	10	4.8
2.1.3	Sensitive log files are stored	I	8	8	4	5	10	7
3	**External entity**	**SR**						
3.1	(2.3. Apps)	S						
3.1.1	Get access to user credentials and connect with app	S	8	5	2	2	10	5.4
4	**Process**	**STRIDE**						
4.1	(1. Camera)	STRIDE						
4.1.1	Telnet service is accessible through weak standard credentials	STRIDE	10	6	4	8	10	7.6
4.1.2	Local control of camera through Tuya service	STRIDE	9	6	4	6	10	7
4.1.3	Old and vulnerable packages are used	STRIDE	10	8	4	6	10	7.6
4.1.4	Sensitive data is logged on the UART interface	I	8	9	4	4	10	7
4.1.5	Gain access to the firmware through the UART interface	STRIDE	8	8	3	8	10	7.4
4.1.6	Gain code execution through the UART interface	STRIDE	10	8	3	4	10	7

Figure 7.2: Threat Traceability Matrix

Chapter 8

Penetration tests

In this chapter, the penetration tests are performed on the selected threats from the Threat Modeling. The DREAD ratings for each threat presented in Figure 7.2 was used to decide which threats to investigate with penetration tests, as described in Section 7.3.

8.1 Sensitive log files are stored on SD-Card

8.1.1 Threat ID

2.1.3

8.1.2 Method

An SD-Card is first inserted into the camera. The camera is then powered on, reset and paired using the app. Different features of the camera are also used. The camera is then powered off and the SD-Card is moved into a computer and searched for files written by the camera.

8.1.3 Results

The camera writes files to the SD-Card. Except for the media files written to the camera, there is also a file named baselog0 in the folder IPC_DEV_LOGS. This is a log file which is written to during boot and while the camera is running. It contains many device identifiers, version numbers of tools, SDKs among other interesting data. The service set identifier (SSID) and Wi-Fi of the configured access point is also saved, even when an already paired camera is booted. There are also some kind of credentials: DevAccount: admin, DevPassword: admin.

8.1.4 Discussion

The information present in the log file is a great first source of information. It can be used to get an initial idea about the architecture of the camera and the installed tools. The main value is information gathering about the camera, it doesn't contain much information that by itself can be used to compromise the camera. Access to sensitive data like the SSID and password of the Wi-Fi network are of limited value since an adversary needs physical access to the camera to access the SD-Card, but most Wi-Fi access points have this information printed on the device, which would also be accessible with physical access.

8.2 Sensitive data is logged on the UART interface

8.2.1 Threat ID

4.1.4

8.2.2 Method

The voltage of the UART interface is determined using a multimeter. The GND, TX and RX connectors are then connected to a computer using a Universal Serial Bus (USB) to Transistor–transistor logic (TTL) converter with the appropriate voltage. Communication with the UART port is then handled by the tool Minicom[51].

Once connected to the UART interface, the boot log is recorded. While recording the boot log, different operations are performed. For example resetting the device, linking it and using various functions through the app. After this the recorded log is examined and sensitive data is attempted to be identified.

8.2.3 Result

The boot log appears to contain all info present in the log file previously found on the SD-Card. In addition to this is also even more verbose information.

Of special note are multiple encryption keys, including the local key used by the Tuya service, details about the HTTPS requests and MQTT traffic. The MQTT traffic includes username and password used to connect to the MQTT server in the Cloud. There also seemed to be a check for an update file on the SD-Card. There is also information regarding the memory layout of the kernel image.

8.2.4 Discussion

The information provided on the UART interface is even more useful than the log file on the SD-Card since it contains the same content but also even more verbose logging. On the other hand it is more difficult to access the UART interface since the camera needs to be disassembled and specialized hardware is needed for the connection. Regardless of this, the logging data is very useful when gathering information with the intention of finding other vulnerabilities.

8.3 Gain access to the firmware through the UART interface

8.3.1 Threat ID

4.1.5

8.3.2 Method

When connected to the UART interface, pressing the letter Q within 3 seconds of booting will interrupt the booting sequence and display the bootloader U-Boot with various options related to booting the device. The tools made available in the bootloader will be explored by consulting built-in help messages as well as online documentation about U-Boot. Useful information from the boot log or other helpful resources might also be used. The goal is to download the firmware installed on the camera.

8.3.3 Results

One of the options accessible from U-Boot[30] is memory display (md). This function can take a memory address and a number and will then print out that many blocks of 4 bytes in hexadecimal. As previously found in the penetration test in Section 8.2, the boot log shows information about the memory layout of the kernel image. It is located at memory address bc050000 and is 2280412 bytes long. 2280412 bytes is 570103 memory blocks and thus running the command md bc050000 570103 prints the entire firmware image on the serial console. A script was also written to convert the ASCII output into a correct binary format. The tool Binwalk[46] was then used to extract the filesystem of the image. This succeeded and access to the firmware installed on the camera was acquired.

8.3.4 Discussion

Having access to the firmware is very useful. It allows an adversary to analyze the filesystem for installed tools, configuration files, hard-coded credentials and custom scripts and binaries. These things can then be used to gain further access into the system. It might also be possible to modify the firmware and flash it back to the camera to introduce a persistent backdoor if no other points of entries are found.

8.4 Gain code execution through the UART interface

8.4.1 Threat ID

4.1.6

8.4.2 Method

It has previously been shown to be possible to gain shell access by modifying the boot_args in the bootloader interface and continuing to boot[32, 33]. If that doesn't work it might be possible to analyze the installed firmware obtained in the penetration test in Section 8.3 for vulnerabilities. Of note for possible vulnerabilities is the check for an update file on the SD-Card which was located in the penetration test in Section 8.2.

8.4.3 Results

There was an attempt to gain a root shell on the device by modifying the boot_args environment variable and adding "init=/bin/sh" to the end of it which has previously been showed to work on other devices[32, 33]. Upon continuing the boot process a simple root shell was acquired. The normal boot process won't run in this state, so the camera itself does not start and no other file systems gets mounted. The only mounted file filesystem uses SquashFS[52] which is read only. This prevented the modification of the root password which works on some systems[33].

Instead the firmware was analyzed. A boot script was located which starts telnet and two binaries, daemon and aoni_ipc. They were inspected with the strings command[53]. Strings shown in the log message about the update file was of primary interest. The daemon binary was found to contain many strings related to /mnt/sd_card/, including the path /mnt/sd_card/update/fireware/update.sh identified previously. Ghidra was then used to decompile the binary in order to determine the operations performed on this path. Searching for the string within the decompiled source uncovered the function showed in Figure 8.1. This means that upon boot the camera will check for the existence of the file /mnt/sd_card/update/fireware/update.sh and then give it permission 777, meaning anyone can read, write and execute the file, and then execute it. A shell script which opens the port 9090 with a shell attached using netcat[54] was placed on the SD-Card at the corresponding path. After the camera booted the port also opened and upon connecting and running the id command, the shell returned that uid=0(root) gid=0(root) groups=0(root). We have now gained root access on the camera by placing a script on the SD-Card at location "update/fireware/update.sh". Note the misspelling of firmware in the path of the update file.

```
undefined4 sd_burn_updatefile(void)

{
  int iVar1;

  iVar1 = is_file_exist("/mnt/sd_card/update/fireware/update.sh");
  if (iVar1 == 0) {
    system("chmod 777 /mnt/sd_card/update/fireware/ -R");
    func_0x0040bcc0("NOW READY TO UPDATE FORM SD CARD");
    system("/mnt/sd_card/update/fireware/update.sh");
  }
  else {
    func_0x0040bcc0("NO SD CARD UPDATE  FILE ... GOGOGOGOGOGGOGOGOGGOG");
  }
  return 0;
}
```

Figure 8.1: Decompiled code for when the update.sh file is executed

8.4.4 Discussion

Gaining root access on the camera is very valuable. An adversary can modify the system to do almost anything. Credentials could be stolen and modified, the video stream could be intercepted and replaced, the camera could be added to a botnet etc. Having root access also allows for further reconnaissance of the system. Running processes and open network ports can be examined for example, which can't be done with only the firmware.

The value of gaining a root shell by using the UART interface is of limited value in normal situations, since it requires extensive access to the camera. Gaining code execution as root by inserting a malicious SD-Card is of considerably more value since it can be done quickly and discreetly while fully compromising the camera.

8.5 Telnet service is accessible through weak standard credentials

8.5.1 Threat ID

4.1.1

8.5.2 Method

We have access to the firmware running on the camera through the penetration test in Section 8.3. This means we can read /etc/passwd and /etc/shadow which contain configuration details about the users on the camera and password hashes for their accounts. Since Telnet uses the same credentials as the users on the machine, cracking these hashes should let us access Telnet with the corresponding user. In order to find the passwords, dictionary attacks were performed on the hashes using John the Ripper[55].

8.5.3 Results

The /etc/passwd and /etc/shadow files revealed that the root user is the only user with a password. This means that the Telnet credentials has to be that of the root user. The wordlist[56] used by the Mirai botnet was first used with no matches. After this the popular wordlist rockyou.txt[57] was used. Since this didn't work wordlists from Probable-Wordlists[58] and Weakpass[59] was used for about a week without results.

8.5.4 Discussion

The password wasn't found in any of the examined wordlists. There are various methods that could be tested in order to find the password. The bruteforce attempts could be performed on a more powerful machine with a dedicated graphics processing unit (GPU) than the laptop used during this study. There are also many different rule-based techniques where the passwords in the wordlists are transformed according to different rules in order to find variations of the provided passwords. Doing an exhaustive bruteforce search might be possible using more powerful hardware and more time.

It is not known if the password is unique per device or shared between every camera. The password is saved on a write-only file system and can only be changed when flashing the device. If the password is shared between all cameras and the password is found it would have a huge impact on the security of the camera. It would then be possible gain root access to the camera through the network.

8.6 Local control of camera through Tuya service

8.6.1 Threat ID

4.1.2

8.6.2 Method

The Local key is found in the penetration test in section 8.2. TuyaAPI[26] is used to connect to the Tuya service. The library is used in order to attempt to fetch state information about the camera as well as to send commands to it over the local network.

8.6.3 Results

The Local key is able to decrypt and encrypt the messages sent to and received from the Tuya service. Despite this, they don't seem to have any effect. The service doesn't return any state information and none of the updates are reflected in the camera or the app. Whenever a message is sent to the Service the serial console on the UART interface prints out the error messages "can't find device" and "__smt_frm_cmd_prep error:-1".

8.6.4 Discussion

It seems that while the Tuya service is enabled, it hasn't been correctly implemented. The binary that hosts the service could be reverse engineered to figure out if there are any exploits possible when interacting with the broken service, or to figure out how to correctly communicate with it. If the service is broken, it should probably be disabled since it doesn't provide any functionality and the regular broadcasts which are part of it unnecessarily reveals some information about the device.

8.7 Old and vulnerable packages are used

8.7.1 Threat ID

4.1.3

8.7.2 Method

Since we have both the firmware and code execution on the camera, we are able to identify exactly which tools are installed. Of extra interest are tools that are used remotely during normal operations, since exploits in those tools might facilitate code execution remotely. The versions of these tools are found by analyzing the firmware and inspecting the filesystem on the camera. Vulnerabilities for each tool is then identified by consulting known CVEs.

8.7.3 Results

The main tools that were identified to interact with the network are telnet, udhcpc and wpa_supplicant. Most of the tools used by the camera come from Busybox[23]. udhcpc and telnet are both implemented as Busybox applets. Busybox is version 1.27.2. There are many CVEs registered for different versions of Busybox[60]. There does not seem to be any CVEs registered for the Busybox applet telnetd running version 1.27.2. For the Busybox applet udhcpc there seems to be one CVE that affects version 1.27.2, CVE-2018-20679 [61]. This CVE leads to an Out-of-bounds Read triggered by a crafted DHCP message.

There also seems to be CVEs for other Busybox applets running version 1.27.2, but these applets doesn't seem to be used by the camera.

Wpa_supplicant version v2.7-devel_rtw_r26589.20180301 is used to establish the Wi-Fi connection. This version seems to be affected by at least 12 CVEs[62] with ratings ranging from 3.3 to 6.8.

8.7.4 Discussion

The camera is using software that is several years old there are thus many known vulnerabilities that affects it. There could be even more software used by the camera that can be exploited using known vulnerabilities in other tools or libraries. The manufacturer should update the camera with modern versions of all tools and libraries installed.

Chapter 9

Discussion

9.1 Reliability and Validity

The goal of the research methodology used in this thesis was to identify important threats, prioritize which threats are the most worthwhile to explore and then to explore those threats thoroughly.

Time is always a limiting factor, it is not feasible to expect that every possible threat is identified. By identifying threats that has already been identified in the works of others you ensure that prominent threats are discovered. By combining this with the STRIDE framework we can ensure that threats related to each part of the system is identified. This encourages the discovery of vulnerability types that might not have been considered. Augmenting the discovery process with brainstorming also allows for the discovery of new types of threats. This methodology ensures that relevant threats are identified for each part of the system.

Time is also a limiting factor when the identified threats are to be explored. This necessitates prioritizing which threats are the most valuable to explore. By using DREAD each threat can be evaluated based on its characteristics. This evaluation is inherently subjective, especially when considering threats that might or might not actually affect the system. To combat that uncertainty the system and its documentation was explored extensively before the threats were rated. This uncovered details about the system that affected the relevancy of the identified threats. All identified network connections for example were thoroughly examined and the conclusion that they were encrypted and that the encryption seemed to be secure was drawn. This isn't to say that the communications actually are securely encrypted, but it affects the relevancy of committing time to explore that possibility. This also means that the ratings for a threat can vary depending on the explored system, which is a desirable trait since each system is unique and threats relevant in one system might not be relevant in another.

The selected threats were then examined using penetration testing. Each threat has some kind of desirable goal, and the tests were constructed with the aim of reaching that goal. The methods for the tests were constructed with inspiration from how similar goals were achieved in the works of other authors and with tools designed for those tasks. This ensures that the methods used are appropriate for the task. Time is still a limiting factor and some threats could be explored during a very long time. For example when brute-forcing a password. There are countless different methods to try if a password isn't found, but at some point you can't afford to explore a threat any more. It might be necessary to abandon a threat without finding an exploit. It means that a conclusion about the existence of the vulnerability can't be drawn, but the existence of the test that has been performed gives credence to the security in regards to that threat.

9.2 Sustainability

The connection between the work in this thesis and sustainability is limited. You could argue that this work is related to the twelfth sustainability goal as recognized by the UN, "Ensure sustainable consumption and production patterns"[63]. The act of performing this work has required me to purchase the camera which has been examined as well as the necessary hardware components used to examine it. The impact of these purchases is already small, but the camera will additionally be returned to the NSE department at KTH which has hosted this project. There other researchers can continue to use the camera in future research. The tools were purchased privately and will be kept for future projects. The act of finding critical vulnerabilities could also have an effect on consumers. If the camera had been shown to be critically insecure consumers might have opted to purchase a more secure replacement. In this case none of the vulnerabilities that were found are considered critical, since physical access is needed. It is also possible for the manufacturer to update the device to fix vulnerabilities, so getting a replacement shouldn't be necessary.

9.3 Ethics

Swedish law has to be considered when conducting a hacking project like this. The Swedish Brottsbalken [64] has laws which control who is allowed to access different systems and in what way. The NSE department at KTH has an excellent summary regarding this: ”Don’t hack someone else’s property. For IoT devices, for instance, you can hack the actual device if you/KTH own(s) it, but you may not attempt to hack the associated Cloud server, owned by the vendor. You can interact with the server in the intended way, though, of course, and you can monitor the traffic.” [65]. For this project the camera is owned by KTH and I have permission to hack it. Great care has been taken throughout the project to not perform any action which might be considered an attack against the Cloud server. The only connections to the Cloud has been those automatically created by the camera.

There are some complicated interactions with Swedish copyright law in regards to object code decompilation [66]. Since object code is protected under Swedish copyright law the act of decompiling it without permission, which can be considered a form of translation, can be interpreted as breaking the law. However, there is an exception where decompilation is allowed for the purpose of allowing program interoperability.

In the case of this thesis, a part of the installed software was decompiled. The purpose of this decompilation was to understand how the system handles updates through the SD-Card and to allow the system to work with a custom update program, which in this case is a program that starts a root shell on the camera. The decompilation should therefore be legal.

Another ethical consideration is in regards to vulnerability disclosures[67]. There are multiple ways to go about reporting vulnerabilities to a vendor. There is Private Disclosure where the vulnerabilities are disclosed directly to the vendor. It is then up to them to make the vulnerability public. In the case where a vendor ignores a vulnerability or tries to hide it some researchers do a Full Disclosure where they make the vulnerability public without an available fix ready from the vendor. As a compromise between these methods there is Responsible/Coordinated Disclosure. In this case the vulnerability is reported to the vendor which is then given a deadline before the researcher makes the vulnerability public. A CVE is also registered for the vulnerability.

Most of the vulnerabilities found in this thesis requires physical access to the camera, either internally or to the SD-Card. They utilize unsecured system functions. The systems actually works as designed, and aren’t utilizing any unintended bugs, just poor design from a security perspective. These findings don't qualify for disclosure or a CVE.

The old versions of the installed programs also already have CVEs registered for them, they aren't new discoveries. These won't be reported either.

Chapter 10

Conclusions and Future Work

10.1 Conclusions

The camera has many unused services that are enabled unnecessarily. These services expand the attack surface of the camera while not providing much value to consumers. The telnet service is enabled. It is locket with a password, but consumers don't have the possibility to access it. The Tuya service is open, but doesn't seem to be implemented. It also wasn't observed to ever be used by the app. The UART interface is enabled. No consumer is ever expected to use it, but it provides a full system log as well as access to the bootloader. All of these services should probably be disabled for consumer devices to minimize attack surface without compromising on usability. Writing log files to the SD-Card should also be disabled on consumer devices since it leaks sensitive information without adding any value for consumers. Finally the update mechanism using the SD-Card should be properly secured in order to prevent malicious files on the SD-Card from running with root privileges. The camera should also receive more regular software updates to fix vulnerabilities present in the tools and libraries installed on the camera. Since it is possible to gain root privileges with physical access we can draw the conclusion that the smart camera HELO View Indoor Security Camera is not secure against cyber attacks.

10.2 Future Work

More work can be done in order to find the password for the root account. This would let adversaries access a root console over the network. This could also include an analysis on if different units come with their own password hash or if the password is shared between every unit.

More work could also be done by more closely analyzing the tools and libraries installed on the camera and to find a vulnerability that enables an adversary to exploit the camera wirelessly.

The programs and data stored on the camera could also be more thoroughly analyzed, for example to be able to decrypt the locally stored video files as well as the video stream. This should be possible as all encryption keys should be stored somewhere on the device.

Further work can be done by analyzing more of the identified threats. This could also include finding new threats that weren't considered in this thesis.

References

[1] "IoT connected devices by vertical 2030," Statista. (), [Online]. Available: https://www.statista.com/statistics/1194682/iot-connected-devices-vertically/ (visited on 01/25/2023).

[2] "Smart termostat | vattenfall." (), [Online]. Available: https://www.vattenfall.se/fjarrvarme/energitjanster/smart-termostat/ (visited on 01/25/2023).

[3] "Electric blinds - smart blinds," IKEA. (), [Online]. Available: https://www.ikea.com/gb/en/cat/electric-blinds-44531/ (visited on 01/25/2023).

[4] "Smart doorbell Elartiklar," PriceRunner. (), [Online]. Available: https://www.pricerunner.se/sp/smart-doorbell.html (visited on 01/25/2023).

[5] "Övervakningskameror för ditt hem | NetOnNet." (), [Online]. Available: https://www.netonnet.se/art/hem-fritid/smarta-hem/overvakningskamera (visited on 01/25/2023).

[6] M. Stanislav and T. Beardsley, "HACKING IoT: A case study on baby monitor exposures and vulnerabilities,"

[7] B. Cusack and Z. Tian, "Evaluating IP surveillance camera vulnerabilities," *Australian Information Security Management Conference*, Jan. 1, 2017. doi: 10.4225/75/5a84efba95b46. [Online]. Available: https://ro.ecu.edu.au/ism/202.

[8] Y. Seralathan *et al.*, "IoT security vulnerability: A case study of a web camera," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*, Feb. 2018, pp. 172–177. doi: 10.23919/ICACT.2018.8323686.

[9] A. Tekeoglu and A. S. Tosun, "Investigating security and privacy of a cloud-based wireless IP camera: NetCam," in *2015 24th International Conference on Computer Communication and Networks (ICCCN)*, ISSN: 1095-2055, Aug. 2015, pp. 1–6. DOI: 10.1109/ICCCN.2015.7288421.

[10] P. A. Abdalla and C. Varol, "Testing IoT security: The case study of an IP camera," in *2020 8th International Symposium on Digital Forensics and Security (ISDFS)*, Jun. 2020, pp. 1–5. DOI: 10.1109/ISDFS49300.2020.9116392.

[11] F. Loi, A. Sivanathan, H. H. Gharakheili, A. Radford, and V. Sivaraman, "Systematically evaluating security and privacy for consumer IoT devices," in *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*, ser. IoTS&P '17, New York, NY, USA: Association for Computing Machinery, Nov. 3, 2017, pp. 1–6, ISBN: 978-1-4503-5396-0. DOI: 10.1145/3139937.3139938. [Online]. Available: https://doi.org/10.1145/3139937.3139938 (visited on 12/18/2022).

[12] R. Alharbi and D. Aspinall, "An IoT analysis framework: An investigation of IoT smart cameras' vulnerabilities," 47 (10 pp.)–47 (10 pp.) Jan. 1, 2018, Publisher: IET Digital Library. DOI: 10.1049/cp.2018.0047. [Online]. Available: https://digital-library.theiet.org/content/conferences/10.1049/cp.2018.0047 (visited on 01/24/2023).

[13] E. P.
bibinitperiod M. G. Legaspi. "Understanding the UART," Embedded.com. (Jan. 20, 2021), [Online]. Available: https://www.embedded.com/understanding-the-uart/ (visited on 01/26/2023).

[14] "Configuring serial port debug console (linux/u-boot) | toradex developer center." (), [Online]. Available: https://developer.toradex.com/software/development-resources/configuring-serial-port-debug-console-linuxu-boot/ (visited on 01/26/2023).

[15] "Tuya smart - apps on google play." (), [Online]. Available: https://play.google.com/store/apps/details?id=com.tuya.smart&hl=en_US&gl=US (visited on 01/26/2023).

[16] M. Howard, *The security development lifecycle : SDL, a process for developing demonstrably more secure software*, 1st edition, ser. Best practices. Redmond, Washington: Microsoft Press, 2006, Publication Title: The security development lifecycle : SDL, a process for developing demonstrably more secure software, ISBN: 0-7356-3557-9.

[17] ——, *Writing secure code*, 2nd ed., ser. Developer Best Practices Ser. Sebastopol: Microsoft Press, 2004, ISBN: 0-7356-3740-7.

[18] "Wireshark · go deep." (), [Online]. Available: https://www.wireshark.org/ (visited on 11/24/2022).

[19] G. Lyon. "Nmap: The network mapper - free security scanner." (), [Online]. Available: https://nmap.org/ (visited on 11/24/2022).

[20] "What is penetration testing? - pen testing," Cisco. (), [Online]. Available: https://www.cisco.com/c/en/us/products/security/what-is-pen-testing.html (visited on 01/26/2023).

[21] Strong. "Company." (), [Online]. Available: https://eu.strong.tv/se/company (visited on 11/24/2022).

[22] "Scan QR code to pair IPC-IoT app SDK-tuya developer." (), [Online]. Available: https://developer.tuya.com/en/docs/app-development/camera-scan-code-network-configuration?id=Kaq6nynklqxbz (visited on 01/24/2023).

[23] "BusyBox." (), [Online]. Available: https://www.busybox.net/ (visited on 01/26/2023).

[24] "Which system ports do an SDK use?" (), [Online]. Available: https://support.tuya.com/en/help/_detail/Kb6bjb8yyf1nf (visited on 01/24/2023).

[25] "Can smart devices be controlled only through the LAN without using cloud services?" (), [Online]. Available: https://support.tuya.com/en/help/_detail/K9tjtiy33x3qf (visited on 01/24/2023).

[26] M. Isom, *TuyAPI*, original-date: 2017-10-25T14:45:32Z, Jan. 25, 2023. [Online]. Available: https://github.com/codetheweb/tuyapi (visited on 01/26/2023).

[27] "MQTT - the standard for IoT messaging." (), [Online]. Available: https://mqtt.org/ (visited on 01/26/2023).

[28] "What is HTTPS?" Cloudflare. (), [Online]. Available: https://www.cloudflare.com/learning/ssl/what-is-https/ (visited on 01/26/2023).

[29] *STUN*, in *Wikipedia*, Page Version ID: 1122415180, Nov. 17, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=STUN&oldid=1122415180 (visited on 01/26/2023).

[30] *U-boot/u-boot*, original-date: 2014-11-12T13:29:02Z, Jan. 26, 2023. [Online]. Available: https://github.com/u-boot/u-boot (visited on 01/26/2023).

[31] OWASP. "OWASP IoT top 10." (2018), [Online]. Available: https://owasp.org/www-pdf-archive//OWASP-IoT-Top-10-2018-final.pdf.

[32] Patrick Wardle & Colby Moore, "Optical surgery - implanting a DropCam," 2014. [Online]. Available: http://archive.org/details/Defcon22_Talk3 (visited on 01/22/2023).

[33] A. Mike. "UART-to-root: The (slightly) harder way," Exfil.co. (Feb. 15, 2019), [Online]. Available: https://exfil.co/2019/02/14/uart-to-root-the-harder-way/ (visited on 01/22/2023).

[34] A. Borg and C. A. Francke, *IoT Pentesting: Obtaining the Firmware of a Smart Lock*. 2020. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-279543 (visited on 01/24/2023).

[35] "Accessing and dumping firmware through UART." (), [Online]. Available: https://www.cyberark.com/resources/threat-research-blog/accessing-and-dumping-firmware-through-uart (visited on 01/24/2023).

[36] "IPC SDK architecture-IoT app SDK-tuya developer." (), [Online]. Available: https://developer.tuya.com/en/docs/app-development/architecure?id=Ka6nuvwrlt53p (visited on 01/24/2023).

[37] *Tuya embedded SDK for multimedia devices*, original-date: 2020-11-12T08:39:16Z, Jan. 3, 2023. [Online]. Available: https://github.com/tuya/tuya-iotos-embeded-sdk-multimedia (visited on 01/26/2023).

[38] "P2p capabilities-IoT app SDK-tuya developer." (), [Online]. Available: https://developer.tuya.com/en/docs/app-development/android-p2p?id=Kalemnh8lf8yi (visited on 01/24/2023).

[39] "Live video streaming-IoT app SDK-tuya developer." (), [Online]. Available: https://developer.tuya.com/en/docs/app-development/livestream?id=Ka6nuvynnjk7h (visited on 01/24/2023).

[40] "Audio and video features-IoT app SDK-tuya developer." (), [Online]. Available: https://developer.tuya.com/en/docs/app-development/avfunction?id=Ka6nuvucjujar (visited on 01/24/2023).

[41] "TUYA_ipc_sdk/stable/rsdk-6.4.1-linux at master · TuyaInc/TUYA_ipc_sdk," GitHub. (), [Online]. Available: https://github.com/TuyaInc/TUYA_IPC_SDK (visited on 01/26/2023).

[42] J. Li, Z. Li, G. Tyson, and G. Xie, "Your privilege gives your privacy away: An analysis of a home security camera service," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, ISSN: 2641-9874, Jul. 2020, pp. 387–396. DOI: 10.1109/INFOCOM41043.2020.9155516.

[43] J. Ren, D. J. Dubois, D. Choffnes, A. M. Mandalari, R. Kolcun, and H. Haddadi, "Information exposure from consumer IoT devices: A multidimensional, network-informed measurement approach," in *Proceedings of the Internet Measurement Conference*, ser. IMC '19, New York, NY, USA: Association for Computing Machinery, Oct. 21, 2019, pp. 267–279, ISBN: 978-1-4503-6948-0. DOI: 10.1145/3355369.3355577. [Online]. Available: https://doi.org/10.1145/3355369.3355577 (visited on 12/12/2022).

[44] H. Li, Y. He, L. Sun, X. Cheng, and J. Yu, "Side-channel information leakage of encrypted video stream in video surveillance systems," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, Apr. 2016, pp. 1–9. DOI: 10.1109/INFOCOM.2016.7524621.

[45] N. Apthorpe, D. Reisman, and N. Feamster, *A smart home is no castle: Privacy vulnerabilities of encrypted IoT traffic*, May 18, 2017. DOI: 10.48550/arXiv.1705.06805. arXiv: 1705.06805[cs].

[Online]. Available: http://arxiv.org/abs/1705.06805 (visited on 12/12/2022).

[46] *Binwalk*, original-date: 2013-11-15T20:45:40Z, Jan. 26, 2023. [Online]. Available: https://github.com/ReFirmLabs/binwalk (visited on 01/26/2023).

[47] "SD card-stored video playback-IoT app SDK-tuya developer." (), [Online]. Available: https://developer.tuya.com/en/docs/app-development/playback?id=Ka6nuvyi69kwd (visited on 01/24/2023).

[48] "SD card management-IoT app SDK-tuya developer." (), [Online]. Available: https://developer.tuya.com/en/docs/app-development/sdcard?id=Ka6nxw2eufia3 (visited on 01/24/2023).

[49] *Welcome to the OpenSSL project*, original-date: 2013-01-15T22:34:48Z, Jan. 26, 2023. [Online]. Available: https://github.com/openssl/openssl (visited on 01/26/2023).

[50] *Heartbleed*, in *Wikipedia*, Page Version ID: 40563969, Jul. 19, 2017. [Online]. Available: https://sv.wikipedia.org/w/index.php?title=Heartbleed&oldid=40563969 (visited on 01/26/2023).

[51] "Minicom / minicom · GitLab," GitLab. (Nov. 18, 2022), [Online]. Available: https://salsa.debian.org/minicom-team/minicom (visited on 01/26/2023).

[52] "Squashfs 4.0 filesystem — the linux kernel documentation." (), [Online]. Available: https://www.kernel.org/doc/html/latest/filesystems/squashfs.html (visited on 01/26/2023).

[53] "Strings(1) - linux manual page." (), [Online]. Available: https://www.man7.org/linux/man-pages/man1/strings.1.html (visited on 01/26/2023).

[54] "The GNU netcat – official homepage." (), [Online]. Available: https://netcat.sourceforge.net/ (visited on 01/26/2023).

[55] "John the ripper password cracker." (), [Online]. Available: https://www.openwall.com/john/ (visited on 01/26/2023).

[56] D. Miessler, *Danielmiessler/SecLists*, original-date: 2012-02-19T01:30:18Z, Jan. 26, 2023. [Online]. Available: https://github.com/danielmiessler/SecLists/blob/decd3cb559844df67b0adb5f1b743c6d0c861126/Passwords/Malware/mirai-botnet.txt (visited on 01/26/2023).

[57] hackerz, *Rockyou.txt*, original-date: 2022-01-26T15:57:54Z, Jan. 11, 2023. [Online]. Available: https://github.com/redfiles/rockyou.txt (visited on 01/26/2023).

[58] Ben, *Probable wordlists - version 2.0*, original-date: 2017-04-16T17:08:27Z, Jan. 25, 2023. [Online]. Available: https://github.com/berzerk0/Probable-Wordlists (visited on 01/26/2023).

[59] "Weakpass." (), [Online]. Available: https://www.weakpass.com/ (visited on 01/26/2023).

[60] "Busybox CVE - OpenCVE." (), [Online]. Available: https://www.opencve.io/cve?vendor=busybox (visited on 01/25/2023).

[61] "CVE-2018-20679 - OpenCVE." (), [Online]. Available: https://www.opencve.io/cve/CVE-2018-20679 (visited on 01/25/2023).

[62] "W1.fi wpa supplicant : List of security vulnerabilities." (), [Online]. Available: https://www.cvedetails.com/vulnerability-list.php?vendor_id=12005&product_id=29296 (visited on 01/25/2023).

[63] "Goal 12 | department of economic and social affairs." (), [Online]. Available: https://sdgs.un.org/goals/goal12 (visited on 01/25/2023).

[64] "Brottsbalk (1962:700) (BrB) | lagen.nu." (), [Online]. Available: https://lagen.nu/1962:700 (visited on 01/25/2023).

[65] "The law," NSE Lab. (), [Online]. Available: https://nse.digital/pages/thesis_guidelines/the_law.html (visited on 01/25/2023).

[66] "Lag (1960:729) om upphovsrätt till litterära och konstnärliga verk (URL) | lagen.nu." (), [Online]. Available: https://lagen.nu/1960:729 (visited on 01/25/2023).

[67] "Vulnerability disclosure - OWASP cheat sheet series." (), [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Vulnerability_Disclosure_Cheat_Sheet.html (visited on 01/25/2023).

TRITA-EECS-EX-2023:243