Czech University of Life Sciences Prague

Faculty of Economics and Management

Department of System Engineering and Informatics

CZECH
UNIVERSITY
OF LIFE SCIENCES PRAGUE

Diploma Thesis

IoT and Future of Security Threat

Author
Haiylegebrel Wubamlak Senbeta

Supervisor
Ing. Martin Havránek, Ph.D

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

DIPLOMA THESIS ASSIGNMENT

B.Sc. HAIYLEGEBREL WUBAMLAK SENBETA, BSc

Systems Engineering and Informatics
Informatics

Thesis title

IOT and Future of Security Threat

Objectives of thesis

The main objective of this thesis is to perform security analysis on different embedded device software/ IOT (Internet of things) from different vendors and which have different architecture and to propose standard security measures to decrease the risk caused malicious activities of adversary.

Methodology

To achieve the objective there is need to review the technologies, architectures, current security issues and the security future of IOT which is next evolution of internet that will enable network infrastructure to connect large number of devices.

The next step is to identify devices which have their update and firmware is publicly available and Collecting firmware from different vendors and from different architectures.

Thus, the technical security analysis of the firmware with different methodologies (SAST and DAST). For SAST (Static Application Security Testing) by reversing the binary file and then from the source code using different tools and manual review. For DAST (Dynamic Application Security Testing) by emulating this firmware performing security testing.

From the analysis result data performing risk analysis and prepare security measures and best practices for IOT to decrease the risk.

The proposed extent of the thesis

60-80p.

Keywords

IOT,SAST,DAST,Firmware,Embedded device

Recommended information sources

Frustaci, Mario, et al. ”Evaluating critical security issues of the IoT world: Present and future challenges.” IEEE Internet of things journal 5.4 (2017): 2483-2495.

Chiang, Mung, Bharath Balasubramanian, and Flavio Bonomi, eds. Fog for 5G and IoT. Vol. 288. Wiley, 2017.

Li, Shancang, Li Da Xu, and Shanshan Zhao. ”5G Internet of Things: A survey.” Journal of Industrial Information Integration 10 (2018): 1-9.

Mahmoud, Rwan, et al. ”Internet of things (IoT) security: Current status, challenges and prospective measures.” 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST). IEEE, 2015.

Zhang, Zhi-Kai, et al. ”IoT security: ongoing challenges and research opportunities.” 2014 IEEE 7th international conference on service-oriented computing and applications. IEEE, 2014.

Expected date of thesis defence

2020/21 SS – FEM

The Diploma Thesis Supervisor

Ing. Martin Havránek, Ph.D.

Supervising department

Department of Information Technologies

Electronic approval: 3. 8. 2020

Ing. Jiří Vaněk, Ph.D.

Head of department

Electronic approval: 21. 10. 2020

Ing. Martin Pelikán, Ph.D.

Dean

Prague on 07. 03. 2021

Declaration

I declare that I have worked on my diploma thesis titled "IoT and Future of security threat" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the diploma thesis, I declare that the thesis does not break copyrights of any their person.

In Prague on March 30, 2021

Haiylegebrel Wubamlak Senbeta

Acknowledgement

First of all, I would like to express my thankfulness to almighty God, who gave me the strength, commitment and health to finish this master thesis.

Secondly, I would like to pass my appreciation to my supervisor Ing. Martin Havránek, Ph.D. for his support and advice while the process of writing this thesis work and to the Czech Ministry of Sports and Education for the scholarship that made these master’s studies available to me.

Finally, I would like also to pass my gratitude to my family, friends who were always on the side of me to unleash all the barriers and supporting me, while writing this thesis.

IoT and Future of Security Threat

Abstract

This thesis focuses on the rapidly growing Internet of things firmware security analysis and to propose standard security measures and IoT module for detecting threats and malicious activities to decrease the risk and to take appropriate measures.

The introductory part contains the objectives of the thesis and the methodology by which the goals will be achieved. The theoretical or literature part is dedicated to mostly to identify IoT devices, Attack surfaces and IoT security and privacy challenges and the researches which is done related to this work.

The practical part describes the firmware selection criteria's to identify different vendors, devices and firmware's. The next section shows the security analysis and results of security findings on that selected firmware's to show the overall risks of the current IoT environment. Finally, briefly show the proposed system design model and activities for IoT cyber threat detection module (ICTDM) and each subcomponent.

The final section shows the result and discussion, firmware risk analysis of these thesis and best practices to secure the IoT environment. Then concludes this work by describing the future work for IoT Devices.

Keywords: Internet of things, Firmware, Security analysis, Cyber threat, Risk, ICTDM,

IoT a budoucnost bezpečnostní hrozby

Abstrakt

Tato práce se zaměřuje na rychle rostoucí analýzu zabezpečení firmwaru internetu věcí a na návrh standardních bezpečnostních opatření a modulu IoT pro detekci hrozeb a škodlivých aktivit s cílem snížit riziko a přijmout vhodná opatření.

Úvodní část obsahuje cíle práce a metodiku, jichž bude cílů dosaženo. Teoretická část nebo část věnované literatuře se věnuje především identifikaci zařízení IoT, útočným plochám a výzvám zabezpečení a ochrany soukromí v oblasti IoT a výzkumům, které se v souvislosti s touto prací provádějí.

Praktická část popisuje kritéria výběru firmwaru k identifikaci různých dodavatelů, zařízení a firmwaru. Následující část ukazuje bezpečnostní analýzu a výsledky bezpečnostních zjištění u vybraného firmwaru a ukazuje celková rizika současného prostředí IoT. Nakonec stručně ukažte navrhovaný model návrhu systému a aktivity pro modul IoT pro detekci kybernetických hrozeb (ICTDM) a jednotlivé dílčí součásti.

V závěrečné části jsou uvedeny výsledky a diskuse, analýza rizik firmwaru z této práce a osvědčené postupy pro zabezpečení prostředí IoT. Poté uzavře tuto práci popisem budoucí práce pro zařízení IoT.

Klíčová slova: Internet věcí, Firmware, Bezpečnostní analýza, Kybernetická hrozba, Riziko, ICTDM

Contents

List of Tables

List of Figures

Abbreviation

IOT	Internet of Things
SAST	Static application security testing
DAST	Dynamic application security testing
CVSS	Common vulnerability scoring system
LwM2M	Light weight machine to machine communication protocol
MQTT	Message Queuing telemetry transport
HTTP	Hypertext transfer protocol
REST	Representational State transfer
API	Application Package Interface
IETF	Internet Engineering Task Force
RFID	Radio Frequency Identification
DDS	Data Distribution Service
AMQP	Advanced Message Queuing Protocol
XMPP	Extensible Messaging Presence Protocol
CoAP	Constrained Application Protocol
DDoS	Distributed Denial of Service
C&C	Command and Control
OT	Operational Technology
UART	Universal Asynchronous Receiver Transmitter
JTAG	Joint Test Access Group
TCP	Transmission Control Protocol
OSI	Open System Interconnection
SSH	Secure Shell
BIOS	Basic Input Output System
UDP	User Datagram Protocol
TLS	Transport Layer Security
DTLS	Datagram Transport Layer Security

1. Introduction

1.1. Background

IoT devices are omnipresent in our day to day activities of life and they are becoming increasingly used in many computing environments and infrastructures The popularity of Internet of Things has increased rapidly, as these technologies are used for various purposes, including communication, transportation, education, and business development. IoT introduced the hyper connectivity concept, which means organizations and individuals can communicate with each other from remote locations effortlessly. In fact, multiple reports estimate an increase in the number of IoT devices in the next few years. Cisco famously predicted that there will be 75 billion of connected IoT devices by 2025 [3]. Those devices will be produced by many different manufacturers and will be present in many different models. Each will probably have several firmware versions, leading to an overall huge number of firmware releases. Hundreds of thousands of firmware images are already available, which is just a lower bound estimate of publicly observable firmware packages. The number of firmware files will likely only grow with the number of new embedded devices being developed and deployed.

The rapid explosion of IoT has benefitted organizations and in various ways improved market research and business strategies. In a similar way, IoT has changed people's lives by introducing and incorporating automated services. However, such unregulated growth has increased privacy and security challenges. At the same time, the security of an average IoT device's firmware is empirically shown to be often weak [4][5]. This has been frequently shown by independent evaluations [6][7]. Such evaluations often show that the security of many IoT devices and their firmware is very low. That once again proves that many vendors are usually more interested in fastest and cheapest release of new products and features to increase their market share. This practice is usually opposite to building secure products and accurately testing them against current and future security threats. These facts are even more worrying because the security flaws in the embedded devices and their firmware are many times found by security practitioners using approaches that are neither systematic nor automated.

The unconscious use, the failure to change passwords, and the lack of device updates have increased cybersecurity risks and gives access to malicious attackers to the IoT environment sensitive information. Data breaches and other risks are more likely as a result of such bad security practices. Because of its insufficient security standards and policies, most security professionals consider IoT to be a vulnerable point for cyber-attacks. Despite the development of many protection measures to secure IoT devices from cyber-attacks, security guidelines are not well established [8]. As a result, end-users were unable to use security mechanisms to avoid data breaches. Since the emergence of the IoT era, hackers have created various types of malware to infect IoT devices. They invented a number of phishing tactics to entice workers or individuals to divulge confidential information [9]. Moreover, vulnerabilities in the firmware constitute an easy entry point for malicious software and make the embedded devices prone to simple yet devastating attacks. Even worse, the affected embedded devices are hard to diagnose and clean (e.g., no embedded anti-virus solutions, no conventional input/output). Therefore, they often remain exploited for long periods of time. Therefore, corporate workstations and personal devices often face privacy violations due to high-profile attacks. If device manufacturers and security experts assess the cyber threats accurately, they can develop relatively efficient protective mechanism to prevent or neutralize cyber threats.

Every day new technologies emerge, or changes are made to existing ones. [10] Consider the latest advances in the 5G network, for example. 5G is expected to play an essential role in the IoT systems and applications. It is getting the researchers' attention and curiosity about the possible security and privacy risks, with its high frequency and bandwidth. However, the short wavelength necessitates a shift in infrastructure, necessitating the deployment of more base stations to serve the same region as other wireless technologies. Fake base stations, for example, pose a greater threat under this new system. It is essential to understand the security risks and potential solutions.

1.2. Motivation of the study

Since the beginning of IoT era and still the current days it's in major development on different aspects. Different devices that are helpful in our daily life and also in industries and organization playing a great role. Multiple vendors and manufacturers are also developing and deploying different kinds of IoT Devices. But the major concern here is they doesn't have common standards they design and develop based on their requirement. IETF is also still on the phase of requirement and standards. This causes major security and privacy concerns and efficient protection and detection mechanisms are not present until current day even if multiple proposals and prototype are designs by many researchers. As any computer system they don’t have any antimalware and virus detection mechanisms due to their super minimal computational power. And these kinds of software and applications are considered as effective than other types protection mechanisms. Considering this why doesn’t super minimal anti malware and virus detection still not designed and developed?

1.3. Organization of the study

The organization of this thesis organized as follows: In Chapter 2 we will discuss literature review about IoT devices, Attack surfaces and IoT security and privacy challenges and the researches which is done related to this work. In Chapter 3 we will identify different vendors, devices and firmware’s then we will perform security analysis on that selected firmware’s based on the criteria’s that we have identified to show the overall risks of the current IoT environment. In Chapter 4, we will briefly show our proposed system design model and activities for IoT cyber threat detection module and each subcomponents. Chapter 5 presents the result and discussion of these thesis and best practices to secure the IoT environment. Chapter 6 concludes this work by describing The future of the IoT environment on different applications.

2. Objective and Methodology

2.1. Objectives

2.1.1. General Objective

The main objective of this thesis is to perform security analysis on different Internet of Things device from different vendors and which have different architecture/file system and to propose standard security measures and IoT module for detecting threats and malicious activities to decrease the risk and to take appropriate measures which may caused by attacker.

2.1.2. Specific Objective

- Analyzing IoT Systems, Architectures/File system
- Analyzing security issues on selected Device
- Analyzing IoT Communication Protocols
- Designing IoT Cyber detection module

2.2. Methodology

The methodologies to achieve the above described objectives for IoT Devices there is a need to review the IoT technology and environment, their architecture and designs from even if their design is vendor specific, the main challenges in the current IoT environment, the security and privacy issues and their attack surface. This is going to help for better understanding of the overall thigs in the IoT environment to designs the proposed module for IoT cyber threat detection.

To achieve the analysis architecture and security issues of selected device firmware, first we search for different IoT device vendors which have many connected devices on the current worldwide internet connection and we will mainly focus on the most popular widely used IoT devices. Gathering their firmware's for the analysis phase.

In the analysis phase we will use different tools for reverse engineering the binary file, extracting the file system and to inspect the code and the main components of services and modules running on the device. We will use two different kind of approach for analysis which are SAST (Static application security testing) and DAST (Dynamic application security testing) focusing mainly on the SAST part. Sometimes the to analyze the firmware in the DAST approach we have to emulate the firmware or get the original devices for testing. So in these research we will try to emulate the whole firmware or each executable file by itself which will give us partial file emulation using qemu.

The next step is to categorize the result of the security assessment of those selected device or vulnerability to may let an attacker to exploit based on the most popular IoT security framework to differentiate the issues.

At this stage we have all the vulnerabilities that may cause security and privacy issues and we become aware of the most of IoT technologies and architecture, security issues then we will design IoT cyber threat detection system based on those device behaviors.

3. Literature Review

3.1. A brief History of IoT

The concept of IoT devices started in 1832 when the first electromagnetic telegraph [11] was designed. This device is designed to allow communication between two devices by the transfer of electronic signals. However, the real IoT history begins with the time when internet is being widely used in the late 1960s [12], which is then a growing technology over the next decades.

The first IoT device was a Coca-Cola [13] vending machine build at the Carnegie Melon University in 1980s and They integrated controller into the machine and used an internet to see if the machine is cooling the drinks enough and to check the available cans remaining in the machine. This machine inspired many peoples for further studies in these field and the development of interconnected machines.

In the 1990s, John Romkey used a Communication Protocols to connect a toaster to the internet for the first time [14]. Later year, Cambridge University scientists brings new idea to use the first webcam to control the amount of coffee available in their laboratory coffee pot. They configured the webcam to take images of the coffee pot multiple times in minute, then they receive images to their monitoring computer, then this allowing everyone to see if there was coffee available or not.

The term Internet of Things has not been known for very long time. However, the real concept of connected devices is being known for longer period of time. The concept was called “embedded internet” at that time. The actual name “Internet of Things” first used by Kevin Ashton [1] in 1999 during his work at P&G. till 1999, the term “internet of things” wasn’t known. Ashton needed to draw in senior management’s attention to a replacement exciting technology known as RFID. And at that time the internet was the popular new trend, he called his research “Internet of Things”. The name “Internet of Things” did not get viral for the next decades.

3.2. Realizing the concept

Internet of Things is of any device which is composed of its own sensing and actuator device and controller device and connected to the Internet [15]. This covers anything, ranging from phones to any maintenance device to the plane engine. Health monitoring devices, for example anybody controlling implant and a microchip device which can transfer information over a network and are members the Internet of things. If it has a toggle switch with some sensing device integrated and controlling and processing mechanism, then it can, theoretically, be part of the system. Internet of things is a collection of an enormous network of interconnected “things” and devices. One of another example a Ring doorbell which connects with smart phone, provides a good example of a recent addition to the IoT. Ring signals of the door notifies when the doorbell is pressed, and lets the owner see who it is and to speak with them remotely.

Figure 3.1 IoT Devices [source:16]

3.3. IoT Takes off

At the beginning of the 21st century, the name “internet of things” became viral and use by many medias [17]. IoT technology is getting attention gradually increasing, then led’s to the first worldwide summit on the IoT organized in Switzerland in 2008, where participant’s different countries from all over the world discussed RFID, short-range wireless communications, and sensor networks.

In addition, several major developments have contributed to the evolution of IoT. One was the Internet-connected refrigerator that LG Electronics introduced in 2000, allowing users to shop online and make video calls [18]. A tiny rabbit-shaped robot called Nabaztag [19] developed in

2005 was another key invention that was able to report the latest news, weather forecasts, and changes in the stock market. According to Cisco [3], even at that time, the number of interconnected devices surpassed from the total individual on the earth.

In our daily lives, interconnected devices have since become ubiquitous and commonplace. Global technology companies [20] such as Google, Tesla, Apple, Cisco, Samsung and General Motors are concentrating their attention on new IoT sensor technologies and applications ranging from interconnected thermostats and smart glasses to self-driving vehicles. The Internet of Things has expanded almost every industry and technology. For example, transportation, oil & energy, healthcare systems, manufacturing, agriculture, retailing, and many more. This drastic change has persuaded us that right here, right now, is the IoT revolution. As of today, along with virtual assistants, smart homes, and level 4 self-driving cars, IoT system retain and maintains a firm grip on their place among the top trends in this year's Gartner Hype Period.

3.3.1. Architecture of Internet of Things

Basically the current IoT system architecture is divided into three layers [22] which is the perception usually called actuating and sensing layer, the network layer also called communication layer, and the application layer. There also four-layer, five-layer and seven-layer architecture. we will use the widely used three-layer architecture. The following image shows the basic three-layer architecture of IoT system.

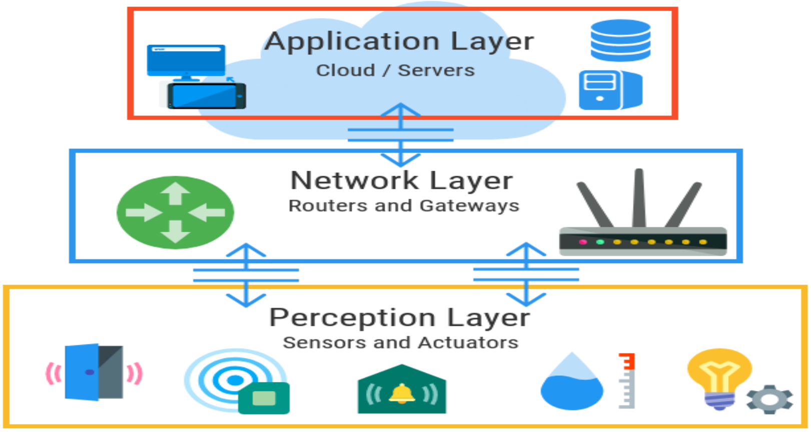


Figure 3.2 Three tire IoT architecture [source:21]

3.3.1.1. Perception Layer

The perception layer is usually used as the sixth senses. this Sensors, actuators and peripheral devices that communicate with the environment. It is used for object recognition, object perception, information processing, and automatic control. The temperature sensor mounted on the air conditioner, for instance, recognizes that the temperature indoors is higher than 30 degrees. It switches on automatically after collecting this information. Cooling air conditioning This degree includes different detection technologies, information gathering technologies, and control technologies. Additionally, they are used cross-wise. this perception can be single and extensive. For example, robots may have or integrated different perception systems. In these layer different sensors and actuators devices used to substitute human senses for better understanding of the physical world and these are usual devices in this layer, it also includes Radio Frequency Identification and two-dimensional code technologies for example barcode, QR code used in the many enterprise information processing.

3.3.1.2. Network Layer

This layer is responsible for discovering, connecting the devices through the network (communication protocol) and interact with the application layer. The network layer mainly transmits information, routing (deciding in which path the information transmits) and manages that the information being transmitting. which is divided into two parts, one part is the communication technology of the Internet of Things, and the other is the communication protocol of the Internet of Things. The communication part is responsible for connecting devices and these devices can exchange information's. The protocol is responsible for managing establishing controlling the communication rules and formatting.

There are different IoT communication protocols which is based and depend on as their communication technologies [23], such as MQTT, DDS, AMQP, XMPP, JMS, REST, CoAP, OPC UA [24]. Equivalent to the human brain and nervous system center, the network layer is primarily responsible for sending and receiving and processing the information received by the perception layer.

3.3.1.3. Application Layer

This layer is the top most layer and also known as “Service” layer. It also consists of another two sub layers, the application service layer and the data management layer [22]. The data management layer provides over all data and information management, combining, directory service, facility management, Quality of Service, etc. The application service layer transforms data into deliverable content and presents a nice user interface for end-users and for different industrial applications input such as agricultural management, environmental, disaster detection and notification, logistics and, production monitoring, etc.

3.3.2. Application of Internet of Things

Smart Home

Smart Home is the most common IoT system feature and is becoming one of the domestic revaluation measures. Many Companies and vendors currently on the production of different on shelf items and devices to provide you more convenient and simple life such as turning lights on/off even you can turn your air conditioner before entering the home. There is a future prediction that smart homes will be widely used and become popular [25]. Those devices and home on shelf appliance promised to save money, energy and time.

Wearables

These devices have software’s and sensors that collect data about their users and extract many information from the user who wears. Google Watch and Glasses [26] are a good examples of Human Wearable. Wearables cover health, fitness and entertainment needs.

Cars

A car which have integrated IoT devices in an automated car that can optimize a car’s operation and maintenance with internet connection and gathers information’s from onboard sensors. Automotive IoT focus is on optimizing automobiles internal functionality. Many car companies [27] like BMW, Tesla, Google, Apple are working on connected car solutions to the next level.

Industrial

Internet of Things is a new attraction in the industries. IoT is allowing many industries business with big data analysis, sensors, and systems to develop advanced equipment’s and machineries. There prospects about IoT which can control production quality and reliabilities [28].

In addition, IoT applications would improve the productivity of the stock chain for timely delivery and trailing items and the sharing of account information between retailers and suppliers in real time.

Smart Cities

A smart city is additionally alternative massive development among individuals [29]. By integrating IoT applications, sensors and software peoples will realize free parking slots, save energy and water with gooder electricity management systems and smart water provides, resolve transport issues with automatic transport and management pollution with environmental management.

Agriculture

Another fastest growing area of IoT is Smart farming. As demand in food, supply is increasing everyday with the increasing population. Many governments and industrial sector are motivating farmers to adopt new technology and research to multiply the yield [30]. Some helpful examples of IoT in agriculture area are controlling unit dominant water consumption, determining fertilizer schedule and analyzing soil for nutrients and moisture. It is a good practice for the farmers using useful IoT sensors and systems to improve growth and investment.

Healthcare

IoT in the Healthcare area is still under development. However, the implementation of smart medical systems and integrated medical system has enormous potential for the welfare of broader population and pharmaceutical companies as well [31].

The Healthcare area is studying on IoT on major scales to empower individuals for a healthier life by sporting connected Healthcare devices. Connected Healthcare devices can collect information on associate individual’s health condition which will facilitate physicians to make customized treatment methods for the sickness.

Smart Retail

IoT has huge potential in the area of retailing. With IoT applications, retailers engage to their client’s to improve their experience. Smartphones also will enable retailers to connect and serve better them in store customers as well as offline customers by offering the best products online.

3.3.3. Advantages of Internet of Things

IoT device enables the communication between different devices, also called as Machine-to-Machine (M2M) or peer to peer communication [32]. Thus, the physical devices are able stay connected therefore with fewer inefficiencies, greater efficiency and complete transparency is available. There is a significant amount of automation and control in this environment due to physical devices being linked and managed digitally and centrally with different network infrastructure. When these devices are communicating each other without human interference, as a result of faster and more reliable production.

Another advantage of IoT is monitoring. Having more information helps with making better decisions. Information is strength, and more knowledge is stronger, whether it's for mundane decisions like choosing what to purchase at the grocery store or assessing whether your business has enough widgets and supplies. Knowing the precise amount of supplies or your home's air quality, will give additional information that would not have antecedently been collected simply [32]. As an example, knowing that you just are low on milk or printer ink might prevent another trip to the shop. Furthermore, controlling the expiration date of the items can and will improve safety. As mentioned in the previous examples, the amount of time saved because of IoT could be quite large. And in today's developing life, we have a tendency to all might use longer.

The biggest advantage of IoT is saving money. The Internet of Things would be generally embraced if the cost of tagging and tracking devices will be less than the amount of money saved. [33] IoT is fundamentally beneficial to people in their daily lives by allowing appliances to interact with one another in an efficient manner, thus reducing and maintaining energy and money. Allowing data to be exchanged and transmitted between devices, and then converting it into the format we need, makes our systems more effective.

Machine-to-machine connectivity improves performance, allowing for faster and more reliable results [34]. As a result, precious time is saved. It encourages people to do other interesting jobs rather than repeating the same activities daily. All of this technology's developments result in improved comfort, convenience, and better management, all of which enhance people's quality of life.

3.3.4. Disadvantages of Internet of Things

There is currently no international standard for IoT device compatibility. In the IoT ecosystem, there are millions and billions of devices which are connected to each other, but they are all manufactured by various companies, which raises the question of product compatibility when it comes to identifying and controlling [33]. It's difficult to persuade any company that their products adhere to a common norm. Even if IoT devices dominate the world, Bluetooth will link different devices, which might lead to compatibility issues. This compatibility issue may force the buyers or users to buy devices from a specific manufacturer which may create a monopoly in the market.

These will lead to more complex environment and there are more opportunities of failure. With the Internet of Things, failures could skyrocket. For example, a software bug can cause send you a message of a shortage of one particular item multiple times when it is needed to send for once only.

There is a risk of losing privacy when data is transferred from IoT devices, so an encrypted communication mechanism is needed. IoT is experiencing security and privacy problems as a result of its rapid development. When data were collected and transferred from the IoT device, IoT devices connected to your computer or laptop increase the chance of leaking users' personal information [32].

IoT is expanding across the globe, and this may result in the replacement of monotonous and unsafe jobs firing off unskilled professionals [32]. All of this may create unemployment issues in society. Daily operations are becoming more automated as a result of IoT, and as a result, there will be less demands on human resources and a lower level of education. As a consequence, developing new ways of doing things will boost or sustain an individual's employability.

IoT has had a major effect on almost everyone's life in some way. Everyone is dependent to technologies for their day-to-day operations, whether it is the older or younger generation. This reliance can grow even more in daily activities with the support of IoT. Every technological application has some flaws, and no application is without flaws. In the event that an IoT infrastructure fails or crashes, relying on IoT devices can cause problems [34].

3.4. IoT Security

IoT security can be described as a security strategy and prevention method that specifically protects IoT devices interconnected to the network and strategically designed for a particular set of features and functions from cyberattacks. Without strong protection, a connected IoT device can be breached and exploited by an intruder, allowing them to steal user data and bring down systems. IoT security should be integrated into common practice, method, and protocol by network security and operations stakeholders to ensure that unmanaged devices have the same level of visibility and control as controlled devices.

The expansion of the threat landscape as more and more different types of IoT devices connected to the network is a reason for problems when it comes to protecting IoT. Furthermore, unless the least protected system is already sufficiently secured, the entire network security status is reduced to the degree of credibility and safety provided to that device. Furthermore, 98% of all IoT computer traffic is unencrypted [35], putting personal and confidential data at danger. The study [36] shows concerns for security in 2020

98% of all IoT network traffic is unencrypted, which exposes sensitive data on the network.	**57%** of IoT devices are vulnerable to medium and high severity attacks	**83%** of medical health devices runs outdated operating systems and components

Table 3.1 Security concerns in 2020 [source:36]

These devices are produced by many different manufacturers and will be present in many different models. Each will probably have several firmware versions, leading to an overall huge number of firmware releases. Hundreds of thousands of firmware images are already available, which is just a lower bound estimate of publicly observable firmware packages [35]. The number of firmware files will likely only grow with the number of new embedded devices being developed and deployed. At the same time, the security of an average embedded device's firmware is empirically shown to be often weak. This has been frequently shown by independent evaluations and security researchers. Such evaluations often show that the security of many embedded devices and their firmware is very low. That once again proves that many vendors are usually more interested in fastest and cheapest release of new products and features to increase their market share. This practice is usually opposite to building secure products and accurately testing them against current and future security threats. These facts are even more worrying because the security flaws in the IoT devices and their firmware are many times found by security practitioners using approaches that are neither systematic nor automated. Below are different devices with highest share of security issue [36].

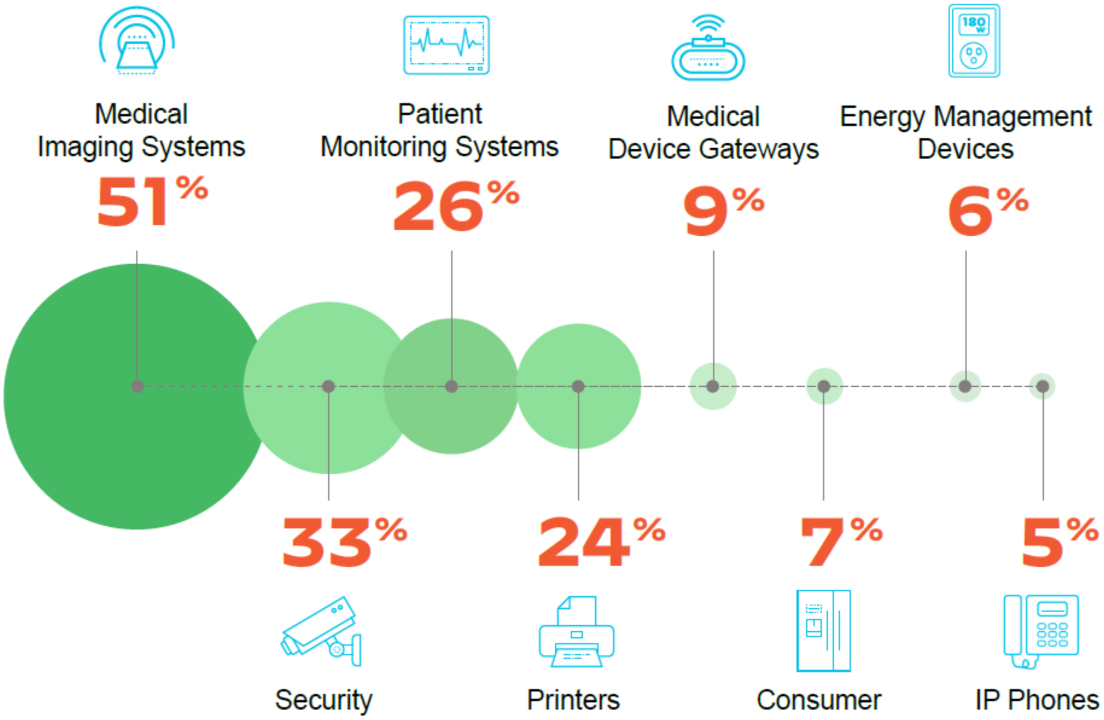


Figure 3.3 Devices with the highest share of security [source:36]

In a network, endpoints are the devices that are connected to the internet, and this includes IoT devices Many IoT device might provide endpoint that can be used by a bad actor to gain access to the network and expose it to external threats. As a result, IoT devices, like other endpoints, can be heavily weaponized. Infected with malware, IoT devices can be used as botnets to launch distributed denial-of-service (DDoS) attacks on the network the bad actor wishes to bring down. Nevertheless, unlike IT devices, IoT devices have such a diverse range of hardware platforms and protecting them all in the same way is difficult. It is impossible to find a single malware prevention agent that works with all IoT platforms.

In addition, firmware vulnerabilities constitute a convenient entry point for malicious software and make the embedded devices vulnerable to simple but devastating attacks. In reality, since 2009, numerous botnets that have exploited various firmware vulnerabilities have been discovered. Such botnets have compromised thousands and millions of online devices. Even worse, the affected embedded devices are hard to diagnose and clean (e.g., no embedded anti-virus solutions, no conventional input/output). Therefore, they often remain exploited for long periods of time. For example, the Carna [45] and Mirai [38] botnet which was used to produce the (in)famous "Internet Census 2012" was operational for more than one year. In addition to this, the rate of embedded devices expected to connect to the Internet is exponential and the speed at which attacks can spread across systems and networks is unimaginable.

Aside from that, weak system and network protection poses to makes IoT devices easy targets, and bad password security practices continue to create password-related attacks on IoT devices. Unpatched IoT devices are becoming more popular, and they operate on outdated operating systems. Exploits using techniques like network scanning, remote code execution, command injection, and others are among the most common attacks on these devices. IT-borne attacks search via network-connected devices in an effort to exploit known bugs, with 41% of attacks leveraging system vulnerabilities [35]. Following the compromise of the first device, lateral movement is enabled in order to gain access to other vulnerable devices and compromise them one at a time.

Peer-to-peer command-and-control (C&C) communication and self-propagating IoT malware worms are two new attack techniques arising on the Cybersecurity horizon, in addition to using some of these time-tested attack tactics thought obsolete by modern IT-based malware prevention. In fact, IoT worms are becoming more common than IoT botnets [35]. Both strategies aim to disrupt enterprise essential business processes by exploiting decades-old legacy OT protocols.

3.4.1. IoT Attack Surface

As part of its Internet of Things Project, the Open Web Application Security Project (OWASP) [37] has published a detailed draft list of IoT attack surface areas, or areas in IoT systems and applications where threats and vulnerabilities may exist. The IoT attack surface areas are summarized below.

Devices and Hardware's

Devices have the potential to be the primary means by which attacks are launched. In the hardware layer, we consider the partial or full control of system hardware and potential subversion or damage. In some cases, an adversary can reach the hardware directly, such that the protection policy may not be very effective. From this perspective, we divide the attack surface into two parts: the hardware (sensors, actuators), and the device physical interface.

On the hardware (sensors, actuators) attack surface, the adversary can use various means to compromise sensors and actuators, instantiating false data to confuse the control system administrator and the decision making process. The system may then carry out incorrect decisions and damage the hardware or components of the system. An example is the Stuxnet worm, which compromises the sensors of the nuclear power generation facility and compromises the control system to destroy the nuclear power plant [40].

On physical attack surface is based on the access interface between hardware and firmware. This attack surface has the following vulnerabilities: *(i) Removal of Storage Media*. This vulnerability is based upon connected portable storage media that could be physically removed, leading to disconnection or disfunction of services, applications, or the device itself, as well as the potential for compromised portable devices to infect the devices and systems. An example is the first version of Stuxnet, which used USB devices to compromise nuclear power plants. Adversaries could also use vulnerabilities in the hardware or firmware to steal important credentials from physically extracted removable storage devices [43] [44]. *(ii) Obtaining Console Access.* Obtaining console access is similar to the firmware and storage extraction vulnerability. Here, an adversary could use some method (like UART or JTAG) to gain full access to a serial interface. Usually, security measures (e.g. Custom bootloaders) are capable of preventing adversaries from going to single user mode, but these have occasionally been bypassed and are not completely secure [43], [44].

Communication channels

The network layer is where the IoT devices transfer data. This data transfer requires the support of various protocols, such as TCP/IP protocol. Based on the Open Systems Interconnection (OSI) network model, we consider attack surfaces to reside primarily in the network application layer. In the network application layer, there exist six attack surfaces. These attack surfaces are the device network service, device web interface, mobile application, cloud interface, privacy and network traffic.

The device network service attack surface is where all communication services are running. This attack surface cause a security issue if it is implemented with *(i) Unencrypted Service.* The unencrypted service vulnerability is one in which data is transferred in clear text, readable by anyone that can receive the data. In this case, an adversary can listen to the communication via Man-In-The-Middle (MITM) [47], attack. *(ii) Poorly Implemented Encryption.* It is a vulnerability in which the encryption implemented in a system is either poorly configured or the is out of date (Like the SSLv2/v3) and therefore ineffective. Just as in the prior vulnerability, the MITM attack can be used here as well. *(iii) Denial of Service.* It controls millions of Compromised devices to send requests to one victim simultaneously over some duration of time. An example is the Memcached Amplification Attack [39] that occurred on Feb 28, 2018, which is called an amplification attack because it exploits a disparity in bandwidth consumption between an adversary and the targeted web resources. This attack requires only a single query and can result in massive attack traffic. DDoS attacks on IoT devices have also demonstrated techniques for affecting IoT devices without damaging them.

Applications and software

Vulnerabilities in Firmware, web applications and related software for IoT devices can lead to compromised systems. we consider programs and applications designed for user and automated machine tasks, interfacing, and interaction. As a primary component of software systems, Application Programming Interfaces (APIs) specify protocols, tools, data structures and communication between multiple subcomponents and subroutines of complex software systems to obfuscate underlying operation and allow for easy interfacing between components. These APIs are a critical component in the software layer and the OS, dividing complex systems into small manageable parts, improving cohesion and reliability between units to improve the system's maintainability and extendibility.

Third-party backend APIs are used for application software running on operating systems, such as the Google Maps API. In this case, if the API is out of date, an adversary could gain unintended application access or data, such as the location in the Google Maps case. On this attack surface, the vulnerability is the insecure third-party components. Cases of third-party components being insecure include out-of-date or unpatched software, such as old versions of BusyBox, SSH, and so on. Old software has a higher likelihood of vulnerabilities and are typically easier to compromise [41].

Unlike third-party APIs, vendor APIs provide software interfaces to get access to hardware. For example, hardware maintenance software uses vendor backend APIs to get hardware information. If these APIs are modified or subverted, users will not receive correct information, such as the hardware data their application depends upon [42].

In distinguishing firmware from the operating system, the firmware can be the simplified startup software, bootloader, or bootstrap program that loads the operating system, and is directly installed on the hardware. As an example, BIOS is a firmware that is used daily, which checks the hardware at startup and loads the operating system. Thus, the firmware is the lowest level software that can directly take full control of the hardware. This attack surface has two kinds of vulnerabilities, the first is the same as one in the operating system layer, called the update mechanism, and the second is the device firmware attack surface.

This vulnerability is based on the directly on the firmware itself. It includes only a single vulnerability, called firmware and storage extraction. Through it, an adversary can use some methods to extract useful information, including the source code, the binary file of a running service, pre-set passwords, and SSH keys.

3.4.2. IoT Major Threats

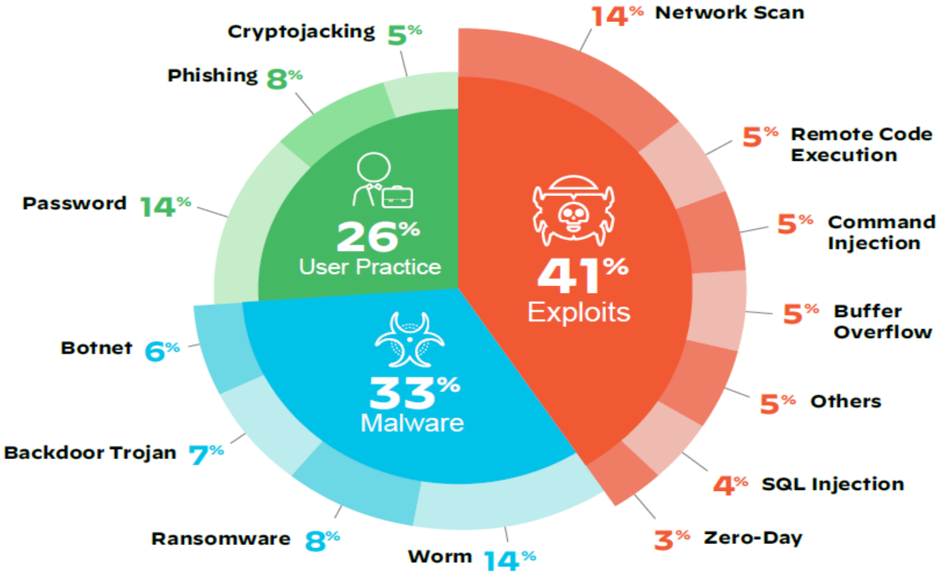


Figure 3.4 IoT Security cyber threats [source:36]

Botnets

Botnets combine many systems to take control of the victim's devices and systems remotely. From here, cybercriminals can harvest confidential data and execute cyber-attacks. These attacks target IoT devices in particular. For instance, the Mirai botnet has infected 2.5 million devices, includes smart cameras, routers, and printers [38]. And the situation is worsening. Cybercriminals produced even more sophisticated IoT botnets as a result of the success of these attacks.

Man-in-the-Middle Attack

Hackers steal messages by breaking into communication networks during Man-in-the-Middle attacks. They quickly seize control of communication and use it to send unauthorized messages. Man-in-the-Middle attacks are a threat to smart refrigerators, industrial equipment, and self-driving cars because IoT devices share data in real-time. Their dependence on this feature has the potential to be catastrophic.

Denial of Service

Through submitting a large number of requests, denial-of-service (DoS) attacks overwhelm networks. DoS attacks seldom steal sensitive information. However, it has the potential to cripple firms, damaging their competitiveness and credibility. Since IoT devices are easy targets, cybercriminals can target them as well. Flooding networks with requests causes them to become overburdened and go offline.

Data theft

We learn about data breaches on almost a daily basis. They put millions of people's data at risk. For the same cause, cybercriminals are now targeting IoT devices such as smartwatches and smart thermostats. It enables them to learn more about specific users and organizations. They will then gain access to company networks after targeting these unsecured computers. This can allow them to gain access to company systems and other resources. These attacks have the ability to spread like a virus. Customers' and employees' data can be harvested by cybercriminals in order to cause even more harm.

3.5. Current Challenges

There are now "things" that communicate with the Internet despite our involvement, in addition to us and our machines. These "things" are constantly interacting with the Internet, whether it's a refrigerator sending a notification on the food inside or our car sending messages to the mechanic about its oil levels. In many ways, the Internet of Things is fantastic. However, technology has not yet developed, and it is not absolutely secure. Numerous security issues of IoT remain for the entire IoT ecosystem, from vendors to users, to overcome, this challenges including:

3.5.1. Manufacturing standards

This is precisely one of the biggest security challenges with IoT. Manufacturers will continue to create products with weak protection because there are no uniform IoT security requirements. Manufacturers who have begun to provide Internet connectivity in their products do not always consider “security” to be a critical component in their product development process. Manufacturing companies had the following security threats in IoT devices:

1. Weak, guessable, or hard-coded passwords
2. Hardware issues
3. Lack of a secure update mechanism
4. Old and unpatched embedded operating systems and software
5. Insecure data transfer and storage

3.5.2. Update management

Insecure systems and firmware are another cause of IoT security risks. Despite the fact that a device can be sold with the most recent firmware update, security problems will almost certainly emerge. Updates are necessary for keeping IoT devices stable. When new bugs are found, they should be fixed as soon as possible. Despite this, some IoT devices are still being used without the required updates, as opposed to smartphones or computers that receive automatic updates. An intruder could capture sensitive data if the communication isn't encrypted and the update files aren't password-protected.

3.5.3. Physical hardening

IoT protection problems are also exacerbated by the lack of physically protecting IoT computers. Some IoT devices function by themselves without any human intervention, so they should have to be physically secured from physical threats. Sometimes, these IoT devices can be located in another different location for longer period of time, and the probability to physically tampered with an attacker is high. for instance, installing malware with a US flash drive. Basically, Ensuring the physical secure location of an IoT device begins from the user. And also, It is difficult task for manufacturers to build physically protected sensor actuators and transmitters in the already low-cost products. Another example for this smart motion sensor or a video camera that placed outside a house could be tampered or disrupted with if not adequately protected.

3.5.4. Users knowledge and awareness

Over the years, people and organizations have become aware that how to protect themselves from spam or phishing emails, some cyber threats, perform virus scans on their computers, and creating strong password for their Wi-Fi networks, online accounts. Internet of Things is a new and yet a growing technology, and many people still do not become aware of it. While vendors pose the majority of the risks associated with IoT security issues, users and business processes may pose greater challenges. The user's ignorance and lack of knowledge of the IoT functionality is one of the main IoT security threats and challenges. In the consequence everyone will be in risk.

3.6. Related Work

Since IoT is a growing technology currently many researches are performing to solve the current problems faced to IoT system. And many papers discussed the challenges when massive devices being connected to the future and with relative technologies development like 5G network, AI and others. Some papers also proposed sample prototype for those challenges described above.

The authors in [50] studied and shows that there are different challenges with in the IoT system, such as network jamming and spoofing attacks and unauthenticated/unauthorized access, which had been compromised the integrity of the user's data. And there are also shown that which could be some potentially different solutions which that the users to implement different security measures to secure their IoT devices. According to [51], different privacy and security threats have emerging every day, and they can exploit IoT systems and their integrated network. It is difficult to manage and control the security of IoT devices on different environment due to their architectures. For all IoT products, organizations can deploy monitoring and threat detection software that can detect any type of security or privacy threat and attempt to minimize the risk of being hacked. Network level communication interceptors and analyzers helps to identify and control the cyber threats.

There are different studies and researches have been conducted on the current design and technologies trend in IoT security [52]. Multiple researches have shown some of the current challenges and various attack surfaces to IoT devices and their mitigations. different simulation tools, prototypes, and the availability of multiple platforms confirms that the security challenges can also help in developing a new IoT security protocol. It is obvious that there is a progress have been made related to IoT security and privacy and multiple simulation tools as well as prototypes which supports this research area. If IoT devices have fails, the issues will be serious.

The authors in [53] shows that, even with the huge advantages the users and organization getting from the IoT, there are challenges of privacy and security that come along with and need to be looked at. And this risks are the primary concerns that have been described. These two pose a real threat to several business organizations as well as public organizations. High severity cybersecurity attacks have demonstrated the weaknesses of IoT technologies. That's because the connectedness of IoT devices enables access to the unknown and untrusted Internet, necessitating the development of new security solutions. On the other hand, it is important to emphasize the standards and basic principles of the IoT Cyber Security Framework when it comes to implementing the IoT security system. According to [54], one of the most important measures to consider is the termination of a contract consisting of different devices with different communication protocols. Deference in protocols obstructs the introduction of independent service contracts and are essential components of every Internet of Things' cybersecurity framework. They showed that some minor measures should be taken to help alleviate the complexities of IoT cybersecurity in order to ensure the stability of the IoT system in the cybersecurity arena. In addition, the authors in [54] showed that scalability is also an essential measure of the success of the cybersecurity Internet of Things framework. According to analysts, the Internet of Things ecosystem must be scalable enough to handle a billion Internet-related and cybersecurity problems. In addition, according to the magazine, the IoT cybersecurity ecosystem can promote testability, such as integration testing, component testing, device testing, and compliance testing, thus reducing challenges and risks. In the same context, the authors in [55] described some of the current IoT cybersecurity solutions. Some basic security measures are implemented by the supplier, and state that it is not profitable for the supplier to produce high-quality solutions. In the case of cybersecurity of the Internet of Things, companies are unlikely to develop the right solution.

Moreover, the authors in [56] describes the current IoT and cyber physical systems can be found everywhere, from critical infrastructure, modern vehicles to industrial control systems. From Current trends, such as Industry 4.0 and IoT [46], with strong communication and the successful use of new generations of IoT devices, they pledge creative business models and new user experiences. These Devices produce, process, and exchange huge amounts of data. Protection and confidential values that make cyber-attacks an enticing target for the framework of the Internet of Things cause physical damage and interrupt the lives of people. As they can pose a threat, cybersecurity and privacy are important. Because of the difficulty of these device and system integration, as well as the possible effects of cyber-attacks, similar industrial IoT systems are now facing new cyber threats. Possible solutions to security and privacy challenges are general IoT devices and systems security frameworks for industrial and every system. The current state of IoT systems and technologies are inadequate to protect the required functionality and reduce risk.

As a result, the research and analysis of various security problems in IoT has been extremely important. One of the main objectives in terms of IoT security is to provide privacy, confidentiality, and to ensure that every user can get better protection, infrastructures, and a guarantee to the availability of various services offered by the ecosystem of IoT. As a result, with the aid of various simulation software and different computing platforms, research in various IoT protection is gaining the requisite momentum.

The author [48] proposed a system that recovers IoT devices within a short period of time even if attackers took full control over the devices in a mass deployment. The recovery technique requires smaller amounts manual intervention. When the user or administrator have been identified or notified the compromise device. And the admin can able to deploy an updated firmware image, then instruct a system to forcibly reset and install the updated firmware on the devices They tried to demonstrate the universality and practicality of this system by deploying it on three different IoT devices ranging from high-end to low-end i.e. (HummingBoard Edge, Raspberry Pi Compute Module 3, and Nucleo-L476RG). According to their results, a system's output overhead is normally negligible.

They used different components to build this system a hardware latch which is responsible for two different states (*open*, *locked*) Its initial state is open and software can cause it to transit into the *locked* state and only a device reset will cause the latch to transition back into the *open* state. Each latch has an associated security function which is enabled if and only if the latch is in the *locked* state. They choose read-write latch that operate on persistent storage *A* read-write latch (RWLatch) that once applied, blocks any read or write access to one or more storage regions. Their system uses this to protect per-device secrets and WRLatches to protect its code against unauthorized modification or deletion. The requires that device reset and power-on provide a clean-state environment in which early boot code can execute deterministically, regardless of the actions of software that was running prior to the reset. they assumed this behavior for CPUs. However, if a CPU is embedded in a platform with additional active devices (e.g., devices that can bus-master or reset the main CPU), then these devices must also be reset when the main CPU is reset. The resets of latched devices must be tightly coupled to resets of the main CPU. In addition to this their system is based on device identifier composition engine (DICE) and a device must be equipped with a 256-bit unique per-device secret which early boot uses this code to enable untrusted code that may run subsequently to perform attestations. DICE code running during early boot reads secret and then latches it so that it becomes inaccessible to later software. DICE then uses a deterministic key-generation algorithm to create two asymmetric key pairs which is the DeviceID key pair and the Alias key pair. The DeviceID key pair is derived solely from the secret and remains the same for the life of the device. The Alias key pair is derived from the secret and the hash of the device firmware. Thus, the Alias key pair will change if the firmware is updated. DICE uses the DeviceID private key to certify the Alias public key and the hash of the device firmware. Before passing control to the firmware, DICE deletes the secret and the DeviceID private key from RAM and registers, but passes the Alias private key and the Alias public key certificate on to the firmware. The firmware can use these keys to make attestation claims to a server by signing a server-generated nonce.

One of their main is to enable the hub to unconditionally recover control of all managed devices even after a complete compromise of the device firmware: a property commonly called availability. With control recovered, the hub may subsequently issue firmware updates to patch the vulnerability or change the security settings that led to the exploit and evict the adversary from the device. It may further request evidence from the device that the updates have been applied correctly. And they used two dominance composition approaches which is Gated boot ensures that the device will boot firmware that is authorized by the hub at that time. If, no such firmware is on the device at boot time then it will first obtain and install an acceptable firmware version on the device before booting into it. And Reset Trigger After gated boot, the firmware has complete control over the device as the firmware was chosen by the hub, it can, in general, cooperate in performing regular maintenance tasks such as installing firmware updates requested by the hub. However, if the device is taken over by an attacker, this will not be the case. In order to prevent the attacker’s code from running on the device indefinitely, the hub needs a mechanism to force a device reset (i.e., a reset trigger). This will preempt the firmware and invoke gated boot, which can examine and update the firmware as requested by the hub.

However, it has several limitations If the reset trigger causes frequent and uncoordinated device resets, it may interrupt the device during a critical operation or cause in-memory state to be lost. Many IoT applications may not be able to tolerate this. Gated boot adds a network interaction with the hub to every boot. This adds a noticeable delay. The gated boot code includes a networking stack which, being large and exposed to attacks from the network, is a potential threat to the integrity of the system.

[57] show and proposed a prototype for secure firmware update using open standards and open libraries which is a building block for firmware's. They did research on different available open standards and protocols and open source libraries, because these are useful generic building components that can be used to enable IoT device firmware updates based on their specific modules. These standards and libraries categorized into the following categories Cryptographic Algorithms, Protocols for transferring updates in the network layer, Firmware Metadata, IoT device management protocols. To ensure the security of firmware updates, cryptographic schemes are needed. The IETF standardized a network stack incorporating Constrained Application Protocol (CoAP) over UDP [58] and CoAP over TCP/TLS [59] to transport firmware over various routes or over heterogeneous minimal power networks.Firmware Metadata provides information about the firmware required to update the device, and a security wrapper to protect the metadata end-to-end. And for Managing IoT Devices Lightweight Machine-to-Machine (LwM2M) protocol [64] designed by OMA to transfer data, LwM2M v1.1 uses CoAP, which can be secured with Datagram Transport Layer Security (DTLS) [65]. A simple data model and several RESTful interfaces for remote management of IoT devices are specified by the LwM2M specifications. The RESTful interfaces enables the IoT devices to provide information updates, communicating with a server, and obtain secret keys. A different number of resources, and objects have already been standardizing to support commonly used IoT device components sensors, actuators, and other resources. Among the standardized objects is the firmware update object.

On their prototype the security assessment result shows that If an attacker may try to update the IoT device with a modified and intentionally flawed firmware image their configurations use digital signatures to ensure integrity of both the firmware and its metadata. Additionally, the device can verify that an authorized maintainer signed the firmware image. An attacker may try to replay a valid, but old (known Tobe flawed) firmware. This threat is mitigated by using a sequence number. their configurations use a sequence number, which is increased with every new firmware update. To prevent firmware mismatch, attack their prototype use device specific conditions which can be verified before installing the firmware.

However, their prototype doesn’t prevent from offline attack if an attacker may cut communication between the IoT device and the update server for an extended period of time. Then, the attacker may try to update the IoT device with a (known-to-be-flawed) firmware image, not provide any mitigation against this threat. The firmware image in transmission can be captured by an attacker for vulnerability analysis. Their prototype doesn’t provide protection against eavesdropping end-to-end (from the maintainer to the IoT device) and doesn’t provide doesn't offer end-to-end security encryption without the extra protection offered by using ITEF Suit [58].

[60] [61] researches shows that a block chain based firmware update prototype. Always checking and keeping up to date of the firmware of IoT devices is one reliable way to protect and prevent the device from cyberattacks. Current firmware update approaches as well as distribution and validation are not ascendible in distribution with the rapid increasing numbers of IoT devices.

A block chain can be used as an open and distribution gateway to continuously distribute a block of data. [62] Each device acts as a normal node, while each device vendor, keeping firmware details, exists to serve as a verification node in its system. both nodes are block chain nodes and form a peer-to-peer directly interconnected block chain network. A normal node checks its own firmware is up to date or not by sending request to other nodes in the blockchain network and then receive information back. If the firmware version of the another node is different or greater than the requestor node, then it requests a detail information file with each node list from the verification node then it will download the latest firmware and update it. If the firmware version of both requestor node and requested normal node is the same, the requestor node checks that other nodes verify firmware integrity. This approach uses blockchain technology to allow for the download of new firmware for IoT devices, as well as the verification of firmware integrity. Their methods, on the other hand, exchange data between both nodes and at least six request messages among the verification nodes in order to achieve firmware version checking. When a large number of IoT devices need firmware upgrades at the same time, this creates a scalability problem.

[60] address this issue by leveraging the strength of smart contracts on a blockchain. A smart contract on a blockchain is a script or an agent that can be executed by its associated member devices when specified triggering conditions are met. Instead of letting end-device initiate

firmware validation, their system uses smart contracts instead to proactively validate all associated nodes for reducing the overhead of repeated message transmission or firmware validation.

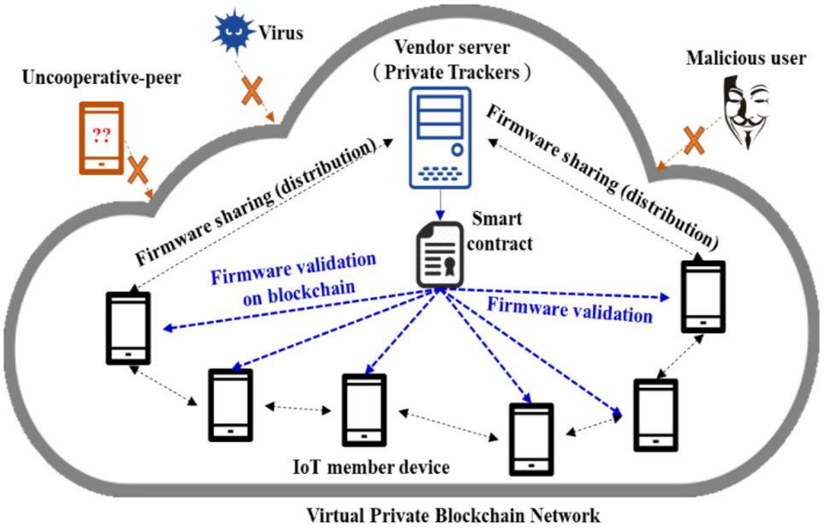


Figure 3.5 Blockchain based firmware update design [source:60]

However, spoofing attacks mainly lead a member device to an incorrect sharing group so that it fails to download the required file. In this study, such attacks were prevented using firmware verification to confirm the integrity of the downloaded firmware.

[63] shows security defect detection system for IoT firmware by measuring the similarities of sub module distance between the codes. As IoT device firmware is heterogeneous, not open source, and big business operation but small security implementation, therefore, there are limited resources and a high code reuse rate. Once the deployed firmware has security vulnerabilities, these vulnerabilities are difficult to find and discover, and the resulting effect spreads rapidly across a broad range.

Firmware code genes are extracted from known security defect sample codes, and then firmware security detection system (FSDS) [63] designed in this study is used to search for the existence of the same or similar codes in the firmware to be detected. A defect sample is the firmware code that contains the vulnerability function, malicious code, or backdoor core function itself and its associated context. FSDS has a good function search matching effect on the firmware binary code under different platforms, compilers, optimization options and matching directions and has high efficiency and robustness for IoT terminal firmware security detection. However, the research on firmware code genes is still in its infancy, related research is still incomplete, and some work still must be undertaken in the future.

3.7. Conclusion

From this research’s we understand that these IoT technology and systems are a growing technology and many researches also under development. This makes it challenging in terms of cyber security and data privacy issue due to heterogeneous environment and lack of international standards.

Keeping the firmware of an IoT device up to date is one feasible way to protect the device against cyberattacks. So different researches are performed in prototype level concerning this issue. Full firmware update using block chain algorithm and partial firmware building block update using open standard and libraries. This method gives hope in the future of IoT patching and updating mechanisms. Currently many IoT vendors do not releases update and patches frequently depending on vendors from two to four times a year. This gives an adversary time to do many damages data theft and more. Even if there is security defect on the devices and if not tracked or reported the vulnerabilities remains unknown and an attacker could also take advantage of it. And this is still the gap in the IoT system and many research’s also conducting in this area. Due to the nature of IoT system and limited capacity of this devices antivirus software’s integration is very difficult.

4. Practical Part

4.1. Firmware Gathering

Currently there are around 31 billion IoT devices connected all over the world. These devices are manufactured by hundreds of different vendors based on their requirement and specifications. And each vendor has different products which may not design with the same architecture and code base offcoures they may use some common known components from third party. Like the kernel webservers etc. In our analysis we use on of the popular platform for discovering Internet of Things connected all over the world. Which is shodan.

Figure 4.1 Shodan search engine [source:68]

This platform used to search anything (computers, servers, IoT devices) around the world. And identifies the basic services which is running on those devices and also suggests exploits if they have publicly available exploit.

For our firmware gathering and analysis basically we will focus on the most popular IoT device vendors and we will narrow it down specific product, device version and firmware version. To gather our firmware first we should have to have some criteria's to identify specific device and firmware. Our criteria would the following to identify specific versions and to limit our scope only to analyze only on selected device firmware's.

1. Firmware and Device Vendor Identification
2. Publicly Available Firmware
3. Larger Number of Device and Firmware
4. Unencrypted Firmware

4.1.1. Firmware and Device Vendor Identification

Thousands of different IoT Device vendors are currently manufacturing different device these devices are ranging from personal wearable, health monitoring devices, telecommunication devices and many IT appliances. As described in Literature review, they all have basically the same architecture and may have different file system based on the vendor or functional requirement of the devices. For example, this are well known popular vendors Tp-Link, D-Link, Netgear, Axis, Cisco, Reolink, Hikvision, Panasonic, Sony etc.

4.1.2. Publicly Available Firmware

This one of the criteria that we considered, there are some devices and firmware's which doesn't have their firmware publicly specifically the one that are released in the latest years and also depends on their functional requirements. Even if they have same kind of architecture based on their requirement, they use different kind of updating mechanism.

4.1.3. Unencrypted Firmware

Some latest firmware's after they have been build, they will encrypted. This is one way of preventing firmware reverse engineering techniques. Mostly encrypted firmware's released publicly in the latest time and the reason behind is many attackers extracting the source code and analyze to take advantage of the vulnerability they found on the analysis. Since we are performing the security analysis of the firmware this should be one the requirement to extract the file system and analyze also to show the vulnerabilities found on the firmware. To check the firmware is whether encrypted or not we will calculate the entropy of the firmware binary. It is used to calculate the randomness of obfuscated file and in our case the value will be between 0 and 1. When the value is getting closer to one, we can say that the file is encrypted. When it is lower the filesystem is in the normal format, we can easily extract the file system.

4.1.4. Larger Number of Device and Firmware

This is one of the important criteria to show the number of cyber threats on the current live available connected device when the numbers are higher the number of risks caused by those threat also increases. The below table shows the number of IoT Devices currently available for the most popular IoT Device vendors.

Device Vendors	**No. active on Shodan**	**Search URL on Shodan**
Net Gear	658703	https://www.shodan.io/search?query=NetGear
TP-LINK	1483376	https://www.shodan.io/search?query=tp+link
D-LINK	243714	https://www.shodan.io/search?query=D-Link
Cisco	4987278	https://www.shodan.io/search?query=cisco

Axis	13575	https://www.shodan.io/search?query=axis
Reolink	19863	https://www.shodan.io/search?query=reolink

Table 4.4.1 Famous IoT device and search engine result [source: own]

As we seen from the above table the search result is based on their vendor brand name for IoT devices. Off course their product type and models may have differences. And the next step is to determine which of their product or devices and firmware is largely available for specific version. To determine it is required to check manually each product model or device type and firmware build version.

4.2. Firmware Selection

As we seen from the above our firmware gathering criteria millions of devices are there from our primary firmware gathering platform Shodan. In this stage we will narrow it down to specific device hardware version and firmware version which should also fulfils the above criteria.

In addition to the above criteria's we will use some minor criteria to select firmware's. some firmware's are packed in very complicated way without encryption. For example, CISCO device firmware's, this firmware's off course can be unpacked but I would take significant amounts of time to inspect manually the binary file, studying the file system structure and extracting the file. Due to the time constraint and since they have same kind of architecture with other devices, we will drop this firmware and mainly focuses on easily unpackable firmware's.

To narrow down the device and the firmware's its required to check manually to go through all vendors products model and firmware version. Yes, this manual checking took significant amounts of time to find the specific device and firmware version.

After searching on Shodan based on the above criteria's we come with the following four specific device and firmware's for analysis. Most of the firmware's from same manufacturer even have different firmware versions and device model type the probability to use same code base for each model and version is high. Therefore our analysis and result this device and firmware's are more than enough to show the risks and cyber threat faced in the IoT environment. With other constraints in mind.

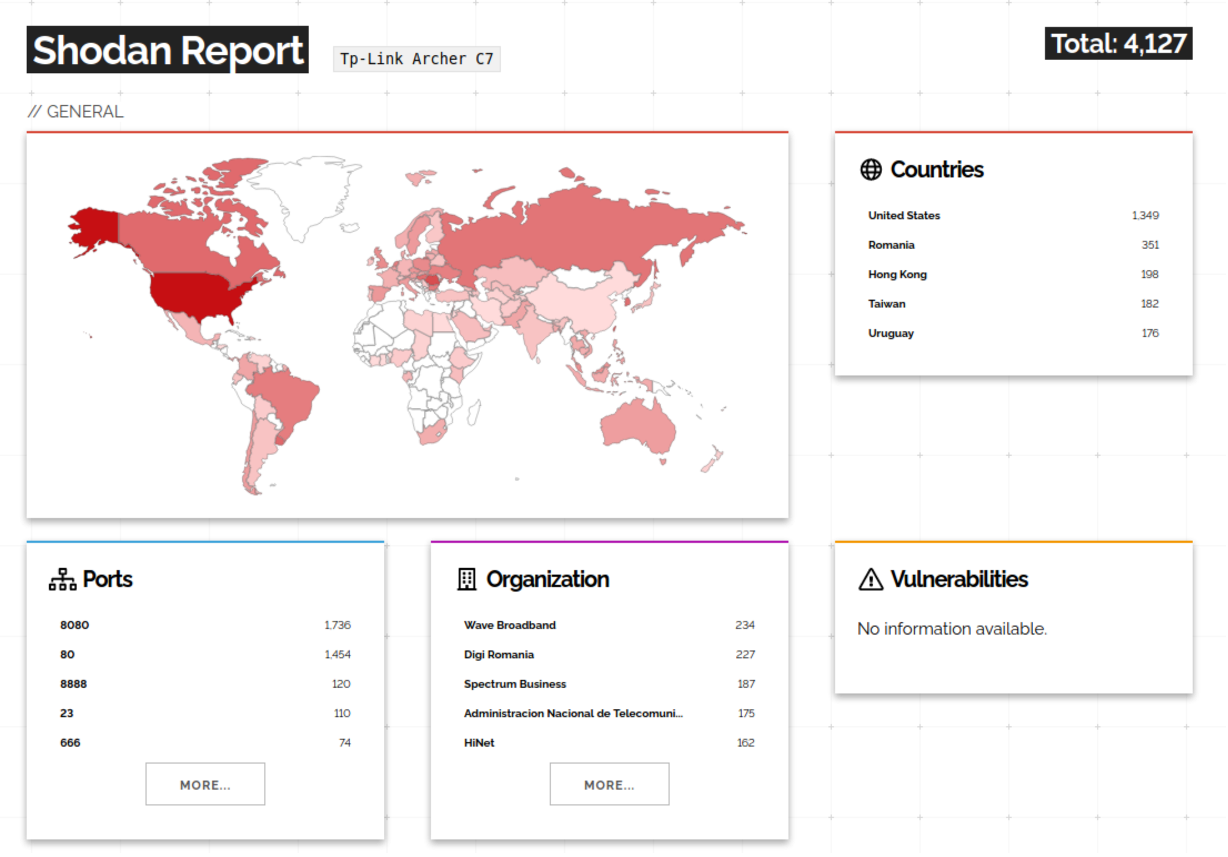
Device: **TP-LINK Archer C7 v1 0.0.3**

Architecture: **Mipsel**

NO. in Shodan: **4124**

Binary File Size: **7MB**

Firmware Source: https://www.tp-link.com/us/support/download/archer-c7/v1/#Firmware



Shodan Report Tp-Link Archer C7

Total: 4,127

// GENERAL

Countries

United States	1,349
Romania	351
Hong Kong	198
Taiwan	182
Uruguay	176

Ports

8080	1,736
80	1,454
8888	120
23	110
666	74

Organization

Wave Broadband	234
Digi Romania	227
Spectrum Business	187
Administracion Nacional de Telecomuni...	175
HiNet	162

Vulnerabilities

No information available.

Report Shodan https://beta.shodan.io/search/report?query=Tp-Link+Archer+C7

Table 4.1.2 Tp-Link Archer C7 search engine result [source: own]

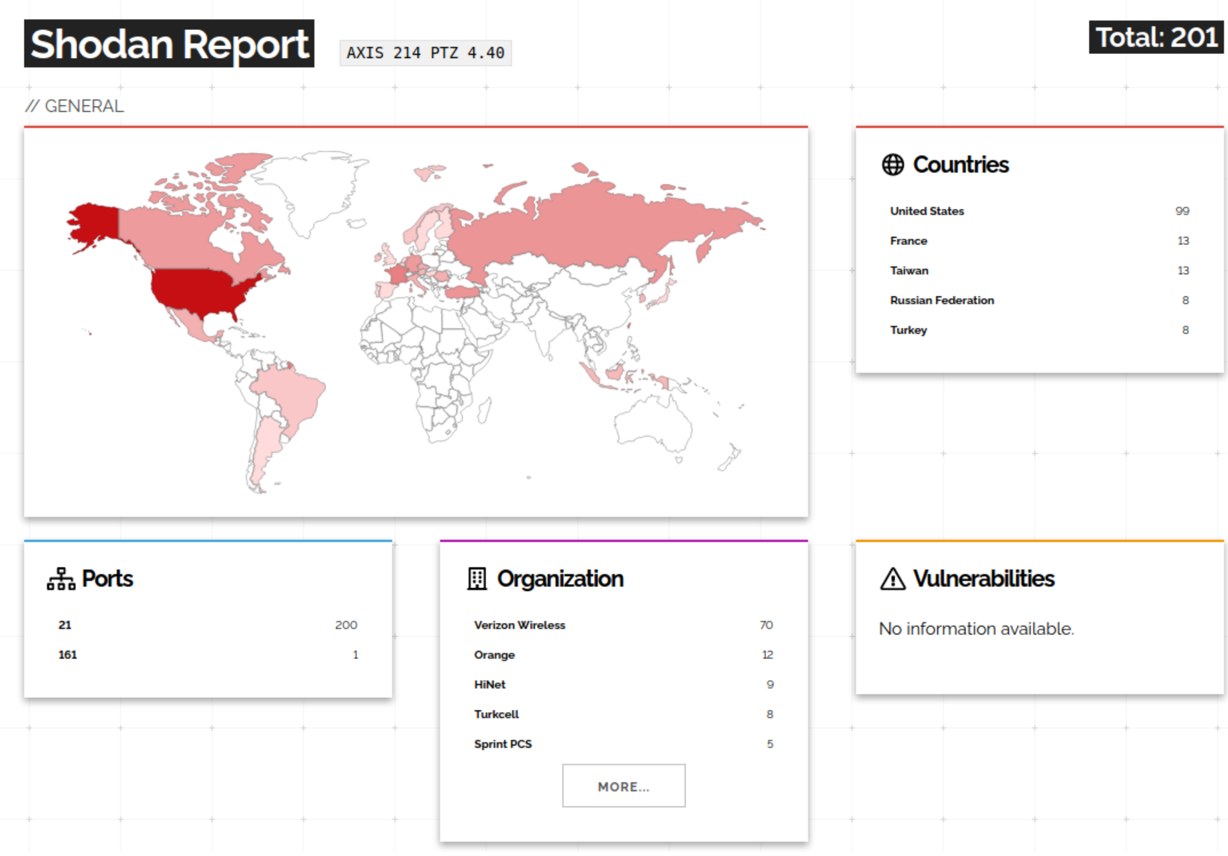
Device: **AXIS 214 PLZ v4.40**

Architecture: **Mipsel**

NO. in Shodan: **201**

Binary File Size: **9MB**

Firmware Source:
https://www.axis.com-/ftp/pub_soft/cam_srv/cam_214/4_40/



Shodan Report AXIS 214 PTZ 4.40

Total: 201

// GENERAL

Countries

United States	99
France	13
Taiwan	13
Russian Federation	8
Turkey	8

Ports

21	200
161	1

Organization

Verizon Wireless	70
Orange	12
HiNet	9
Turkcell	8
Sprint PCS	5

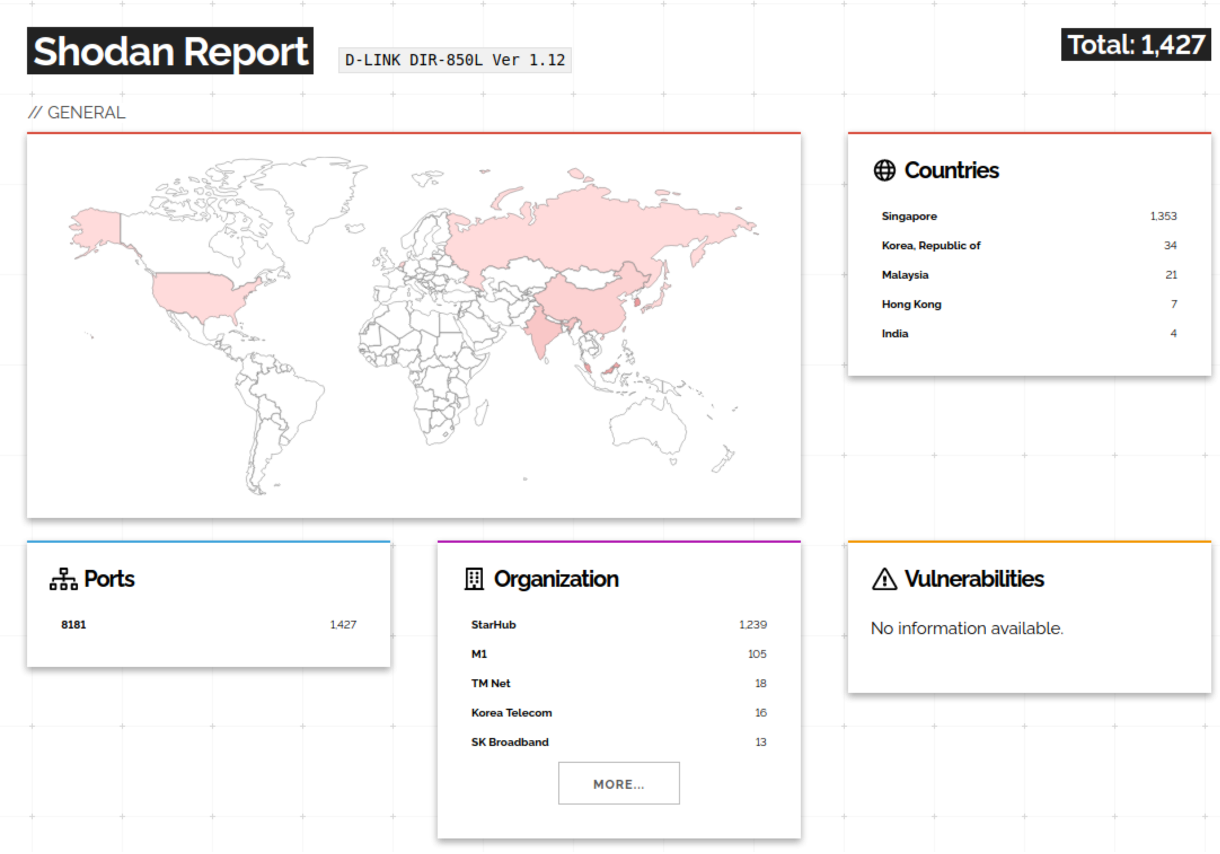
Vulnerabilities

No information available.

Report Shodan https://beta.shodan.io/search/report?query=AXIS+214+PTZ+4.40

Table 4.1.3 Axis 214 PLZ search engine result [source: own]

	Device: **D-LINK DIR-850L v1.12w** Architecture: **Mipsel** NO. in Shodan: **1427** Binary File Size: **10MB** Firmware Source: https://ftp.dlink.ru/pub/Router/DIR-850L/Firmware/



Shodan Report

D-LINK DIR-850L Ver 1.12

Total: 1,427

// GENERAL

Countries

Singapore	1,353
Korea, Republic of	34
Malaysia	21
Hong Kong	7
India	4

Ports

8181	1,427

Organization

StarHub	1,239
M1	105
TM Net	18
Korea Telecom	16
SK Broadband	13

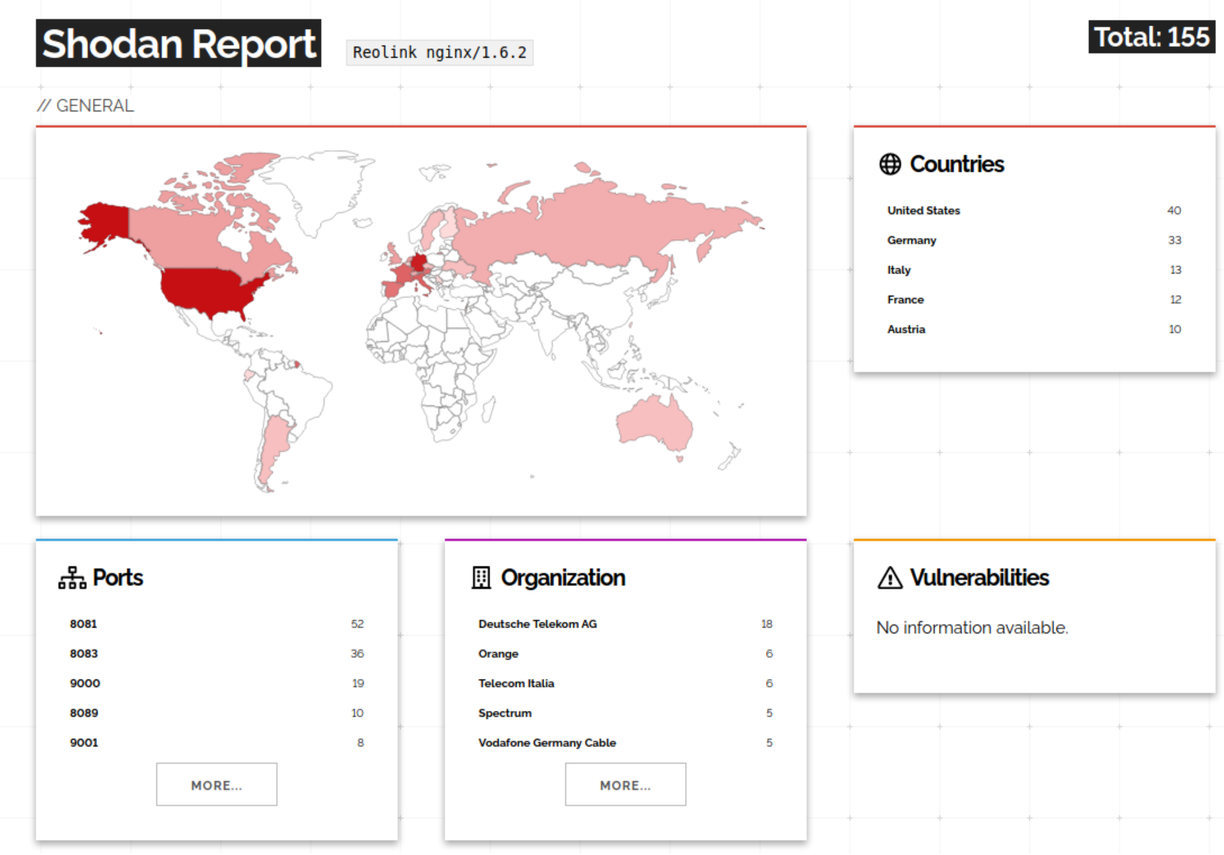
Vulnerabilities

No information available.

Report Shodan https://beta.shodan.io/search/report?query=D-LINK+DIR-850L+Ver+1.12

Table 4.1.4 D-Link DIR-850L search engine result [source: own]

	Device: **REOLINK RLC-410 v1.0.242** Architecture: **Mipsel** NO. in Shodan: **155** Binary File Size: **18MB** Firmware Source: https://s3.amazonaws.com/reolink-storage/website/firmware/20190321firmware/RLC-410_1441_19032101.zip



Shodan Report Reolink nginx/1.6.2 Total: 155

// GENERAL

Countries

United States	40
Germany	33
Italy	13
France	12
Austria	10

Ports

8081	52
8083	36
9000	19
8089	10
9001	8

MORE...

Organization

Deutsche Telekom AG	18
Orange	6
Telecom Italia	6
Spectrum	5
Vodafone Germany Cable	5

MORE...

Vulnerabilities

No information available.

Report Shodan https://beta.shodan.io/search/report?query=Reolink+nginx%2F1.6.2

Table 4.1.5 Reolink RLC 410 search engine result [source: own]

4.3. Firmware Analysis

In this stage we will perform detail security analysis the above selected firmware and summarize the analysis security vulnerability findings. This include publicly available exploit for the device itself from https://www.cvedetails.com/ and other source.

4.3.1. Enumeration

In this stage we will perform general information gathering about the firmware's from the binary file this includes

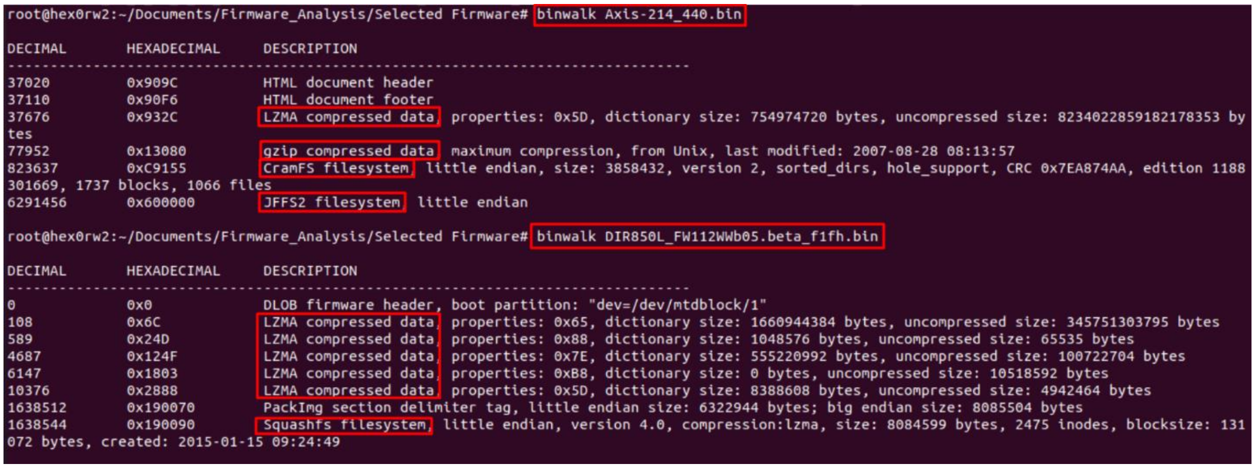
- The firmware build date: this reveals that when the firmware is build and if the firmware is new or old deploy. Most of the time the firmware's which have built long years ago probably have public CVE (Common Vulnerability Exposure) or exploits that can an attacker easily get the proof of concept and by using against the device may compromise it based on the severity of the vulnerability.
- The kernel Version: this reveals the device file system and architecture and also from the kernel version we could find public exploit but this depends on two conditions, first mostly to take advantage on the kernel the attacker first should compromise the device and the second one is if the kernel version is older or newer the probability to get public exploits varies accordingly newer have low or none and the older one may contain a lot of vulnerabilities.
- File System type: this reveals what kinds of file system the device contains for example mostly the file system in IoT devices are squashfs, cramfs, jffs2. Therefore, we can decompress the file system.
- The firmware compression method: this reveals what kinds of compression method the firmware's uses for compressing files. If we identify the compression method on this stage, we can find easily the decompression algorithm to extract the file system.
- Entropy of the firmware: this will help us to identify if the firmware is encrypted or not the entropy value of firmware usually from our analysis tool gives the range between 0 and 1 when the number is getting closer to 1 we can conclude that the firmware is encrypted.

The tool that we use primarily for enumeration is binwalk [66]. this tool is super powerful tool which is used to analyze extract and perform entropy analysis. And supports different filesystem and compression algorithms. The commands and the enumeration result screenshot presented as follows.

First, we have to install all the packages and dependencies from the source. The following command are as an example for a single firmware and the same goes for other firmware's.

Basic firmware Information Gathering	`binwalk Axis-214_440.bin`
Entropy Analysis	`binwalk -E Axis-214_440.bin`

Table 4.1.6 Binwalk enumeration command [source: own]



```
root@hex0rw2:~/Documents/Firmware_Analysis/Selected Firmware# binwalk Axis-214_440.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
37020         0x909C          HTML document header
37110         0x90F6          HTML document footer
37676         0x932C          LZMA compressed data, properties: 0x5D, dictionary size: 754974720 bytes, uncompressed size: 8234022859182178353 by
tes
77952         0x13080         gzip compressed data, maximum compression, from Unix, last modified: 2007-08-28 08:13:57
823637        0xC9155         CramFS filesystem, little endian, size: 3858432, version 2, sorted_dirs, hole_support, CRC 0x7EA874AA, edition 1188
301669, 1737 blocks, 1066 files
6291456       0x600000        JFFS2 filesystem, little endian

root@hex0rw2:~/Documents/Firmware_Analysis/Selected Firmware# binwalk DIR850L_FW112WWb05.beta_f1fh.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             DLOB firmware header, boot partition: "dev=/dev/mtdblock/1"
108           0x6C            LZMA compressed data, properties: 0x65, dictionary size: 1660944384 bytes, uncompressed size: 345751303795 bytes
589           0x24D           LZMA compressed data, properties: 0x88, dictionary size: 1048576 bytes, uncompressed size: 65535 bytes
4687          0x124F          LZMA compressed data, properties: 0x7E, dictionary size: 555220992 bytes, uncompressed size: 100722704 bytes
6147          0x1803          LZMA compressed data, properties: 0xB8, dictionary size: 0 bytes, uncompressed size: 10518592 bytes
10376         0x2888          LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes, uncompressed size: 4942464 bytes
1638512       0x190070        PackImg section delimiter tag, little endian size: 6322944 bytes; big endian size: 8085504 bytes
1638544       0x190090        Squashfs filesystem, little endian, version 4.0, compression:lzma, size: 8084599 bytes, 2475 inodes, blocksize: 131
072 bytes, created: 2015-01-15 09:24:49
```

Figure 4.2 Binwalk firmware enumeration [source: own]

The information gathering for the selected device summarized as follows.

Device	**Enum Result**
TP-LINK Archer C7 v1 0.0.3	Architecture: Mipsel Filesystem: SquashFS Compression: LZMA Entropy: 0.067946 Date: 2014-12-04 Kernel: Linux 2.6.31
AXIS 214 PLZ v4.40	Architecture: Mipsel Filesystem: CramFS, JFFS2 Compression: LZMA Entropy: 0.310936

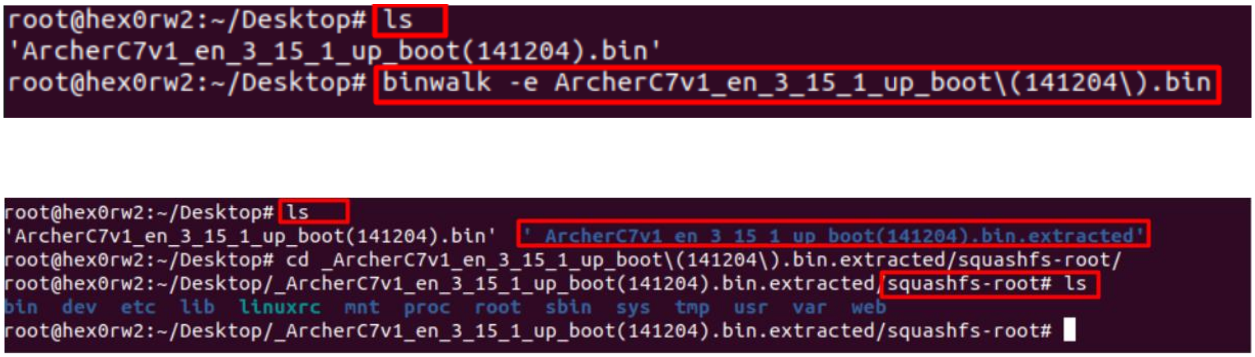
	Date: 2007-08-28 Kernel: Linux 2.6.18
D-LINK DIR-850L v1.12w	Architecture: Mipsel Filesystem: SquashFS Compression: LZMA Entropy: 0.292844 Date: 2015-01-15 Kernel: Linux
REOLINK RLC-410 v1.0.242	Architecture: Mipsel Filesystem: SquashFS Compression: LZMA Entropy: 0.344397 Date: 2019-06-14 Kernel: Linux 4.1.0

Table 4.1.7 Summary of firmware enumeration [source: own]

4.3.2. Firmware Unpacking

Now we have all information about the firmware the in this stage we will extract the file system from the firmware for the analysis by using the binwalk firmware analysis toolkit. The command to extract the firmware and sample extraction step presented as follows. Binwalk automatically detects all the file structure in the firmware and extracts them with proper decompression algorithms.

Extraction **`binwalk -e ArcherC7v1_en_3_15_1_up_boot(141204).bin`**



```
root@hex0rw2:~/Desktop# ls
'ArcherC7v1_en_3_15_1_up_boot(141204).bin'
root@hex0rw2:~/Desktop# binwalk -e ArcherC7v1_en_3_15_1_up_boot\(141204\).bin
```

```
root@hex0rw2:~/Desktop# ls
'ArcherC7v1_en_3_15_1_up_boot(141204).bin'  '_ArcherC7v1_en_3_15_1_up_boot(141204).bin.extracted'
root@hex0rw2:~/Desktop# cd _ArcherC7v1_en_3_15_1_up_boot\(141204\).bin.extracted/squashfs-root/
root@hex0rw2:~/Desktop/_ArcherC7v1_en_3_15_1_up_boot(141204).bin.extracted/squashfs-root# ls
bin  dev  etc  lib  linuxrc  mnt  proc  root  sbin  sys  tmp  usr  var  web
root@hex0rw2:~/Desktop/_ArcherC7v1_en_3_15_1_up_boot(141204).bin.extracted/squashfs-root# 
```

Figure 4.3 Binwalk firmware extraction [source: own]

As we seen from this file system extraction stage finally, we are able to extract the squash file system its Linux like file system structure. Now we are ready to analyze the file system for security issues.

4.3.3. Static Application Security Testing

At this stage we have all required file system for analysis of all selected firmware's. in the file system there are different components i.e. third-party software's, configuration files, web applications etc. In this section we mainly focus on configuration and sensitive information's and web application analysis.

4.3.3.1. Manual Testing

In manual testing mainly we will focus of searching vulnerabilities, service misconfiguration and other sensitive information this could be hardcoded password, API keys, private certificate keys.

To start testing first we run firmwalker [67] toolkit to identify the file that containing the above sensitive information. This tool based on common static configuration searches for patterns and sensitive file in the file system. Then from manual inspection of the files the following results found.

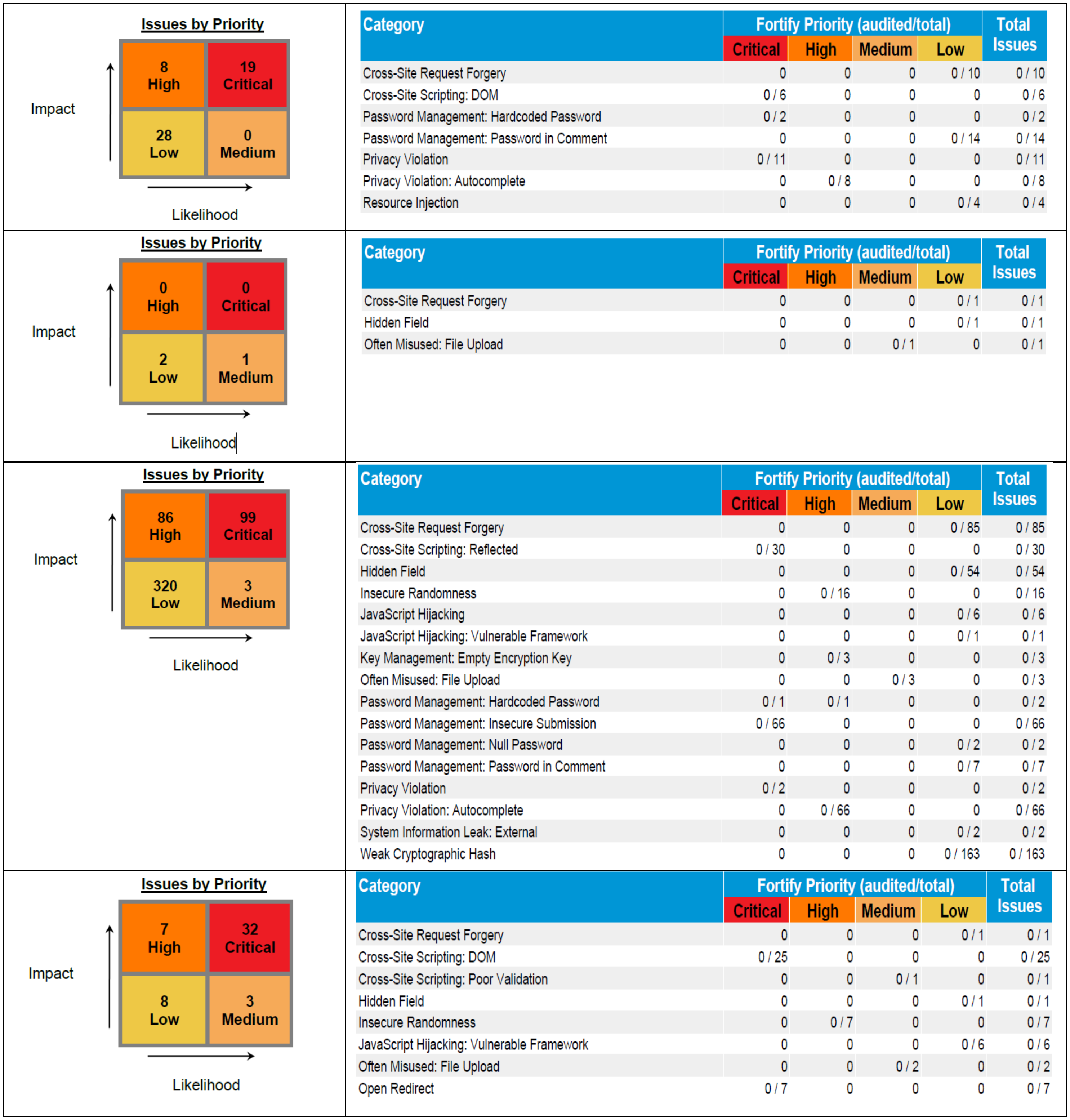
Device	Password/hash	API keys	Certificate keys
TP-LINK Archer C7	1	0	0
AXIS 214 PLZ	2	0	0
D-LINK DIR-850L	9	2	1
REOLINK RLC-410	6	1	2

Table 4.1.8 Summary of manual analysis [source: own]

4.3.3.2. Automated Testing

For automated testing we will use Fortify Static source code analyzer. For this automated testing we mainly focus on web application services source code that we extracted from the firmware. Fortify code analyzer mainly supports almost all web programing and scripting languages.

The table below shows a summary of all issues, which are organized vertically by Fortify Category. Fortify Priority Order displays the total number of issues for each category, as well as information about the number of audited issues.



Issues by Priority

Impact	8 High	19 Critical
	28 Low	0 Medium
	Likelihood	

Category	Fortify Priority (audited/total)				Total Issues
	Critical	High	Medium	Low	
Cross-Site Request Forgery	0	0	0	0 / 10	0 / 10
Cross-Site Scripting: DOM	0 / 6	0	0	0	0 / 6
Password Management: Hardcoded Password	0 / 2	0	0	0	0 / 2
Password Management: Password in Comment	0	0	0	0 / 14	0 / 14
Privacy Violation	0 / 11	0	0	0	0 / 11
Privacy Violation: Autocomplete	0	0 / 8	0	0	0 / 8
Resource Injection	0	0	0	0 / 4	0 / 4

Issues by Priority

Impact	0 High	0 Critical
	2 Low	1 Medium
	Likelihood	

Category	Fortify Priority (audited/total)				Total Issues
	Critical	High	Medium	Low	
Cross-Site Request Forgery	0	0	0	0 / 1	0 / 1
Hidden Field	0	0	0	0 / 1	0 / 1
Often Misused: File Upload	0	0	0 / 1	0	0 / 1

Issues by Priority

Impact	86 High	99 Critical
	320 Low	3 Medium
	Likelihood	

Category	Fortify Priority (audited/total)				Total Issues
	Critical	High	Medium	Low	
Cross-Site Request Forgery	0	0	0	0 / 85	0 / 85
Cross-Site Scripting: Reflected	0 / 30	0	0	0	0 / 30
Hidden Field	0	0	0	0 / 54	0 / 54
Insecure Randomness	0	0 / 16	0	0	0 / 16
JavaScript Hijacking	0	0	0	0 / 6	0 / 6
JavaScript Hijacking: Vulnerable Framework	0	0	0	0 / 1	0 / 1
Key Management: Empty Encryption Key	0	0 / 3	0	0	0 / 3
Often Misused: File Upload	0	0	0 / 3	0	0 / 3
Password Management: Hardcoded Password	0 / 1	0 / 1	0	0	0 / 2
Password Management: Insecure Submission	0 / 66	0	0	0	0 / 66
Password Management: Null Password	0	0	0	0 / 2	0 / 2
Password Management: Password in Comment	0	0	0	0 / 7	0 / 7
Privacy Violation	0 / 2	0	0	0	0 / 2
Privacy Violation: Autocomplete	0	0 / 66	0	0	0 / 66
System Information Leak: External	0	0	0	0 / 2	0 / 2
Weak Cryptographic Hash	0	0	0	0 / 163	0 / 163

Issues by Priority

Impact	7 High	32 Critical
	8 Low	3 Medium
	Likelihood	

Category	Fortify Priority (audited/total)				Total Issues
	Critical	High	Medium	Low	
Cross-Site Request Forgery	0	0	0	0 / 1	0 / 1
Cross-Site Scripting: DOM	0 / 25	0	0	0	0 / 25
Cross-Site Scripting: Poor Validation	0	0	0 / 1	0	0 / 1
Hidden Field	0	0	0	0 / 1	0 / 1
Insecure Randomness	0	0 / 7	0	0	0 / 7
JavaScript Hijacking: Vulnerable Framework	0	0	0	0 / 6	0 / 6
Often Misused: File Upload	0	0	0 / 2	0	0 / 2
Open Redirect	0 / 7	0	0	0	0 / 7

Table 4.1.9 Summary of automated analysis [source: own]

4.3.4. Dynamic Application security testing

In this stage we will emulate the firmware binaries partially. Since we have all the source code and file system performing DAST is not that much relevant. Because DAST approach is without having any information about the device testing through the running services. In this case we will use this approach for emulating a single binary file to gather information, version and looking for publicly available exploits. The following command shows the emulation process using QEMU. QEMU is a free and open-source emulator and virtualizer that can perform hardware virtualization.

- The first step is downloading QEMU image for each device architecture in our case all firmware's are MIPSEL (Mips architecture with little Indian) then runthe image with the following command

  -
    ```
    qemu-system-mips -M malta -kernel vmlinux-3.2.0-4-4kc-malta
    -hda debian_wheezy_mips_standard.qcow2 -append
    "root=/dev/sda1" -nographic -net user,hostfwd=tcp::9001-:22
    -net nic
    ```
- Next copying the extracted firmware file system to the machine with SCP in the above command we have port forwarded the ssh service on port 9001 the password will be root.

  -
    ```
    scp -P 9001 ./squashfs-root.tar.gz root@127.0.0.1:/root
    ```
- Then extracting the tarball on the running VM

  -
    ```
    tar zxf squashfs-root.tar.gz
    ```
- Most of the libraries and dependencies of the extracted file system is dynamically linked. The easy way to force the linker to use the correct libraries is to start an instance of the build in shell inside chroot using firmware filesystem.

  -
    ```
    cd squashfs-root/
    ```

  -
    ```
    chroot ../bin/sh
    ```
- Now we can run the executable files in the firmware for analysis.

From DAST examining all executable components from the extracted firmware and from publicly available CVE for each firmware model the total number of vulnerabilities presented as follows for each device.

Device	**Total CVE (all components and devices)**
TP-LINK Archer C7	376
AXIS 214 PLZ	566
D-LINK DIR-850L	19
REOLINK RLC-410	178

Table 4.1.10 Summary of DAST [source: own]

4.3.5. Summary of Finding

The summary table shows from all analysis result the findings of summary based on OWASP (Open Web Application Security Project) Internet of Things framework [37]. This framework basically defines all the security issues in IoT device in ten categories. Based on our analysis approach from those top ten security issues we have selected 7 categories as described below.

Based on the impact and the severity of each security issues found from analysis and described on each OWASP categories has different risks. Some of them can be exploited remotely by using publicly available exploit. Some of the security issues can be used together to compromise the devices. Some of the finding are not exploitable and an attacker may need to have local access these kinds of security finding categorized under Use of Insecure or Outdated Components.

Some of the security issues are informative which means if an attacker can learn from the information about the device the running services and other components in the firmware and he can craft, or search publicly exploit to take advantage.

And Finally, from our analysis the following table shows the number of security issues finding on those analyzed firmware's. the SAST section is based on OWASP category. And the DAST section is based on running executable components and by checking the version of those components and also by searching publicly available CVE.

OWASP Category	**TP-LINK Archer C7**	**D-LINK DIR-850L**	**REOLINK RLC-410**	**AXIS 214-PLZ**
SAST				
Weak Guessable, or Hard Coded Passwords	16	9	6	2
Insecure Network Services	2	2	1	1
Insecure Ecosystem Interfaces	20	303	42	2
Use of Insecure or Outdated Components	8	5	7	10
Insufficient Privacy Protection	19	68	1	11
Insecure Data Transfer and Storage	2	2	1	1
Insecure Default Settings	2	2	2	2
DAST				
Publicly Available Exploit	1	5	3	0
Firmware Components	375	14	175	566
TOTAL	**445**	**410**	**238**	**595**

Table 4.1.11 Summary of all analysis finding [source: own]

4.4. IoT Cyber Thereat Detection Module

4.4.1. Description of the Module

The proposed IoT Cyber Threat Detection Module (ICTDM) is a simple antivirus or antimalware like module or plugin for IoT devices which basically used to monitor the device state, monitor the network communication for intrusion and other unknown requests and periodically monitor the running processes in the IoT device to check whether the device is infected with malware or not. Since the behavior of the IoT devices is super minimal in terms of processing speed or computational power and limited memory and also other aspects this IoT cyber threat detection module designed to fit all the minimal requirement of the IoT Devices.

Basically this module designed to integrate or build together with the firmware binary file image. Currently there are different designs and prototypes to monitor IoT devices and IT system to monitor and detect the intrusions and different attacks by adversary. But they work with their dedicated machine or instance on the same network without integration with the device, this implementation has its own pros and cons.

One of the advantages of implementing IDS in the same network is it will allow us to monitor large number of devices in the enterprise organizations and give easily management interface for only network Intrusions. But sometimes this detections mechanism can be bypassed with different methods by tricking the IDS the communications as a normal protocol. In such scenarios the IDS will not protect others device behaviors. For example, if it is infected with malware and checking the devices is functioning properly. And also if we check with the normal users of the IoT devices for personal or home purpose, almost all of them doesn’t implement such kinds of detection and protection mechanisms. So the probability of getting hacked or infected by an attacker is very high. In order to fulfil this gaps, it is necessary to design a plugin or modules that would be integrated or build with the device itself for better cyber threat detection.

The Proposed ICTDM addressed this issues by implementing those additional features with the device requirements. For each functional requirement and subcomponent, it will have its own device specific pre-defined configuration and rules to monitor analyze and detect malicious activities. This static configuration will help the device and the plugin to function with minimal resources. Because the IoT devices are resource limited operation specific devices. In addition to this the only activity that could take a little bit storage on the detection phase is the logging of unknown or malicious activities then in the same time the alert or the notification of that activity will be transferred to the vendors or the users based on its configuration using REST API interfaces.

4.4.2. Functional Requirement

The proposed ICTDM module provides the following functionalities

- **Device State Monitoring**: this function checks whether the device required services are running or not. If one of those required services are not running the plugin detects, logs and notify to the user that he devices is not functioning properly.
- **Network Monitoring**: this function monitors the incoming and outgoing traffics from the device. In addition, it inspects the bandwidth and the packets that are transmitted based on the pre-defined device specific configuration.
- **Process Monitoring**: this function monitors in the device whether unknown process is running or not. If unknown process found running on the device, it will check the properties by sending its hash sum to the third party API to check the file is malware or not and in both cases it will log and notifies the user.
- **REST API**: this function is mainly responsible for sending and receiving the activities and anomaly behaviors detected by ICTDM to the vendors or the users based on configuration.

4.4.3. ICTDM Stages

ICTDM have basically three different stages

1. Monitoring Stage
2. Analysis Stage
3. Detection Stage

Monitoring Stage

In this stage the ICTDM in real-time and periodically checks the device state, network traffic and running process in the device based on the configuration. Once some of the required process not running, unknown behavior and unknown processes or network traffic detected it will send it to the analysis stage.

Analysis Stage

On this stage the detected behaviors, unknown processes and network traffic will be inspected in detail and matched with the pre-defined configuration file. And will be gathered additional information from the detected activity for logging. On the unknown process analysis phase the hash sum value will be generated sends to third party platform to check if the file is known as malware or not. Then send all the detail information to the detection phase.

Detection Stage

In this stage all information received from the analysis phase will be logged and sends notification to the user or vendor.

4.4.4. Implementation requirements

The basic concept of ICTDM is to provide all required functionalities with minimal resource with all IoT device specification. With this in mind the computational power and other resources of the IoT devices may different from device to device. And each vendor may design the components and the services of the devices based on their requirement's. for example, some vendors use different programing language for the back end processing and web management interfaces like Perl, CGI, PHP and others. Some vendors use built in shell script command from the kernel.

In addition, this module can be Implemented according to each vendor's functional requirement components i.e. programing language like C, C++ and others basically most of the IoT devices have Linux kernel and Linux kernel is developed with C language. Implementing with C programing language would give more minimalistic module when I compared to other programing languages which may requires additional dependencies this would led to more resource usage when compared to C.

ICTDM can be implemented with considering the following requirement to fulfill and meet with the device specification and to use minimal resources.

- ICTDM Can be designed with the components that are already built with firmware i.e. built in libraries for network communication in the Kernel

- ICTDM can be designed the programing language that the vendor uses for the backend activity for management console without installing additional software's dependencies and libraries.

Each IoT devices developed by different manufacturing companies uses different kinds implementation design and programing language so they don't have common standards. And this is still the challenge faced for IoT environment. In this module design, its shows the high level of design specification of ICTDM that each vendor may integrate with their own functional and non-functional requirement's for detection of cyber threats.

4.4.5. Initial Assumptions

The design of ICTDM and how it works based upon on the following initial assumptions

- The initial configuration for those three monitoring (device state, network and process) components will be statically set by the manufacturer of the devices based on the required services protocols and network protocols.
- When the devices boot this module boots together with privileged mode to able to perform all required functionality. Some activities i.e. networking and process may need root privileges to get required detail information.
- The communication for notification and alerting uses HTTP protocol to transfer and receive information to and from the user or vendor. HTTP is categorized under Low End Machine to Machine protocol and widely used in Internet of things.
- This module only gives REST API interfaces for communication so locally doesn't store to much information to meet the IoT Device specification and architecture.
- The information and alert that would be sent upon the detection is to the vendor or users which is based on the statically configuration file and the address.
- The notification and alerts can be sent to vendor's cloud infrastructure for tracking the cyber threats happening to the devices and the users of the device can access his device status from the vendor infrastructure. Since this module give only communication interface this concept is out of our scope.

4.4.6. Architecture Design

4.4.6.1. Proposed ICTDM Architecture

The proposed ICTDM high level technical architecture, the description of each components and the communication between each component described as follows.

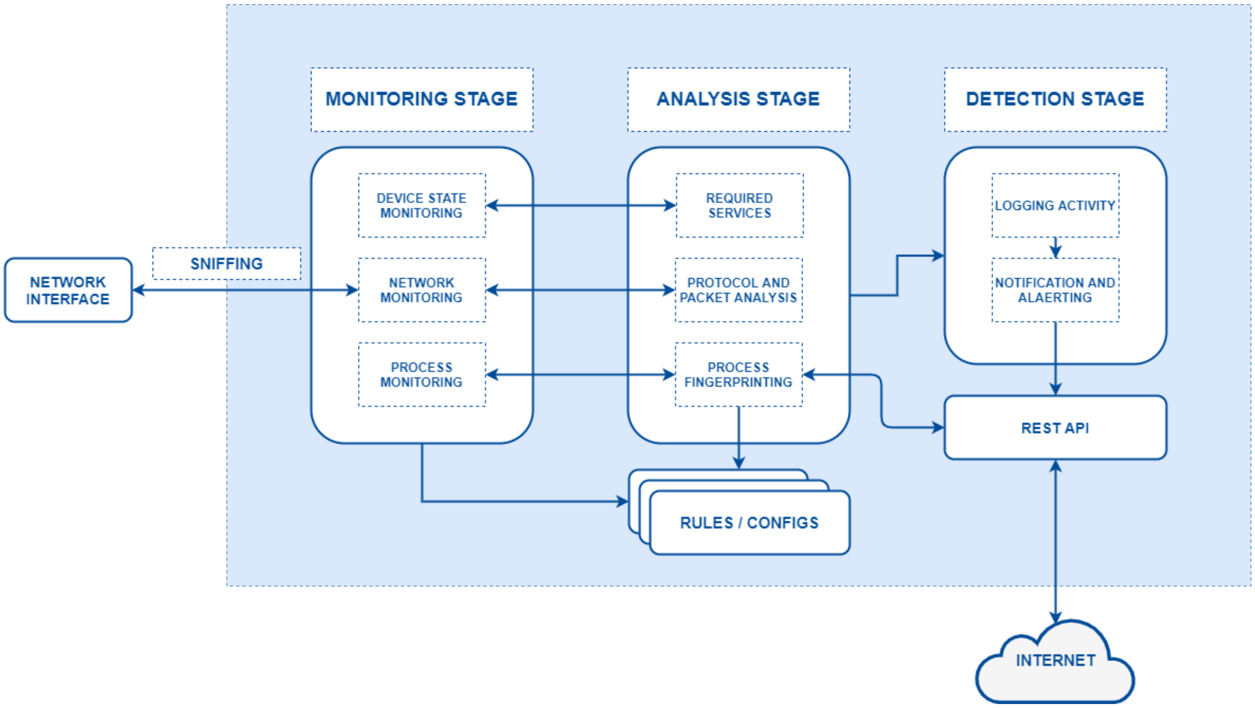


Figure 4.4 Proposed ICTDM architecture [source: own]

4.4.6.2. Rules and Configuration

The rules and configuration files of the proposed ICTDM is the main components of this module. It contains manually configured different information and variables for each ICTDM components which the module uses for monitoring, analysis and detection. In addition, it provides the device and the module to operate with minimal resources because all required information for monitoring analysis and detection stored statically. This configuration file should be stored securely with higher privilege access permission because it contains different sensitive device specific information. Based on the vendor design approach all the configuration variables can be stored in local file storage or in memory with all required protection mode. In this design we considered the configuration file will be stored in local file system. And the following table shows all required configuration variables and example values with their description.

COFIG VARIABLE NAME	TYPE/VALUE	DESCRIPTION
NOTIFY_TARGET_URL	*STRING* *E.g. http://www.cloud.com*	The address of user or vendor where the notification and alerts sent upon the detection of malicious activity.
NOTIFY_TARGET_AUTH_SECRET_KEY	*STRING* *E.g. <RANDOM_24_BYTE>*	Secret key for the device to authenticate to send notification and alert.
AUTHENTICATION_METHOD	*STRING* *E.g. BASIC,API*	Authentication type that the endpoint uses for exchanging information.
PROTOCOL	*STRING* *E.g. HTTP*	The protocol used for communication which is supported by both endpoints.
SSL_KEY_ALG	*STRING* *E.g. RSA*	Algorithm used by SSL for verification and secure communication.
SSL_KEY_PATH	*STRING* *E.g. <FILE PATH>*	Local SSL static key path in the device.
DEVICE_SECRET_KEY	*STRING* *E.g. <RANDOM_24_BYTE>*	Device secret key which is used by other third party applications for authentication and exchanging information.
LOG_FILE_NAME	*STRING* *E.g. <ictdm.log>*	The file name which stores all the malicious activities detected on the analysis phase.

LOG_FILE_PATH	*STRING* *E.g. <FILE PATH>*	Statically configured the path of the log file in the file system.
LOG_FORMAT	*STRING* *E.g. [DATETIME TYPE DETAIL FILE]*	The format which is used to be stored in the log file for detected malicious activity.
PROCESS_VERIFICATION_URL	*STRING* *E.g. http://www.virustotal.com*	The address which used to check the detected file information whether its malicious or not.
PROCESS_VERIFICATION_URL_KEY	*STRING* *E.g. <RANDOM_24_BYTE>*	The key which is used to authenticate the third party application.
DEVICE_STATE_CHECK_TIME	*INTEGER* *E.g. 5 min*	A value in a minute which periodically to check the device state.
DEVICE_STATE_MATCH_FLAG	*BOOLEAN* *E.g. true/false*	A flag which is used to store the state after the device state check completion.
DEVICE_PROCESS_CHECK_TIME	*INTEGER* *E.g. 5 min*	A value in a minute which periodically to check the device if unknown process are running or not.
DEVICE_PROCESS_MATCH_FLAG	*BOOLEAN* *E.g. true/false*	A flag which is used to store the state after the device state check completion.

DEVICE_STATE	*INTEGER* *E.g. 0/1/2*	Stores the device state after the completion of the device state check.
REQUIRED_PROCESS_LIST	*ARRAY OF STRING* *E.g. [httpd,vsftpd]*	A list of running required process in the device configured manually.
REQUIRED_PROCESS_HASH	*ARRAY OF STRING* *E.g. [md5(process)]*	A list hash for the running process which is used to check the integrity of the process. Configured manually.
OTHER_PROCESS_LIST	*ARRAY OF STRING* *E.g. [httpd,vsftpd]*	A list of running other process i.e. kernel processes in the device configured manually/or up on the device booting.
OTHER_PROCESS_HASH	*ARRAY OF STRING* *E.g. [md5(process)]*	A list hash for the running process which is used to check the integrity of the process. Configured manually or up on the device booting.
HASHING_ALGORITHM	*STRING* *E.g. MD5,SHA1*	Hashing algorithm for process and running services fingerprinting.
NETWORK_INTERFACE_NAME	*STRING* *E.g. eth0*	The network interface name which is used to in communication in networking.

SUPPORTED_PROTOCOLS	*STRING* *E.g. HTTP, HTTPS, FTP, MQTT*	The overall supported and accepted communication protocols list used in network monitoring.
MIN_BANDWIDTH_BYTE	*INTEGER* *E.g. 1024 BYTE*	The minimum size of the communication packet in networking in byte.
MAX_BANDWIDTH_BYTE	*INTEGER* *E.g. 4096 BYTE*	The maximum size of the communication packet in networking in byte.
ACCEPTED_ADDRESS	*ARRAY OF STRING* *E.g. [http://www.cloud.com]*	An optional list of address for whitelisting the address which only allowed to access the device.

Table 4.1.12 ICTDM Configuration Variables [source: own]

4.4.6.3. Sub Components

Device State Monitoring and Analysis Stage

This component mainly responsible for monitoring the device state by checking all required services from the configuration file and from the kernel and checking if there is a missing service are there or not finally sets the DEVICE_STATE variable and get full information i.e. process name, process type, date time for logging and notification. The pseudocode and activity diagram described ab below.

```
LOAD_DEVICE_REQUIRED_SERVICES_CONFIGURATION

LOAD_RUNING_SERVICESS_FROM_CURNEL

  COMPARE_AND_MATCH_SERVICES
   IF MATCHED

         CONTINUE;
  ELSE
         GET_DETAIL_INFORMATION

         PASS TO DETECTION PHASE
```

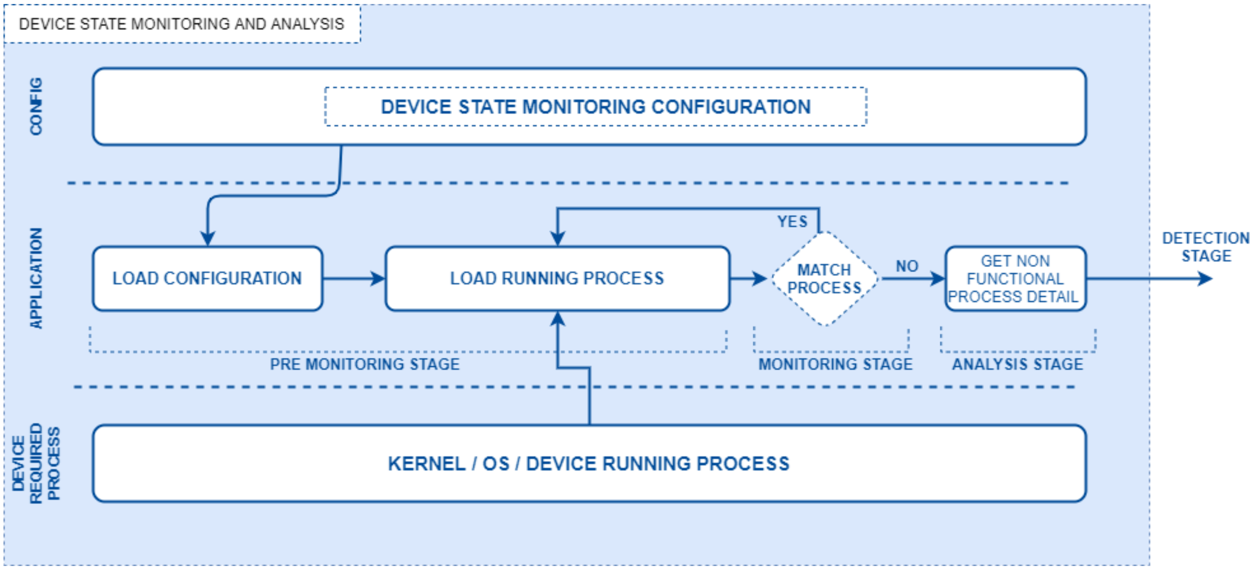


Figure 4.5 Device state monitoring and analysis activity [source: own]

Network Monitoring and Analysis Component

This component responsible monitoring and analyzing network communications, packets and protocols by sniffing the traffic from the network interface and compare the packets information and behavior from statically defined in the configuration. The pseudocode and the activity diagram described as follows.

```
LOAD_NETWORK_CONFIGURATION

SNIFF_THE_TRAFFIC_FROM_INTERFACE

  IF PROTOCOL NOT MATCHED

           GET TRAFFIC INFORMATION

           PASS TO DETECTION PHASE

  ELSE
           IF MIN AND MAX BANDWITH MATCH

                     CONTINUE

           ELSE

                     GET TRAFFIC INFORMATION

                     PASS TO DETECTION PHASE
```

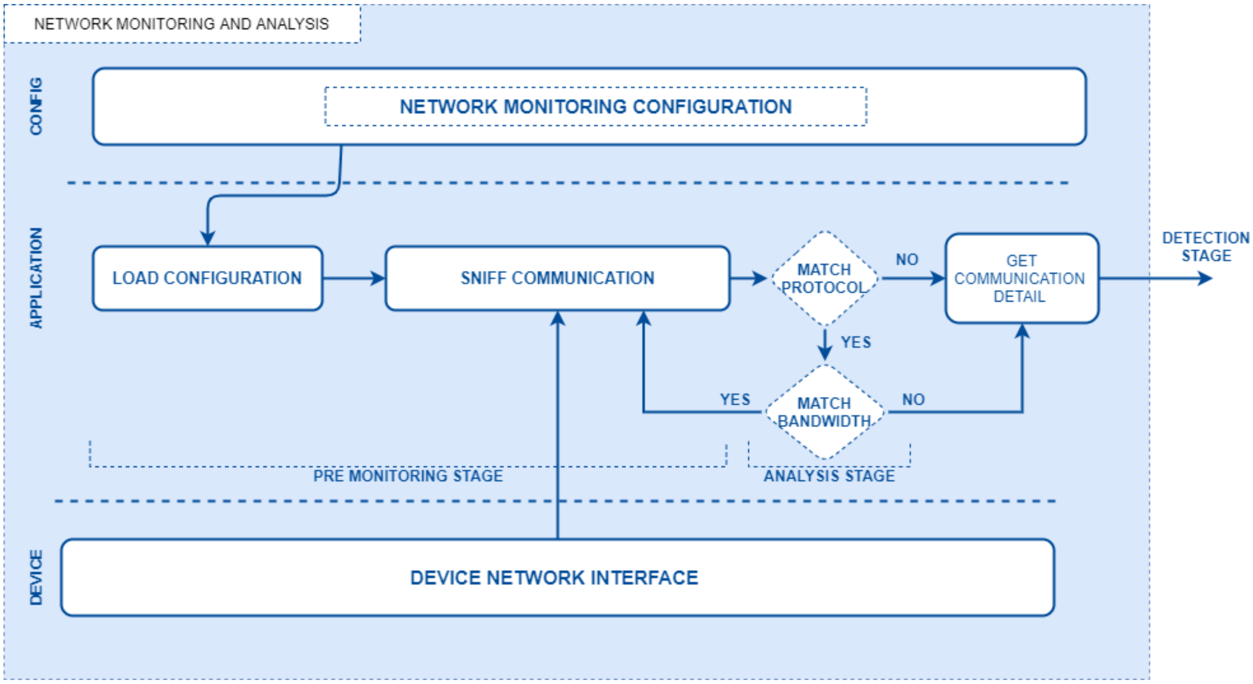


Figure 4.6 Network traffic monitoring and analysis activity [source: own]

Process Monitoring and Analysis Component

This component responsible for monitoring and analyzing unknown processes and checks for the file characteristics whether it is a malware or not by sending the file hash to the third party api i.e http://www.virustotal.com then pass the result to the detection phase. The pseudocode and activity diagram described as follows.

```
LOAD_PROCESS_LIST_AND_CONFIGURATION
LOAD_KERNEL_RUNNING_PROCESS
    IF PROCESS MATCHED
        IF FILE HASH SUM MATCHED
            CONTINUE
        ELSE
            SEND HASH VALUE TO API
            IF FILE INFECTED

                GET PROCESS DETAIL
                PASS TO DETECTION PHASE
            ELSE
                GET PROCESS DETAIL
                PASS TO DETECTION PHASE
    ELSE

        GET PROCESS DETAIL
        PASS TO DETECTION PHASE
```

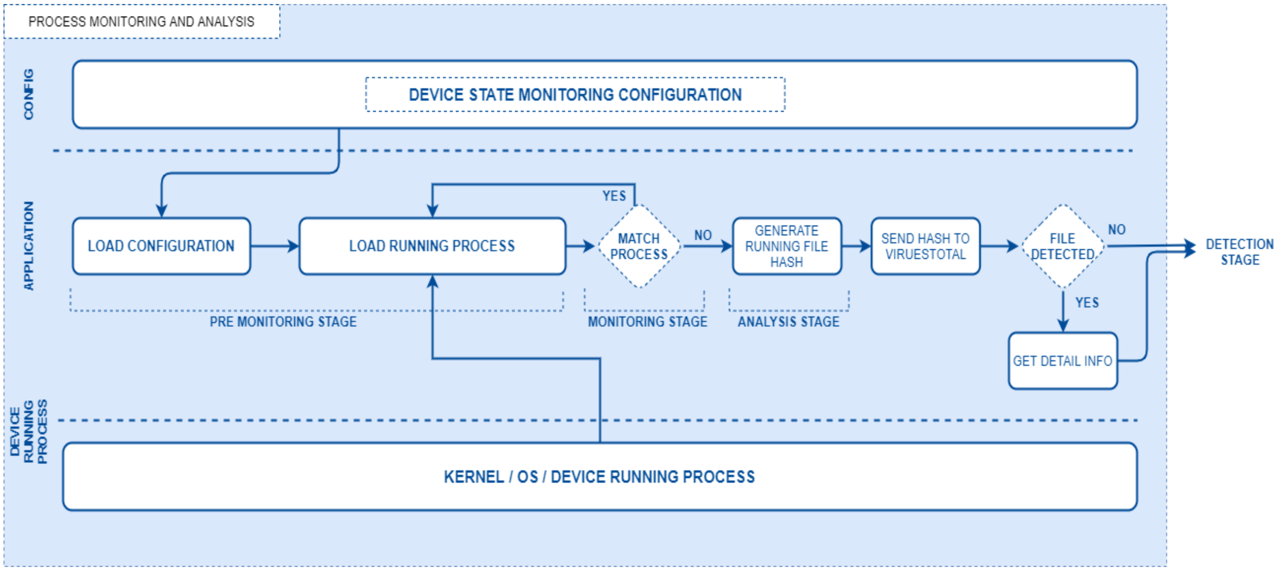


Figure 4.7 Device process monitoring and analysis activity [source: own]

Detection component

This component responsible for logging the malicious activities and sending notification to the user or vendor by using HTTP REST API. The pseudocode and the activity diagram described as follows.

```
LOAD_DETECTION_CONFIGURATION

PARSE_ ANALYZED_MALICIUS_ACTIVITY

WRITE _TO_LOG_FILE

SEND_NOTIFICATION_AND_ALERT
```

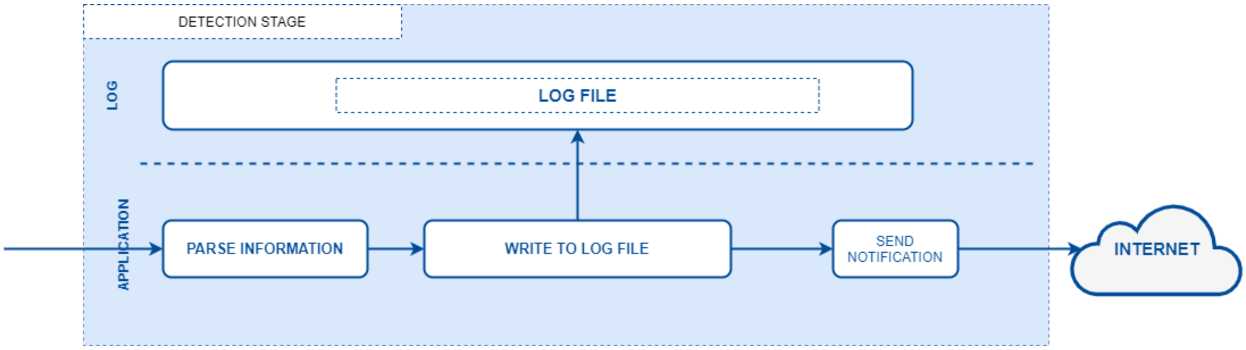


Figure 4.8 Detection stage activity [source: own]

5. Result and Discussion

5.1. Firmware Risk Analysis

From our analysis result we have found several security issues and having different severity rate. And the impact of each vulnerability varies. In this section we have shown the risk for the analyzed firmware's and potentially those having the same models but with different versions with different perspectives. We assumed for those devices models and firmware versions which is not included in this research have same code base for their devices. Off course they have some minor changes on each firmware version releases but the probability to have the same code base is high.

Which devices is affected?

For this research as shown we have selected four devices. But the number of devices that we found from Shodan platform with same specification model and firmware version is much higher.

Device	**Total number**	**Total Vulnerabilities**
TP-LINK Archer C7	4124	455
AXIS 214 PLZ	201	595
D-LINK DIR-850L	1427	410
REOLINK RLC-410	155	238
Total	5907	1698

Table 5.1 Affected device and vulnerabilities [source: own]

As we see the total number of devices and the total number of security issues found from the analysis is relatively very high. This is due to most the firmware still functioning and available on the internet is very old or outdated. From this analysis we can conclude that most of the devices currently connected on the internet including which is not analyzed in this research and those which have outdated firmware releases and their components potentially have multiple security issues.

What data is at risk?

From this perspective and based on our analyzed devices we can categorize in two part which is the routers and IP cameras. When we consider the risks of data loses in the router it is very high when compared to the cameras. Routers can be used as an internet gateway for the users and any organization. Every users' network traffic is passes through the router. Which means users sensitive information i.e. online accounts, credit card information totally everything is at risk. Not only the compromization of the device leads to the data looses also the communication between the device and the user is not encrypted communication in most of the devices. This will lead also MIMT attack which is an attacker monitoring the traffic intercepting sensitive information will potentially a Couse for any data loses.

When we consider the IP cameras the potential data or information loses will be primarily the audiovisual contents and streaming activity which may revels sensitive information's from the camera.

Severity of the potential attack?

Basically, we have categorized the severity of the vulnerability discovered and found from the analysis in to four categories. Which is critical high medium and low. And the severity of each vulnerability calculated using CVSS (Common Vulnerability Scoring System) [69].

Severity	**CVSS**	**Issues**	**Description**
Critical	9.0 – 10.0	150	Critical severity vulnerabilities allow a remote attacker to easily gain privileged access to a system and execute arbitrary code or take over an entire system. They would include any attacks that have no or very low user* interaction to exploit the vulnerability.
High	7.0 – 8.9	101	High severity vulnerabilities allow a remote attacker to gain privileged access to a system and execute arbitrary code or take control of a system. These would include any attacks that require at least some level of user* interaction for an attacker to gain entry to a system. Under limited circumstances, it may be upgraded to a Critical vulnerability if the exploit is being used by attackers at the time the patch is announced or shortly thereafter
Medium	4.0 – 6.9	7	Medium severity vulnerabilities allow an attacker to perform a denial of service, to read privileged information, and/or under certain circumstances it may become a High vulnerability. They would include any attacks that require a high level of user* interaction or physical access to the system is needed to gain control of the system.
Low	0.1 – 3.9	358	Low severity vulnerabilities usually refer to unnecessary services or services providing information leakage which could be useful to an attacker to execute other more sophisticated attacks

Table 5.2 Severity of the vulnerability SAST [source: own]

Potential damage as a result of vulnerability?

The potential damage which is caused by the vulnerabilities depends on each security issues and severity. From the analysis result for those critical security issues may lead to the full compromise and takeover of the device. In addition, the attacker may gather sensitive information of the device users and organization and using the device his own intended purpose. This leads the organization and users to business disruption and non-functionalities including devices which is not covered in this analysis.

5.2. Significance of ICTDM

As we seen from the risk analysis the impact of the vulnerability could cause significant amounts of security and privacy losses. To decrease and detect those security issues and to take appropriate mitigation actions before causing too much damage the proposed model ICTDM plays a great role on the IoT environment. ICTDM provides the detection of cyber threat on the following different approaches.

- Network Level threats: many attacks primarily started from the network communication. The proposed model monitors and analyzes the traffic which comes to the device. And notifies to the users or a vendor. So, it provides early detection of the cyber threats.
- Application Level threats: some cyber threats primarily target the application and the running services. Even if the communication medium is the network protocols. Based on the applications and the running services some complicated threats may bypasses the first detection phase. But the proposed model monitors and analyze the behavior of the unknown process and services. Same here as previous provides an early detection of malicious file and malwares.

Finally implementing and integration the proposed module by the vendors with the necessary IoT devices give a great benefit for users and organization with their business process and day to day activities. In addition, it gives a competitive advantage for the vendors and manufacturers of IoT device with their product quality and with the ability to detect cyber threat on the early stage. To properly manage and mitigate the from known and unknown vulnerabilities.

5.3. Security Measures

In addition to the proposed model there are different best practices from the manufacturer and the user side to be comply on the IoT Devices and environment. This best practice and measures the risks the cyber threat in the IoT environment. Even if still the security requirement for IoT device is still under development with IETF the following requirements are necessary the risks.

- **Keeping the firmware and its component up to date**: this is one way preventing cyber threats older firmware and components may have security issues and vulnerability.
- **Assigning unique device identifier and credential**: this prevents the massive exploitability of the devices in the IoT environment and Internet.
- **Applying strong authentication and access control**: this includes some thrust boundaries and decreases the risks of authentication attacks i.e. brute forcing.
- **Using secure communication protocols**: with the latest updates and usages of cryptographic network protocols to decrease and prevent network level traffic attacks i.e. MITM.
- **Deploying security monitoring and analysis tools**: this provides an early detection of known and unknown security issues in the device to take a measure. i.e. our proposed module.
- **Minimize the attack surfaces on the device**: this includes unnecessary services and processes that are running on the device. i.e. smb, ftp, ssh.

In addition, this major security measure and practices, it is required to have a policy to enforce a user’s and organization to change the device default username and password. And others device specific and operation-based measures are required to decrease the risks caused by attackers.

6. Conclusions

Internet of things and environment is still on the early and developing stage. And this technology provides a great advantage on our day to day life's. from modernizing our day to day life to monitor and operate large infrastructures. Also, it is taking attentions of multiple manufacturers and companies and they are investing and joining the IoT environment.

With the massive development and growth of this technology there are other concerns that comes with its development. Security and privacy issues. These issues could lead to multiple damages and risks to the users and organizations. And should be addressed and take appropriate measures to decrease those cyber threats.

Our proposed model plays a great role on detecting the cyber threat at the early stages for the challenges currently faced in the IoT environment. But doesn't provides the protection mechanisms for against cyber threat. This protection and for the better performance and detection of this module will be the future works.

7. Reference

[1] K. Ashton, That ―Internet of Things‖ Thing, RFiD Journal. (2009).

[2] “State of Enterprise IoT Security in 2020.” Palo Alto Networks, 2020, www.paloaltonetworks.com/resources/infographics/state-of-enterprise-iot-security-in-2020. Accessed 3 Feb. 2021.

[3] “Internet of Things (IoT) - the Future of IoT Miniguide: The Burgeoning IoT Market Continues.” *Cisco*, July 2019, www.cisco.com/c/en/us/solutions/internet-of-things/future-of-iot.html. Accessed 3 Feb. 2021.

[4] Alaba, Fadele Ayotunde, et al. “Internet of Things Security: A Survey.” Journal of Network and Computer Applications, vol. 88, June 2017, pp. 10–28,

[5] M. Conti, N. Dragoni and V. Lesyk, "A Survey of Man In The Middle Attacks," in IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 2027-2051, thirdquarter 2016, doi: 10.1109/COMST.2016.2548426.

[6] Alrawi, Omar & Lever, Chaz & Antonakakis, Manos & Monrose, Fabian. (2019). SoK: Security Evaluation of Home-Based IoT Deployments. 1362-1380. 10.1109/SP.2019.00013.

[7] Medeiros, Lohana & Zuvanov, Fabio & Mello, Flávio & Strauss, Edilberto. (2018). IoT Information Security Evaluation for Developers and Users. Journal of Information Security and Cryptography (Enigma). 4. 16. 10.17648/enigma.v4i1.63.

[8] Conti, Mauro, et al. “Internet of Things Security and Forensics: Challenges and Opportunities.” Future Generation Computer Systems, vol. 78, Jan. 2018, pp. 544–546

[9] Aldwairi, Monther & Tawalbeh, Loai. (2020). Security techniques for intelligent spam sensing and anomaly detection in online social platforms. International Journal of Electrical and Computer Engineering (IJECE). 10. 10.11591/ijece.v10i1.pp275-287.

[10] Shafique, Kinza, et al. “Internet of Things (IoT) for Next-Generation Smart Systems: A Review of Current Challenges, Future Trends and Prospects for Emerging 5G-IoT Scenarios.” IEEE Access, vol. 8, 2020, pp. 23022–23040, 10.1109/access.2020.2970118. Accessed 6 Feb. 2021.

[11] Henry, Joseph, and David Hochfelder. Inventor of the Telegraph

[12] Prince, Brumancia. (2019). A Survey on IoT Technologies, Evolution and Architecture.

[13] Alaba, Fadele Ayotunde, et al. “Internet of Things Security: A Survey.” Journal of Network and Computer Applications, vol. 88, June 2017, pp. 10–28, https://www.ibm.com/blogs/industries/little-known-story-first-iot-device/. Accessed 3 Feb. 2021.

[14] Romkey, John. (2017). Toast of the IoT: The 1990 Interop Internet Toaster. IEEE Consumer Electronics Magazine. 6. 116-119. 10.1109/MCE.2016.2614740.

[15] “A Brief History of the Internet of Things - DATAVERSITY.” DATAVERSITY, 16 Aug. 2016, www.dataversity.net/brief-history-internet-things/. Accessed 6 Feb. 2021.

[16] E-Zigurat.com, 2021, www.e-zigurat.com/innovation-school/wp-content/uploads/sites/5/2019/03/20190326-iot-in-five-years2-1.jpg. Accessed 6 Feb. 2021.

[17] Yamin, M. Information technologies of 21st century and their impact on the society. Int. j. inf. tecnol. 11, 759–766 (2019). https://doi.org/10.1007/s41870-019-00355-

[18] Luo, Suhuai & Li, Jiaming. (2009). A smart fridge with an ability to enhance health and enable better nutrition. International Journal of Multimedia and Ubiquitous Engineering. 4.

[19] Huang, Chung-Ching & Bardzell, Jeffrey & Terrell, Jennifer. (2011). Can your pet rabbit read your email?: a critical analysis of the Nabaztag rabbit. 10.1145/2347504.2347532.

[20] Angelova, Nadezhda & Kiryakova, Gabriela & Yordanova, Lina. (2017). The great impact of internet of things on business. Trakia Journal of Science. 15. 406-412. 10.15547/tjs.2017.s.01.068.

[21] Readthedocs.io, 2021, mpython.readthedocs.io/en/master/_images/three-layer-iot-architecture.png. Accessed 7 Feb. 2021.

[22] Kumar, Nallapaneni Manoj, and Pradeep Kumar Mallick. “The Internet of Things: Insights into the Building Blocks, Component Interactions, and Architecture Layers.” Procedia Computer Science, vol. 132, 2018, pp. 109–117, 10.1016/j.procs.2018.05.170. Accessed 7 Feb. 2021.

[23] S. Al-Sarawi, M. Anbar, K. Alieyan and M. Alzubaidi, "Internet of Things (IoT) communication protocols: Review," 2017 8th International Conference on Information Technology (ICIT), Amman, 2017, pp. 685-690, doi: 10.1109/ICITECH.2017.8079928.

[24] Burak, Han, et al. Comparative Analysis of IoT Communication Protocols Secure Fog Computing in IoT Based Systems View Project the Effects of Eye-Tracking Based and Parent Mediated Joint Attention Early Intervention Program on Social Communication Skills of Children with ASD View Project. 2018, 10.1109/ISNCC.2018.8530963.

[25] Hussein, AbdelRahman H. “Internet of Things (IOT): Research Challenges and Future Applications.” International Journal of Advanced Computer Science and Applications, vol. 10, no. 6, 2019, 10.14569/ijacsa.2019.0100611.

[26] “Google Glass.” Wikipedia, Wikimedia Foundation, 19 Jan. 2021, en.wikipedia.org/wiki/Google_Glass. Accessed 7 Feb. 2021.

[27] Impact of IoT Enabled Service Solutions in the Downstream Automotive Supply Chain. https://www.diva-portal.org/smash/get/diva2:838916/FULLTEXT01.pdf Accessed 7 Feb. 2021.

[28] Boyes, Hugh, et al. “The Industrial Internet of Things (IIoT): An Analysis Framework.” Computers in Industry, vol. 101, Oct. 2018, pp. 1–12,

[29] Zanella, Andrea & Bui, Nicola & Castellani, Angelo & Vangelista, Lorenzo & Zorzi, Michele. (2012). Internet of Things for Smart Cities. Internet of Things Journal, IEEE. 1. 10.1109/JIOT.2014.2306328.

[30] Stočes, Michal & Vaněk, Jiří & Masner, Jan & Pavlík, J.. (2016). Internet of Things (IoT) in Agriculture - Selected Aspects. Agris on-line Papers in Economics and Informatics. VIII. 83-88. 10.7160/aol.2016.080108.

[31] Gamage, Rashmika & Madushan, Rashmika. (2020). A Review on Applications of Internet Of Things (IOT) in Healthcare. Journal of the American Society for Information Science and Technology.

[32] Linkedin.com "The advantages and disadvantages of Internet Of Things", 2021, www.linkedin.com/pulse/advantages-disadvantages-internet-things-iot-tommy-quek/. Accessed 15 Mar. 2021.

[33] Browning, Jack. "5 Benefits of the Internet of Things for SMBs." Impact Networking, Impact Networking, 11 Feb. 2021, www.impactmybiz.com/blog/blog-5-benefits-of-the-internet-of-things-for-smbs/. Accessed 15 Mar. 2021.

[34] Varun Bhagat. "What Are Pros and Cons of Internet of Things? Let's Check Out!" PixelCrayons, PixelCrayons, 23 June 2019, www.pixelcrayons.com/blog/what-are-pros-and-cons-of-internet-of-things/. Accessed 15 Mar. 2021.

[35] "What Is IoT Security?" Palo Alto Networks, 2018,
www.paloaltonetworks.com/cyberpedia/what-is-iot-security. Accessed 15 Mar. 2021.

[36] "State of Enterprise IoT Security in 2020." Palo Alto Networks, 2020,
www.paloaltonetworks.com/resources/infographics/state-of-enterprise-iot-security-in-2020. Accessed 15 Mar. 2021.

[37] "OWASP Internet of Things Project - OWASP." Owasp.org, 2018,
wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=IoT_Attack_Surface_Areas. Accessed 27 Jan. 2021

[38] "Mirai (Malware)." Wikipedia, Wikimedia Foundation, 12 Dec. 2020,
en.wikipedia.org/wiki/Mirai_(malware). Accessed 28 Jan. 2021.
[39] Memcached DDoS Attack | Cloudflare. "Cloudflare." Cloudflare, Cloudflare, 2021,
www.cloudflare.com/learning/ddos/memcached-ddos-attack/. Accessed 28 Jan. 2021.

[40] "Stuxnet." *Wikipedia*, Wikimedia Foundation, 23 Jan. 2021, en.wikipedia.org/wiki/Stuxnet. Accessed 28 Jan. 2021.

[41] Ballano, Mario, and Barcena Candid. SECURITY RESPONSE Insecurity in the Internet of Things. https://docs.broadcom.com/doc/insecurity-in-the-internet-of-things-en Accessed 28 Jan. 2021

[42] M. Stanislav and T. Beardsley," HACKING IoT: A Case Study on Baby Monitor Exposures and Vulnerabilities" https://www.rapid7.com/globalassets/external/docs/Hacking-IoT-A-Case-Study-on-Baby-Monitor-Exposures-and-Vulnerabilities.pdf

[43] M. Stanislav and T. Beardsley, ``HACKING IoT: A case study on baby monitor exposures and vulnerabilities," Rapid7, Boston, MA, USA,Tech. Rep. 6-7, Sep. 2015. [Online]. Available: https://www.rapid7.com/docs/Hacking-IoT-A-Case-Study-on-Baby-Monitor-Exposuresand-Vulnerabilities.pdf

[44] T. Xu, J. B. Wendt, and M. Potkonjak, ``Security of IoT systems: Design challenges and opportunities," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD), Nov. 2014, pp. 417_423.

[45] "Carna Botnet." Wikipedia, Wikimedia Foundation, 2 Jan. 2021, en.wikipedia.org/wiki/Carna_botnet. Accessed 15 Mar. 2021.

[46] "Industry 4.0: Fourth Industrial Revolution Guide to Industrie 4.0." I-SCOOP, 29 Jan. 2021, www.i-scoop.eu/industry-4-0/. Accessed 9 Feb. 2021.

[47] "Man-In-The-Middle Attack." *Wikipedia*, Wikimedia Foundation, 28 Jan. 2021, en.wikipedia.org/wiki/Man-in-the-middle_attack. Accessed 28 Jan. 2021.

[48] Xu, Meng, et al. Dominance as a New Trusted Computing Primitive for the Internet of Things. , 2019.

[49] Tawalbeh, Lo'ai, et al. "IoT Privacy and Security: Challenges and Solutions." *Applied Sciences*, vol. 10, no. 12, 15 June 2020, p. 4102, 10.3390/app10124102. Accessed 30 Jan. 2021.

[50] Y. Meng, W. Zhang, H. Zhu and X. S. Shen, "Securing Consumer IoT in the Smart Home: Architecture, Challenges, and Countermeasures," in *IEEE Wireless Communications*, vol. 25, no. 6, pp. 53-59, December 2018, doi: 10.1109/MWC.2017.1800100.

[51] Siby, Sandra, et al. *IoTScanner: Detecting Privacy Threats in IoT Neighborhoods*. 10.1145/3055245.3055253. Accessed 30 Jan. 2021.

[52] Mohamad Noor, Mardiana binti, and Wan Haslina Hassan. "Current Research on Internet of Things (IoT) Security: A Survey." *Computer Networks*, vol. 148, Jan. 2019, pp. 283–294,

[53] Leloglu, Engin. "A Review of Security Concerns in Internet of Things." *Journal of Computer and Communications*, vol. 05, no. 01, 2017, pp. 121–136,

[54] Liu, Xiruo, et al. "A Security Framework for the Internet of Things in the Future Internet Architecture." *Future Internet*, vol. 9, no. 3, 28 June 2017, p. 27, 10.3390/fi9030027. Accessed 30 Jan. 2021.

[55] Ali, S.; Bosche, A.; Ford, F. Cybersecurity Is the Key to Unlocking Demand in the Internet of Things; Bain and Company: Boston, MA, USA, 2018.

[56] Sadeghi, Ahmad-Reza, et al. "Security and Privacy Challenges in Industrial Internet of Things." *Proceedings of the 52nd Annual Design Automation Conference on - DAC '15*, 2015, 10.1145/2744769.2747942. Accessed 1 Mar. 2020.

[57] Zandberg, Koen, et al. “Secure Firmware Updates for Constrained IoT Devices Using Open Standards: A Reality Check.” IEEE Access, vol. 7, 2019, pp. 71907–71920, 10.1109/access.2019.2919760. Accessed 31 Jan. 2021.

[58] “The Constrained Application Protocol (CoAP).” *Ietf.org*, 2014, tools.ietf.org/html/rfc7252. Accessed 31 Jan. 2021.

[59] Naito, Katsuhiro. “A Survey on the Internet-of-Things: Standards, Challenges and Future Prospects.” *Journal of Information Processing*, vol. 25, 2017, pp. 23–31, www.jstage.jst.go.jp/article/ipsjjip/25/0/25_23/_pdf/-char/en, 10.2197/ipsjjip.25.23]. Accessed 31 Jan. 2021.

[60] Lu, Ching-Hu, et al. “Secure and Efficient Firmware Update for Increasing IoT-Enabled Smart Devices.” Journal of Ambient Intelligence and Humanized Computing, 4 Sept. 2020, 10.1007/s12652-020-02492-z. Accessed 31 Jan. 2021.

[61] Hu, Jen-Wei, et al. “Autonomous and Malware-Proof Blockchain-Based Firmware Update Platform with Efficient Batch Verification for Internet of Things Devices.” *Computers & Security*, vol. 86, Sept. 2019, pp. 238–252, 10.1016/j.cose.2019.06.008. Accessed 31 Jan. 2021.

[62] Lee, Boohyung, and Jong-Hyouk Lee. “Blockchain-Based Secure Firmware Update for Embedded Devices in an Internet of Things Environment.” The Journal of Supercomputing, vol. 73, no. 3, 13 Sept. 2016, pp. 1152–1167, 10.1007/s11227-016-1870-0. Accessed 2 Feb. 2021.

[63] Zhu, Xinbing, et al. “Research on Security Detection Technology for Internet of Things Terminal Based on Firmware Code Genes.” IEEE Access, vol. 8, 2020, pp. 150226–150241, 10.1109/access.2020.3017088. Accessed 2 Feb. 2021.

[64] OMA Lightweight Machine to Machine Technical Specification. , 2018.
http://www.openmobilealliance.org/release/LightweightM2M/V1_0_2-20180209-A/OMA-TS-LightweightM2M-V1_0_2-20180209-A.pdf Accessed 9 Feb. 2021.

[65] “Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things.” Ietf.org, 2016, tools.ietf.org/html/rfc7925. Accessed 9 Feb. 2021.

[66] Heffner, Craig. “Binwalk.” Kali.org, 2014,
tools.kali.org/forensics/binwalk#:~:text=Binwalk%20is%20a%20tool%20for,for%20the%20Unix%20file%20utility.. Accessed 28 Mar. 2021.

[67] Smith, Craig. “Firmwalker.” Kali.tools, 2020, en.kali.tools/all/?tool=466. Accessed 28 Mar. 2021

[68] “Shodan.” Shodan.io, 2013, www.shodan.io/. Accessed 28 Mar. 2021.

[69] “CVSS V3.1 Specification Document.” FIRST — Forum of Incident Response and Security Teams, 2019, www.first.org/cvss/specification-document. Accessed 28 Mar. 2021.

8. Appendix

Identified Outdated and Vulnerable Firmware Components

Device Name	File Name	Number of CVE
Tp-Link Archer C7 V1	Linux kernel 2.6.31	360
	busybox 1.20	3
	iptables 1.4.5	1
	net-snmp-5.4.2.1	7
	ppp-2.4.3_ipv6	2
	sysstat-6.0.1	1
	vsftpd-2.3.2	2
AXIS 214 PLZ V 4.40	Linux kernel 2.6.18	553
	busybox 1.1.3	5
	iptables 1.2.1a	3
	OpenSSL 0.9.7f	32
	sysklogd 1.3	1
	termcap 1.2.4	1
Reolink RLC-410 v1.0.2	Linux Kernel 4.1.0	146
	nginx/1.6.2	3
D-LINK DIR-850L	Httpd 2.0	18
	busybox 1.0	6

Table 8.1 Identified firmware components [source: own]

Identified Sensitive Files in file system (Most Important)

Device Name	File
Tp-Link Archer C7 V1	/squashfs-root/etc/shadow
	/etc/wpa2/hostapd.eap_user
	/web/userRpm/NasUserCfgRpm.htm
AXIS 214 PLZ V 4.40	/jffs2-root/fs_1/etc/passwd
	/usr/etc/param/par_https.conf
	/usr/etc/param/par_network_ftp.conf
	/usr/etc/param/par_prop_api_http.conf
	/usr/sbin/axisns.sh
	/usr/html/axis-cgi/admin/restart.cgi
	/usr/html/axiscgi/pwdroot/pwdroot.cgi
Reolink RLC-410 v1.0.2	/squashfs-root/etc/passwd
	/squashfs-root/mnt/app/www/self.key
	/fastcgi.conf
	/api.cgi
	/squashfs-root/bin/api.cgi
D-LINK DIR-850L	/squashfs-root/etc/passwd
	/squashfs-root/etc/stunnel_cert.pem
	/squashfs-root/etc/stunnel.key
	/squashfsroot/etc/defnodes/mfc_config.xml

Table 8.2 Identified sensitive files [source: own]