Literature Review & Project Proposal

IoT Cyber Security Analysis and R&D Investigation

Authors[a]*:*
Yewei Yuan, Kexi Huang

[a]Supported by BMT

Supervisor:
Dr. Daniel Lawson

School of Mathematics
University of Bristol

May 21, 2020

Contents

1 Introduction

Accroding to Statista, the total installed base of connected devices is expected to reach 75.44 billion by 2025, which means that it will increase five-fold in ten years[1]. These huge and growing number of devices devices connected through the Internet have created the Internet of Things (IoT). The IoT landscape includes Industrial Internet of Things (IIoT) and Industrial Control System (ICS) in the context of Information Technology (IT) and Operational Technologies (OTs). IoT devices are now ubiquitous and widely used in smart cities, homes, manufacturing, automotive, transportation and logistics, retail, public sector and healthcare.

However, most of IoT devices are vulnerable to intrusion and therefore have become the main targets of cybercriminals such as Mirai malware, which swept the Internet through massive distributed denial of service (DDoS) at the end of 2016 and the attack overwhelmed several high-profile targets[2]. A larger "surface area" would be introduced by these IoT devices to attack and expose systems connected to them[3]. Therefore, critical infrastructure systems such as water treatment or power plants as well as our daily lives will all be threatened by security vulnerabilities in IoT devices.

Against this worrying and challenging context, our project will first assess the IoT landscape from the perspective of cybersecurity. This activity includes a review of:

- IoT key technologies in the commercial and enterprise/industrial sectors;
- Review of major IoT vulnerabilities, threats and security issues documented in available online literature and papers, especially for webcams;
- Review of major IoT cyber-attacks in the last 5-10 years.

Then we will proceed to the R&D practical Lab work. There will be a cyber investigation and experiment utilising an IP (wi-fi) enabled camera as a significant example of IoT device. This R&D work would include:

- Utilise an IP webcam that supports a wide range of web connectivity protocols (port forwarding, UPnP [1], P2P [2], etc.). Then build a mini-test environment including a router with web access and IP camera which will support packet capture and sniffing of traffic generated by the camera;

[1] described in section 2.3
[2] described in section 2.1.1

- Investigate P2P communication protocols and potentially others by using open source packet capture tools Wireshark. Capture network traffic generated by the web camera, especially when the web camera is remotely accessed from a mobile web application; capture utilised protocols (e.g. HTTP/HTTPS), security level, exchanged, information;
- Assess, from a security viewpoint, utilised protocols and exchanged data during these interactions (e.g. P2P) by identifying any potential issues (e.g. unsecure use of protocols/HTTP, potential unprotected exposure of confidential data, etc.)

Starting from this project description, this report gives a literature review and a project plan. In Section 2, we give a summary of the relevent literature that will be used in the project about the IoT landscape assessment from the perspective of cybersecurity. The content includes a three-layer model of the Internet of Things system as well as major security vulnerabilities and cyber attacks. Section 3 includes main approaches in R&D assessment part. We will talk about how machine learning(further, deep learning) is connected to IoT security, and how we are going to use it in our lab work to assess utilised protocols and exchanged data. At the same time we will try to use some statistical methods to model our data streams. Using the methods described in this section to assess utilised protocols and exchanged data is the core of our project. Section 4 introducts relevent datasets for IoT, and where is the data from that we are going to use in our project. The data will be mainly based on our lab work and will also use resources on Internet for comparison. Section 5 is the project proposal. It contains 1) the high level structure for the project outlining any major units of lab work, any dependencies, additional software and computing infrastructure requiredand and the timeline; 2)An explanation of who will be doing these units of work and our tools including Github page and Colab; 3)Risk assessment and mitigation plans. Finally, in Section 6 we give a conclusion.

2 IoT Landscape

2.1 IoT Technologies and The Architecture

2.1.1 Protocols

Internet of Things (IoT) can achieve many different functions including intelligent identification, positioning, tracking, monitoring and management[4] by using infrared sensors, global positioning systems (GPS), laser scanners and other information sensing equipment to connect any item to the Internet for information exchange and communication according to standard and effective protocols.

These IoT networks use various radio technologies, such as radio frequency identification (RFID), WLAN (IEEE 802.11), WPAN (IEEE 802.15) and WMAN (IEEE 802.16) for lower-level communications in the data link layer[5]. And in the application layer, unlike the Web, which uses only the messaging protocol HTTP, IoT cannot use only a single standard to meet all its needs[6]. It mainly has MQTT, CoAP, AMQP and HTTP, four protocols currently, which are shown at the top of the protocol stack for IoT systems[7] in [figure 1].

Figure 1: Protocol stack for IoT systems[7]. The four layers from top to bottom are the application layer, transport layer, network layer, and link layer of the Internet protocol stack. Each layer contains different network protocols.

As shown in the figure, except for the application layer, the protocols used by the Internet of Things are generally the same as the traditional Internet protocol stack. These top-level protocols will also be one of the main research contents of our project. The following is a description of the four messaging protocols MQTT, CoAP, AMQP and HTTP:

- **MQTT (Message Queuing Telemetry Transport Protocol)** MQTT is a lightweight communication protocol[8] based on the client-server model of the publish/subscribe mode. It is built on the TCP/IP protocol. The biggest advantage of MQTT is that it can provide real-time and reliable message service for connecting remote devices with little cost and limited bandwidth, which make it has a wider application in the IoT systems.

- **CoAP (Constrained Application Protocol)** COAP is a lightweight protocol specifically applied to constrained nodes and constrained networks[9]. Constrained nodes usually work with small ROM and RAM, and constrained networks usually have a high packet loss rate and a small throughput. The COAP protocol is designed for M2M (machine-to-machine) applications like smart energy and smart buildings. CoAP uses UDP as the transport protocol and DTLS to improve security[10].

- **AMQP (Advanced Message Queuing Protocol)** AMQP provides various functions related to messaging. Clients and message middleware based on the AMQP protocol can deliver messages and are not restricted by different client / middleware products, different development languages, and other conditions. Both request/response and publish/subscribe architecture are supported by AMQP. AMQP uses TCP as the transport protocol and uses TLS / SSL and SASL for security protection[11].
- **HTTP (Hyper Text Transport Protocol)** HTTP is mainly a Web messaging protocol. HTTP is a text-based protocol, depending on the Web server or programming technology. It provides multiple functions such as persistent connection, request pipeline and block transmission coding. HTTP uses TCP as the default transport protocol and uses TLS / SSL to ensure security[12].

The above four protocols all belong to the client-server architecture. Unlike the client-server architecture in which customers do not exchange information with each other but only communicate with the corresponding server, another important point we want to study is the P2P architecture in IoT systems. Peers, can be IoT devices, are both suppliers and consumers of resources in P2P architecture[13]. Therefore, P2P communication technology is also widely used in IoT.

2.1.2 Three-layer model

In order to describe the architecture of a general IoT system better.There is a three-layer model [14], as shown in [figure 2], widely used to describe IoT systems.

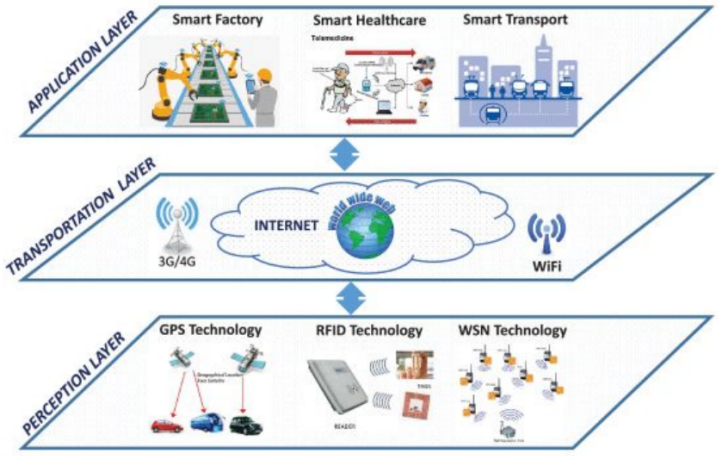

Figure 2: The three-layer model[14].The three layers from top to bottom are the perception layer, transportation layer and application layer. The figure shows the representative items contained in each layer such as 3G/4G at transportation layer.

In the model, an IoT system is represented and described by three main key layers: 1) *Perception Layer*; 2) *Transportation(or Network) Layer*; and 3) *Application Layer*. Each layer has some special technologies with different functions.

The main function of the perception layer is to collect data from physical IoT sensors, process the data and send it to the application layer via the transportation layer. RFID tags and WSN hardware are the two most important IoT hardware infrastructure resources in the perception layer and the entire IoT system as well. RFID system and WSN are fused in this layer which called EPC sensor network. RFID tags used to mark and identify objects. RFID technology realizes non-contact information transmission and identification of radio frequency signals through spatial coupling and realize the real-time tracing[15]. WSN hardware includes sensor interface, processing unit, transceiver unit and power supply[16]. This layer also contains sensors that provide the most primitive information, as well as embedded technology, which can make IoT devices more intelligent.

The transmission layer mainly transmits the collected information received from the perception layer to the application layer through the existing communication network. Cloud computing platforms, Internet gateways, switching and routing equipment operate by using some of the latest technologies (eg WiFi, LTE, Bluetooth, 3G, Zigbee, etc.) at this layer[17]. The network gateway aggregates between different sensors, filters and transmits data to act as an intermediary between different IoT nodes[18].

The application layer receives processed information from devices, analyzes the information and returns feedbacks to the devices. This layer can provide the computing platforms used to analyze the data and the applications designed based on the platforms to meet users' requests.

All three layers should guarantee the confidentiality, integrity and authenticity of the data.

2.2 Threats and Vulnerability

Most IoT network security challenges stem from inherent system vulnerabilities including firmware, hardware (devices), system applications, data, and network interfaces or ports[16]. However, it is difficult to directly perform security analysis on a great quantity of components included in a complete IoT system. Therefore, the three layer model can continue to be used to help us understand these threats and vulnerability in different layers. Some possible solutions are also attached.

2.2.1 Perception Layer

Internet of Things devices usually operate in an outdoor environment, which makes both the signal and the device itself have certain physical security risks. Moreover, RFID tag information security and information storage and processing security should be noticed at perception layer. The limited node resources and distributed organized structure at this level cause attacks as follows.

- **Physical attacks** To make the attacks working, the attacker should be physically close to the IoT system. As an example, the attacker can cause damage to the physical sensor, or even electronically interrogating the nodes to get access to sensitive information. In this case, it is necessary to hide the equipment in the public space as much as possible and enhance the safety awareness of the users. In order to cope with possible node tampering, that is, by physically replacing certain hardware parts to gain access and change sensitive information, or malicious code injection, it requires enhanced hardware security through the use of cryptographic coprocessors or anti-tampering technologies (eg, chip or memory protection, self-destruction, etc.)[14]

- **Impersonation** Careless design of authentication in IoT device also cause fatal vulnerability to the system[19]. Almost all webcam mobile applications do not provide strong authentication scheme like two-factor authentication, and most of them do not even enforce the users to change default password. The Mirai botnet attack, the largest DDoS (Distributed Denial of Service) attack was carried out based on the 61 weakest passwords [20]. Attackers use these Mirai-controlled devices to conduct DoS attacks on the network while finding and controlling new IoT devices with no/weak passwords. In this case, access control and authentication system need to be strengthened. Users also should be forced to set an initial strong password and develop a good habit of regularly changing passwords.

- **Data transit attacks** Attackers can obtain sensitive information of data in the traffic, due to the unsecure code in the device[21]. It was very surprising that the TRENDnet Webcam transmitted user login credentials in clear, readable text over the Internet, and attackers who obtained a camera's IP can easily get the login details from the traffic. Data encryption should be strengthened at this point.

2.2.2 Transportation Layer

In transportation layer, threats come from network transmission and wireless communication. According to [22], the open and heterogeneous architecture of an IP-based LTE network cause

more security threats compared to the 3G networks. The security weaknesses coming from this layer are summarized as follows.

- **Routing attacks** Attackers can get access to the intermediate nodes in the data transmitting path, inject malicious code and modify the right routing path.
- **DoS attacks** Due to the limited processing ability of the nodes, the attackers can cause denial of service by sending flood of packets to exhaust the resources.
- **Data transit attacks** Attackers can break the confidentiality and integrity during data transmit. Careless encrypted text can leak information of plaintext. The Man-In-The-Middle (MiTM) attacks also targets the transportation layer, as the result of faulty authentication process. An example for MiTM is the hacking of a Jeep Cherokee[23]. Hackers like this example can access and control the basic functions of the vehicle, such as braking, steering, and acceleration, which can be very dangerous. The best way to avoid man-in-the-middle attacks is to use a strong encryption method between the client and the server: the server verifies the client's request by verifying the digital certificate before the connection can be established.

2.2.3 Application Layer

Most IoT user interactive applications are based on the Web or mobile devices. Since Internet of Things still has no global policies and standards for managing interactions and application development, there are many issues related to application security. Different applications have different authentication mechanisms, which makes it difficult to ensure data privacy and authentication when all are integrated together[17]. The application design mainly uses the application programming interface (API) as the interface code to use PHP, Java, XML and HTML architecture. Unpatched APIs are vulnerable to attacks and the entire system suffers malicious attacks[16]. In addition, this layer serves as a window for connecting users so that its security is easily affected by human factors. The potential security threats within the application layer are as follows.

- **Faulty coding** Attackers can hack a computing platform as the same way they do to a website, using techniques such as SQL Injection, XSS etc. Amazon-owned Blink XT2 security camera systems were once hacked by attackers due to the unsecure code, and the attackers can remotely view the camera footage and listen to audio output. This requires application developers to consider security when they developing.
- **DoS attacks** Similar to the DoS attacks described in the transportation layer, attackers can destroy the availability of the service by sending large quantity of requests in a short

time. Besides, most IoT applications are designed based on Linux kernel, which has been found vulnerable to buffer overflow attacks. For instance, in CVE-2016-8658, attackers can cause a DoS attack via a long SSID Information Element in a command to a Netlink socket[24]. Also, in CVE-2016-5344, multiple integer overflows in the MDSS driver for the Linux kernel 3.x allows attackers to cause DoS attack using a large size value to exploit the buffer overflow vulnerability. Therefore, these applications have a high risk of being attacked by DoS and a powerful intrusion detection system is a must.

- **Malicious code injection** This attack happens when the application does not check the uploaded files carefully, and attackers can upload malware coded files including malicious code to the network. For instance, Hewlett Packard All-In-One computers once allowed the attackers to steal data through a company's network using a phone line and a fax number. Attackers can send coded files to the targeted network and decoded there[20].

2.3 Webcams

Among all IoT devices, webcam is one of the most popular IoT devices. Webcams will be the main research object in our project since they are perceived as poorly secured devices in particular when utilising Peer-to-Peer (P2P) protocols, enabling remote access of IP webcams (installed at home/office) from the Internet, by using smartphone applications.

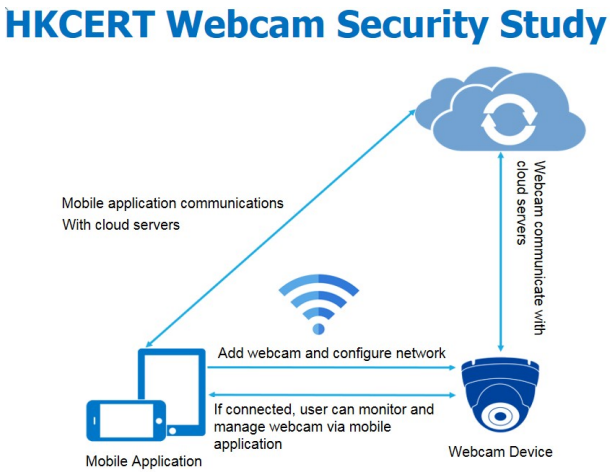

Figure 3: HKCERT webcam security study[19]. The figure shows main objects of research including mobile application security level, the degree of security of data transmission between the webcam, mobile application and cloud server and the security level of the network camera itself

.

An research conducted by Hong Kong Computer Emergency Response Team Coordination Centre[19] has assessed the security level of webcams available in Hong Kong, from all aspects of the operation of the webcam system in [figure 3].

It's easy to summarize the result according to the three-layer model.

2.3.1 Perception Layer

The main threat in the perception layer comes from the poor design for the authentication process. According to the survey, most webcams have open ports that can be exploited in the network. Users can login the device through these ports by the username and password. However, some can be logged in with the default username and password, and one webcam can even be logged in without login credentials. In addition, several webcams did not provide protection against brute force attack.

2.3.2 Transportation Layer

It is mentioned in the result of survey that most data transmissions between mobile applications and webcams are found unencrypted. Although mobile applications are forced to encrypt the data when connecting with the platform due to enforcement of HTTPS connections, there is no strict standard for the embedded code inside webcams. The manufacturers may consider the local networks as safe networks, which is not under most circumstances, and take few actions to protect data transmitted between the applications and webcams.

Some webcams also set their UPnP function enabled by default, which permits other devices discover the webcam on the network. Universal Plug and Play (UPnP) enables p2p connections between PCs, smart devices and wireless devices. It is a distributed, open network architecture, which uses TCP/IP and Web technology to achieve control and data transmission between network devices anywhere[25]. For example, if you are in the office and control the camera at home through the remote control application on your mobile phone, this process may use UPnP. Risk of UPnP is given in [figure 4]. UPnP can let internal devices being accessed from the internet without user instruction. However, most users have no knowledge about this function but their devices are exposed to other devices, leading to high probability of being attacked by hackers.

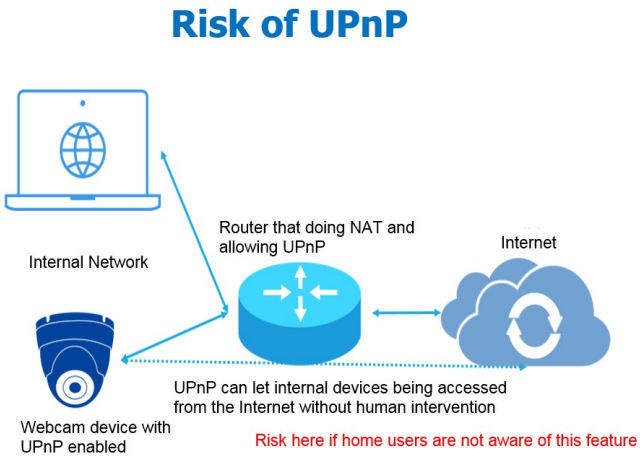

Figure 4: Risk of Upnp[19]. The internal network and devices using UPnP establish a gateway through network address translation (NAT) technology and share a globally unique public IP address. NAT help external programs obtain the shared global routable IP address of the UPnP capable NAT device and configure port mapping. At the same time, devices on the internal network can be accessed from the Internet. It is worth noting that turning on NAT does not mean using UPnP.

.

2.3.3 Application Layer

As mentioned above, there is enforcement in the security level of applications for IoT devices. Therefore, it is not easy for an attacker to exploit vulnerabilities in the applications themselves.

However, similar to the perception layer, weak authentication scheme brings the main threat in the application layer. According to the research, sixty percent of the webcams do not require users to set a complicated password for their accounts, and only one webcam featured good security practice of re-authentication. Apart from the requirement to the password, all webcam applications do not provide strong authentication schemes like 2FA.

3 Approaches Exploration

Traditional protection mechanisms, intrusion detection system for IoT systems are found too passive and therefore inadequate for evolving attacks. A common strategy for service provider is to patch the devices to protect against existing attacks, and there is no chance to detect new types pf hacking techniques. This problem can be conquered by Intrusion Detection System (IDS) that are designed based on mathematical models, which make it possible to detect

potential threats in traffic actively.

CIDF("Common Intrusion Detection Framework") [26], a working group created by DARPA in 1998, has come up with a general definition of IDS architecture that includes four functional modules, as shown in [figure 5].

- **E blocks ("Event-boxes").** This module consists of sensor elements, collecting data from the monitored environment and send the data to other boxes.
- **D blocks ("Database-boxes").** This module is a database, storing data from E blocks and prepare them for A blocks and R blocks.
- **A blocks ("Analysis-boxes").** This module is the core part of the whole system. A blocks analyze the information collected by E blocks and calculate how likely an event is malicious. The result will be sent to R blocks.
- **R blocks ("Response-boxes").** This module receives the estimation from A blocks, and if an event is labelled as malicious, R blocks executes a response to the event.

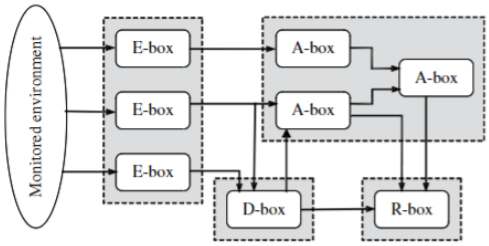

Figure 5: A general IDS architecture defined by CIDF, including four functional modules. Arrows indicate the flow of data from the monitoring environment to R-blocks through each module.

There are two most used types of IDS in real systems, rule-based IDS and anomaly-based IDS. The details of these two types of IDS will be given in each subsection.

3.1 Rule-based IDS

A rule-based intrusion detection system is a simple system using only IF/THEN programming to distinguish whether a request is normal according to artificial rules stored in a repository. A rule-based IDS maintains an extensive database of attack signatures, normally create by security experts. Then the system sniffs every packet passing through the monitored network, compares the packet to the signatures in the database, and raises an alert if a packet matches

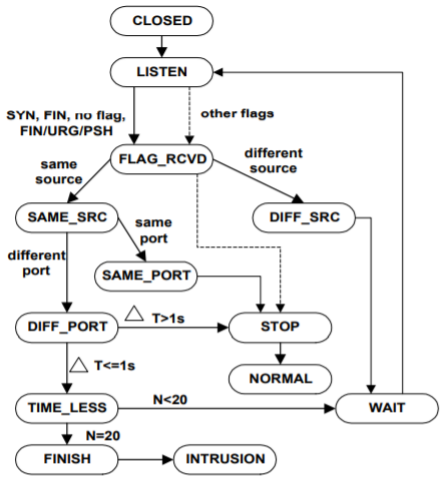

Figure 6: Detection state diagram of the rule-based IDS proposed by Urupoj, which shows how the system detects an intrusion based on given features and rules.

an attack signature.

The core part of this type of IDS is the construction of the signature database. It will be time-consuming if the database is too large, while a small database may lead to a low accuracy of detecting process. There are typically two methods to construct the signature database.

1. **Artificial database.** The signature database is totally determined by humans. Experts build the rules based on their knowledge and experience.
   Urupoj proposed an rule-based approach for port scanning attack in 2000[27]. In his paper, the system featured each packet as a five-tuple of parameters $K = (F, S, P, \Delta T, N)$, referring to the flags marking on the TCP header, the source IP address, the destination port number, the interval time of two consecutive packets and the number of suspecious packets, respectively.
   The detection state diagram is given below in [figure 6], showing the clear logic of this rule-based IDS.

2. **Algorithm-based database.** This type of database construction is completed by algorithms. Compared to the artificial database, the algorithm-based database is more reliable and generally shows a better complexity when facing new types of intrusions. A rule-based IDS has been proposed by Ojugo in 2012, which uses genetic algorithm (GA) to generate its signature database[28]. GA is a search method that finds an approximate solution to an optimization task. In his work, the system begins with randomly gener-

ated populations. The chromosome has a number of genes, whose quality presents the quantitative representation of each rule's adaption. When the population evolves, the stronger genes dominates the weaker ones and the model ends up with the fittest genes in the population, i.e. the best rules for the system.

In Conclusion, rule-based IDS is the earliest form of IDS in history, because it is the most intuitive method for humans. People create rules, and computers help us check the packets according to the rules. However, this type of IDS relies too much on human's knowledge and experience, and shows a low capability when detecting new types of attacks, while the speed of intrusion evolving can easily grow beyond our imagination. Although to some extend, rule-based IDS with algorithm-based database improve this situation, this type of IDS seems out-dated when facing our rapid developing environment.

3.2 Anomaly-based IDS

Compared to rule-based IDS, anomaly-based Network IDS (A-NIDS) focuses on estimating the "normal" behavior of a system. This type of IDS first builds a mathematical model for the system, then uses benign data to "teach" the model what kind of events are likely to be normal. Once the training is done, the model monitors every packet passing through the system, estimates the anomaly of each packet, and raises an alert if the packet is labelled as malicious.

A survey carried out by Garcıa-Teodoro explains that although there are different types of A-NIDS, in general their architectures are similar, including the following three modules, as shown in [figure 7].

- **Parameterization.** In this stage, the system collects data from monitored environment, and then transforms it into a pre-established form, like the five-tuple defined in Urupoj's model, for instance.
- **Training stage.** This stage is the most important part of A-NIDS, in which a detection model is built for the system, learning "normal" behaviors from benign data. This stage can be done in different ways and we will discuss each type of A-NIDS in the following paragraph
- **Detection Stage.** When the training stage has been finished, the detection is ready to detect anomaly in the network. It compares the parameterized data monitored in the environment to the normal data, and generates an alarm if the different exceeds a threshold.

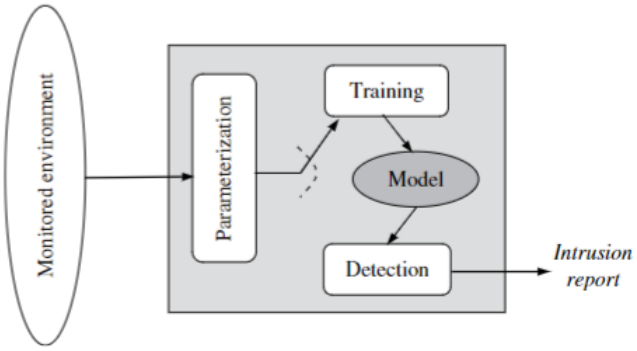

Figure 7: Generic A-NIDS architecture with four functional modules. The raw data collected from the environment should be first converted to a list of parameters. Then the embedded model uses this parameterized data to learn "normal" patterns of events. According the estimation from the model, the detection stage generates an intrusion report.

Depending on the different technologies of training stage in the architecture, A-NIDS can be further classified as statistical-based, knowledge-based and machine-learning based systems.

- **Statistical-based A-NIDS.** This type of A-NIDS estimates a traffic flow in a random pointview. Events like users logging in a system can be modelled as a Poisson process, for instance, then the system can analyze the time interval sequences between consecutive events, and estimate the possibility that the system is being attacked comparing to Poisson process properties.

- **Knowledge-based A-NIDS.** This type of A-NIDS is similar to the rul-based IDS with artificial database described in the previous subsection. The system sets up a database, storing "anomaly" behaviours in the traffic, and compares any monitored event to the data in this database. Once an event has been matched with a abnormal data, the system gives out an alarm.

- **Machine-learning based A-NIDS.** This type of A-NIDS uses machine-learning techniques to analyze the patterns of events. The system establishes an explicit or implicit model, provides labelled data for the model, and once the training has been done, the model is able to estimate the patterns of monitored events.

One early example for statistical-based A-NIDS is the multivariate host-based A-NIDS proposed by Ye et al.[29] in 2002. This A-NIDS used audit data from a UNIX-based host machine with the Polaris operating system, which has an auditing facility called Basic Security Module. The BSM monitors the instructions executed by the processor, records the events as a combi-

nation of event type, user ID, group ID, process ID, session ID and so on. After capturing the data, this BSM builds a long-term profile of normal activities, and uses both T^2 test and X^2 test to estimate the anomaly.

Li et al.[30] has proposed an A-NIDS technique to monitor the operation of IoT devices and detect cyber attacks. They adopted linear regression, NNs and recurrent NNs (RNNs) models to model the IoT system, and then measured the anomaly score considering the differences between observations and predictions.

Finally, many network gateway and router devices that can carry NIDS do not have enough memory or processing power to train the model. More importantly, training in a supervised manner means that labeled network traffic must be labeled and the model needs to be updated manually from time to time[31]. Kitsune[31]is a novel NIDS based on ANN, which is online, unsupervised and efficient,as shown in [Figure 8]. Kitsune has a set of small neural networks (autoencoders) that are trained to imitate (reconstruct) network traffic patterns, and their performance will gradually improve over time.

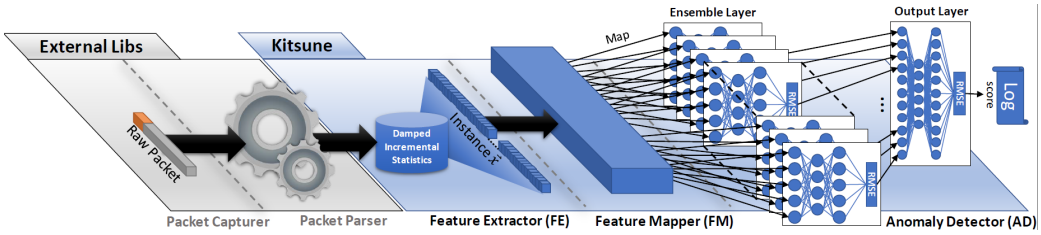

Figure 8: The architecture of Kitsune[31]. First, a feature extraction framework tracks the patterns of every network channel and extracts a feature vector for each packet. Next, the features are mapped to the visible neurons of an ensemble of autoenoders (KitNET). Then each autoencoder reconstruct the instance's features and computes the reconstruction error in terms of root mean squared errors (RMSE). Finally, the RMSEs are forwarded to an output autoencoder, which acts as a non-linear voting mechanism for the ensemble.

This article will also be mentioned in the following dataset section since it provides a good IoT dataset.

3.3 summary

Rule-based IDS set up a database storing "rules", that are generated by human or machine-learning models, and compare the monitored events to the rules to estimate the anomaly.

Similar to the idea of the blacklist firewall, this type of IDS knows how malicious events "look like", and if an event seems suspicious, the system intercepts it and generate an alarm. On the other hand, Anomaly-based IDS spends more effort on detecting unseen attacks. This type of IDS try to set up a "normal" pattern, and treat any event that does not match this pattern as an attack. From this point of view, there is an analogy between A-NIDS and whitelist firewall. They allow only "familiar guests" to pass through, but intercept every unseen or unfamiliar events, until further analysis.

There are advantages and drawbacks in both types of IDS. Rule-based IDS are simple and cost less time and computation, while anomaly-based IDS provide higher accuracy and flexibility. The environment and the requirement of security level decides which type of IDS should be used in the system. Rule-based IDS fit small devices like webcams, which have low computing ability, and require lower security level, while anomaly-based IDS can be used in cloud platforms, which provide stronger capability of computation, and can be the target of millions attackers.

4 Dataset

Websites such as UC Irvine Machine Learning Repository (UCI)[3] provide a large number of original Pcap file sets related to IoT devices and tagged datasets after processing. These data sets will be used in conjunction with the data captured by ourselves to analyze related vulnerabilities. Further, we can compare them.

4.1 Dataset Collection and Description

First, we will build a test environment using webcams and routers (possibly via hotspots on laptops). The specific hardware support and experimental steps will be described in the project proposal. Then use wireshark to capture the network traffic generated when accessing the webcam remotely from the mobile web application to get the pcap file to get utilised protocols and exchanged data. After this, we will try to carry out some common attacks such as DOS and password blasting on the experimental equipment, and even try to use Mirai's source code [4]to test and verify the vulnerabilities in webcams and get more abnormal network flows.

As a supplement, we have currently found two datasets for IoT devices that would be used in our project:

[3]http://archive.ics.uci.edu/ml/index.php

[4]https://github.com/jgamblin/Mirai-Source-Code

Detection_of_IoT_botnet_attacks_N_BaIoT Data Set[32] provided by Meidan et al.[33] solves the problem of lack of public botnet datasets, especially for the Internet of Things. It is based on real traffic data, which was collected from 9 commercial IoT devices including webcams that were infected by Mirai and BASHLITE. It contains 7062606 instances with 117 feature labels and the malicious data can be divided into 10 attacks carried by 2 botnets.

Kitsune Network Attack Dataset Data Set[34] contains nine different network attacks on a commercial IP-based surveillance system and an IoT network[31]. There are 27170754 instances in total including attack types following. Several of these attack methods can be tried when we collect our test data.

- Recon
  - OS Scan: Use Nmap to scan the hosts and operating systems on the network to find possible vulnerabilities.
  - Fuzzing: Search for vulnerabilities in the camera's web server by sending random commands to the camera's cgis.
- Man in the Middle
  - Video Injection: Inject recorded video clips into the live video stream.
  - ARP MitM: Intercept all LAN communications through ARP poisoning attacks.
  - Active Wiretap: Intercept all LAN communications through active wiretapping (bridge) secretly installed on an exposed cable.
- Denial of Service
  - SSDP Flood: Send spam emails to the camera to UPnP advertisements, thereby overloading the DVR.
  - SYN DoS Hping3: Disables a camera's video stream by overloading its web server.
  - SSL Reneg: Disable the camera's video streaming by sending many SSL renegotiation packets to the camera.
- Botnet Malware
  - Mirai: Infects IoT with the Mirai malware and then scans for new vulnerable victims network.

In addition, a network intrusion detection system (NIDS), Kitsune, which is proposed in the original article[31] ,using an ensemble of neural networks can also provide a good reference. It

has been described in the previous section.

All the data above captured in network will be provided 1) csv file where each row is a network packet with the corresponding label [benign, malicious]; 2) The original network capture in truncated pcap format. Therefore, we can either use the ready-made csv data set for assessment directly or analyze the possible vulnerabilities in protocols by observing the original pcap file features.

4.2 Data Processing

The main problem of data processing this time in the project is to convert the pcap file obtained by capturing the packages into a csv file for easy processing. We found a software that can create a network intrusion dataset in CSV format, Customizable Network Intrusion Dataset Creator[35]. It can run and create a data set on a local server, or read PCAP directly from other sources and convert it to CSV format based on selected attributes. However, we currently found that some of the dependent packages could not be installed when we tried to use it initially. We will find a way to solve this problem in the subsequent project implementation process. We also found an alternative solution. There is another Pcap_Features_Extraction tool[36] can extract 26 features from pcap files and save them in a csv file.

After obtaining the csv data set, data preprocessing will be routine including method such as visualization of data, feature engineering and vectorization steps as usual.

5 Project Proposal

A detailed project plan has been formulated under the guidance of the previous content.

5.1 Hardware and Software Support

Main devices we plan to use in this project:

- **a Webcam for each member:** The item name is "ieGeek Security Outdoor Camera 1080P Waterproof Home Security CCTV WiFi Surveillance Outdoor Bullet IP Camera With 25m Night Vision, Remote Viewing Motion Detection and Push Alerts for Android/iOS/PC" [5] Main functions has been described by the name. It supports a wide range of web connectivity protocols (port forwarding, UPnP, P2P, etc.)

[5] https://www.amazon.co.uk/dp/B073GQ8T2L/ref=psdc$_1$330828031$_t$1$_B$078Y446LX?$th = 1$

- **a Laptop for each member:** The laptop will be used as a router. Although they are mainly used to capture WLAN data and write reports, the coding work will be carried out in Colab, thus Google 's server will be the hardware for our coding work.
- **some cables**

Main softwares and platforms we plan to use in this project includes **Wireshark3.2.3** will be used to capture data. Coding work will be mainly on the **Colab** with **Google Drive** using **Python3**. At last, this project will be collaborated in **Github** [6] .

5.2 Major Units with Work Division and Time Line

There is a project plan with some additional notes attached. The project has been split into sets of distinct units with time milestones. Since the number of members in our group is the smallest prime number, most things will be done by the whole group except the attempts of different machine learning models. However, we still try to detail everyone's work as much as possible.

In our schedule, we will plan our progress on a weekly basis. Monday, June 1st is the beginning of week1, and the date before this is marked as week0. It is worth noting that there will be a final exam in the first two weeks of June. We have also taken this into account in the project plan. Only a few tasks are arranged in week1 and week2. Most of the experimental work will be completed in week3 and week4. This would be followed by two weeks of code and evaluation work. Finally, the report writing, which is planned to be completed in week8. We would submit our first draft by then.

Here is the detailed plan.

5.2.1 Stage1-Preparatory Work

The first is the preparatory work, including the accumulation of background knowledge and experimental preparation. Actually most of the work in this stage has been completed by the time of writing words here.

Unit1, week0: IoT landscape assessment from the perspective of cybersecurity
This unit includes a review of IoT key technologies in the commercial and enterprise/industrial sectors; Review of major IoT vulnerabilities, threats and security issues documented in available online literature and papers, especially for webcams; Review of major IoT cyber-attacks

[6] https://github.com/YeweiYuan/Iot-Security-Project

in the last 5-10 years. All relevant literature must be read by all staff and provide notes. At the same time, this part of the report is also completed during this period.

As for the writing of this part report, IoT technology section would be done by Yewei, the threat and vulnerability parts would be written together, and analysis of the webcam system section would be done by Kexi.

Unit2, week0: Dataset (get online) and model preparation
This unit includes dataset collection and pro-processing and finding and current popular network flow analysis methods such as suitable models. Yewei would collect available data sets and upload to our Github then conduct preliminary tests on them. Kexi would be responsible for collecting IDS methods, especially those based on statistics and deep learning.

Unit3, week0: Equipment preparation
The project would be created on Github, Colab and Overleaf. Dr.Daniel Lawson and Mr.John Newby would mail all required equipment. After the camera arrives, we would learn how to use it, including downloading the matching mobile phone application.

Milestone: Sunday, 31 May 2020

5.2.2 Stage2-Lab Work and Coding

The main work in the mid-term is experimental work, we will obtain experimental data at this stage. Most of the work in this stage would be done in June.

Unit4, week1 and week2: Environment construction
We would build a mini-test environment including a router with our IP cameras. This work would be done by all staff, Yewei and Kexi.

Unit5, week3 and week4: Data collection
Capture network traffic generated by the web camera, especially when the web camera is remotely accessed from a mobile web application. At the same time, the data is processed into a format that is convenient for our model to use. This work would be done by all staff, Yewei and Kexi.

Unit6, week3 and week4: Preliminary data packet analysis
Investigate P2P communication protocols and potentially others by using open source packet

capture tools Wireshark. Analyse utilised protocols (e.g. HTTP/HTTPS), security level, exchanged, information. This work would be carried out simultaneously with unit5. This work would be done by all staff, Yewei and Kexi.

Milestone: Sunday, 28 June 2020

5.2.3 Stage3-Assessment and Report

The last is the evaluation stage after the experiment. The final stage work would be done in July.

Unit7 , week5: Statistical assessment We would try to use statistic method to assess the vulnerability in this unit. The specific method would be determined by the results of unit2. This work would be done by all staff, Yewei and Kexi.

Unit8 , week6-8: Machine learning assessment Yewei and Kexi would use a different machine learning / deep learning model for assessment respectively. Then make comparison. The specific method would be determined by the results of unit2. The code will be completed on Colab then uploaded to Github.

Unit8 , week8: Writing report While other units are conducting, we are gradually writing some draft reports. So the final report will not take much time to complete. In fact, this report was completed within three days. This work would be done by all staff, Yewei and Kexi on overleaf.

Milestone: Sunday, 25 July 2020

5.3 Risk Analysis and Mitigation Plans

Risk	Likelihood	Severity	Prevention
Unable to or spend too much time on environment set up	2	4	Read related write ups, try different methods of connecting
Unable to capture or identify the packets in the local network	1	4	Learn how to use Wireshark or similar tools, and the identifier of packets transmitted between webcams and applications
Unable to use collected data sets	3	3	Read relevant papers about the data sets, change the format if necessary
Fail to build a statistical model or machine learning model for the project	2	5	Plan how we build our model in advance, provide a model outline if necessary
Fail to use colab to work remotely, or unable to set up proper environment on it	3	2	Try to work out why, if the problem cannot be solved, try to use other platforms to work remotely
No prior experience of creating a similar model, not knowing which parameters should be chosen or which data set should be used	2	2	Collect information from relevant works, try different parameters and find the best solution for the project
Unable to launch cyber attacks due to the lack of computer knowledge	4	1	Talk to students in CS department, or read write ups and follow others' experience step by step
Cannot understand and use existing attacking code like Mirai	4	2	Comment on the source code, chat with each other and try to learn the logic of the code

6 Conclusion

By reviewing the current literature, we understand the IoT system structure and get a general landscape of current IoT cybersecurity. At the same time, main related vulnerability and solutions are given. Machine learning is considered to be a way to analyze and solve IoT vulnerabilities, and we plan to apply deep learning and statistic model to our security assessment. Moreover, we got two data sets that fully fit this project, and developed our own experimental plan for making a network streaming data set utilising webcams.

In order to ensure that the project is completed on time, we have developed an adequate and reasonable time schedule. Finally, we absorbed the above work to formulate a detailed project plan, including experimental equipment, division of labor, and risk assessment.

References

[1] URL: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.

[2] Manos Antonakakis et al. "Understanding the mirai botnet". In: *26th {USENIX} Security Symposium ({USENIX} Security 17)*. 2017, pp. 1093–1110.

[3] Ben Heubl. *How to hack an IoT device*. URL: https://eandt.theiet.org/content/articles/2019/06/how-to-hack-an-iot-device/.

[4] Rolf H Weber. "Internet of things–Need for a new legal environment?" In: *Computer law & security review* 25.6 (2009), pp. 522–527.

[5] Vangelis Gazis et al. "A survey of technologies for the internet of things". In: *2015 International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE. 2015, pp. 1090–1095.

[6] Nitin Naik and Paul Jenkins. "Web protocols and challenges of web latency in the web of things". In: *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE. 2016, pp. 845–850.

[7] Nitin Naik. "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP". In: *2017 IEEE international systems engineering symposium (ISSE)*. IEEE. 2017, pp. 1–7.

[8] Soma Bandyopadhyay and Abhijan Bhattacharyya. "Lightweight Internet protocols for web enablement of sensors using constrained gateway devices". In: *2013 International Conference on Computing, Networking and Communications (ICNC)*. IEEE. 2013, pp. 334–340.

[9] Zach Shelby, Klaus Hartke, and Carsten Bormann. "The constrained application protocol (CoAP)". In: (2014).

[10] Alessandro Ludovici, Pol Moreno, and Anna Calveras. "TinyCoAP: A novel constrained application protocol (CoAP) implementation for embedding RESTful web services in wireless sensor networks based on TinyOS". In: *Journal of Sensor and Actuator Networks* 2.2 (2013), pp. 288–315.

[11] Ngoc Son Han. "Semantic service provisioning for 6LoWPAN: powering internet of things applications on Web". PhD thesis. 2015.

[12] Ilya Grigorik. "Making the web faster with HTTP 2.0". In: *Communications of the ACM* 56.12 (2013), pp. 42–49.

[13] Rüdiger Schollmeier. "A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications". In: *Proceedings First International Conference on Peer-to-Peer Computing*. IEEE. 2001, pp. 101–102.

[14] M. Frustaci et al. "Evaluating Critical Security Issues of the IoT World: Present and Future Challenges". In: *IEEE Internet of Things Journal* 5.4 (2018), pp. 2483–2495.

[15] Handong Zhang and Lin Zhu. "Internet of Things: Key technology, architecture and challenging problems". In: *2011 IEEE International Conference on Computer Science and Automation Engineering*. Vol. 4. 2011, pp. 507–512.

[16] Knud Skouby, Reza Tadayoni, and Samuel Tweneboah-Koduah. "Cyber Security Threats to IoT Applications and Service Domains". In: *Wireless Personal Communications* (May 2017). DOI: 10.1007/s11277-017-4434-6.

[17] R. Mahmoud et al. "Internet of things (IoT) security: Current status, challenges and prospective measures". In: *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*. 2015, pp. 336–341.

[18] Mario Frustaci et al. "Evaluating critical security issues of the IoT world: Present and future challenges". In: *IEEE Internet of things journal* 5.4 (2017), pp. 2483–2495.

[19] Hong Kong Computer Emergency Response Team Coordination Centre. *IoT Device (Webcam) Security Study*. 2019. URL: https://www.hkcert.org/my_url/en/guideline/19012401.

[20] Rudra Srinivas. *10 IoT Security Incidents That Make You Feel Less Secure*. 2020. URL: https://www.cisomag.com/10-iot-security-incidents-that-make-you-feel-less-secure.

[21] *The 5 Worst Examples of IoT Hacking and Vulnerabilities in Recorded History*. 2017. URL: https://www.iotforall.com/5-worst-iot-hacking-vulnerabilities/.

[22] Sabina Baraković et al. "Security issues in wireless networks: An overview". In: *2016 XI International Symposium on Telecommunications (BIHTEL)*. IEEE. 2016, pp. 1–6.

[23] Christian Simko. *Man-in-the-Middle Attacks*. 2016. URL: https://www.globalsign.com/en/blog/man-in-the-middle-attacks-iot.

[24] Knud Skouby, Reza Tadayoni, and Samuel Tweneboah-Koduah. "Cyber Security Threats to IoT Applications and Service Domains". In: *Wireless Personal Communications* (May 2017). DOI: 10.1007/s11277-017-4434-6.

[25] B. A. Miller et al. "Home networking with Universal Plug and Play". In: *IEEE Communications Magazine* 39.12 (2001), pp. 104–109.

[26] S Stainford Chen, B Tung, and D Schnackenberg. "Common intrusion detection framework". In: *Proc. of Information Survivability Workshop*. 1998.

[27] Urupoj Kanlayasiri, Surasak Sanguanpong, and Wipa Jaratmanachot. "A rule-based approach for port scanning detection". In: *Proceedings of the 23rd electrical engineering conference, Chiang Mai Thailand*. 2000, pp. 485–488.

[28] AA Ojugo et al. "Genetic algorithm rule-based intrusion detection system (GAIDS)". In: *Journal of Emerging Trends in Computing and Information Sciences* 3.8 (2012), pp. 1182–1194.

[29] N. Ye et al. "Multivariate statistical analysis of audit trails for host-based intrusion detection". In: *IEEE Transactions on Computers* 51.7 (2002), pp. 810–820.

[30] F. Li et al. "System Statistics Learning-Based IoT Security: Feasibility and Suitability". In: *IEEE Internet of Things Journal* 6.4 (2019), pp. 6396–6403.

[31] Yisroel Mirsky et al. "Kitsune: an ensemble of autoencoders for online network intrusion detection". In: *arXiv preprint arXiv:1802.09089* (2018).

[32] URL: `http://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT#`.

[33] Yair Meidan et al. "N-baiot—network-based detection of iot botnet attacks using deep autoencoders". In: *IEEE Pervasive Computing* 17.3 (2018), pp. 12–22.

[34] URL: `https://archive.ics.uci.edu/ml/datasets/Kitsune+Network+Attack+Dataset`.

[35] Nadun Rajasinghe, Jagath Samarabandu, and Xianbin Wang. "INSecS-DCS: A Highly Customizable Network Intrusion Dataset Creation Framework". In: *2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE) (CCECE 2018)* (2018).

[36] URL: `https://github.com/lucadivit/Pcap_Features_Extraction`.