A Secure Network Switch for IoT Devices using Whitelist Approach

Porapat Ongkanchana

03-170425

A thesis presented for the Bachelor Degree

Department of Information and Communication Engineering
The University of Tokyo, Japan
February 8, 2019

Abstract

In this paper, we have discussed the possibility and the capability of IoT devices, it is undeniable that IoT devices will become a part of human society. However, IoT system security is considerably weak and the risk of problem like privacy intruding, spying, data stealing, occurring is high. In order to help solving the IoT security problem, we have proposed the security system using the whitelist method and switch-level security. The system has three operation stage: Preparation, Analyzation and Operation stage. System traffic data is captured during Preparation stage. In Analyzation stage, the system discovers all IoT devices in the system, and examine their communication pattern to extract the whitelist. During Operation stage, only packet from hosts is the whitelist is accepted. We have conducted an experiment to find a way to implement the mentioned functions. We found that the best way to discover devices is to analyze ARP request packet. The secure hosts of each device are the host which our devices initiate the connection to. We have implemented filtering function using "iptables" command. The result showed that this approach worked well with the tested IoT subject, discovering all devices and extracting most of device's secure host.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

IoT, the internet of thing, is a concept of implementing commutable daily objects. These objects are connected to the internet, allowing the transmission of gathered information and controller commands to be done anytime, anywhere. Problems that used to take time and effort now be done easier and faster because of IoT [1][2].

IoT technology is developing and growing at the exponential rate. The global economic impact of IoT is estimated to reach trillion dollars by 2025 and more than 50 billion devices are expected to be deployed in 2020 [3]. Many of world leading technology companies are pouring their resource into developing the future of IoT [4]. It is natural to think that IoT will soon be integrated into a part of our life.

IoT applications can be used in almost every aspects of our life. For example; in healthcare system, some of patient's medical asset information can be collected through a smart wearable, reducing tasks of medical staff. In electricity power system, "Smart grid" implementation allows a better energy management and more durable against blackout. "Smart house", where your house can automatically turn off the light when nobody is at home. This is just a tip of an iceberg of what IoT can do for us.

With the number of IoT device and its application growing, our world has never been more comfortable, however it has also never been scarier. IoT device has access to our privacy information, if this information falls into the wrong hand, it can lead to privacy breaching, data forging or even worse a matter of life and death[5][6]. The security of IoT is something we really need to put our attention to.

In 2018, it is reported that "Telnet attack" is the most frequent attack on IoT system, followed by "SSH attack"[7]. The telnet attack and SSH attack is an attack vector that exploit user's behavior of not changing device's default username and password. Attackers try to get device's root permission by brute-forcing all user, password combination. If adversary has acquired device's root permission, he can use that IoT device as their desire. This means accessing to all data in the device, installing and uninstalling any software, downloading and uploading any privacy data, attacking target servers or even spreading malware to other devices. The infamous "MIRAI" virus also used telnet to spread over six hundred thousand devices causing one of the biggest DDOS attack ever in the history of mankind [8].

Vulnerabilities in IoT device are existed and in much needed for the solution. For the better world and for the better tomorrow, we can not allow the IoT devices to get compromised. The pursue for the improvement of IoT devices security is critical.

1.2 Purpose

The aim of this paper is to construct a system that can secure user's IoT devices, protect them from various attack surfaces and alleviate the troublesome task of system management.

1.3 Outline

In chapter 2, we described technologies related to this research along with previous attempt to secure IoT system. We presented our IoT security system in chapter 3. In chapter 4, we conducted various experiment to investigate system's functions performance, along with how to improve it. Chapter 5 is this thesis summary and discussion.

Chapter 2

Related Technology

2.1 IoT System Architecture and Its Security Threat

There are many framework architectures used in IoT system. The simplest and most well-known model is the three-layer model (figure 2.1): perception layer, network layer, application layer.

- **Perception Layer** or recognition layer: This layer's main responsibility is to percept useful data from environment, translate those analog data into digital and send the data to the server. The devices in this layer are: WSN node, RFID label, smart sensor. (We mentioned the threats of perception layer in section 2.2)
- **Network Layer** : The network layer of IoT is established on top of current internet structure. Its main features are conveying collected data, processing data and also providing a storage to those data. Because of its long history, the security status of this layer is relatively safe. However, threats like computer virus, network congestion causing the latency or unavailable in the system, Man-in-the-Middle attack can occur, and security mechanism is something that should not be ignored [9].
- **Application Layer** : The application layer uses the precepted data to create services. The range of services which IoT can provide are impressive: authentication, real time data visualization, smart house, smart building, automatic temperature controller and more. The main threat in this layer is the application itself. A software implemented by ignorant developers can be exploited to invade user's privacy, cause damage to IoT devices or even harm users [9].

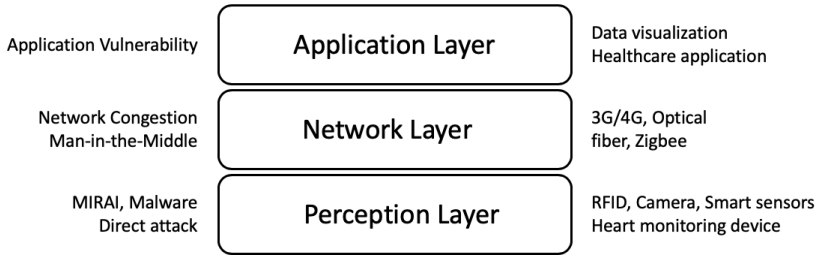


Figure 2.1: 3-layers Architecture

2.2 Smart Device

Smart device is an electronic device, connected to other devices or networks. It is generally deployed at the perception layer of IoT system and used for collecting environment data or performing task from the command center. The characteristics of smart device can be summarized as below:

- **Deployed into the physical environment:** For example, electricity consumption meter is placed around electric cable to tracking how much electricity spent. Temperature meter embedded into the wall. Tsunami detection sensor located along the sea course. These devices are vulnerable to direct-attack. For example, using USB cable to connect into the system, and installing malicious software.
- **Expected to operate without user's recognition:** After the initialization, devices are often deployed sparsely into outdoor environment. Normally, if no unexpected behavior has occurred, devices are simply left there without any attention.
- **Low computational power:** In order to reduce the manufactured cost, these devices can only operate with a very limited set of functions. For example, smart sensor has only sensing and transmitting function. The device's computational power is insufficient to perform task like encryption.
- **Communicate with a limited number of hosts:** IoT Smart device is different from our daily smart widget, it only needs to communicate with host we specified.

2.3 Whitelist and Blacklist

Whitelisting and Blacklisting are network filtering techniques. Blacklist is a list contained with IP or domain name of malicious hosts, while whitelist is a list of secured hosts. A whitelist filter would allow all hosts from its list to communicate with device's network. A blacklist filter drops all packet from its list.

Blacklist is normally used, when there is no specific communication pattern in the traffic. However, when communication pattern can be identified, it is more efficient to use whitelist.

The filtering is normally performed at layer 3 of the network, on the router. The wellknown firewall uses blacklist as its filtering protocol.

2.4 Related Researches

The security issues of IoT system arise from its heterogeneous nature. There was a time when IoT was developed with neither actual definition nor the standard. The companies which foresee the prosperity of IoT industry, developed and provided services as fast as they can, without paying much attention on the security system. The challenges of securing IoT are various: Authentication, Privacy protection, Access control, Cryptosystem and Security protocol, Software Vulnerability, Malware protection [10][11].

A well implemented authentication method can help securing the privacy of IoT system, while protecting the integrity of data. Many researchers have tried to create a proper authenticate system for IoT system. M. Saadeh *et al.*[12] have classified authentication

methods into 4 categories: Centralized, Distributed, Hierarchical and Flat. Each method has its own benefit, the selection of authenticate system depends on which kind of attack user wish to hinder. However, authentication do not focus on the security of the device, hence it would not work if the authenticated devices are compromised by adversaries. The hacked devices can still authenticate into the server and add any data.

The main concern about IoT system is the privacy. We place these smart devices closer to ourselves than any other kind of devices: from smart watch, security camera to even heart rate monitoring. The privacy breach, data leakage, is something that cannot allow to happen. G. Kalogridis *et al.* [13] have proposed a model to hind the electric consumption data in smart mater, while T. Song *et al.* [14] have introduced a scheme of data transmission with chaotic-generated symmetric key and secret key, guaranteeing the confidentiality and integrity. The privacy preserving data aggregation scheme has been presented by P. Yang *et al.* [15], their approach can protect the privacy of data even when compromised nodes are available in the system.

With the heterogeneous nature of IoT system, it seems implausible to create the system which can protect IoT system from all possible attack surfaces. C. Liu *et al.* [16] challenged this believe, and proposed that IoT security system should have an adaptive feature like immunity system. They have proposed a model which deal with the security threats according to its detector. J. Pacheco and S. Hariri [17] also came up with a similar idea. They have inspected smart house/smart building sensor's transmitted data, then used Anomaly Behavior Analysis (ABA) to identify the abnormality and perform the right recovery actions.

Most of previous researches seem to focus on data confidentiality, while left the integrity and the availability unresolved. We believe that the key to improving IoT system's integrity and availability is lied within the IoT device. Hence, in this reaseach, we want to contribute in the improving of IoT security by finding a way to secure IoT devices.

Chapter 3

Proposed System

In this chapter, we presented our novel-approach of how the IoT securing system should operate. However, we didn't complete the system. We only implemented, tested and evaluated the performace of each funtions the system used, under the assumption that if these functions perform well, then the system itself should also perform well too.

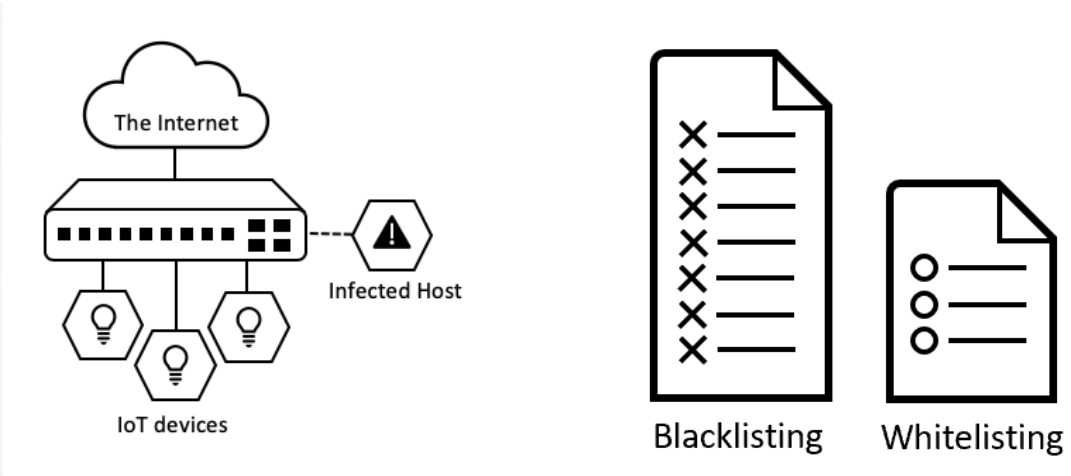
3.1 System requirements

This research aims to improve the security of smart device (especially the smart sensor) in IoT system. The security system we want to implement must have the following requirements.

- **Guarantee device safety:** The focus of the system is to secure the perception layer of IoT system, the sensor.
- **Independent from device:** System should be able to operate without any additional setups on IoT devices. This design help reducing the additional work when dealing with a large-scaled IoT system.
- **Can implement on top of an existed system:** Users can easily install system without having to change their network topology.
- **Do not affect system performance:** After system is added to the workspace, it should not impede on the performance: system latency, packet drop.
- **Reduce system manager's task:** Normally, the system manager can investigate the system integrity by looking its traffic pattern, system performance, CPU temperature and more. However, considering that system manager can come and go, without well written document, it is quite burdensome for user, or new manager to understand the system by themselves. We believe that the system must perform partially automatic without any help from user.

3.2 Idea behind the proposed system

We wanted to create the system with all characteristics mention above. The approach we came up consists of two main idea: Switch level security and Whitelist.



(a) LAN Level Security (b) Whitelist

Figure 3.1: System's key characteristics

3.2.1 Switch level security

Switch level security is securing IoT at the lower level of network protocol. The security measurement is done outside of the edge device, between the internet and device (figure 3.1a). The benefit of using this method is that there is no need for additional configuration at edge devices. Moreover, it is security at low level where transmission protocol is more restricted, therefore can be managed more easily. Another benefit of layer 2 security is even when malicious host or infected host has entered the local area network, it can prevent the infection from spreading.

3.2.2 Whitelist

Consider that the number of hosts, sensor normally connect to is limited: NTP (Network Time Protocol) server for clock synchronization, DNS (Domain Name System) Server for translating between host name and IP address, DHCP (Dynamic Host Configuration Protocol) server for dynamic IP assigning, HTTP (Hypertext Transfer Protocol) server for its application usage and vender specific communication protocol. Whitelist approaching is considerably more ideal in IoT sensor.

3.3 System Architecture

System infrastructure is shown in figure 3.2. The operation of this system is divided into 3 stages: Preparation stage, Analyzation stage and Operation stage. Preparation stage and Analyzation stage would be performed at the system's initialization, while Operation stage is used after that for the rest of system operating period.

3.3.1 Preparation Stage

In this stage, all packet going through switch would be captured. This system is developed under the assumption that during Preparation Stage, the network, edge devices, servers, middleware is secure and no malicious software has entered into the system yet.

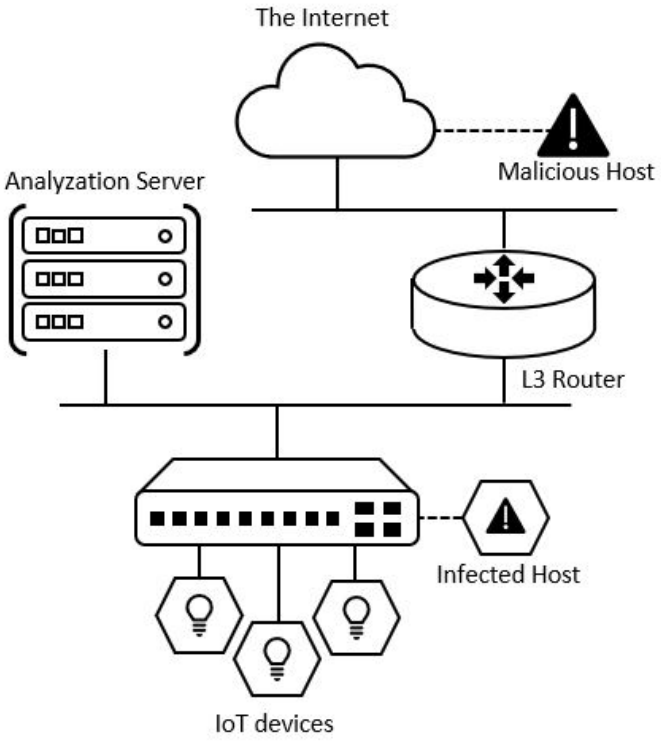


Figure 3.2: System Architecture

3.3.2 Analyzation Stage

In Analyzation stage, data collected by network plane is then further investigated to find.

1. All devices in the LAN, **Host discovery**.
2. All hosts, each device needs to operate, **Whitelist Extraction**.

Host Discovery

We believe that it is possible to find all devices in LAN by analyzing captured ARP (Address Resolution Protocol) packets.

Whitelist Extraction

Whitelist Extraction program use packet data to create whitelist profile for each discovered host. In this research, we use rule-based algorithm to extract whitelist.

Definition of Secured host and Whitelist Rule In this research, we define secure host as "host that device needs in order to fully operate" and we assume that secure host can be extracted using the following rule.

"Only IP addresses to which the device initiates connection is considered secured host"

3.3.3 Operation Stage

After Analyzation Stage, switch would start filtering network traffic, letting device to only communicate with hosts in its whitelist, guarantee the integrity of IoT system.

3.3.4 System Operation Flow

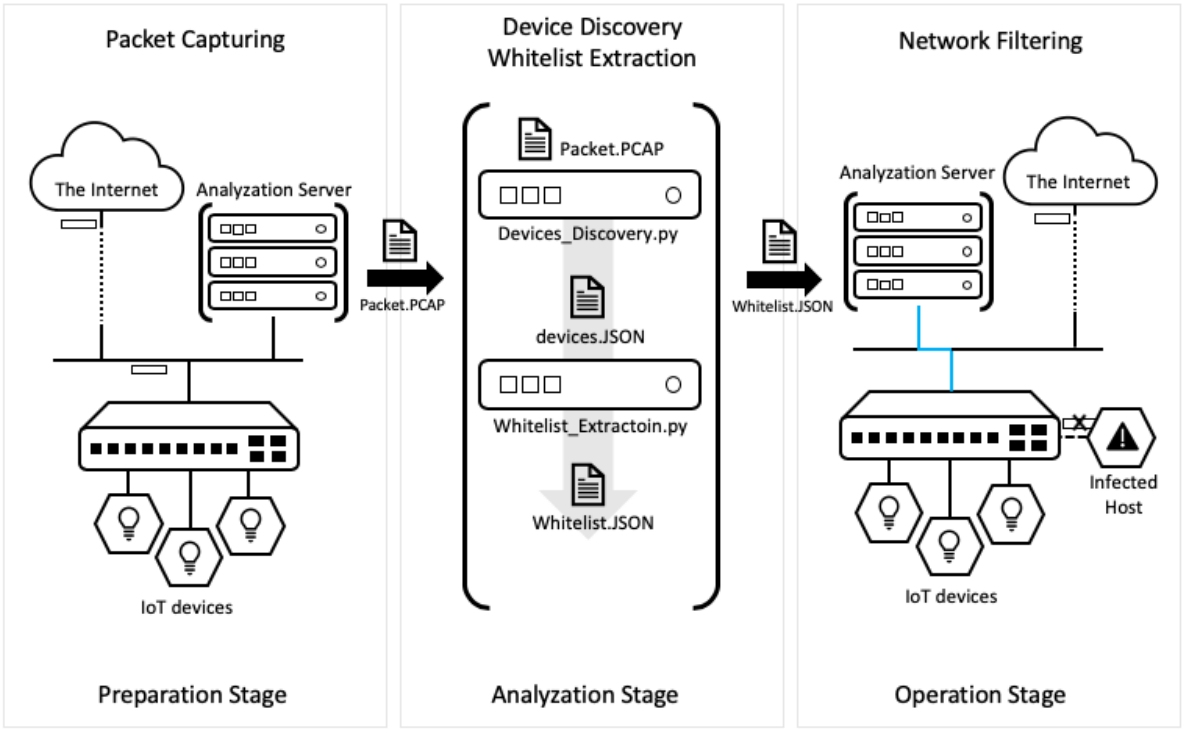


Figure 3.3: System Overview

First, we start capture devices' packet under assumption that there is no on-going attacks or infected host in the LAN. After packet data is sufficiently captured, we pass gathered packet to analyzation server, where packet is inspected to find devices in the system. Then we create a whitelist for each device and start filtering traffic according to it.

Chapter 4

Experiment

Various experiments were conducted to find the best way to implement functions mentioned in section 3: device discovery, whitelist extraction and network filtering. In the next section, we would like to explain the IoT systems used as the test subject in this experiment.

4.1 Experiment Environment

Load leveling is a system that usually found in renewable sources electricity generator. System's function is to balance the excess energy, for example in solar panel case; it can either store or sale the excess electricity produced during day time, while discharge electricity during night.

eroom is an IoT system that connects load levelling with battery, control panel, air conditioners and solar panel. The network topology of eroom IoT is shown in figure 4.1.

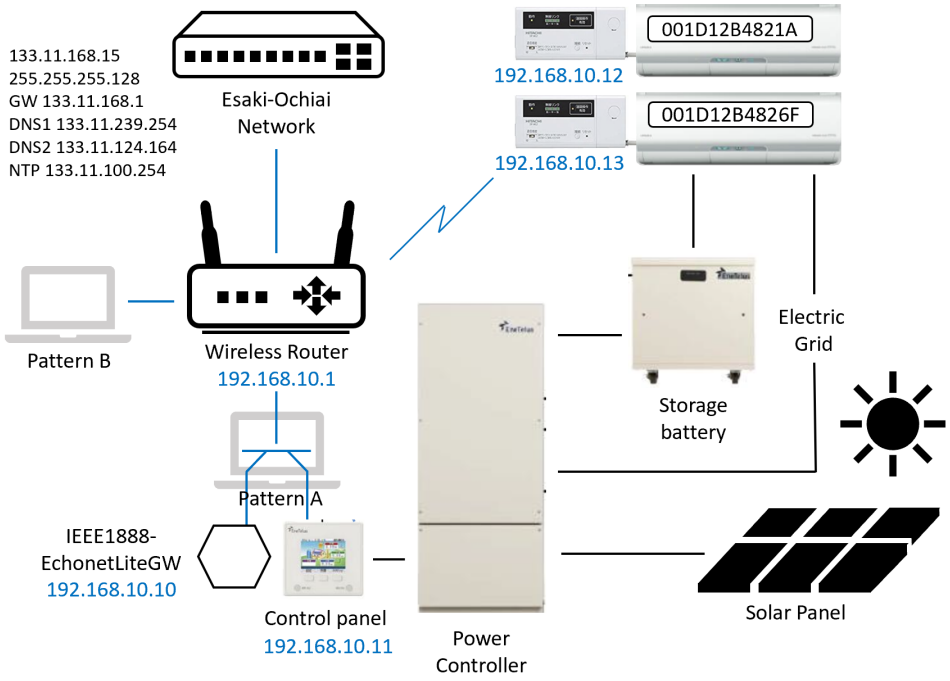


Figure 4.1: The network topology of eroom.

Blue line indicates that devices is connected through ethernet cable, while black line represents the electrical cable. Devices are connected to the internet through the network of Esaki-Ochiai laboratory. Electricity produced by the solar cell is sent to the load levelling unit. Load levelling unit then provides the necessary energy to the air conditioners, while stores the excess energy at the battery. During night time, when the electricity produced from solar cell is insufficient, load leveling can provide the needed electric from battery. User can also configure the way load leveling operate through the control panel. The consumption data, along with command code is passed to EchoLiteGW for taking record.

- The control panel (192.168.10.11). This control panel is used as user interface, it can take command of how load leveling should operate, visualize the collected data: how much electricity has been produced, how much electricity did air conditioner has consumed, and more.
- IEEE1888-EchonetLiteGW (192.168.10.10). EchonetLite is communication protocol standard for smart devices used in house (air conditioner, lighting equipment or electricity sensor). In this IoT system, air conditioner controllers and the control panel communicate with each other using Profinet (UDP/3610), IEEE1888-EchonetLiteGW serves as a gateway adapter, translate Profinet protocol data into XML for storing data on GUTP server.
- 2 Air conditioners with controller (192.168.10.12, 192.168.10.13). This air conditioners are operating according to the setting from the control panel.
- (Wireless Router (192.168.10.1). It is used to connect all 4 devices and the internet)

4.2 Packet Capturing

4.2.1 Purpose

In this experiment, we captured system's traffic data, then analyzed it to find the pattern of this IoT system communication. This packet data capturing method is also used in both "Device discovery experiment" and "Whitelist extraction experiment".

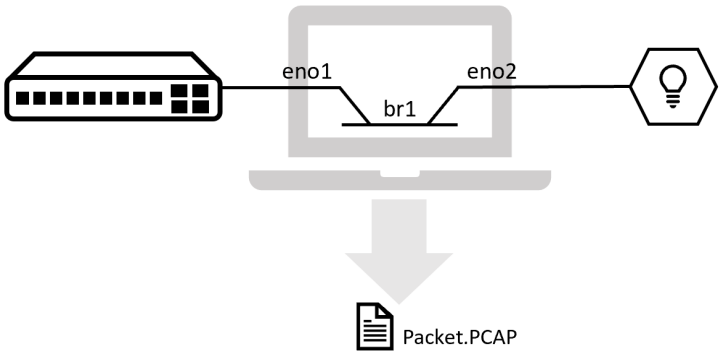
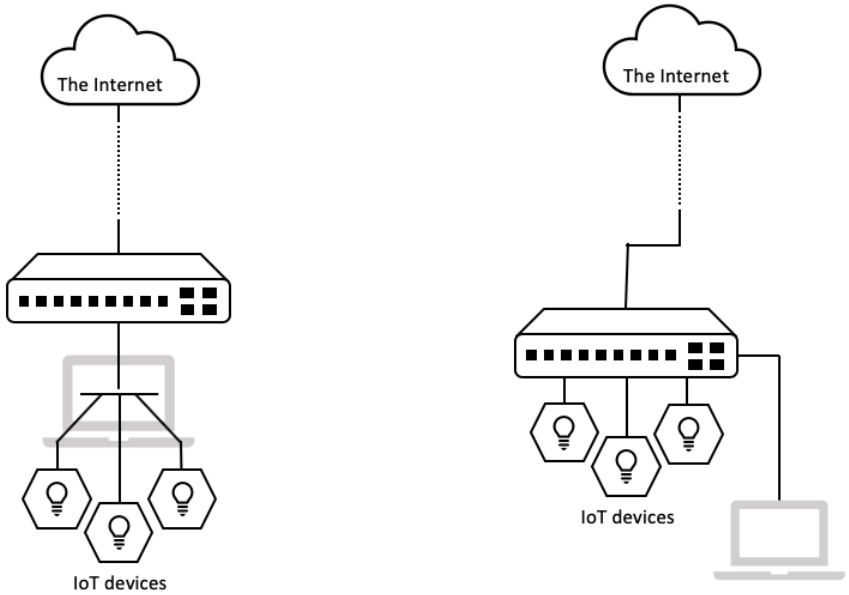


Figure 4.2: Packet Capturing

4.2.2 Method

First, we inserted a computer in between Switch and any IoT devices, we aim to capture its packet. As a requirement, computer used in this procedure must has at least 2 network interfaces. One network interface was connected to the switch, while another was connected to IoT device (Figure 4.2). After that, we created network bridge to connect two of inserted computer network interfaces. Any computers run with Linux kernel can initiate a bridge using "brctl" command. Then packet would be captured using tpcdump command. (tcpdump is a TCP/IP packet sniffing command) Collected data was saved into PCAP format.

In this experiment, we had tested capturing devices' packet in 2 patterns. In pattern A (Figure 4.3a), we created a computer bridge between all devices and switch, while in pattern B (Figure 4.3b), we connected PC to one of IoT devices switch's ports, and capturing packet coming through.



(a) Pattern A: Capturing packets between switch and IoT devices

(b) Pattern B: Capturing packet from one of switch's port

Figure 4.3: Packet capturing method

4.2.3 Result

After we have sufficiently taken to traffic data, we used Wireshark to perform primary investigation. The summary of each captured data is shown in table 4.1.

Table 4.1: Captured data summary

	Captured period	Number of packets
Pattern A	2018-11-29 07:21:26 - 2018-12-04 05:40:04	905454
Pattern B	2019-01-20 12:52:30 - 2019-01-20 17:49:45	6424

Pattern A (Switch-Bridge-IoT)

We analyzed network traffic data (Pattern A) and summarized each host necessary traffic in table 4.2.

Table 4.2: Pattern-A Packet Summary

Source	Destination	Protocol	Description
Router(.1)	224.0.0.1	IGMPv2	Multi-cast group participation protocol
	239.255.255.250	SSDP(UDP/1900)	Simple Service Discovery Protocol
	EchonetLite (.10)	DNS(UDP/53)	DNS Response
EchonetLite (.10)	160.16.109.189	HTTP/POST(TCP/80)	For saving data to the database
	133.243.238.243, 133.243.238.244, 133.243.238.164, 133.243.238.163	NTP(UDP/123)	Network Time Protocol
	Router (.1)	DNS(UDP/53)	DNS Query
	Air Conditioner and Control panel (.12/.13 and .11)	Profinet(UDP/3610)	EchoNet Protocol
Control panel (.11)	54.65.232.105	HTTPs(TCP/443)	TLSv1.2
	224.0.23.0	IGMPv2	Multi-cast group participation protocol
	EchonetLite (.10)	Profinet(UDP/3610)	DNS Response
Air Conditioner (.12/.13)	EchonetLite (.10)	Profinet(UDP/3610)	EchoNet Protocol

Pattern B (Switch Port)

For pattern B, we could only detect broadcast packets. We found that packet data we have captured is mainly contained with ARP request packet and SSDP packet.

From the result above, we knew which host is necessary for device and which is not. We would then use table 4.2 as the evaluator for our "Whitelist Extraction" experiment (section 4.4)

4.3 Device Discovery

4.3.1 Purpose

In order for proposed system to have an efficient filtering function. First, it must be able to percieve how many devices are operating. The purpose of this experiment is to find a way to extract all available devices in the system using only system traffic data.

4.3.2 Method

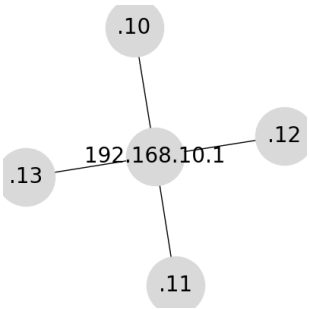
We have tested our host detection algorithm constructed with three approaches: Detecting hosts by ARP request, by ARP reply and by gratuitous ARP.

- **ARP request** is Arp packet with opcode equal to 1 and Target MAC address equal to "00:00:00:00:00:00". We considered Sender of this packet to be host in LAN and added tuple of its MAC address (SHA) and IP address (SPA) to host list.
- **ARP reply** is Arp packet with opcode equal to 2. We added both Target and Sender of this packet to host list.
- **Gratuitous ARP** is Arp packet which has is sender IP address equal to target IP address, its opcode equal to 1, and has target MAC address equal to "00:00:00:00:00:00". We added Sender of this packet it to host list.

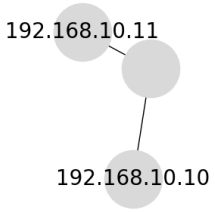
All three approaches were examined using packet data pattern A and B captured in "Packet Capturing" experiment (Section 4.2).

4.3.3 Result

Result of device discovery is shown in figure 4.4. From the result we knew that Gratuitous ARP approach performed poorly in most case, detecting no device or unrelated device. While, both ARP Reply approach and ARP request approach were able to detect some hosts.



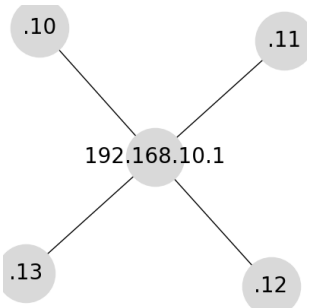
(a) Pattern A + ARP Request



(b) Pattern A + ARP Reply

No Device Found

(c) Pattern A + Gratuitous ARP



(d) Pattern B + ARP Request

No Device Found

(e) Pattern B + ARP Reply

No Device Found

(f) Pattern B + Gratuitous ARP

Figure 4.4: The result of device discovery

4.3.4 Discussion

From the result, we found that the best approach to find devices in the system is to analyze ARP Request packet. The reason ARP reply didn't perform as well as the Request pattern is because air conditioners were connected to the router wirelessly, therefore we couldn't collect it ARP reply packet directly. We believe that if all devices were connected to the switch through ethernet cable. The ARP reply approach should perform as well as the ARP request pattern.

ARP reply might seem like a not efficient method, considering that our bridge PC needs to have network interface at least equal to the number of devices plus the switch. However, if the system was implemented with SDN-switch (Software Defined Network switch), we can easily sniff through all packet. We can set a rule in SDN-controller plane to copy all ARP reply packet in network plane.

4.4 Whitelist Extraction

4.4.1 Purpose

In order to secure our IoT system, the suitable whitelist for each device is crucial. By implementing whitelist, not only can we guarantee that system can withstand the attacks from outside, but we can also prevent our infected device from attacking other servers. In this experiment, we want to find an algorithm to extract the secure hosts.

4.4.2 Method

Program and Algorithm We implemented program to extract whitelist for each device, extracted from Device Discovery. Next, we would like to explain the algorithm behind this program.

1. Program takes network traffic data as an input.
2. Traffic data is divided into a smaller group of time τ. Only IPv4 packet with TCP/UDP as its transmission protocol was considered.
3. In each group, program looks at each host's first interaction (packet) with edge device.
    - If packet is originated from device, we decide that this host might be secure.
    - If packet is destined to device, we decide that this host might be insecure.
4. If in most group (over threshold value), host is identified as "might be secure", we add it to device's whitelist.
5. The output of program is devices' whitelist (saving in JSON format).

In this experiment, we used *eroom - Pattern A (Switch-Bridge-IoT)* as the program's input. The experiment conditions were set to: threshold $> 90\%$ and $\tau = 60$ seconds.

```
porapatongkanchana@pop-os:~/Desktop/whitelist$ python3 _print_whitelist.py result/eroom-switch-bridge-
ID : 1
IP Address  : 192.168.10.10
MAC Address : ['00:11:0c:12:44:8e']
-----------------------------------------------------------------------------------------------------
IP Address          Port/Protocol               SECURE         ALL            RATIO
192.168.10.11       3610/UDP                    6867           6868           0.9998543972044264
192.168.10.13       3610/UDP                    6893           6894           0.9998549463301422
192.168.10.12       3610/UDP                    6892           6893           0.9998549252865225
192.168.10.1        domain(53)/UDP              236            237            0.9957805907172996
133.243.238.243     ntp(123)/UDP                59             60             0.9833333333333333
133.243.238.244     ntp(123)/UDP                58             59             0.9830508474576272
133.243.238.164     ntp(123)/UDP                60             61             0.9836065573770492
133.243.238.163     ntp(123)/UDP                59             60             0.9833333333333333
160.16.109.189      http(80)/TCP                6891           6898           0.9989852131052479
=====================================================================================================
ID : 2
IP Address  : 192.168.10.1
MAC Address : ['34:76:c5:a1:fa:e2']
-----------------------------------------------------------------------------------------------------
IP Address          Port/Protocol               SECURE         ALL            RATIO
239.255.255.250     1900/UDP                    489            490            0.9979591836734694
192.168.10.10       domain(53)/UDP              236            237            0.9957805907172996
=====================================================================================================
ID : 3
IP Address  : 192.168.10.13
MAC Address : ['00:1d:12:b4:82:1a']
-----------------------------------------------------------------------------------------------------
IP Address          Port/Protocol               SECURE         ALL            RATIO
192.168.10.10       3610/UDP                    6894           6895           0.9998549673676578
=====================================================================================================
ID : 4
IP Address  : 192.168.10.12
MAC Address : ['00:1d:12:b4:82:6f']
-----------------------------------------------------------------------------------------------------
IP Address          Port/Protocol               SECURE         ALL            RATIO
192.168.10.10       3610/UDP                    6887           6888           0.9998548199767712
=====================================================================================================
ID : 5
IP Address  : 192.168.10.11
MAC Address : ['a0:12:db:00:89:b5']
-----------------------------------------------------------------------------------------------------
IP Address          Port/Protocol               SECURE         ALL            RATIO
192.168.10.10       3610/UDP                    6868           6869           0.999854418401514
54.65.232.105       https(443)/TCP              54             56             0.9642857142857143
=====================================================================================================
```

Figure 4.5: Screenshot of Whitelist Extraction Program (eroom)

4.4.3 Result

The screenshot of program's output is shown in figure 4.5 The list contained the IP address of hosts, the protocol and port number which our devices started the connection to (IP addresses in yellow frame of figure 4.5 is the whitelist of 192.168.10.10). "SECURE" column show the number of time subgroup was judged as secure (First packet was from device). "ALL" column show number of subgroup packet of this host was found.

We compared our program's output with our previous inspect in section 4.2 and found that hosts extracted with this algorithm were indeed the secured hosts, proving that the program is working.

4.4.4 Discussion

The result shows that the algorithm can perform well with our tested IoT devices, taking the advantage of device's limited communication pattern. However, this didn't mean that this approach would work well on all IoT system, each IoT system has its own protocol, network topology and communication pattern.

In our tested IoT system, air conditioner controllers take command from the control panel. In this case the control panel is the one initiates the connection to air conditioner controller; therefore, air conditioner should not consider control panel as the secure host and would not add it to the whitelist. However, our tested IoT system is designed so that both control panel and controller are in the same LAN, so we were able to extract both IP addresses. In the case that, control panel locates outside the LAN, we don't think that this algorithm would work.

Another worth mentioned topic is that we have conducted this experiment under the assumption that during Preparation stage, no malicious host has already intruded in and all our captured packet is secure. If there were malware in the captured traffic, there is a possibility that our algorithm would recognize it as a secure host and couldn't create a proper whitelist.

For the whitelist extraction algorithm improvements, in this current version, we only investigated IPv4 packets, while left the IPv6 untouched. We didn't consider hosts' P2P connection, which might happen outside of the switch.

4.5 Packet Filtering

4.5.1 Purpose

After whitelist has been extracted, we can keep the network traffic of IoT system secure by filtering unwanted traffic. The purpose of this experiment is to test our filtering method, `iptables`

4.5.2 Method

In this experiment, we created a filter by combining `NetfilterQueue-python` and `iptables`. `iptables` is a Linux command that allows user to filter network packets by configure tables of IP packet filter rules in the Linux kernel. We wrote a program that take the JSON output of whitelist extraction program as an input a create an `iptables` iptables script as its output. Next is the template of our script.

```
1 # iptables -F
2 # iptables -P ACCEPT
3 # iptables -A FORWARD -p [udp|tcp] -dport [port number] -d [Host
      IP] -s [Device IP] -j ACCEPT
4 # iptables -A FORWARD -p [udp|tcp] -dport [port number] -d [
     Device IP] -s [Host IP] -j ACCEPT
5 # iptables -A FORWARD -p [udp|tcp] -j DROP
```

- 1st line, -F command is to clean the exist rule in iptables.
- 2nd line states that our default policy to ACCEPT. In other words, packet that doesn't match with any rule in iptables will be forwarded.
- 3rd and 4th line states that only device's hosts with specific protocol, port number can communicate with the device.

- 5th line means any tcp,udp packet that doesn't match above rules will be dropped.

Then we ran this script on computer bridging between the internet and the switch. This allowed us to filter all unnecessary packets and kept our IoT system secure. Next step, we wanted to evaluate the performance of our whitelist. Therefore, instead of dropping and accepting at the iptables, we configured iptables to pass all dropped packets to our "NetfilterQueue" python program. NetfilterQueue is a module that provides access to packets matched by iptables rule, we can analyze those packets using "kamene" (also known as "scapy"). (Figure 4.6)

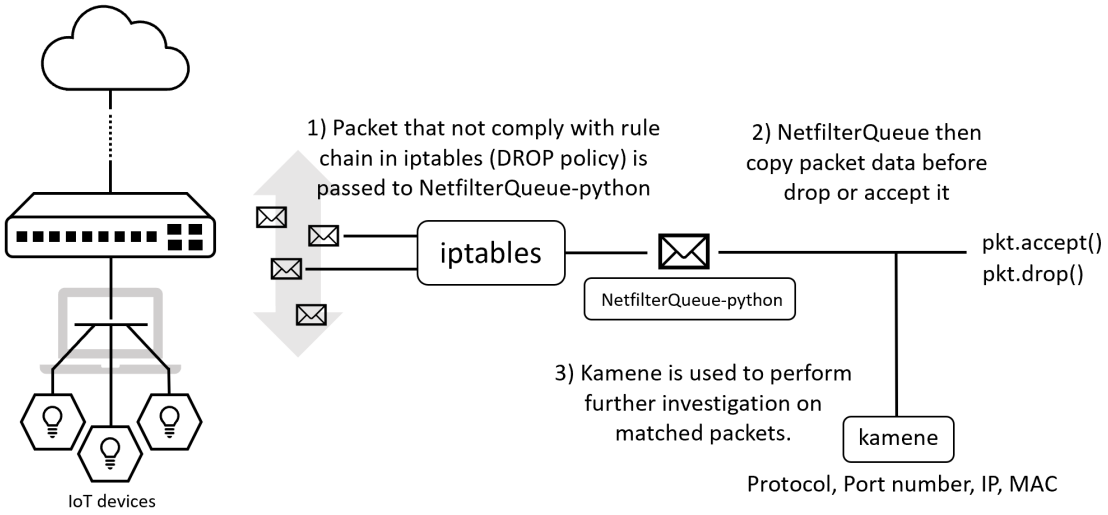


Figure 4.6: Analyzing matched packet using NetfilterQueue-python module and kamene

We had filtered the network traffic of eroom from 24 January 2019 to 30 January 2019. During this period no intentional attack was conducted.

4.5.3 result

We summarize all packet data that was sent to NetfilterQueue. The dropped packet is divided into 3 category:

- Host-dropped packet : A packet which is destinated for devices but doesn't comply with the rule.
- Device-dropped packet : A packet which is originated from our devices but doesn't comply with our iptables rule.
- Unrelated packet : A packet which source or destination IP isn't marked for our IoT devices.

As a primary evaluation method, we believe that the system should has low Device-dropped packet number while all the Host-dropped packet should be from unknown source.

Host-dropped packet

Table 4.3: Host-dropped packet

Protocol	Source	Host Location
TCP/22	75.151.236.50	USA
	78.128.112.62	Bulgaria
	114.156.2.98	Japan (JPNIC)
	185.209.0.12	Latvia (VPN)
tcp/1886	185.222.211.146	UK
TCP/1888	85.93.20.22	Bulgaria
	114.156.2.98	Japan (JPNIC)
	198.108.67.32	USA
	198.108.67.48	USA

Table 4.3 summarizes, packets from host which is not in device's whitelist. All the packet is destined for EchoNetLiteGW (192.168.10.10). TCP/22 is the SSH port (Secure Shell), adversaries normally guess the password and try logging in to the device through this port. If the 22 port is opened and device's configuration allow logging in by password, there is a chance that our devices might get hacked.

Device-dropped

Table 4.4: Device-dropped packet

Protocol	Source	Destination
TCP/3610	192.168.10.11	224.0.23.0
	192.168.10.12	
	192.168.10.13	

The device-dropped packets are TCP/3610 packets originated from Control panel (.11) and two air conditioners (.12 .13) destined for 224.0.23.0.

In EchoNetLite protocol, 224.0.13.0 is the broadcast IP address, so we believe that this IP should be added to the whitelist. After we investigated traffic data, we found that IoT devices used broadcast less than 10 times during filtered period. We assumed that reason it wasn't added to the whitelist is because, our dataset was too short and that it didn't contain broadcast packet.

Unrelated packet

We had UDP/67 source 0.0.0.0, destination 255.255.255.255 as our unrelated packet. This packet pattern can be found in dynamic host configuration protocol. However, our IoT devices have their allocated IP address, therefore there was no necessary for using DHCP.

From this result, we knew the our system was working, and we have protected our device from potential attack.

Chapter 5

Conclusion

5.1 Summary

In this thesis, we have purposed a security system that would protect our devices from any possible attack. We designed our system to operate independently from user and can be used on top of a running IoT system. We then implemented and evaluated functions that would be used in the system, including *Packet capturing*, *Device discovery*, *Whitelist extraction*, *Network filtering*. The best way to detect host is to use analyze ARP request packet, captured from bridge PC between IoT devices and switch. Whitelist of IoT system can be detected by observing who is the one initiated the connection. We can use any PC running with Linux OS with iptables installed as the network filter.

Next, we would like to state our opinion of what the system should be used. We believe that whitelist filtering function should not be set as the default setting. Securing the device from ever getting infected might sound like a promising approach, however we must trade system flexibility for it. Adding new device, changing network topology or introducing new protocol, function would ruin the whitelist. We need to perform the system traffic analysis again, before using it. We think that this system should operate alongside the anomaly detection server. First, we would allow IoT devices to communicate with any hosts, while keep capturing their packet. If the anomaly is detected, we could use the captured traffic (The traffic before the anomaly is detected) to create the whitelist, and then start blocking the invalid communication until the issue is resolved.

5.2 Future Work

In this research, we have conducted all the experiment on the bridge PC, from capturing packet, analyzing packet to filtering packet. However, when the IoT system become bigger, with more devices and more packet, performing all the task at the bridge PC might not be the best implementation. We consider the SDN (software defined network) switch as the solution for this problem. Decentralizing the task, let the control plane capturing and filtering packet, while let another server run the analyzation.

For user experience, we think that users should has easy time understanding the system. Instead of reading through program outputted JSON format, the IoT device's data and its whitelist should be visualized and put on the web server. User should be able to see and edit device's whitelist as they demand.

In this experiment, we have tested our approach on only one IoT system. However, the IoT system is so diversity; each system has its own communication protocol, connection

pattern, architecture. Without further testing on more system, it is hard to claim that our approach work. For future work, we want to test our approach on more system from various country, vendor.

Acknowledgement

I would first like to thank my advisors, professor Hiroshi Esaki and assistant professor Hideya Ochiai for advising, supporting and encouraging me throughout this entire year. Thank you for answering my every little question, for teaching me even the simplest things.

Beside from my advisors, I would also like to thank doctor Manabu Tsukada and research associate Seiichi Yamamoto for prepping me for the presentation and for sharing their profound point of view. This helps me reflect on my own work and improve my research.

Thanks to doctor Satoru Kobayashi for help structuring and organizing my presentation. Thanks to Mr. Tomoki Suga for helping investigate the tested IoT subject. Thanks to Mr. Yudai Aratsu for answering all my questions. Thanks to Mr. Hideyuki Nagashima and Mr. Matsuoka Katsuya for providing me the much-needed equipment. Thanks to Mr. Genta Imai for help proofreading this thesis.

I also would like to thank to all my seniors and my friends for encouraging me, for helping in both academic life and social life, for understanding and bearing with my poor Japanese, for making this year such a precious year of my life.

Finally, I would like to express my gratitude to my family who always be there to me. Thank you.

Porapat Ongkanchana
February 8, 2019

References

[1] Yesha Bhatt and Chintan Bhatt. "Internet of Things in HealthCare". In: *Internet of Things and Big Data Technologies for Next Generation Healthcare*. Ed. by Chintan Bhatt, Nilanjan Dey, and Amira S. Ashour. Cham: Springer International Publishing, 2017, pp. 13–33. ISBN: 978-3-319-49736-5. DOI: 10.1007/978-3-319-49736-5_2. URL: https://doi.org/10.1007/978-3-319-49736-5_2.

[2] A. Zanella et al. "Internet of Things for Smart Cities". In: *IEEE Internet of Things Journal* 1.1 (Feb. 2014), pp. 22–32. ISSN: 2327-4662. DOI: 10.1109/JIOT.2014.2306328.

[3] D. Reinsel J. Gantz. "The digital universe in 2020: Big data bigger digital shadows and biggest growth in the far east". In: *IDC iView: IDC Anal. Future*. Vol. 2007. Dec. 2012, pp. 1–16.

[4] A. Al-Fuqaha et al. "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications". In: *IEEE Communications Surveys Tutorials* 17.4 (Fourthquarter 2015), pp. 2347–2376. ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2444095.

[5] Muddy Waters LLC. *MW is short St. Jude Medical (STJ: US)*. Aug. 2016. URL: https://www.muddywatersresearch.com/research/stj/mw-is-short-stj/.

[6] Cesare Garlati. *Owlet Baby Wi-Fi Monitor "Worst IoT Security Of 2016"*. Oct. 2016. URL: https://www.informationsecuritybuzz.com/expert-comments/owlet-baby-wi-fi-monitor-worst-iot-security-2016/.

[7] Vladimir Kuskov Mikhail Kuzin Yaroslav Shmelev. *New trends in the world of IoT threats*. Sept. 2018. URL: https://securelist.com/new-trends-in-the-world-of-iot-threats/87991/.

[8] C. Kolias et al. "DDoS in the IoT: Mirai and Other Botnets". In: *Computer* 50.7 (2017), pp. 80–84. ISSN: 0018-9162. DOI: 10.1109/MC.2017.201.

[9] H. Suo et al. "Security in the Internet of Things: A Review". In: *2012 International Conference on Computer Science and Electronics Engineering*. Vol. 3. Mar. 2012, pp. 648–651. DOI: 10.1109/ICCSEE.2012.373.

[10] Z. Zhang et al. "IoT Security: Ongoing Challenges and Research Opportunities". In: *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*. Nov. 2014, pp. 230–234. DOI: 10.1109/SOCA.2014.58.

[11] 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)*. Dec. 2015, pp. 336–341. DOI: 10.1109/ICITST.2015.7412116.

[12] M. Saadeh et al. "Authentication Techniques for the Internet of Things: A Survey". In: *2016 Cybersecurity and Cyberforensics Conference (CCC)*. Aug. 2016, pp. 28–34. DOI: 10.1109/CCC.2016.22.

[13] G. Kalogridis et al. "Privacy for Smart Meters: Towards Undetectable Appliance Load Signatures". In: *2010 First IEEE International Conference on Smart Grid Communications*. Oct. 2010, pp. 232–237. DOI: 10.1109/SMARTGRID.2010.5622047.

[14] T. Song et al. "A Privacy Preserving Communication Protocol for IoT Applications in Smart Homes". In: *IEEE Internet of Things Journal* 4.6 (Dec. 2017), pp. 1844–1852. ISSN: 2327-4662. DOI: 10.1109/JIOT.2017.2707489.

[15] P. Yang et al. "An Efficient Privacy Preserving Data Aggregation Scheme with Constant Communication Overheads for Wireless Sensor Networks". In: *IEEE Communications Letters* 15.11 (Nov. 2011), pp. 1205–1207. ISSN: 1089-7798. DOI: 10.1109/LCOMM.2011.092911.111598.

[16] C. Liu, Y. Zhang, and H. Zhang. "A Novel Approach to IoT Security Based on Immunology". In: *2013 Ninth International Conference on Computational Intelligence and Security*. Dec. 2013, pp. 771–775. DOI: 10.1109/CIS.2013.168.

[17] J. Pacheco and S. Hariri. "IoT Security Framework for Smart Cyber Infrastructures". In: *2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W)*. Sept. 2016, pp. 242–247. DOI: 10.1109/FAS-W.2016.58.