Hindawi
Security and Communication Networks
Volume 2021, Article ID 6610863, 14 pages
https://doi.org/10.1155/2021/6610863

Research Article

Enhancing Digital Certificate Usability in Long Lifespan IoT Devices by Utilizing Private CA

Daiki Yamakawa,[1,2] **Takashi Okimoto**,[1,3] **Songpon Teerakanok**,[1,4] **Atsuo Inomata**,[1,4,5] **and Tetsutaro Uehara**[1,3]

[1]*Cyber Security Laboratory, Ritsumeikan University, Kusatsu, Shiga 525-8577, Japan*
[2]*Graduate School of Information Science and Engineering, Ritsumeikan University, Kusatsu, Shiga 525-8577, Japan*
[3]*College of Information Science and Engineering, Ritsumeikan University, Kusatsu, Shiga 525-8577, Japan*
[4]*Research Organization of Science and Technology, Ritsumeikan University, Kusatsu, Shiga 525-8577, Japan*
[5]*Osaka University, Suita, Japan*

Correspondence should be addressed to Daiki Yamakawa; yamakawa@cysec.cs.ritsumei.ac.jp

Received 4 December 2020; Revised 22 January 2021; Accepted 7 February 2021; Published 16 February 2021

Academic Editor: Chalee Vorakulpipat

Copyright © 2021 Daiki Yamakawa et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Today, smart devices and services have become a part of our daily life. These devices and services offer a richer user experience with a much higher quality of services than before. Many of them utilize sensing functions via cloud architecture to perform remote device controls and monitoring. Generally, the security of the communication between these devices and the service provider (e.g., cloud server) is achieved by using the TLS protocol via PKI standard. In this study, we investigate the risk associating with the use of public certificate authorities (CAs) in a PKI-based IoT system. An experiment is conducted to demonstrate existing vulnerabilities in real IoT devices available in the market. Next, the use of a private CA in the cloud-centric IoT architecture is proposed to achieve better control over the certificate issuing process and the validity period of the certificate. Lastly, the security analysis pointing out the strengths and drawbacks of the proposed method is discussed in detail.

1. Introduction

Emerging of the Internet of Things (IoT) brings a big leap in technological advancements to today's information system. IoT technologies introduce a new way of connecting IoT devices (so-called "things") to the Internet. It allows many devices to form networks for various purposes. By collecting a large amount of data from these devices, IoT provides users with a richer experience and higher quality of services. On the other hand, these new advancements in technology come with new challenges regarding security and privacy. In this paper, the network security aspects of cloud-based IoT devices are studied and discussed.

Unlike the communication between a web browser and the webserver, in the cloud-centric IoT architecture, IoT devices require predetermined server information to successfully initiate communication with the cloud server. During communication with external servers, transport layer security (TLS) is often used as a primary method for authenticating and providing confidentiality of data via encryption and digital signatures. Generally, TLS requires server certificates issued by public certification authorities (CAs) during the handshake period. However, there often are reports about vulnerabilities due to the lack of appropriate and consistent verification methods of the certificate chain [1, 2].

Furthermore, the TLS end-entity certificate has a maximum validity period of 398 days, according to Baseline Requirement version 1.7.3 [3]. With each update of the Baseline Requirement, the validity period of the TLS certificate is getting shorter every time, while the cost of certificate renewal is getting higher. This poses threats to some IoT systems in which IoT devices may not be connected to the network for a long period.

To achieve TLS security, many manufacturers currently use public CA to issue certificates for their systems. In this work, we point out that the use of public CAs, however, may not be an optimal choice for IoT systems due to the long lifespan of IoT devices, certificate cost, and inflexible requirement of the issuing process. In contrast, using a company's owned private CA may be an attractive way to tackle these problems in IoT. By using private CA, it allows the company to flexibly issue certificates for all their servers and devices with customizable details, such as the validity period. This will help to reduce the risk of certificate verification failure for some long lifespan IoT devices. Furthermore, since the private CA can issue any number of certificates for the company without additional costs, the use of private CA also provides scalability to the business. Lastly, without relying on trusted third parties such as public CAs, some unnecessary risks regarding external factors can be removed.

There are 2 main contributions in this research. First, we introduce a risk assessment method to analyze the security aspects regarding the use of TLS in various scenarios, especially when the IoT devices are not connected to the network for a long time. Second, a PKI-based IoT architecture utilizing a private certificate authority is presented. The proposed method is designed to solve the security problems according to the analyzed results obtained from the risk assessment process.

The rest of this paper is organized as follows. Section 2 gives some background information regarding the pinning process and also some examples of past security incidents. Sections 3 and 4 present a risk analysis of applying TLS to the IoT and a demonstration of an attack showing vulnerabilities in today's IoT devices, respectively. Next, the proposed method of utilizing private CA over the public one is introduced in Section 5. In Section 6, we discuss the security aspects of the proposed method and compare it against the traditional PKI-based architecture. Finally, we briefly conclude this paper in the last section.

2. Background

In the following subsections, some background knowledge and information are provided. First, the concept of cloud-centric IoT architecture is described. We then discuss the severity of the attack against TLS communication by addressing past incidents caused by operational errors in TLS. Next, a brief introduction to pinning techniques is presented. Finally, some related work and past research are presented and discussed.

2.1. Cloud-Centric IoT System Architecture. According to the 2019 White Paper on Information and Communications [4] published by the Ministry of Internal Affairs and Communications, Japan, the number of IoT devices is expected to be approximately 44.8 billion worldwide by 2021, and this number is expected to increase in the future. Generally, in cloud-centric architecture, a user uses his/her control device (typically a smartphone) to send remote commands or receive the operational status of an IoT device (e.g., home appliances) from the cloud. Regarding user security and privacy, PKI-based SSL/TLS communication is often used to establish secure communication in cloud-centric IoT architecture by authenticating the sender and encrypting transmitted data to ensure data confidentiality. Figure 1 shows an example of a communication between a smartphone application with an IoT device via a cloud server.

2.2. Past Incidents. Regarding the use of SSL/TLS, there is also a possibility of system failure due to server certificate expiration. The cause of this problem may come from human error or technical problem in system infrastructure. In this section, we present examples of past incidents involving inappropriate certificate update handling which ultimately resulted in system failures.

The first incident is the case study of a communication disturbance and services disruption of SoftBank, a large Japanese telecommunications company, on December 6, 2018 [1], which involved 36 million customers rendering them unable to make a phone call or accessing the Internet. As a result, SoftBank had to emergently downgrade the mobility management entity (MME) to the non-TLS version to cope with the situation.

Second, there is a malfunction in a device called "Unko Button" [2], an IoT device made by 144Lab [5] designed to keep track of new-born babies' stool information. In this incident, the device simply could not connect to the server due to the expired certificate. Generally, we can solve such kind of problem by updating the server certificate as usual. However, in the case of the mentioned device, the fingerprint of the server certificate was hard-coded inside the machine and could not be recovered in a typical way. Furthermore, when executing the Over the Air (OTA) update, the device was configured to forcefully use TLS without any alternative. As a result, the company had to recall all of its devices.

Lastly, there are two security reports, in 2018 and 2019, on an android and IOS application regarding lacking proper verification of server certificates. First, a MITM vulnerability was found on an older version (before version 3.0.0) of the Android application "NTV News24" [6] caused by lacking X.509 certificate verification from SSL servers. In addition, another report about MITM vulnerability was found on a LINE application (version 7.1.3 to 7.1.5) [7] on the IOS platform due to the same reason, i.e., no X.509 certificate verification.

2.3. Pinning. Pinning is a technique of embedding a server's certificate-related information to the applications or devices. This technique allows clients to determine the authenticity of the servers they are talking to without the risk of a MITM attack. There are several types of pinning depending on what information being stored (pinned) inside the client application/machine (e.g., CA certificates, end-entity certificates, and public keys). In this subsection, we discuss two pinning technologies: HTTP Public Key Pinning (HPKP, rfc7469 [8]) and DNS-Based Authentication of Named Entities (DANE, rfc6698 [9]).

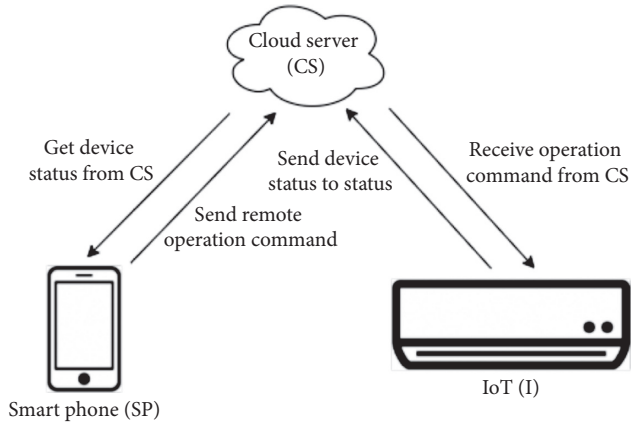

Figure 1: An overview of the cloud-centric IoT system architecture.

HPKP [8], started by Google and first developed on Chrome, is a pinning technique which pins servers' public keys information to clients using the following procedure. First, when the client connects to the server for the first time, the server sends a response back to the client with a special header. This special response header contains the public key (identical to the public key in the server certificate) and the information regarding how long the client should keep the pinned information. Therefore, after the first contact with the server, the client can now verify the certificate chain sent by the server by itself. This helps the client from being tricked by a fake/malicious server in the MITM attack. However, during the updating of the certificate information, if the updating of pinning information results in failure or problems, the client may no longer be able to connect to the server until the end of the mentioned period. Therefore, HPKP comes with such risk and operational difficulties. As a result, Google decided to abolish the support for HPKP from Chrome72 [10].

Next, DNS-Based Authentication of Named Entities (DANE) [9] is a technique that enables the utilizing of both domain name and certificates which are published for each domain. Utilizing DANE, the clients use DNS entry and TLSA Resource Record. The TLSA record contains four of the following fields: certificate usage, selector, matching type, and certificate association. The certificate usage field contains information regarding how to verify the certificate. It specifies the location where the clients will pin the information in the certificate chain and also the certificate verification method. A selector field refers to the entire certificate or "SubjectPublickKeyInfo" field in the certificate. Next, the matching type specifies whether the entire certificate association information or only a hashed version of it will be used. Lastly, the certificate association data field contains the raw data for verification; the data stored here are changed according to the information contained in the other fields.

Using DANE, clients verify the certificate with TLSA Resource Record (RR) received from the DNS server. In case that the verification process succeeds, the client will continue with the TLS handshake. Should the verification results in failure, clients will abort the process. To prevent the pinned information from being tampered, DNSSEC [11] is usually implemented to overcome this problem. However, there is still a controversy about whether DNSSEC can truly solve the problems of DNS security or not. Also, there is a problem regarding the insufficient number of end-user making DNSSEC difficult to become widespread.

2.4. Related Work. Sánchez et al. [12] provides a comprehensive technical review of several security aspects of TLS and PKI focusing on the certificate pinning mechanism. In this paper, several attacks against SSL/TLS protocol such as MITM attack using insecure renegotiation (discovered by Rescorla, E. in 2009) and BEAST attack (discovered by Duong, T. and Rizzo, J. in 2011) are discussed. The paper points out that the length of the authentication key tends to become longer due to advancements in today's computing technology. To enhance the credibility of the PKI, the use of the pinning technique is introduced and analyzed in various aspects against multiple scenarios. As a result, the author concludes that DANE is currently the best choice for the pinning technique both in terms of its functions and the cost of management.

In [12], the paper presents a way to enhance the security of the system by utilizing public CA. On contrary, in this paper, we proposed the use of private CA to gain full control of the certificate issuing process and to strengthen the security of PKI-based IoT systems. In Sections 3 and 4, we discuss the risk associated with today's PKI-based IoT system in detail.

3. Risk Analysis for IoT Devices Using TLS

There are possibilities that some problems may arise if an IoT device tries to establish TLS communication with the cloud server after disconnecting from the network/Internet for a long time. IoT devices are generally produced in factories and then sold to customers via retailers. An example of a problem (so-called "dead stock") is given in which some currently unused/unsold IoT devices being kept in the inventory/warehouse of retailers for a long time.

The causes of the dead stock problem may vary from country to country. Also, it depends on the size of the company and its inventory management policies. For example, large retail enterprises and manufacturers, which are well-established and are existed for a long time, usually have a significantly lower risk of going bankrupt comparing to small venture companies due to differences in management and cash flow.

Business models of small manufacturers or retail companies are usually limited by their smaller cash flow encouraging them to lower the number of products in the inventory and urge them to sell their products more quickly. Although "dead stock" is not something desirable, big enterprises still have more options available. They can wait and keep their products inside the inventory for some time if they need to. Hence, the inventory problems are more likely to occur to these enterprises rather than small venture companies that are urged to sell things to generate cash flow very quickly.

When the IoT products are kept in the inventory or remained unused for a long time, these devices become old and can potentially cause several security issues due to their outdated security configurations.

An excellent example of a very recent vulnerability reported on Cisco devices in 2021 is given [13]. On January 2021, a vulnerability report was founded on several Cisco devices, i.e., Cisco RV110W Wireless-N VPN (2011), RV215W Wireless-N VPN Router (2012), and RV130 VPN (2014). A vulnerability found on these old Cisco devices allows an unauthenticated, remote attacker to execute arbitrary code on the target Cisco device due to improper validation of incoming UPnP traffic. Regarding this report, an attacker can exploit the vulnerability by sending crafted UPnP request to the affected target to execute arbitrary code as the root which can end up causing the affected device to reload, finally resulting in denial of service (DoS) condition [14]. In this example, the affected devices are old Cisco equipments which mostly are 7 to 10-year-old. Even though the information of the end-of-support date displayed on the Cisco website indicates that Cisco intends to support these devices for 3 more years (until 2024) [15], there are possibilities that these devices may prone to future attacks after the end-of-support date.

Furthermore, in 2020, a vulnerability has been reported against an older model of Nintendo game console (i.e., Nintendo64). Exploiting such vulnerability allows attackers to execute malicious code on the mentioned game console. This vulnerability report is also a good example showing that even though the device is old, it still has a possibility of becoming a target of the attack.

Since the current generation of IoT device has relatively higher capabilities than electronic devices in the past, IoT devices have now become an attractive target for attackers. Concerning previously mentioned examples, we cannot deny the possibility that a similar situation may happen to these IoT devices in the future when they become old.

In this research, we studied problems which might occur when IoT devices did not connect to the network or the Internet for a long time, focusing on problems associated with TLS certificates and their validity period. We first discuss the peculiar situation of IoT devices compared with the traditional devices, such as PC and smartphone, which usually are preinstalled with web browsers having TLS capability. Finally, we point out a problem that occurred in this situation and discuss the solutions to the problem.

First, during the TLS handshake, a web browser utilizes trust anchors stored in the device to perform a certificate path (so-called "certificate chain") validation from an end entity to the root entity. Generally, these traditional devices (laptops, for example) that come with built-in browsers can connect to the network. These devices are usually connected to the network or Internet to utilize various services including obtaining new patches and updates. Hence, it is relatively safe to assume that these devices are usually connected to the network, and there is a slim chance that these devices are disconnected from the network for a very long time, e.g., 5–10 years.

Let us consider the case of an electrical shop where the traditional devices such as PCs and laptops are kept in boxes and stored in the inventory for a long period. Generally, OS supports for these devices are usually no longer than 10 years. Also, the average lifespan of PCs and laptops is approximately 3–5 years. Therefore, there is a relatively low possibility that the root certificate stored in these devices will be expired before the end of their lifespan. Therefore, we can assume that typical devices like laptops, smartphones, and PCs are not likely to face the problems of digital certificate expiration.

On the contrary, IoT devices are different. IoT devices usually consist of two primary functions: base functions and auxiliary functions. Base functions represent the main functions of the devices. Without these main functions, the IoT device can be considered useless or unusable. On the other hand, auxiliary functions are optional functions that are designed to deliver smarter and higher quality of services. For example, an IoT air conditioner's primary function is to adjust (increase or decrease) temperature via cooling and heating process, while it may have an auxiliary feature allowing users to turn it on/off through the Internet.

Auxiliary features of IoT devices usually come in the form of network/internet-related services which allows users to control and monitor their own devices via the Internet. However, as we will see, many IoT devices can be used properly only with their base functions without using any auxiliary features. Therefore, some unused/unsold IoT devices that are kept in storage for a very long time may encounter security problems caused by expired root certificates, even though the devices can still properly function.

Typically, an IoT device communicates with a cloud server for two major reasons: (1) performing services and (2) conducting maintenance (e.g., firmware update). The firmware update is a way for enterprises to continue supports for their IoT products. The update is performed to provide the device with new or improved features or eliminate some existing problems including known vulnerabilities. In terms of security, a firmware update usually involves updating the trust anchor information stored inside an IoT device.

After some time, companies usually decide to stop providing supports to some of their older products resulting in the older model of IoT devices will no longer be able to receive any further updates (including the certificate update). At this point, the user has generally two main options: replace IoT devices with the newer version or keep using them.

Replacing all IoT devices in the system is theoretically a best practice in terms of security, especially for a home user since changing one or two IoT devices is relatively easy. However, the same may or may not hold true for enterprises looking from the business perspective where the cost of implementation and disruption of continuing services do matter. Furthermore, in the case of the Industrial IoT (IIoT) system in which a large number of IoT devices and sensors are deployed across the site/factory, replacing all devices and sensors just because the maker decided to stop providing the firmware support may not be an optimal choice since the devices are still practically usable. Although no further firmware update may lead to these devices being exposed to future vulnerabilities, however, after performing risk prioritization and business impact analysis, these companies may consider accepting the risk or decide to find another way to mitigate the risk without replacing all IoT devices.

Since the certificate issued by public CAs is relatively short-lived compared with certificates issued by private CA, this can cause service disruptions in some IoT devices rendering them unable to use their auxiliary features, not working properly, or even stop working completely after the certificate expires. On the other hand, utilizing a TLS certificate issued by private CAs with customizable expiration periods can help mitigate such problems. Figure 2 shows the comparison of the operational lifetime of an IoT device between the use of public and private CA's certificates. As we will see, in case of a certificate issued by the public CA (Figure 2(a)), the device is guaranteed to work properly until the root certificate expires. On the other hand, the certificate issued by a private CA offers more flexibility to this problem. Using private CA can help to significantly extend the operational lifetime for an IoT device (Figure 2(b)). Note that during this extended period, users can also decide for themselves whether to replace or not to replace their devices with the new ones.

In the following subsections, we perform a risk analysis by first discussing the problem and attack scenario against the previously mentioned scenario where an IoT device has not been connected to the network for a long period. Then, the details of risk identification and countermeasures are finally presented.

3.1. Attack Scenario. In this section, we discuss attack surfaces and problems that might occur in the scenario in which an IoT device has been disconnected from the network for a very long period. Figure 3 shows an overview of the IoT system utilizing public CA that we used in risk identification.

In this scenario, the IoT device is connected to the network after being unused for a very long time. The very first thing an IoT device does is to send a request to the NTP server to acquire the current time information. The device will then try to contact the cloud server to receive the certificate chain information. Next, it verifies the certificate chain to ensure that the certificate signature is valid (correct and not expired) and also checks whether the Certificate Policy (CP) is matched and satisfies constraint conditions.

Since we focused on the scenario that an IoT device is not connected to the network for a long time, therefore, we classified the term "long time" into three cases: *A* (less than a year), *B* (longer than 1 year but less than 10 years), and *C* (more than 10 years). Table 1 shows the risks associated with each case.

According to Table 1, in case *A*, the period of disconnection is shorter than most software supporting periods, device lifespan, and root certificate lifespan. Therefore, there are no particular risks during this period. However, in the case of *B*, since the certificate validity period is approximately 1 year (according to Baseline Requirement), there is a risk that the end-entity certificate is already expired. Lastly, in period *C*, because the period is considerably long, there are chances that the TLS cipher suites become insecure or deprecated or the root CA's certificates are expired. For example, TLS 1.3 [16] was introduced after the elapse of ten years from the TLS 1.2 [17] release which comes with various changes in underlying handshake protocol and cipher suites. Furthermore, Cryptography Research and Evaluation Committees (CRYPTREC) initiated by the Japanese government also published a list of recommended cryptographic techniques called "e-Government Recommended Ciphers List" [18] which are expected to be secure within ten years.

3.2. Risk Identification and Countermeasure. In this section, risks and their countermeasure are explained and discussed. First, we identify risks and measures to be considered in case there are problems with the CA's root certificates rendering them to become invalid or unusable. These problems may be caused by the public CAs become bankrupt, go out of business, getting hacked, or the root private key is leaked or compromised.

An excellent example of security problems caused by an attack against public CA is the case of DigiNotar. DigiNotar is a Dutch certificate authority owned by VASCO Data Security International, Inc. founded in 1998. The company was responsible for issuing certificates to the private sectors and handled the PKI part of the Dutch government's e-government program called "PKIoverheid" (https://cryptosense.com/wp-content/uploads/2014/11/black-tulip-update.pdf). In 2011, an attacker performed unauthorized access to DigiNotar's CA server which allows the attacker to unlawfully get his/her hands on the private certificate information. As a result, all DigiNotar's certificates were revoked and the company finally went bankrupt in September 2011 [19].

To deal with this incident, the countermeasure to this problem is to update the trust anchor in every machine. Before updating a trust anchor, each device needs to perform a firmware update. However, the device also needs to

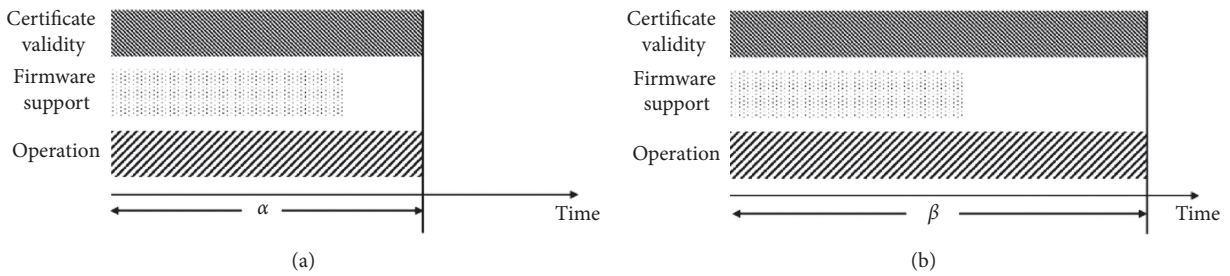

Figure 2: Operational lifetime of an IoT device.

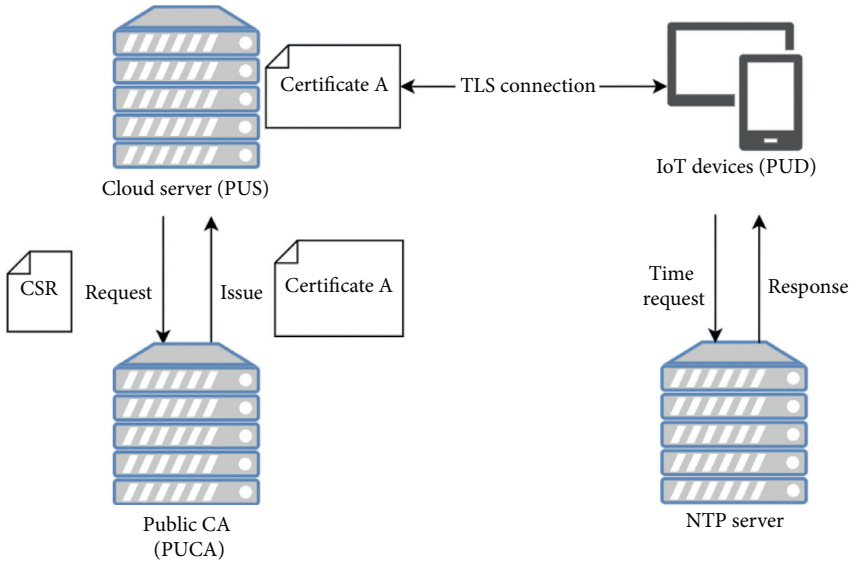

Figure 3: The system constitution using public CA.

Table 1: Risks associating with each period.

Period	*A* (less than a year)	*B* (less than 10 years)	*C* (more than 10 years)
Risk	None	The end-entity certificate is expired	TLS and its cipher suites are compromised Root certificate is expired

confirm the integrity of the firmware before the update is installed. In case that the integrity of the firmware is not guaranteed, there is a risk that the device might be running a tampered firmware making it vulnerable to some attacks. Therefore, the server is also required to perform code signing on the firmware to ensure its integrity. Next, there is a possibility the cipher suites become insecure. For example, some vulnerabilities may be found in some cryptographic algorithms rendering many systems and protocols utilizing such algorithms become vulnerable against particular types of attacks. A good example of vulnerability found in cipher suites is the case of an attack called the "Lucky 13" attack. Lucky 13 attack is a cryptographic timing attack against TLS protocol targeting the cipher block chaining (CBC) mode of operation, discovered by AlFardan, N.J. and Paterson, K. in 2013 [20]. Lucky 13 attack utilizes the padding that is not protected in CBC mode in the construction of integrity verification of TLS. Should an attacker succeed in executing the Lucky 13 attack, he/she would be able to decrypt the message and obtain the confidential information inside. Fortunately, the researchers who found this attack exercised the vulnerability disclosure policy and worked with software vendors to the creating updates and patches to mitigate this problem and also made them available at the time of publication.

To deal with this type of problem (i.e., vulnerabilities found within cipher suites), the countermeasure in this situation is to update the firmware of each IoT device as soon as it connects to the network. When an IoT device tries to connect to the server using vulnerable cipher suites, the server should turn down the request/connection and allow the IoT device to download and install a new firmware to prevent any known vulnerabilities.

Lastly, there is also a risk associating with the use of network time protocol (NTP) [21]. Generally, an IoT device will attempt to query the time information from the NTP server as soon as it connects to the network. However, the authenticity of this time information becomes a critical issue

because it involves the verification of the certificate validity period. An attacker can execute a MITM attack by creating a fake NTP server and sending the manipulated time information to the clients (i.e., IoT devices).

To prevent the MITM attack, we need to consider the use of any reliable information other than the time information to prevent the IoT device from being tricked by the fake NTP server. GPS [22] information is considered a reliable choice for any IoT device equipped with GPS-related functions. If such functions are implemented, IoT devices also have an option to receive the time information from the satellites. Another way to overcome the mentioned problem is to use the pinning technique to store the public key information of the service server in advance. In this case, the IoT device can establish a secure connection with the cloud server using the pinned public key. After TLS handshake, the secure communication channel is established; the IoT device can then ask the cloud server for the correct time information.

Unfortunately, if the devices have already been affected by the MITM attack, the IoT device has no choice but to repeatedly try to reconnect to the real server. In case that, after many attempts, the client is still unable to establish secure communication with the real server, the IoT device is suggested to disconnect from the network and operate in an offline mode. If the IoT device somehow cannot establish a secure connection with the server and also cannot request the firmware update from the server, it is also advised to operate the device in offline mode.

4. Vulnerabilities in Today's PKI-Based IoT Security

In this section, the experiment and security analysis of today's PKI-based IoT security are tested and discussed. In this study, an experiment to show the vulnerability in today's IoT system was conducted. During the experiment, a man-in-the-middle (MITM) attack was carried out in the testing environment to demonstrate how a real IoT device with the improper implementation of security controls (i.e., PKI-related modules) may fall prey to such attacks. The following subsections present details of the testing environment and the results of the MITM attack against various IoT devices.

4.1. Testing Environment. There are four main components in the test system: target IoT devices (I), cloud server (CS), a fake (malicious) cloud server (FS), and a fake DHCP and DNS Server (FD). Figure 4 shows an overview of the test system.

According to Figure 4, the DHCP and DNS server was deployed using Raspberry Pi 3 (Model B, v1.2) [23] running dnsmasq [24] (v2.76) module with Raspbian 9 OS. The fake cloud server for executing a MITM attack is implemented using a PC running Nginx v1.18.0 [25] on Ubuntu 20.04.1 OS [26]. Lastly, the IoT devices component represents IoT devices from different vendors. Table 2 shows the details of IoT devices used for evaluation. There are five devices used during the test: a smart air conditioner, an air purifier, an IoT control device, a camera, and a smart plug. Each device comes with a different TLS certificate validity period. Note that the specific details of each device (e.g., device model and manufacturer) in Table 2 cannot be disclosed and were intentionally omitted due to security reasons.

In this section, we demonstrate how to exploit weaknesses in IoT devices focusing on the MITM attack. All devices being tested are typical IoT home appliances. However, the same exploit can be found and can also be applied to the industrial field (i.e., IIoT) as well.

Under normal circumstances when the network connection between I and CS is active, the IoT devices can securely communicate and exchange information with the cloud server without a problem. On the other hand, the target IoT device is, however, considered prone to attack when the connection between itself and the cloud server is lost. In the following subsection, we demonstrate how a MITM attack can be executed under this condition.

4.2. MITM Attack. In this work, we assess the security of today's IoT system against a MITM attack. Generally, the IoT devices are expected to properly verify the certificate chain from an end entity to the root entity using the public key of each CA. However, there are also possibilities that some IoT devices available in the market are not practically doing this or do not verify each certificate within the chain properly. In this experiment, we found that some IoT devices verify the TLS certificate using only subject or issuer fields. This allows us to bypass the certificate verification of these IoT devices and finally complete the TLS handshake.

To demonstrate such an attack, first, the risk assessment against a MITM attack was conducted. In this assessment, an IoT device is considered to have a MITM-related vulnerability if the attacker can bypass the certificate verification and successfully complete the handshake process. There are two phases in performing a MITM attack: data collection and experimentation.

4.3. Data Collection. First, we begin the data collection phase by collecting cloud server (CS) related information (e.g., IP address) by observing network traffic between target IoT devices (I) and the server. To achieve this goal, the IoT devices are connected to the local access point (AP) which is also connected to the local network via a layer-2 switch (S1). Utilizing the port mirroring function of the network switch, the malicious server (PC) is connected to the mirror port on S1 to gather information. This allows the PC to observe the communication between I and CS. Using the obtained information with the client function of the TLS connection via OpenSSL [27], we can successfully retrieve the server certificates. Figure 5 shows the network topology used in the testing against the MITM attack.

4.4. Experimentation. Next, we set up the attack environment by first removing C2 from the layer-2 switch to simulate a situation where the connection to the cloud server C2 is not available. Next, we plug a fake DHCP and DNS server FD into the network. Then, a fake TLS certificate (FC)

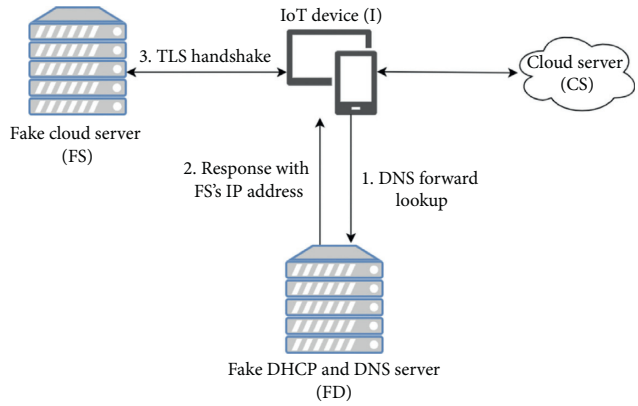

Figure 4: Overview of the test system.

Table 2: Details of IoT devices used during the test.

Corporate	A	B	C	D	E
IoT devices	Air conditioner	Air purifier	Controller	Camera	Plug
Issuer type	Private	Public	Public	Public	Public
Period of validity	24 years	2 years	2 years	1 year	1 year

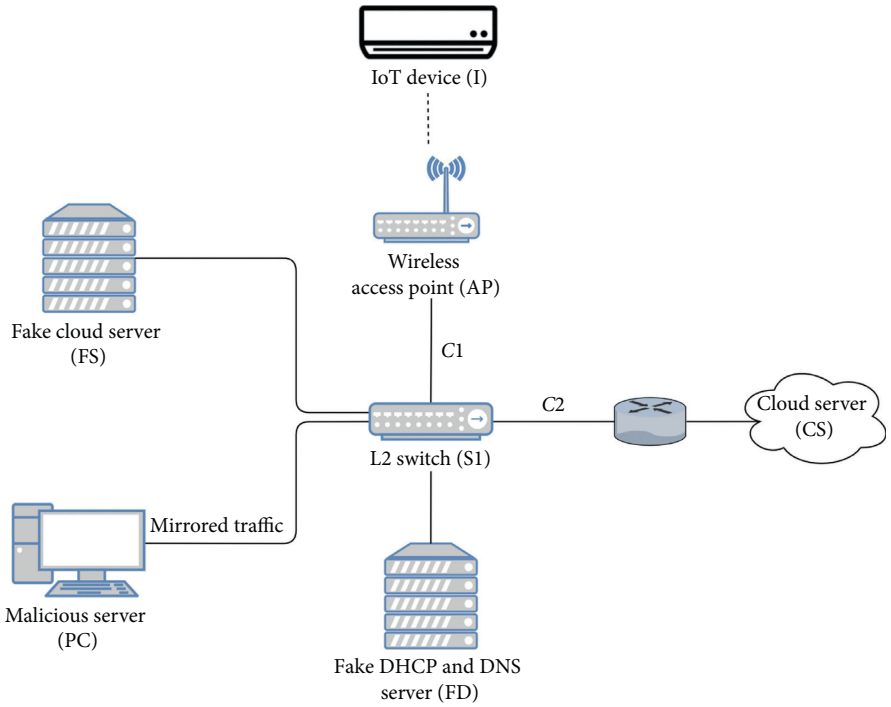

Figure 5: Network topology used for a MITM attack.

imitating the one retrieved from the cloud server is forged. This certificate FC will be used later to deceive the target IoT device.

The MITM attack begins with an IoT device sending a forward DNS lookup request to FD. The fake DNS server FD then replies with the IP address of the fake cloud server FS. Next after resolving the server IP, the IoT device will try to initiate a TLS handshake with the server (which is the fake one) by sending a "Client Hello" message. FS then replies to the IoT device with a "Server Hello" message using the previously forged TLS certificate. This marks the completion of the MITM attack.

5. Results

During the data collection stage, we obtained the certificate information from the cloud server by observing the traffic coming in and out of the ethernet switch S1. Depending on

the type of certificate authority (CA) used to issue the certificate, the collected certificate information can be classified into two types: certificates signed by public CA and certificates signed by the private CA. Table 2 shows the type of certificates associated with each device. In this work, we generally focus on the difference between a certificate issued by public and private CA. According to the TLS Baseline Requirements document [3], the TLS certificate validity period is approximately 2 years. On the other hand, there is no such rule/requirement in the case of certificates signed by the private CA. In our experiment, the validity period of the certificate issued by private CA is set to roughly 24 years. Regarding the forged TLS certificate, Table 3 shows information that appeared on the certificate and also information of fields that can and cannot be imitated/manipulated. Table 4 shows the supported X509v3 [28] extensions on each device. As shown in Table 4, we can imitate generally most of the extensions with the exceptions of the one which requires CA information or the secret key information from the cloud server. Although we can not create a perfect copy of the certificate since it requires CA's private key to perform cryptographic operations, the information that appeared in the forged certificate is enough to deceive some IoT devices to believe that the certificate is real since some devices inadequately verify the authenticity of a certificate using only subject and issuer fields. Hence, as a result, we found that 2 devices (i.e., B and C) were prone to MITM because they allowed us to bypass the certificate verification and complete the TLS handshake, while the other 3 devices (A, D, and E) did not.

6. Proposed Mechanism

In this study, we proposed the use of private CA in the cloud-centric IoT architecture, which is shown in Figure 6. Using private CA allows companies to issue digital certificates themselves without using any trusted third party (i.e., public CA). This approach allows manufacturers to have full control over the digital certificates pinned to their devices.

There are three primary components in the proposed architecture. First, the IoT device (PRD) represents a cloud-connected IoT device. A certificate issued by the private CA and verification key (VKF) was pinned to this device at the manufacturing stage. The cloud server (PRS) and the private CA (PRCA) are operated by the company (i.e., IoT device manufacturer) that makes PRD. Using the certificate issued by PRCA, a secure communication channel between PRS and PRD can be established. Furthermore, the cloud server (PRS) uses the predefined signature key (SKF) during the firmware updating process. Regarding the private CA, PRCA is the private certificate authority owned and managed by the manufacturing company to provide security (via digital certificate) to the systems. Having a self-signed certificate, PRCA is responsible for issuing the certificate to PRS. Since PRCA does not have to operate according to the Baseline Requirement, PRCA can flexibly decide the validity period of the issued certificates.

In this work, first, PRS makes and sends Certificate Signing Request (CSR) to the private CA. PRCA then issues a certificate B based on the obtained CSR. The manufacturer then pinned PRCA's self-signed certificate and electronics signature verification key (VKF) to their products (e.g., IoT devices). When a user purchases these devices and turns them on, the IoT devices (PRD) will try to initiate a TLS secure communication with the cloud server (PRS) (i.e., sending a "Client Hello" message). The server then accepts the request and sends the certificate chain including certificate B back to the client. Next, PRD verifies the certificate chain received from the server using the pinned certificate (PRCA's self-signed certificate). In case that the device receives any certificate that is not signed by PRCA, the TLS connection is terminated.

Regarding firmware update, a verification key (VFK) is another key pinned to an IoT device at the time of manufacturing to be used for the firmware updating purpose. The use of a separate key pair for firmware updating operation allows the IoT devices to perform an update (including security updates) even another key pair is compromised.

6.1. Benefits. There are three main benefits of using a private CA in issuing certificates in the IoT system. First, there is a low possibility of errors caused by the expiration of the certificate during the firmware updates because the validity period of the certificate issued by the private CA can be customized. Second, there is no need to change the certificate chain in case of intermediate or root CAs having technical problems or being attacks. Lastly, the cost of issuing and updating is almost free, excluding operational and labor costs.

As mentioned in Section 2, there are incidents caused by a failure in updating certificate information which is mostly due to certificate expiration. To tackle this problem, we should make the validity period of the certificate in IoT-related system longer. This will reduce the number of times an IoT device needed to update the certificate during its lifetime which will also reduce the chance of errors that occurred during the updating process. Using the proposed method, the validity period of the certificate can be customized. Since the company owns the CA, therefore, it has full control over the certificate issuing process and the certificate specification.

On contrary, using the public CAs increases the attack surface that adversaries can utilize. For example, public root and intermediate CAs can now become the target of attacks. On the other hand, if certificates are signed by the private CA and are kept as trust anchors, the range of attack targets can be reduced.

OpenSSL allows the manufacture to build the private CA for free because the software is open-source software (OSS). The firmware update server can prevent the attackers from tampering with the firmware updating process by using the client verification. In case the attackers can get their hand on the firmware, they can freely analyze and discover vulnerabilities within the firmware. There is a report of an OS command injection attack on the firmware of some web cameras in CVE-2013-1599 [29]. Thus, to prevent the system

Table 3: Contents of the certificate which can be imitated.

Field of certificates	Can be imitated/manipulated?
Version	Yes
Serial number	Yes
Signature algorithm	Yes
Issuer	Yes
Validity	Yes
Subject	Yes
Subject public key info	Partially (except the public key information)
X.509v3 extensions	Almost everything (except the fields that require a secret key or CA certificate information)
CT precertificate timestamp	No

Table 4: X.509v3 extensions.

X.509v3 extensions	A	B	C	D	E
Authority key identifier	×	×	×	×	×
Authority information access	—	○	○	○	○
Subject alternative name	—	○	○	○	○
Certificate policies	—	○	○	○	○
Extended key usage	—	○	○	○	○
CRL distribution points	—	○	○	○	○
Key usage	○	○	○	○	○
Subject key identifier	×	×	×	×	×
Basic constraints	—	—	○	○	○

"○" and "×" indicate fields that can and cannot be imitated, respectively, while "—" represents fields that are not being used/presented in certificates of some particular devices.

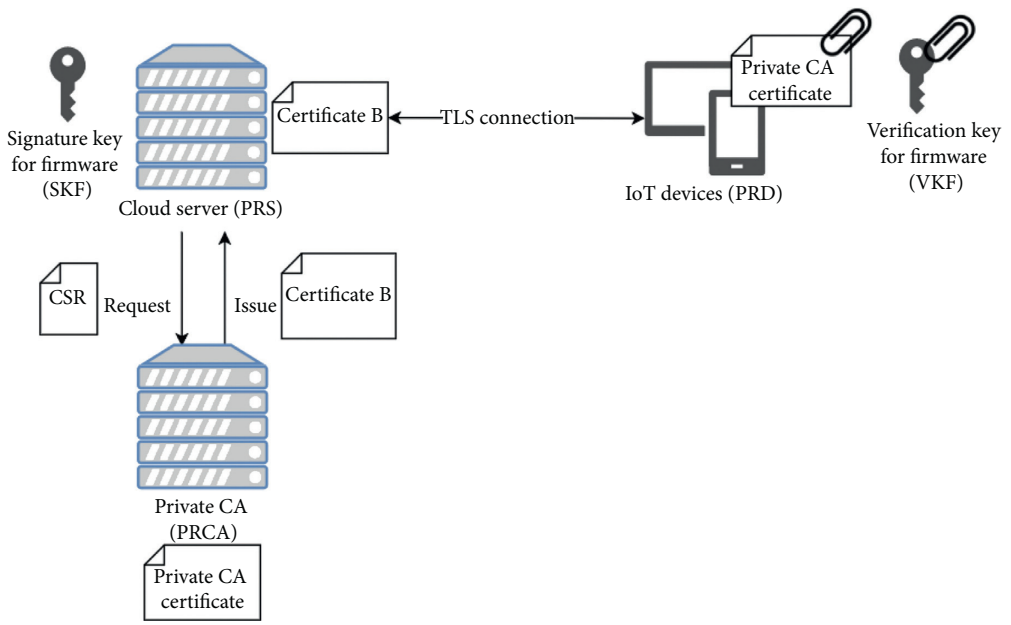

Figure 6: An overview of the proposed framework.

from such risk, the firmware must be downloaded only by the authorized/intended IoT devices. To achieve this goal, TLS client verification is considered an efficient way to overcome the problem. TLS client verification allows the server to verify whether the client is actually who it is claimed to be. In this case, each client (i.e., IoT device) must have client certificates. However, the cost of implementing such an approach is considered very expensive (or almost practically impossible in the IoT industry), if the manufacturer decides to use public CA for issuing client certificates. However, issuing client certificates can be done for free (or much less cost) by using the private CA owned by the company.

6.2. Drawbacks and Limitations. On the downside, the proposed method of using a private CA in the IoT system suffers from two main drawbacks. First, the cipher suites or

cryptographic algorithms used in the system may become deprecated or vulnerable to attacks. In the proposed system, there are possibilities that the vulnerabilities of the underlying cryptographic methods will be discovered during the lifetime of an IoT device since the certificate validity period is long.

Second, from the user perspective, there are also reliability issues when using private CA. Generally, the public root CAs gain many trusts because the trust anchor at the top of the certificate chain is operated under the root CA certificate program. For instance, Mozilla Root Store Policy [30] requires the CAs to get audited by third-party organizations such as Web of Trust for CA [31] and ETSI [32]. However, regarding the proposed system, since the private CA is used, the companies are not required to get audited. Therefore, the proposed method still requires good and secure implementation of the entire system to make it practical.

7. Security Analysis and Discussion

In this section, the security aspects of the proposed method are analyzed and discussed. Table 5 shows a comparison between the proposed method and traditional approaches. We compare our proposed method with two different architectures: A1 and A2. Table 5 shows the comparison between the proposed architecture and the existing techniques.

In the first architecture A1, public CA is employed. The public CA issues the server certificate conforming to the Baseline Requirement [3]. When the cloud server and the IoT device try to create a secure connection, the IoT device verifies the certificate chain received from the server. Similar to the first architecture, the second architecture A2 utilizes public CAs to issue digital certificates. However, in this architecture, the manufacturer pins verification key information inside the IoT device at the time of manufacture. This verification key information is used to verify code signing to ensure the integrity of firmware during the firmware update process.

The proposed method, on the other hand, uses the private CA to issue server certificates. These certificates have a longer validity period while they are also not required to follow the Baseline Requirement. Moreover, the self-signed certificate of the private CA is also pinned to the products (i.e., IoT devices) to help to prevent a MITM attack. Lastly, in the proposed architecture, a verification key for code signing is also stored inside each IoT device, similar to the A2 case. The following subsections discuss each criterion used in the comparison in detail.

7.1. Issue Cost. Generally, a certificate is required as proof of identity when communicating with TLS. Companies usually have to pay trusted third-party companies to issue their certificates. Therefore, A1 and A2 cost a lot more money than the proposed method to successfully implement the system. Besides, companies will have to pay more money if they need more certificates in the future. In contrast, the proposed method offers a cheaper way to deal with the certification fee problem using private CA. Also, the private CA can be implemented using free open-source software, e.g., OpenSSL. Since the company has a CA of its own, therefore, the company can issue any number of certificates cheaply without additional cost.

7.2. External Factors Risk. There are possibilities of external factor risk in which the companies may not be able to directly control, for example, disclosure of CA's secret key and so on. In this case, since A1 and A2 both rely on external organizations (i.e., public CA) to handle all their certification-related issues, therefore, there is a higher external factor risk in A1 and A2 than the proposed method in which private CA is employed.

7.3. Attack Surfaces (Number of CAs That Can be Targeted). A certificate issued by a public CA usually involves the issuance of certificates by many trusted organizations. These trusted parties can be attacked and compromised. Therefore, the case of A1 and A2 has a higher risk of being attacked due to a larger number of CAs that can be targeted. On the other hand, the proposed method has a very small number of CAs in the entire system, typically only one CA for a small company. Hence, the attack surface, in terms of the number of CAs that can be attacked, is relatively small compared with A1 and A2.

7.4. Risk of Certificate Update Failure. There are some incidents in which a client cannot connect to the server using TLS because the administrator of the server failed to update or renew the server certificate. Countermeasures of such incidents involve reviewing operational policies and the use of automatic updates. However, no matter what measures a manufacture take, there is always a chance of making a mistake during an update. Therefore, the most effective measures are to reduce the number of times the manufacture has to update the certificate by issuing the certificate with a longer validity period than the IoT device lifetime.

In this case, A1 and A2 cannot use such countermeasure since both of them use public CA to issue certificates which generally have much shorter validity than the lifespan of the IoT device. On contrary, the proposed method can issue a unique certificate with a long expiration period by using the private CA.

7.5. Risk of Firmware Tampering. Sometimes, an IoT device may be required to perform a firmware update to add some new features or for security reasons. During the firmware updating process, attackers may find a way to tamper with the firmware to install backdoors or create other vulnerabilities to the IoT device. Therefore, the integrity of the firmware used during the update is the utmost crucial factor needed to be concerned.

Code signing can help ensure the integrity of the firmware during the update process. Since the first architecture A1 does not have a verification key and does not utilize code signing; thus, A1 is considered prone to such

Table 5: Comparison between the proposed architecture and the traditional approaches.

Factors	A1	A2	Proposed architecture
Issue cost	High	High	Low
External factors risk	High	High	None
Attack surfaces (number of CAs that can be targeted)	High	High	Low
Risk of failure during certificate update	High	High	Low
Risk of firmware being tampered	High	Low	Low
Management cost (operate, labor cost)	Low	Low	High
CA reliability	High	High	Relatively low
Internal factor risk (insider threats)	Low	Low	Relatively high
Verification time	Slow	Slow	Fast

attack. On the other hand, both A2 and the proposed method have a lower risk against such attacks due to having a verification key pinned to each device at the time of manufacture.

7.6. Management Cost (Operational and Labor Cost). If the company decides to implement a private CA of its own, they unavoidably have to pay additional costs, i.e., operational and labor costs. Since A1 and A2 use public CAs to issue their server certificates, therefore, all operational and labor costs are already included in the amount they pay to the third-party companies from the beginning. Therefore, there is no extra cost in the case of A1 and A2. The proposed method, however, have to pay a full price of implementing a private CA including server fee and some hardware security modules (HSMs) for protecting the secret key. Hence, the proposed method is considered more expensive to manage and maintain than the other architectures.

7.7. CA Reliability and Internal Factor Risk. Many public CAs are well-known in terms of security and reliability because many of them are required to periodically perform security auditing to ensure the security of the system. For this reason, public CA is generally trusted by many companies and organizations. However, it is not likely to be the case for the private CA. Since the security of the private CA depends on each company implementing the system, therefore, it is very difficult to determine whether the private CA of which company is secure or insecure unless it performs security auditing conforming with the industrial standard.

7.8. Verification Time. Generally, many IoT devices are facing the problem of limited computational power due to energy constraints. This causes a time-lag and slower computational speed in IoT devices, comparing to traditional devices such as laptops or smartphones. Hence, one of the primary goals in designing and implementing an IoT security control is to limit resource consumption. Regarding the certificate verification process, end-entity certificates are generally issued by the intermediate CA. Also, the certificate of this intermediate CA is issued by another intermediate or the root CA. This process forms the concept of a certificate chain in which the destination server sends such hierarchical information (a.k.a. certificate chain) to the device (including IoT device). Upon receiving the certificate chain, the device verifies the validity of this certificate chain starting from the end-entity certificate to the root CA certificate.

The verification of a certificate chain usually involves the decryption of each certificate within the chain. Therefore, the number of operations required to verify a certificate chain increases in proportion to the length of the chain. Asymmetric-key cryptographic operations (e.g., decryption) are usually computationally expensive. Thus, verifying several certificates, especially in the case of a complex certificate chain, can pose a significant challenge for the IoT. On contrary, using a private certification authority allows the end-entity certificate to be issued directly from the root CA, except when the company also utilizes its own intermediate CA. This results in a flatter hierarchical structure of the certificate chain which can significantly help to reduce computational power consumption and time for verification in the computational resource-limited IoT devices.

Furthermore, using private CA with pinning technique can help getting rid of some unnecessary processes such as checking certificate validity through OCSP or checking against the certificate revocation list (CRL). Although this reduction of unnecessary complexity is not a major factor in improving the efficiency of the system, it can still help to save some computational and network resources which can ultimately end up improving the overall performance of the IoT.

8. Conclusion

In this paper, we study the standard secure communication model of the cloud-centric IoT architecture using TLS and PKI. We identify the security flaws and vulnerabilities of the public CA-based IoT architecture due to errors in the certificate verification process. Next, risk identification and countermeasures regarding the situation when an IoT device tries to connect to the network after disconnecting for a very long period are discussed. Following with a simulation of a MITM attack, we demonstrate an attack on actual devices available in the IoT market. As result, it is shown that using public CA may not be an optimal solution for IoT. To overcome such a problem, finally, we proposed the use of private CA together with the separate verification key for firmware updating. Lastly, the benefits, drawbacks, and design limitations of the proposed method are discussed and compared with today's existing PKI-based approaches.

Data Availability

This research and all experiments contain information on vulnerabilities in real IoT devices available in the market. Publicizing this sensitive information will allow adversaries to take advantage of the mentioned vulnerabilities, which can affect a large number of users. Due to this reason, we decided to exercise a nondisclosure policy on the experimental data of this research.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] SoftBank Corp, *SoftBank Press Conference*, SoftBank Corp, Tokyo, Japan, 2018, https://www.youtube.com/watch?v=Gu7xMCllCbM, in Japanese.

[2] 144Lab Co., Ltd., *144Lab: Unko Button*, 144Lab Co., Ltd., Tokyo, Japan, 2020, https://unkobtn.com/, in Japanese.

[3] CA/Browser Forum, "Baseline requirements for the issuance and management of publicly-trusted certificates version 1.7.3," 2020, https://cabforum.org/wp-content/uploads/CA-Browser-Forum-BR-1.7.3.pdf.

[4] Ministry of Internal Affairs and Communications Japan, "The evolving digital economy and society 5.0 beyond," in *White Paper on Information and Communication*Ministry of Internal Affairs and Communications Japan, Tokyo, Japan, 2019, https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r01/pdf/01honpen.pdf, in Japanese.

[5] 144Lab, *Light Up Your Science*, 144Lab, Tokyo, Japan, in Japanese, 2020.

[6] National Institute of Standards and Technology, "CVE-2019-6032," in *National Vulnerability Database*National Institute of Standards and Technology, Gaithersburg, MD, USA, 2019, https://nvd.nist.gov/vuln/detail/CVE-2019-6032.

[7] National Institute of Standards and Technology, "CVE-2018-0518," in *National Vulnerability Database*National Institute of Standards and Technology, Gaithersburg, MD, USA, 2018, https://nvd.nist.gov/vuln/detail/CVE-2018-0518.

[8] C. Palmer, R. Sleevi, and C. Evans, *RFC 7469-Public Key Pinning Extension for HTTP*, Internet Engineering Task Force (IETF), Fremont, CA, USA, 2015, https://tools.ietf.org/html/rfc7469.

[9] J. Schlyer and P. Hoffman, *RFC 6698-the DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA*, Internet Engineering Task Force (IETF), Fremont, CA, USA, 2012, https://tools.ietf.org/html/rfc6698.

[10] Google LLC, "Google, remove HTTP-based public key pinning (removed)," 2020, https://www.chromestatus.com/feature/5903385005916160.

[11] R. Austein, M. Larson, D. Massey, S. Rose, and R. Arends, *RFC 4033-DNS Security Introduction and Requirements*, IETF Network Working Group, Fremont, CA, USA, 2005, https://tools.ietf.org/html/rfc4033.

[12] D. Sánchez, A. M. Lopez, F. A. Mendoza, P. A. Cabarcos, and R. S. Sherratt, "TLS/PKI challenges and certificate pinning techniques for IoT and M2M secure communications," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 4, pp. 3502–3531, 2019.

[13] National Institute of Standards and Technology, "CVE-2021-1360," in *National Vulnerability Database*National Institute of Standards and Technology, Gaithersburg, MD, USA, 2021, https://nvd.nist.gov/vuln/detail/CVE-2021-1360.

[14] Cisco systems, Inc., *Cisco Small Business RV110W, RV130, RV130W, and RV215W Routers Remote Command Execution and Denial of Service Vulnerabilities*, Cisco systems, Inc., San Jose, CA, USA, 2021, https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-rv-overflow-WUnUgv4U.

[15] Cisco systems, Inc., *Cisco RV130W Wireless-N Multifunction VPN Router*, Cisco Systems, Inc., San Jose, CA, USA, 2021, https://www.cisco.com/c/en/us/support/routers/rv130w-wireless-n-multifunction-vpn-router/model.html.

[16] E. Rescorla, *RFC 8446-the Transport Layer Security (TLS) Protocol Version 1.3*, Internet Engineering Taks Force (IETF), Fremont, CA, USA, 2018, https://tools.ietf.org/html/rfc8446.

[17] T. Dierks and E. Rescorla, *RFC 5246-TLS Protocol Version 1.2*, IETF Network Working Group, Fremont, CA, USA, 2020, https://www.ipa.go.jp/security/rfc/RFC5246EN.html.

[18] National Institute of Information and Communications Technology and Information-Technology Promotion Agency Japan, *CRYPTREC Report 2019*, Information-echnology Promotion Agency Japan, Tokyo, Japan, 2020, https://www.cryptrec.go.jp/report/cryptrec-rp-2000-2019.pdf.

[19] B. V. Fox-IT, "Black tulip report of the investigation into the DigiNotar certificate authority breach," 2012, https://www.researchgate.net/publication/269333601_Black_Tulip_Report_of_the_investigation_into_the_DigiNotar_Certificate_Authority_breach.

[20] N. J. AlFardan and K. G. Paterson, "Lucky thirteen: breaking the TLS and DTLS record protocols," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 526–540, Berkeley, CA, USA, May 2013.

[21] D. Mills, U. Delaware, J. Martin, J. Burbank, and W. Kasch, *RFC 5905-Network Time Protocol Version 4: Protocol and Algorithms Specification*, Internet Engineering Task Force (IETF), Fremont, CA, USA, 2020, https://tools.ietf.org/html/rfc5905.

[22] National Coordination Office, *GPS: the Global Positioning System*, National Coordination Office, Washington, DC, USA, 2020, https://www.gps.gov/.

[23] Raspberry Pi Foundation, *Raspberry Pi 3 Model B*, Raspberry Pi Foundation, Cambridge, UK, 2020, https://www.raspberrypi.org/products/raspberry-pi-3-model-b.

[24] S. Kelley, "Simon kelley: Dnsmasq," 2020, http://www.thekelleys.org.uk/dnsmasq/doc.html.

[25] Nginx, Inc., "Nginx," Nginx, Inc., San Francisco, CA, USA, 2020, https://nginx.org/en.

[26] Canonical Ltd., *Ubuntu 20.04.1 LTS (Focal Fossa)*, Canonical Ltd., London, UK, 2020, https://releases.ubuntu.com/20.04.

[27] OpenSSL Software Foundation, *The OpenSSL Project: OpenSSL Cryptography and SSL/TLS Toolkit*, OpenSSL Software Foundation, Adamstown, MD, USA, 2020, https://www.openssl.org.

[28] S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, and D. Cooper, *RFC 5280-Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, IETF Network Working Group, Fremont, CA, USA, 2008, https://tools.ietf.org/html/rfc5280.

[29] National Institute of Standards and Technology, "CVE-2013-1599," in *National Vulnerability*

*Database*National Institute of Standards and Technology, Gaithersburg, MD, USA, 2020, https://nvd.nist.gov/vuln/detail/CVE-2013-1599.

[30] Mozilla, *Mozilla Root Store Policy Version 2.7*, Mozilla, Mountain View, CA, USA, 2020, https://www.mozilla.org/en-US/about/governance/policies/security-group/certs/policy.

[31] CPA Canada, *Principles and Criteria and Practitioner Guidance*, CPA Canada, Toronto, Canada, 2020, https://www.cpacanada.ca/en/business-and-accounting-resources/audit-and-assurance/overview-of-webtrust-services/principles-and-criteria.

[32] European Telecommunications Standards Institute (ESTI), *Welcome to the World of Standards!*, European Telecommunications Standards Institute (ESTI), Sophia Antipolis, France, 2020, https://www.etsi.org/.