Risk Mitigation in PKI Implementation within IoT

Myriam Benkoussa

Advised by Kris Micinski
Dept. of Computer Science
Haverford College

Abstract

This thesis aims to review the risks of public key infrastructure (PKI) implementation within the internet of things (IoT) as well as their potential solutions. This thesis first introduces IoT and its history and expansion, and presents an overview of general security architecture as well as security considerations within the context of IoT. Both cryptographic and non–cryptographic security measures are presented, and the security risks within the authorization/certification process (i.e., non–cryptographic security measures of IoT) are outlined and assessed. Currently, the largest security threats lie within the implicit trust between IoT devices and in their communication. There are also issues with the interoperability of systems and scalability of the PKI implementation, which increase in conjunction with IoTs rapid growth. Potential solutions proposed at the end of this thesis consist of the bridge approach to interoperability and scalability of systems, policy constraints in certification, and pinning, which associates devices with their expected certificate or public key.

Contents

Motivation

Recent years have witnessed the tremendous growth of the internet of things (IoT) and its many applications – in the field of medicine, banking services, "smart" wearables, and autonomous vehicles. However, the rapid growth of these connected devices is accompanied by an increase in their attack surface. There have been various attempts by researchers to review and mitigate the security risks in PKI within IoT, and the ever–evolving security improvements (such as SSL to TLS, and the development of key exchange methods) show that there is still research to be done in terms of the security and improvements of PKI. Currently, the largest security threats lie within the implied trust among IoT devices and in their communication. There are also issues with the interoperability of systems and scalability of the PKI implementation, which increase in conjunction with IoTs rapid growth. The applicability of IoT to all sectors (business, medicine, home, auto, etc.) means that the security breaches to these applications will also impact all sectors for example, a hacker could break into a home by accessing the data for a smart door, or obtain someones banking information. Ultimately it is important to evaluate and mitigate these security risks as IoT continues to grow and become an integral part of the daily transactions of human life.

Introduction

What is IoT?

The internet of things, or IoT, is defined as a network of interrelated computing devices or systems that enables connectivity, as well as the collection and exchange of data among them. The "things" in IoT can range from a person with a heart monitor implant to a home security system, to a self–driving car. All of these things are assigned a unique IP address and have the ability to collect information and autonomously share the information between devices. The AFCEA International Cyber Committee characterizes IoT devices as physical, connected, and smart [1]:

Physical: these devices are used every day by people and organizations to manage their lives and business processes.
Connected: IoT devices have the capability to connect users to processing services and shared data, as well as to one another.
Smart: devices, services, and networks receive, exchange, and analyze data in order to operate to some degree of autonomy

Many are calling the current boom of IoT the "Second Economy" or the "Industrial Internet". It is estimated that the current rise of IoT will bring about a market increase the size of trillions of dollars, and an estimated 50 billion of new connected devices. The rise of IoT is creating a new network of connectivity solutions that will improve healthcare, home energy consumption, transportation, and various other economic sectors [1].

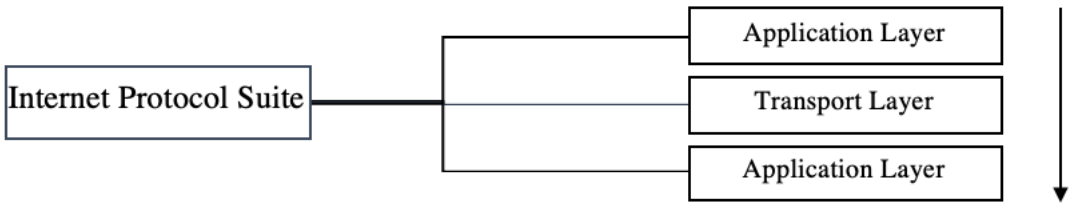
History

The idea of connected devices has existed since around the 1970s; known as pervasive or ubiquitous computing. This involved embedding microprocessors with some sort of computational capability into everyday objects [2]. Pervasive computing allowed for communication between

devices in a way that minimized a users need to interact with them. The goal of pervasive computing is to create devices that can receive, exchange, and analyze data – much like the "smart" aspect of IoT devices as defined by the AFCEA. IoT has primarily evolved out of pervasive computing, however, it is distinguished by the physical and connected characteristics defined above [5].

IoT Expansion and Vulnerabilities

The connected aspect of the devices in IoT is one of the most complex characteristics, as well as what compromises the majority of the research for this thesis. The internet protocol suite is the model and protocols used on the internet and other networks. Within the internet protocol suite are the application layer (top layer), transport layer (end–to–end communication services for applications – this will be further discussed in the SSL/TLS section), and finally the internet layer [5]. All internet transport protocols within the internet layer use the Internet Protocol (IP) to carry data from a given source host to a destination host. Host addressing and identification is accomplished using the IP addressing system, which is hierarchical [1].



IPv6 (Internet Protocol version 6) has contributed greatly to the expansion of IoT. IPv6 is the most recent version of the Internet Protocol, and it introduced the extension of IP addresses from 32 bits to 128 bits, resulting in trillions of unique available IP addresses. The adoption of IPv6 will allow larger address space for connected IoT devices rather than constraining them to the IPv4 (IPv6s predecessor) space [1, 5].

It is forecasted that there will be 50 billion to 200 billion connected things in use by 2020 – these IoT devices will span several categories such as transportation, consumer, and business [6]. The rise in IoT will bring about several new platform options, new variations on IT/IoT integration, new industry standards, and new applications. The increase of billions of devices in the IoT network correlates directly to an increase in the attack surface of these systems. Additionally, as IoT applications collect more data and information, security becomes a major challenge.

Security

Definition of Security

The term security subsumes the basic provision of security services, namely authentication, authorization, confidentiality, integrity, and non–repudiation [12, 13]:

Authentication: the process of verifying the identity of a device or entity at the end of a communication.
Authorization: once the identity is verified, it is necessary to check whether the entity has appropriate permissions to access the requested information, service, or system. Access control determines the respective rights of access for each entity or device.
Confidentiality: relates to the protection of the large capacities of information possessed by organizations and institutions. This information is to be disclosed exclusively to authorized parties.
Integrity: ensures that corruptions and attacks to a service or application are easily detected.
Non–repudiation: the aspect of security that ensures that no user of a service can falsely deny having used it, or deny accountability. Non–repudiation guarantees the legality of a transaction or process.

In terms of the interrelation of the security services, consider an employee at a company with an ID badge. This is an example of *authentication* – the badge likely consists of name, DOB, an ID number, and other personal information that serves to identify the employee. *Authorization* occurs when the employee successfully using their ID badge to enter and ultimately gain access to the company building. The badge may only work during certain days (i.e. the weekday), or it may be limited to certain times. If a badge is used at an incorrect time, it may beep or trigger a flashing red light, thus maintaining *integrity*. Finally, the employee must keep the ID badge to themselves and not share with others. If the badge is stolen, it must be reported immediately – this is *non–repudiation*.

Security services can be implemented either cryptographically or via non–cryptographic mechanisms. For the former, a solid key management infrastructure is fundamental cryptographic keys are strings of bits used for encoding and to ensure secure communications. This is further discussed in the section on encryption. For the latter, authorization and authentication must be properly codified as a function of device roles [7].

Security Considerations within the Context of IoT

Within the context of IoT, security must not only focus on the required services, but also on how these encompass the overall system and how the security functionalities are executed. IBM describes a secure IoT device as having the following capabilities [3]:

1. Ability to prevent system breaches or compromises: each respective layer of the IoT application must implement effective security measures.

2. Ability to support continuous monitoring: supplementing security measures with constant monitoring and system upgrades to protect against various types of attacks.

3. Ability to remain resilient: if a security attack is successfully executed, a system must attempt to mitigate damage and recover quickly.

Public Key Infrastructure

Cryptographic Security Measures (Encryption)

SSL/TLS

IoT is defined as a network of connected computing devices or systems and the collection and exchange of data among them. The transport layer of IoT provides the end-to-end communication services between these systems. SSL (secure sockets layer) are cryptographic protocols designed to provide secure communications and prevent breaches of data sent between two systems. These systems can be servers, devices, and clients. Ultimately, SSL, or the newer generation version, TLS (transport layer security) attempts to guarantee the privacy of data transmission between these two endpoints. This is done by making the data impossible to read between the endpoints by using various encryption algorithms [4, 5].

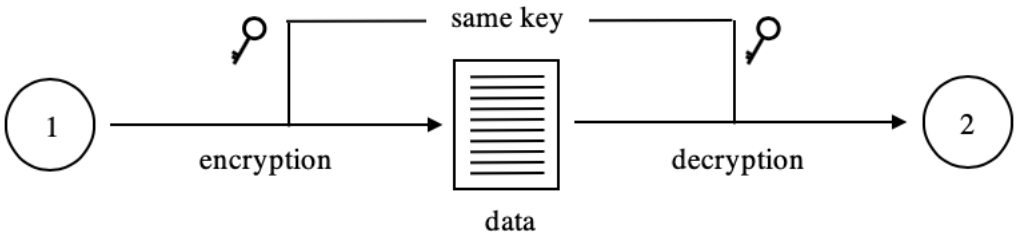
Introduction to Encryption Algorithms

Algorithms to authenticate and protect the privacy of large networks are a crucial part of SSL security and maintain the confidentiality and integrity of a connection. These digital encryption algorithms, known as public key cryptography, are encompassed by the greater public key infrastructure, or PKI. PKI is comprised of both cryptographic (ie encryption) and non–cryptographic security measures as outlined above in the types of security service implementations. Authentication and certification are the non–cryptographic security measures within PKI, and are discussed in their respective later section [10]. According to the 2018 Global PKI Trends Study sponsored by the Ponemon Institute LLC, the rapid growth in the use of IoT devices has had a directly proportional relationship with the use of PKI technologies. The study found that IoT is becoming a major driver for the use of PKI by surveying 1,688 IT and IT security practitioners in several countries across the world. This growing recognition indicates that PKI provides important core encryption and authentication technologies for IoT [8].

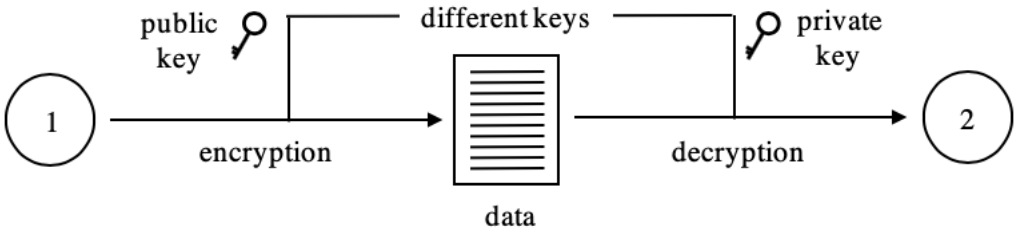
There are two different types of encryption – symmetric and asymmetric. Both methods involve keys, which are strings of bits used for encoding and to ensure secure communications. Asymmetric encryption is what is used in PKI/public key cryptography [7].

Symmetric Encryption vs. Asymmetric Encryption

Symmetric encryption is the simpler of the two techniques, and it uses shared private keys for the encryption and decryption of data sent from one party to another.



Asymmetric encryption, on the other hand, involves a public key that is a coded value with both public and private parts, that allows for encryption/decoding by one party [11].



One of the major drawbacks of symmetric encryption is that both parties involved in the communication are required to have access to the shared private key.

Asymmetric Encryption: Background

The first instance of asymmetric encryption was introduced theoretically in 1976, by Whitfield Diffie and Martin Hellman. Initially, a secure exchange of keys could only be done physically, such as via a paper key list. The Diffie–Hellman key exchange, on the other hand, allows two parties/systems with no prior knowledge to establish a shared secret exchange over an insecure channel or public network [8].

The Diffie–Hellman method was implemented in 1977 with the invention of the RSA Public Key Cryptosystem, which uses a matched pair of encryption and decryption keys. As their respective names indicate, the RSA public key is made publicly available, while the private key is kept private – not even shared among the two parties. For a private exchange to occur, the message sent from an author is encoded with the recipients public key using a mathematical algorithm that involves the factorization of the product of two large prime numbers, which makes it difficult for anyone other than the author or recipient to reverse engineer access to the private information [8, 11].

Asymmetric Encryption: Security

Once it is encrypted, the message can only be decoded with the recipients private key. RSA keys can work in either direction between two endpoints, and it is difficult for an outside party or observer to reverse engineer to obtain the private message [11]. In a public key cryptosystem, both a messages author and recipient must place an encryption procedure in a public file. This public file is a directory that contains the encryption procedure of each user. In contrast, the users details of their respective decryption procedure are kept secret. Both encryption and decryption procedures usually consist of two parts – a general method and an encryption key. The general method enciphers a message to obtain the enciphered form of the message, known as the ciphertext using a series of mathematical procedures. While the general method can be used by everyone, the security of the key exchange depends on the security of the key. Revealing the encryption algorithm ultimately reveals the key [11]. It is extremely inefficient to compute the decryption method despite the encryption method being revealed, because the number of messages required to test is incredibly large [12]. This property classifies the encryption and decryption methods as "trap–door one–way" functions. These are functions for which it is easy to compute in one direction but difficult in the other ("one–way") and for which the inverse functions are easy to compute with certain private information ("trap–door") [11, 12].

As mentioned, encrypting data and messages between endpoints directly offers solutions to two of the security services: confidentiality and integrity [10].

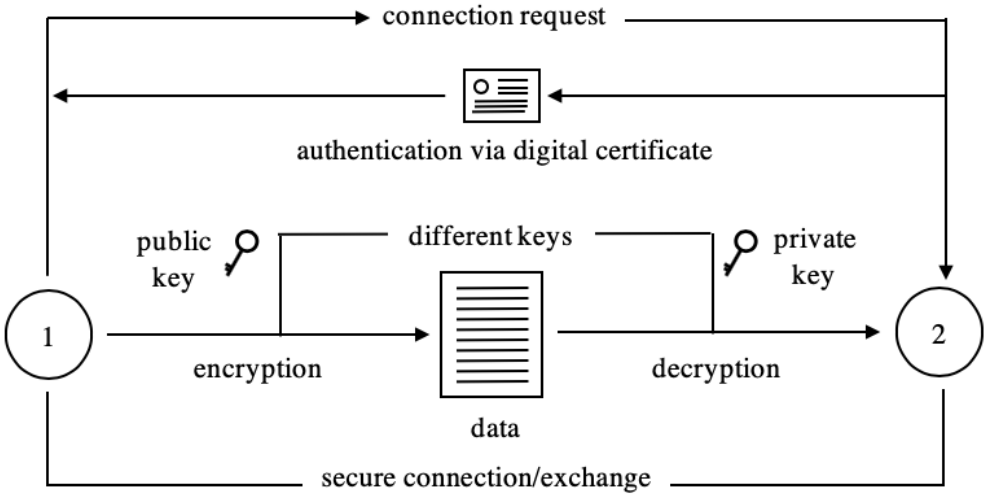
Confidentiality: the contents of the data/message are encrypted with an individuals public key, and can therefore only be decrypted with the individuals respective private key. This guarantees that only the intended recipient has the ability to view the message contents via decryption.
Integrity: part of the decryption process involves verifying that the contents of the original encrypted message from the sender and the decrypted message to the recipient. Any change to the original content of the encrypted message would therefore cause the decryption process to fail and maintaining the integrity of the communication.

Non-Cryptographic Security Measures (Authentication)

Certificates/Digital Signatures

While the public and private keys maintain secure communication and safeguard the data, certificates are used to establish the identities of the communicating parties. The digital certificate is used to certify an individual or institutions access to a respective public key – it is a small file that combines a cryptographic key with an individuals details. It also includes a validity period and digital signature, which confirm the contents of the certificate. If and only if the certificate is valid, and if examining software trusts the certificate issuer, the key can then be used to communicate with the subject of the certificate [3, 12].

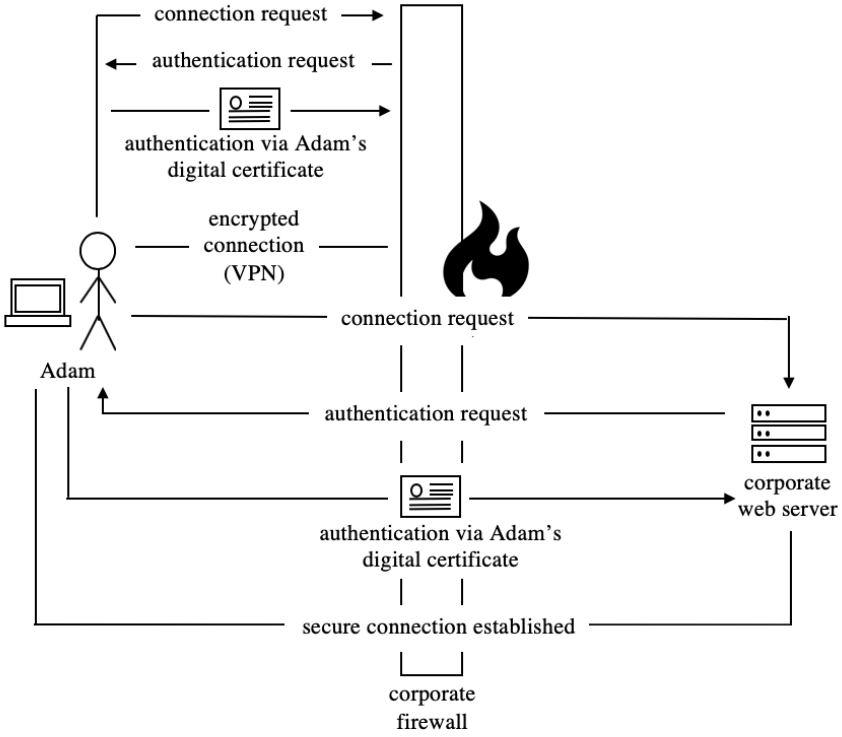
It is most common to access IoT devices and applications with a username and password. However, a PKI–enabled device will demand the digital certificate of an entity requesting access, rather than login information. When secure, certificates are stronger authentication mechanisms than the standard username/password combination – this is because unlike a password, a private key cannot be stolen. It would take years of brute force to compute with the length of a private key, in contrast to a standard 8–character password. Below is an addition to the asymmetric encryption diagram discussed in the previous section, extending the exchange between the two parties to include authentication:



Entities 1 and 2 in the diagram can be used to represent several scenarios – an iPhone and Apple Watch communicating a users fitness data, a Google Home connecting with a home security system to obtain security footage, or a remote client obtaining access to a private corporate web server. The systems capable of communication in the PKI framework within IoT can be classified as servers, devices, or clients [13].

Consider the example of a remote client obtaining authorized, secure access to a private corporate web server via digital certification – this is a client to server instance of communication. Oftentimes, these communications also involve a firewall, which is a security system that both monitors and controls network traffic. A firewall is at the boundary between two networks, and all traffic between the two networks must pass through the firewall [19]. Assume a client, Adam is working from home and would like to access the corporate web server for the business he works at using a personal computer [12,13]:

- Adam initiates a connection to the businesss network, which reaches the corporate firewall.
- The firewall requires authentication to allow the connection. Rather than entering a username/password, Adam's PC provides the firewall with his digital certificate.
- The firewall verifies the validity of the certificate, and ensures that Adam has authorization to access the businesss network.
- The connection is allowed to proceed through the firewall. If the firewall was instructed to do so, the connection between Adam and the firewall is encrypted, creating a VPN.
- Adam attempts to connect to a web server on the business's network. Because of the VPN, this connection goes through the firewall.
- The web server requires authentication to allow the connection, and again Adam's digital certificate is presented by the PC.
- If Adam has a valid certificate and is an authorized user, the connection is complete.



Constituents of a Certificate/X.509

Certificates are the backbone of authentication in PKI, the components for which are typically defined by the X.509. The X.509 certificate is the standard format for digital certificates. The X.500 standard was first issued in 1988 by the ITU Telecommunication Standardization Sector, and is now currently up to version 3 of X.509. A strict hierarchical system of CAs is assumed for issuing the certificates [14].

The process of issuing an X.509 certificate first begins with a request, known as a certificate signing request (CSR). A public/private key pair is generated; the private key is used for the signature, and the public key is used to verify said signature. Once verified, the CA issues a certificate binding the public key to the specific distinguished name associated with the request [13].

The actual contents of a certificate are expressed via Abstract Syntax Notation One (ASN.1), an interface description language for defining data structures that is easily serialized and deserialized, and are as follows [13, 14]: * = *optional field*

```
Certificate {
        Version Number
        Serial Number
        Signature Algorithm ID
        Issuer Name
        Validity Period {
                Not Before
                Not After }
        Subject Name
        Subject Public Key Information {
                Public Key Algorithm
                Subject Public Key }
        Issuer Unique Identifier*
        Subject Unique Identifier*
        Extensions* {
                E1
                E2
                ... }
        Certificate Signature Algorithm
        Certificate Signature }
```

Core Components of the Authentication Process

The certificate is not the only part of the authentication process; there are other core components to authentication within the PKI framework. These are [12, 13, 14]:

Certification Authority (CA): the CA is essentially the "center of the trust model supported by a PKI" [13]. CAs must digitally sign and issue certificates in order to bind the identity of an entity to its public key. CAs also check whether certificate requests conform to the policy under which they will be issued, and that they are submitted by an authorized RA (described below). Finally, CAs also handle certificate revocation, which is the process by which invalid certificates are withdrawn.

Registration Authority (RA): RAs are used in conjunction with CAs. The main obligation of the RA is to collect requests and examine and verify the credentials of end entities for authentication. This typically involves proof that the communicating entities actually possess the private key corresponding to the private key relating to the requested certificates. Approved requests are then passed on to the CA.

Certificate Revocation Lists (CRL): Certificate revocation lists are issued when a certificate is invalidated before it reaches its expiry date (ie before the end of the validity period). There are several instances for which a certificate may potentially become invalidated prior to expiry, such as if the private key is compromised. In such a case, a CRL issuer issues a certificate revocation list in order to inform users about certificates that have been revoked.

Certificate Repository (CR): the certificate repository is a directory service for storage and retrieval of all PKI-related information. Specifically, the information stored within a CR includes the certificates themselves and certificate revocation lists.

End Entity: the end entity is either the user of the digital certificate or the end user system that is the subject of the certificate.

The end entities in PKI are classified as *PKI users* because they either directly use the digital certificate or the system that is the subject of the certificate. Likewise, *PKI management entities* are the components of the authentication process that manage the issuance of certificates. Specifically, these are the CA, RA, and CRL issuer.

Certificate Issuance

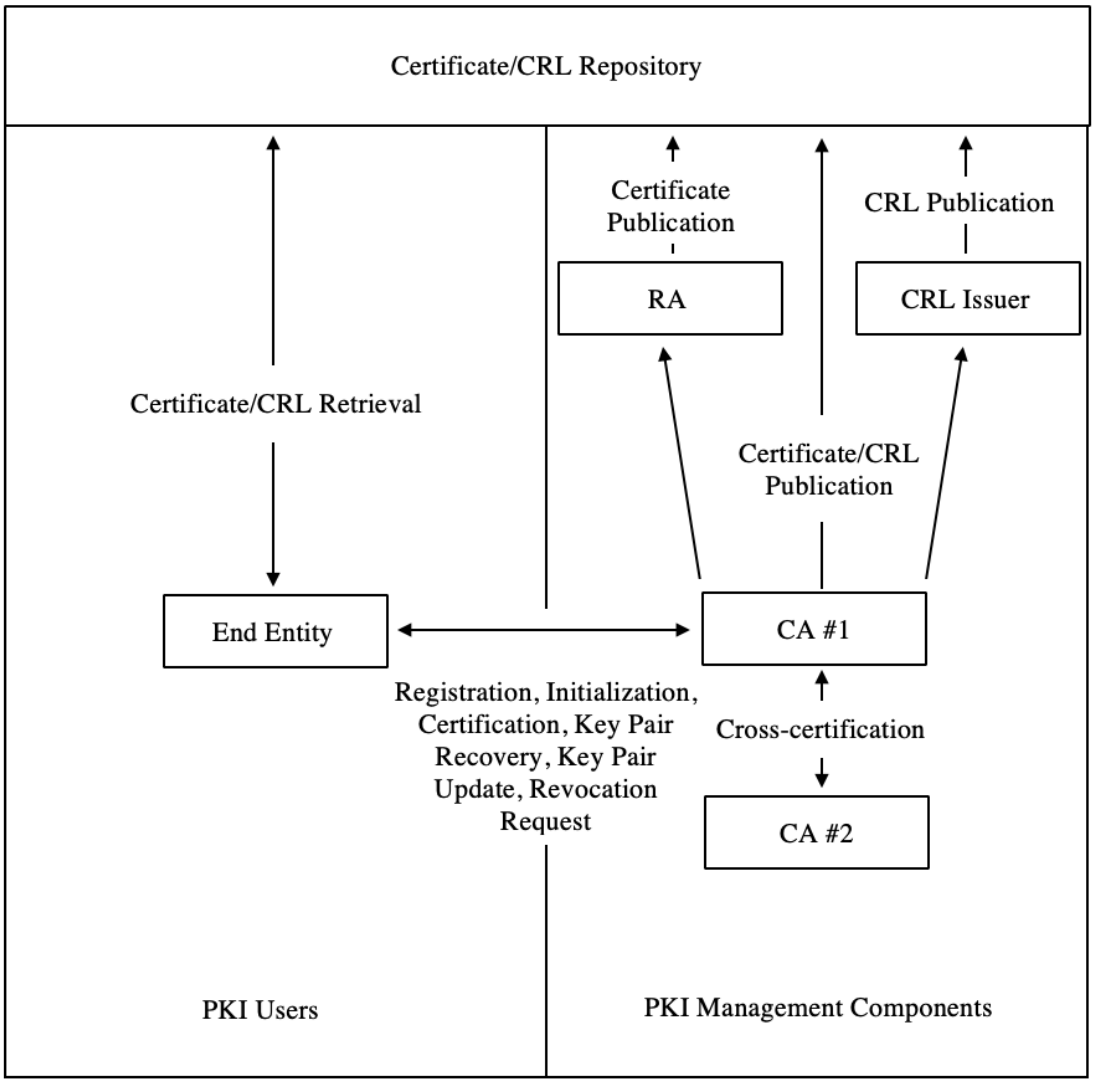
There are two main protocols involved with issuing an X.509 certificate which relate to the operational transactions (transactions between the PKI users and CR/CRL) and management transactions (transactions between PKI users and PKI management entities [illustrated below]) – these protocols are the operational protocols and management protocols, respectively [14].

Operational protocols are required to deliver certificates, CRLs, and status information to client systems that user certificates. These protocols require conventions on certificate and CRL delivery, such as distribution procedures. Other specifications such as message formats and environment support are also included within the operational protocols [14].

Management protocols are used to support interactions between PKI user and management entities. There are seven main functions that are shaped by management protocols [13]:

Registration: registration must occur prior to a CA issuing a certificate; i.e. the process by which a user first makes itself known to a CA. This is done directly or through an RA.
Initialization: the process of installing all materials required for the client system to operate securely, including initializing a client with its key pairs.
Certification: the process in which a CA issues a certificate for a user's public key and either posts it in the CR or returns the certificate to the users client system.
Key pair recovery: optional backup of user client key materials for recovery purposes; requires an online protocol exchange in order to support recovery.
Key pair update: involves the regular update of key pairs and the issuance of new certificates.
Revocation Request: a revocation request for an invalid certificate.
Cross–certification: a cross–certificate is a certificate issued by one CA to another that contains a CA signature key for issuing certificates.

A visual of the interactions between the core components of the authentication process, and the operational/management protocols are illustrated below [12,13,14]:



Security Risks within PKI

The basic security services, "confidentiality, authentication, integrity, authorization, non–repudiation, and availability, as well as augmented services, such as duplicate detection" [12] are theoretically subsumed within PKI, however it is far from being the sole solution to communication and connectivity within IoT. Furthermore, although research indicates that PKI is the leading source of encryption and authentication technologies for IoT [8], the current status of PKI still has problems, specifically related to interoperability and scalability, certification path processing, and identity in certificates [13, 16]. The implicit trust, or unquestionable reliance on the components of the authentication framework, also poses security risks [15, 17].

Before outlining specific issues and disadvantages, it is crucial to conceptualize trust as applied to IoT systems as well as components of PKI. In *An Anatomy of Trust in Public Key Infrastructure*, the following definition is used for the idea of trust between two parties:

> "Trust is a mental state compromising (1) expectancy the trustor expects a specific behavior from the trustee (such as providing valid information or effectively performing cooperative actions); (2) belief the trustor believes that the expected behavior occurs, based on the evidence of the trustees competence, integrity, and goodwill; (3) willingness to take risk the trustor is willing to take risk for that belief" [15].

Interoperability and Scalability within PKI

By definition, IoT is a network of interrelated computing devices or systems that enables connectivity, as well as the collection and exchange of data among them [1]. Interoperability, or the ability of computer systems or software to exchange and make use of information, is crucial to the communication and connectivity within IoT systems. However, "the lack of interoperability across systems is one of the most significant hurdles to PKI's widespread adoption" [13]. The user-defined extensions that are a part of the X.509 certificate allow users to extend the original format to fit their needs, however these extensions have led to communication problems among different PKIs with varying certificate formats. The lack of uniformity has led to encoding/decoding issues, boundary and range issues, naming mismatches, and overall inconsistency with certificates and CRLs [13, 14].

Compatibility and interoperability are also important when looking at the management protocols for cross–certification. Although the idea of cross–certification is simple (a certificate issued by one CA to another that contains a CA signature key for issuance), ensuring that two certificates are consistent requires models for trust management, i.e. how different CAs relate and extend trust to each other. There are two main approaches to cross–certification with different trust models (i.e. different distributions of trust), each with advantages and disadvantages [13].

One such approach is the Hierarchical approach, which is modeled around a central CA that is the primary source of trust. The central CA can delegate trust to secondary CAs. As a result of this hierarchy and the importance of the central CA, its security procedures must ensure that it deserves the trust being placed in it. The biggest problem with the hierarchical approach is that if there were to be an attack on the private key of the central CA, the results would be widespread. Additionally, the hierarchical approach is often hard to scale.

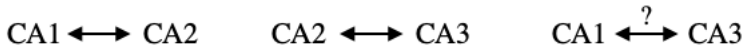
Another approach is the Mesh approach. The Mesh approach assumes that each pair of CAs establishes a cross–certification between themselves. This is also sometimes known as a peer-to-peer CA model. Like the hierarchical approach, the mesh approach is also hard to scale given that each CA has a cross–certificate relationship with multiple CAs, the number of cross certificates required expands geometrically with the addition of CAs. The computation required by end entities to construct and validate certificate paths also increases as a result.

Certification Path Processing (CPP)

Like the distribution of trust discussed in the previous section, the processing of trust is equally as complex. "Before a certificate can be used, it must be validated. In order to validate such a certificate, a chain of certificates or a certification path between the certificate and an established point of trust must be established, and every certificate within that path must be checked. This process is referred to as certification path processing" [13]. CPP can be characterized by two phases: certificate path construction and validation.

The starting CA for validation is known as the trust anchor – it can be viewed as the root of the hierarchical tree. Unilateral cross–certification occurs when a root CA certifies its subordinate or child. In the peer–to–peer/mesh model on the other hand, the trust anchor is the entitys local CA, which is autonomous and does not depend on a root like in the hierarchical model. This is known as bilateral cross-certification [13].

The issue that arises with CPP is how to decide whether or not a path exists between one CA and another, i.e. whether the certificate from a particular CA can be trusted. This involves integrating intelligence into what should be an automated process. For example, consider three certificate authorities CA1, CA2, CA3. "Undesirable trust cascades" must be prevented, e.g. if CA1 trusts CA2 and CA2 trusts CA3, CA1 should not necessarily nor implicitly trust CA3 [13, 15].



Vulnerabilities with Implicit Trust

Implicit trust is a recurring component not just with the communications between the components in PKI, but within the trustworthiness of the components themselves. The trust between end entities and CAs as well as the basis for this trust, the trust between end entities and their respective keys, and the fully confirmed identity of an end entity are all assumed to be valid. The security of the verification process is also implicitly trusted [17]. Protection of the keys is particularly important due to the significance of digital non–repudiation, and all of these components must be authorized as per the definition of security. This implicit trust within the PKI components makes it easier for attackers to impersonate CAs and end entities, and to issue fraudulent certificates.

Proposed Solutions

The following are potential solutions and recommendations to the above problems within the PKI infrastructure:

Interoperability and Scalability within PKI – Certificate Policies/ Bridge Approach

In response to the inconsistencies and interoperability problems with certain user-defined components within X.509, certain profiles have been developed that enable the notification, identification, and understanding of certificate extensions. Another improvement to interoperability has been the improved use of Certificate Policies (CP), and policy mappings, which are used to verify the interoperation between organizations with different CA policies.

A CP is formally defined as "a named set of rules that indicates the applicability of a certificate to a particular community and/or class of applications with security requirements" [20]. CPs do so by either setting forth requirements for certificate usage and requirements on members of a community, or by identifying a set of applications or uses for certificates with a certain level of security, respectively. An example CP might indicate the applicability of a particular certificate to the authentication of two businesses engaging in transactions that are within a given price range. CPs are represented in certificates by object identifiers, or OIDs, which are registered unique numbers that are assigned to certain organizations.

Policy mappings, which are extensions that are only used in CA certificates, allow a CA to indicate that certain policies in its own domain can be considered equivalent to certain other policies in the subject certification authority's domain.

In terms of cross-certification, both the Hierarchical Approach and the Mesh Approach for cross-certification have their advantages, as well as a distinct set of disadvantages. The Bridge Approach is a hybrid of the two that makes use of the benefits from both. The Bridge Approach involves a trusted third party "that acts as a hub linking the CAs together, thus avoiding the need for CAs to enter into bilateral contracts with each CA that it wants to interoperate with" [13]. Bridge Approach is an improvement from the Hierarchical Approach in that there is no trust anchor, so a change in the public key of the root node does not impact as many CAs in the Bridge Approach as it would in the Hierarchical Approach. The Bridge Approach also reduces the overhead involved with the Mesh Approach, which requires disproportionate management of cross–certificate pairs. Finally, the bridge model allows for several autonomous PKIs to interoperate without a loss of their autonomy [13].

Certification Path Processing (CPP) – Policy Constraints

Consider the previous example involving the three certificate authorities CA1, CA2, CA3. One way to prevent the undesirable trust cascades outlined in the example is to put in place a set of policies for each CA. So, CA1 would have a set of policies put in place that determine whether or not it should trust CA3, rather than automatically trusting it due to its trust with CA2.

Specific policy constraints are accomplished using path, name, and policy constraints [13, 20]:

Path length constraints: path length constraints can be used to limit the depth of the tree by limiting the number of children/subordinate CAs that can be added to the hierarchy, or to define the maximum number of times that trust can be placed in CAs.
Name constraints: these allow trust to be limited on the basis of all or part of the specific entity name that the certificate has been issued to. With name constraints, a CA can conform to certain conditions based off of their respective name.
Policy constraints: policy constraints can limit a CA's trust to only CAs that match predefined criteria with respect to the policy fields within their certificates.

Ultimately, these constraints provide organizations with a way to tailor the trust between CAs to reflect explicit rather than implicit relationships.

Vulnerabilities within Implicit Trust – Pinning

Public Key Pinning (PKP) is a security mechanism for detecting and blocking man–in–the–middle attacks, as well as the mitigation of fraudulent communications and components such as fraudulent CAs or certificates [18]. Pinning effectively removes the "conference of trust" [16].

Pinning works by "associating a host with their expected X.509 certificate or public key. Once a certificate or public key is known or seen for a host, the certificate or public key is associated or 'pinned' to the host. If more than one certificate or public key is acceptable, then the program holds a 'pinset' containing the acceptable information" [16]. A pin directive is what specifies the way to indicate the cryptographic identity that should be bound to a particular host. As mentioned, both public keys and certificates can be pinned, ultimately leveraging the knowledge of the pre–existing relationship between a user and an organization [16, 18].

Certificates are easier to pin than public keys, but nonetheless come with disadvantages. When certificates expire, the pinning application must be updated. However, if a certificate is regularly rotated, then the pinning application must be regularly updated as a result. Google, for example, rotates its certificates, although the underlying public keys within the certificate remain static. Public key pinning is more flexible in contrast to certificate pinning but it involves the extra steps necessary to extract the public key from the certificate itself [18, 20].

Conlusion

It is evident that the rapid growth in the use of IoT devices has had a directly proportional relationship with the use of PKI technologies, and the rise of interconnected systems and devices has led to an increase in their attack surface. Currently, the largest security threats lie within the implied trust among IoT devices and in their communication. There are also issues with the interoperability of systems and scalability of the PKI implementation, which increase in conjunction with IoT's rapid growth.

The applicability of IoT to all sectors (business, medicine, home, auto, etc.) means that the security breaches to these applications will also impact all sectors – for example, a hacker could break into a home by accessing the data for a smart door, or obtain someones banking information.

Upon research, the following recommendations have been made in terms of the three main risks:

- Interoperability and scalability within PKI
  - Use of Certificate Policies (CP) and policy mappings to verify the interoperation between organizations with varying CA policies.
  - Bridge approach, which involves trusted third party; hybrid between hierarchical and mesh approaches
- Certification Path Processing (CPP)
  - Policy constraints – provide constraints on path length, name, and policy
- Vulnerabilities with Implicit Trust
  - Public key pinning removes the "conference of trust," and associates a host with its respective X.509 certificate or public key, i.e. "pinning" them. Both public keys and certificates can be pinned.

It is evident that most of the security solutions to the risks within PKI involve further specifications and constraints i.e. the Bridge Approach, use of CP and CPP, and the additional process of pinning hosts and either certificates or keys. Ultimately, the improvement of PKI warrants further investigation from security researchers with the growth of IoT and as PKI becomes an increasingly integral security solution for IoT systems and devices.

Bibliography

[1] Chris Folk et al., *The Security Implications of the Internet of Things,*, AFCEA National Cyber Committee

[2] Maede Zolanveri, *IoT Security: A Survey*, Washington University in St. Louis Department of Computer Science and Engineering

[3] Amitranjan Gantait et al., *Securing IoT devices and gateways*, IBM Developer

[4] Internet Engineering Task Force (IETF), *Requirements for Internet Hosts Communication Layers*

[5] Noura Aleisa and Karen Renaud, *Privacy of the Internet of Things: A Systematic Literature Review (Extended Discussion)*, School of Computing Science, University of Glasgow

[6] Gartner, Inc. Newsroom, *In 2020, 25 Billion Connected "Things" Will Be in Use*

[7] O. Garcia Morchon et al., *Security Considerations in the IP-based Internet of Things*, Philips Research, University of Glasgow, Struik Consultancy

[8] *2018 Global PKI Trends Study*, Thales eSecurity, Ponemon Institute LLC.

[9] David Adrian et al., *Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice*, Microsoft Research, University of Pennsylvania, Johns Hopkins, University of Michigan

[10] *What is Public Key Cryptography?*, GlobalSign GMO Internet Group

[11] RL. Rivest et al., *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*

[12] SANS Institute, *A Business Perspective on PKI: Why Many PKI Implementations Fail, and Success Factors to Consider*

[13] Tejan Balakrishnan, *Current Status of Public Key Infrastructures*, University of Leeds Minerva

[14] D. Cooper et al., *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, NIST, Microsoft, Trinity College Dublin, Entrust, Vigil Security

[15] Jingwei Huang, David M. Nicol, *An Anatomy of Trust in Public Key Infrastructure*, Old Dominion University, University of Illinois at Urbana-Champaign

[16] Jeffrey Walton et al., *Certificate and Public Key Pinning, Open Web Application Security Project (OWASP)*

[17] Carl Ellison, Bruce Schneier, *Ten Risks of PKI: What Youre not Being Told about Public Key Infrastructure*, Computer Security Journal vol. XVI no. 1

[18] C. Evans et al., *Public Key Pinning Extension for HTTP*, Google, Inc.

[19] Kenneth Ingham, Stephanie Forrest, *A History and Survey of Network Firewalls*, Kenneth Ingham Consulting, University of New Mexico

[20] S. Chokani et al., *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Policies Framework*, Verisign, Grion Security Solutions, Cooley Godward LLP, McCarter English, Infoliance