7-2-2020

Efficient Key Management Schemes for Smart Grid

mumin cebe
mcebe002@fiu.edu

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

EFFICIENT KEY MANAGEMENT SCHEMES FOR SMART GRID

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL AND COMPUTER ENGINEERING

by

Mumin Cebe

2020

To: Dean John Volakis
College of Engineering and Computing

This dissertation, written by Mumin Cebe, and entitled Efficient Key Management Schemes for Smart Grid, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

A. Selcuk Uluagac

Ahmed S. Ibrahim

Jason Liu

Kemal Akkaya, Major Professor

Date of Defense: July 2, 2020

The dissertation of Mumin Cebe is approved.

Dean John Volakis
College of Engineering and Computing

Andres G. Gil
Vice President for Research and Economic Development
and Dean of University Graduate School

Florida International University, 2020

DEDICATION

To my wife, son, and daughter.

ACKNOWLEDGMENTS

I want to send my most heartfelt and sincere thanks to my advisor Dr. Kemal Akkaya, for his valuable leadership and huge support during all these years at FIU. I would not be able to make this work happen without his help. I am also indebted for the chance he gave me to work for Advanced Wireless and Security Lab (ADWISE).

I want to extend my thankfulness to Dr. Selcuk Uluagac. His dynamic approach and valuable encouragement helped me to determine my path.

I also want to reveal my appreciation to Enes Erdin, who has provided meaningful help to improve my writing and communication skills. Besides, I am always indebted to him for his sincere friendship and invaluable support.

I also appreciate Dr. Ahmed S. Ibrahim and Dr. Jason Liu for serving on my dissertation committee and generously giving their time to advice and help.

Last, but not least, I want to thank my wife, Nuran. There was a long period when I was pretty directionless, and she always had a way of putting a positive spin on things. Whenever I got down on myself, she could convince me to continue for success.

I also want to acknowledge the support by Department of Energy. This dissertation is completely based on the work supported by Department of Energy under the Award Numbers, DE-OE0000779.

ABSTRACT OF THE DISSERTATION

EFFICIENT KEY MANAGEMENT SCHEMES FOR SMART GRID

by

Mumin Cebe

Florida International University, 2020

Miami, Florida

Professor Kemal Akkaya, Major Professor

With the increasing digitization of different components of Smart Grid by incorporating smart(er) devices, there is an ongoing effort to deploy them for various applications. However, if these devices are compromised, they can reveal sensitive information from such systems. Therefore, securing them against cyber-attacks may represent the first step towards the protection of the critical infrastructure. Nevertheless, realization of the desirable security features such as confidentiality, integrity and authentication relies entirely on cryptographic keys that can be either symmetric or asymmetric. A major need, along with this, is to deal with managing these keys for a large number of devices in Smart Grid. While such key management can be easily addressed by transferring the existing protocols to Smart Grid domain, this is not an easy task, as one needs to deal with the limitations of the current communication infrastructures and resource-constrained devices in Smart Grid. In general, effective mechanisms for Smart Grid security must guarantee the security of the applications by managing (1) key revocation; and (2) key exchange. Moreover, such management should be provided without compromising the general performance of the Smart Grid applications and thus needs to incur minimal overhead to Smart Grid systems. This dissertation aims to fill this gap by proposing specialized key management techniques for resource and communication constrained Smart Grid environments. Specifically, motivated by the need of reducing the revocation management overhead, we first present a distributed public key revocation management scheme for Advanced Metering Infrastructure (AMI) by utilizing distributed hash trees (DHTs). The basic idea is to enable sharing of the burden among smart meters to reduce the overall overhead. Second, we propose another revocation management scheme by utilizing cryptographic accumulators, which reduces the space requirements for revocation information significantly. Finally, we turn our attention to symmetric key exchange problem and propose a 0-Round Trip Time (RTT) message exchange scheme to minimize the message exchanges. This scheme enables a lightweight yet secure symmetric key-exchange between field devices and the control center in Smart Gird by utilizing a dynamic hash chain mechanism. The evaluation of the proposed approaches show that they significantly out-perform existing conventional approaches.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF FIGURES

FIGURE PAGE

CHAPTER 1

INTRODUCTION

The existing power grid is currently going through a major transformation to enhance its reliability, resiliency and efficiency by enabling networks of intelligent electronic devices, distributed generators, and dispersed loads [Far10], which is referred to as *Smart(er) Grid*. The ongoing transformation also enhances with IoT integration [LTCP16,SCRC19,KGM^{+}19] due to benefits from the information provided by IoT devices such as voltage, current, temperature, etc. for real-time monitoring of electricity generation, transmission lines, and distribution. There are two enablers in this transformation: 1) enabling new devices and communication technologies, and 2) integration of the new smart devices to the existing communication infrastructure.

Advanced Metering Infrastructure (AMI) network is one of the renewed components of Smart Grid that falls into the first category where devices and communication infrastructure are being updated together. For instance, utilities are upgrading their old powerline-based communication infrastructure to a wireless mesh infras-

AMI	Advanced Metering Infrastructure	**IEEE**	Institute of Electrical and Electronics Engineers
AODV	Ad-hoc On-demand Distance Vector	**IETF**	Internet Engineering Task Force
CA	Certificate Authority	**IoT**	Internet of Things
CC	Control Center	**LTE**	Long-Term Evolution
CRL	Certificate Revocation List	**MAC**	Message Authentication Code
DH	Diffie–Hellman	**NIST**	National Institute of Standards and Technology
DHT	Distributed Hash Table	**OCSP**	Online Certificate Status Protocol
DoS	Denial of Service	**PKI**	Public Key Infrastructure
DTLS	Datagram Transport Layer Security	**PSK**	Pre-Shared Key
ECC	Elliptic Curve Cryptography	**RSA**	Rivest–Shamir–Adleman
ECDSA	Elliptic Curve Digital Signature Algorithm	**RTT**	Round Trip Time
FAN	Field Area Network	**RTU**	Remote Terminal Unit
HAN	Home Area Network	**SCADA**	Supervisory control and data acquisition
HES	Head-End Server	**TCP**	Transmission Control Protocol
HMAC	Hash-based Message Authentication Code	**TLS**	Transport Layer Security
IED	Intelligent Electronic Device	**UC**	Utility Company
HWMP	Hybrid Wireless Mesh Protocol	**UDP**	User Datagram Protocol
IED	Intelligent Electronic Device	**WAN**	Wide Area Network

Table 1.1: List of Abbreviations

tructure to utilize two-way communication capability for AMI [SAU12]. This new communication infrastructure brings advantages in terms of costs and management and thus is becoming more common. Besides that, smart meters are the new devices of the transformation which are connected to each other and the utility company using this new mesh-based communication infrastructure. AMI's upgraded structure is generally used to facilitate reduction of peak demand by monitoring the power demands over short periods and providing various ratings and effective management based on remote metering data.

On top of AMI, power grid also has an Supervisory Control and Data Acquisition (SCADA) control system to convey information and control deployed Intelligent electronic devices (IED) and remote terminal units (RTU). SCADA uses existing communication infrastructure that covers all the geographic areas where these devices are planted with low density. This is typically through a Wide-Area Network (WAN) technology such as 2.5G, and other 900 MhZ low-bandwidth radio communications [GSK$^+$11]. While the addition of the new devices (e.g., IoT devices) to the grid is relatively easier due to their accessibility, upgrading the underlying communication infrastructure is much harder due to sparse and distributed nature of SCADA. Therefore, this transformation is carried out by mostly not touching the existing communication infrastructure. Since the communication infrastructure relies on some very low-bandwidth technologies where the bandwidth is only in the order of kilobits, any new application introduced by IoT devices to improve the efficiency of Smart Grid needs to take this restriction into account.

Considering the increasing number of new applications by the on-going transformation of Smart Grid, meeting the security of these applications has become critical. As a matter of fact, Smart Grid does not have different requirements from the conventional systems as confidentiality, authentication, message integrity, access control, and non-repudiation are all needed to secure it. For instance, confidentiality is required in order to prevent exposure of customer's private data to unauthorized parties while integrity is necessary to ensure that power readings are not changed for billing fraud. Furthermore, authentication is crucial to prevent any devices from communicating with other Smart Grid components and devices. As in the case of conventional networks, these requirements can be met by using either symmetric or asymmetric key cryptography. However, in both cases management of the keys is a major issue in terms of efficiency and cost. The efficiency and cost of key management are directly related to creation, renewal, distribution, and revocation of those keys. Thus, in this dissertation, by considering the resource-limited smart devices and their limited communication infrastructure, we propose effective and efficient key management mechanisms that focus on two areas: (1) the lightweight revocation management of keys in public-key settings (i.e., asymmetric keys); (2) the lightweight key-exchange (i.e., key renewal) mechanism in symmetric key settings.

1.1 Lightweight Revocation Management

According to National Institute of Standards and Technology (NIST), Public Key Infrastructure (PKI) is the only proper way to provide the security of AMI considering the number of possible communicating pairs of devices [NIS14]. Thus, companies such as Landis&Gyr and Silver Spring Networks use PKI in AMI to provide security for millions of smart meters in the US [lan15]. In such PKI settings, the public-keys for smart meters are in the form of *certificates*, which are issued by Certificate Authorities (CAs). The employment of PKI in AMI requires management of these certificates, which include their distribution and revocation. In particular, the management of certificate revocation and its associated overhead for AMI is critical. [MMAS15].

Problem: Several reasons necessitate revoking certificates, such as key compromise, excluding malicious meters, renewing devices, etc. Besides, if there is a vulnerability in the algorithms or libraries that are used in certificate creation, a massive number of revocations may additionally occur. For instance, a recent discovery of a chip deficiency on RSA key generation caused revocation of more than 700K certificates of devices that deployed this specific chip [NSS$^+$17] and renowned heartbleed vulnerability caused the revocation of millions of certificates, immediately [DKA$^+$14]. Thus, to establish secure communication, a smart meter should check the status of the other smart meter's certificate against a certificate revocation list (CRL) that keeps all revoked certificates. Considering the large number of smart meters in an AMI and the fact that the expiration period can be even lifelong in particular applications [lan15], the CRL size will be huge. Consequently, revocation management becomes a burden for the AMI infrastructure which is typically restricted in terms of bandwidth. This overhead is particularly important since the reliability and efficiency of AMI data communication are crucial for the functionality of the Smart Grid. Considering the potential impact on the performance of AMI applications [MMAS15], handling the overhead of revocation management is essential.

Existing Solutions: Certificate revocation management is commonly handled by utilizing CRL that is stored in the smart meters. The status of a smart meter is determined by checking whether its certificate is listed in the CRL or not. An alternative method would be to store the CRL in a remote server as in the case of Online certificate status protocols (OCSPs) [GSM$^+$13]. In OCSP, an online and interactive certificate status server stores revocation information. Thus, each time a query is sent to the server to check the status of the certificate. While OCSP-like approaches can be advantageous on Internet communications, employing them for AMI is not attractive since it will require access to a remote server each time. In this regard, another alternative would be to use OCSP *stapling* [Pet13], where the smart meters query the OCSP server at certain intervals and obtain a signed timestamped OCSP response which is directly signed by the certificate authority is included (”stapled”) in the certificate. Again, this approach also needs frequent access to a remote server. Moreover, the ’stapled’ certificates should be downloaded frequently by smart meters to ensure security, and this will create additional traffic overhead on the AMI, which affects applications such as demand response or outage management.

Our Solutions: To address the aforementioned problem, in this dissertation, we introduce our proposed solutions for two different use cases to handle the associated revocation management overhead as follows:

- In the first use case, we consider Advanced Metering Infrastructure (AMI), which is used in the service of many Smart City applications such as gas and water data collection or electric vehicle charging. As security of communications between AMI and Smart City applications can be provided by employing PKI, there are still challenges regarding the revocation management considering the potential size of CRLs. Motivated by the need to keep the CRL distribution and storage cost-effective and scalable for such use case, we present a distributed CRL management scheme by utilizing distributed hash trees (DHTs) [SMK$^+$01], in Chapter 4. DHTs which have been widely employed in P2P networks serve as a quick lookup service where the data is distributed to multiple nodes. Our solution utilizes DHTs to provide scalable and efficient lookup service for a revoked certificate. The basic idea is to share the burden of storage of CRLs among all the smart meters by exploiting the convenient wireless communication capability of the smart meters among each other. Using DHTs not only reduces the space requirements for CRLs but also makes the CRL updates more convenient.

- In the second one, smart meters have issued certificates to accomplish a typical Smart Grid *Demand-response* application which requires mutual authentication if multi-hop transmission is in place. Since the certificate revocation is critical and has potential to impact the performance of AMI applications significantly, this time, we focus on the management of the revoked certificates of smart meters in AMI. We propose a cryptographic accumulator based approach, in Chapter 5, to decrease the related revocation overhead. The accumulator is a cryptographic tool which is able to digest a set into a single value like well-known cryptographic hash functions. But, it also provides a mechanism to check whether an individual element is in the set or not. This property differs it from the conventional hash functions by enabling to be used for membership testing. Our approach utilizes this unique feature of the accumulator and employs it to reduce the space requirements for revocation information significantly and thus provides efficient distribution of such information within AMI.

1.2 Efficient Key Agreement Protocol

As stated before, the security of Smart Grid is provided either utilizing asymmetric or symmetric keys. The employed type of keys solely depends on the requirements of the application and deployed system. For instance, utility companies employs symmetric keys to ensure the security of their SCADA systems. Considering the increasing number of newly integrated IoT devices to SCADA, the overhead related to key management is consistently increasing since it is a requirement to employ proper key management while new devices are being added to the grid. In addition to that, when the low-bandwidth of the communication infrastructure is taken into consideration, the overhead of the applied key management reaches a critical level that significantly affects the health of Smart Grid.

Problem: As the communication between IoT devices and control center needs to guarantee (at least) integrity and authentication, key management becomes a major challenge due to its additional overhead on narrow-band communication infrastructures that are part of the current power grid [LTQ13]. Hence, using existing key management protocols that are designed for resource-rich communication networks is not feasible as they will congest the links easily, hindering the actual data transfer and eventually causing longer delays which may not be acceptable for time-sensitive power flow control. In general, effective security mechanisms for Smart Grid domain must guarantee the security of any applications running on it without compromising their performance. Due to such communication infrastructure challenges to run security algorithms/protocols in general and key management schemes in particular, the utilities follow a naive solution by using the same key for a long period of time to avoid overhead of updating symmetric keys. Obviously, this is very problematic as compromising one key means compromising forward secrecy (future data) during that period.

Existing Solutions: Looking at the literature, while there has been a lot of focus on designing computationally efficient security solutions in any resource-constrained domain [Raz13], the extremely constrained infrastructure has never been considered since broadband links are becoming part of the cyberspace whether it be wireless or wired. Nonetheless, recently Google introduced the QUIC [Goo16] protocol for efficiency in its client-server session key management. This protocol ensured 0-RTT, meaning that without a complete round trip message from a client to server, the encryption can start with the new shared key. QUIC also triggered new developments in the Transport Layer Security (TLS 1.3) and Datagram Transport Layer Security (DTLS 1.3) [Res, RTM20] came with similar features. Despite their efficiency, these protocols have issues with certain attacks such as replay attacks that prevent them from being directly used in Smart Grid domain. In particular, if 0-RTT is desired, replay attack resistance is not possible with current solutions.

Our Solution: Considering a legacy radio communication infrastructure with bandwidths in the order of kilobits, in Chapter 6, we aim to enable essential security services in Smart Grid via a lightweight key agreement scheme. Specifically, the proposed scheme provides mutual authentication, key agreement, and key refreshment by utilizing a 0-Round Trip Time (RTT) message exchange that relies neither on certificates nor session resumption. It depends on dynamic hash chains concept to enable authentication and prevent any replay attacks between field devices and control center. The evaluations results show that the proposed scheme significantly out-performs other conventional approaches and is suitable for Smart Grid legacy infrastructure.

1.3 Organization of the Dissertation

The remainder of the dissertation is organized as follows. In the following chapter, we give a brief background about the dissertation's primary building blocks. It is followed by a thorough literature review on which every single work in this dissertation depends. In Chapter 4, we propose a DHT based revocation management for HAN and AMI integration. In Chapter 5, we present our cryptographic accumulator based solution to relieve the overhead of CRLs for AMI. In Chapter 6, we introduce a 0-RTT key-exchange protocol and define its characteristics. Finally, we conclude the dissertation and discuss some future works for follow-up studies in Chapter 7.

CHAPTER 2

PRELIMINARIES

In this chapter, we establish some background about Smart Grid particularly its communication infrastructure and a general overview of cryptographic keys and their management.

2.1 Multi-tier Network Structure of Smart Grid

Before moving into the how key management in Smart Grid can be achieved, we first briefly explain the network structure of Smart Grid that consists of three major subnetworks: 1) home area network (HAN); 2) Advanced Metering Infrastructure (AMI); and 3) Field Area Network (FAN). A typical Smart Grid network showing the multi-tier structure of it is depicted in Figure 2.1. Under this network structure, different applications run simultaneously. Thus, it is critical to ensure the security

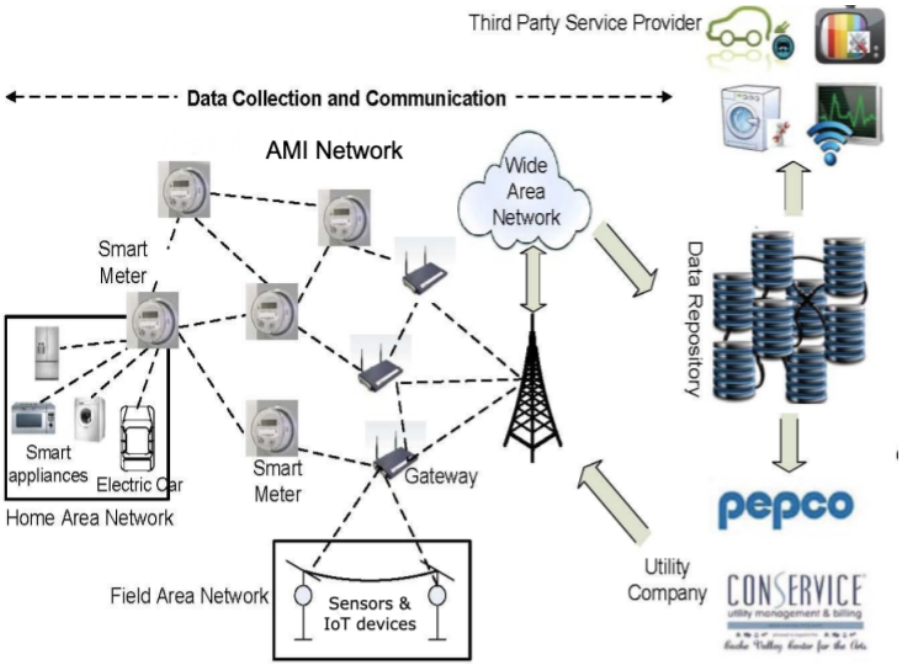


Figure 2.1: A sample multi-tier communication network of Smart Grid.

of these applications while taking into account the characteristics of each network tier of Smart Grid.

2.1.1 Home Area Network (HAN)

HAN is located within the perimeter of the customer domain and provides a control/monitoring ability to home appliances by the utility company. In a futuristic point-of-view, each home appliance sends its power demands to the smart meter for different Smart Grid applications, which enables an automation infrastructure to allow efficient monitoring and control applications, and demand-response applications.

The data generated from each device provide an active base for managing the load profile of the power grid. For instance, in a smart city setting, it enables a *controllable load* for large appliances such as air conditioners, washers and dryers, stoves. With the help of detailed information from these devices, such as the expected demand, duration of the usage, and availability of the appliance, the utility company can manage the load by managing their demand.

2.1.2 Advanced Metering Infrastructure (AMI)

Smart meter's data collection and communication with the homes are done through the AMI by considering several wired and wireless network technologies [SAU12]. With the recent developments in Smart Grid technologies, more utilities are moving to wireless infrastructure for AMI (as opposed to a powerline-based communication). Such wireless infrastructure is usually a stand-alone mesh network owned and governed by the utilities. This brings advantages in terms of costs and management and thus is becoming a viable option and adapting throughout in US and Europe [GH15].

Basically, AMI implemented using a wireless mesh infrastructure where smart meters form a connected network and send their data to a utility company or a third-party data collector. While there are a number of options to implement a wireless mesh network [UIA12], all of them accommodate a multi-hopping mesh network capability for communication. The nodes in AMI are given names based on their roles. All nodes are mesh points (MP) and are able to provide connectivity at the data link layer between other MPs. If an MP also provides connectivity to another network such as the Internet, it is called a mesh portal point (MPP)/gateway. An MP becomes a mesh access point (MAP) if it provides access to wireless clients which are referred to as mesh stations (mesh STA). In a typical AMI, all the smart meters will act as MPs/MAPs. There will also be some additional nodes acting as relay MPs when there is no smart meter available. Note that, we may have mesh STAs such as appliances from HANs or water/gas meters in a Smart City environment that can connect with smart meters and have them act as MAPs. The gateway node which will be connected to the utility will be MPP. The connection can be via GPRS, 3G. 4G/LTE.

Hybrid Wireless Routing Protocol

The routing protocol used in the wireless mesh network can be categorized as proactive, reactive and hybrid [GDGV13]. In proactive routing, a routing path is established between two nodes before any flow of data traffic. In fact, routing protocols maintain routing tables to keep routes to all destinations, regardless of whether or not these routes are needed. In reactive routing protocols, a path is established only when the source needs to communicate with a destination. This certainly reduces the routing overhead but introduces a route setup delay.

Hybrid routing protocols combine both reactive and proactive routing to increase the overall scalability of routing in networks. The basic idea behind hybrid routing protocols is to use proactive routing mechanisms in some areas of the network at certain times and reactive routing for the rest of the network.

IEEE 802.11s defines an HWMP as its basic routing protocol. HWMP combines reactive and proactive modes. The proactive part of HWMP is based on tree-based routing centered on a root node called the gateway. HWMP proactive mode ensures that every node has the best possible path to its root node, and the root has all the path information toward any destinations.

The reactive routing event occurs when a source node wants to send data to a destination node but has no path to the destination in its routing table. The reactive part of HWMP is based on Ad hoc On-Demand Distance Vector (AODV) routing. The source sends the data packets to the root node. The root forwards the data packet to the destination together with an indication that both the source and destination are in the same mesh network. This data packet activates the destination to initiate a path discovery for itself. Eventually, this procedure will establish the least cost path between the source and destination nodes, and the subsequent data will be forwarded on this path.

2.1.3 Field Area Network (FAN)

One of the major transformations in Smart Grid to enhance its reliability, resiliency, and efficiency is by integrating IoT devices for comprehensive sensing and processing abilities. The ongoing Smart grid and IoT integration process [LTCP16, SCRC19, KGM^{+}19] benefits from the information provided by IoT devices such as voltage, current, temperature, etc. for realtime monitoring of electricity generation, trans-

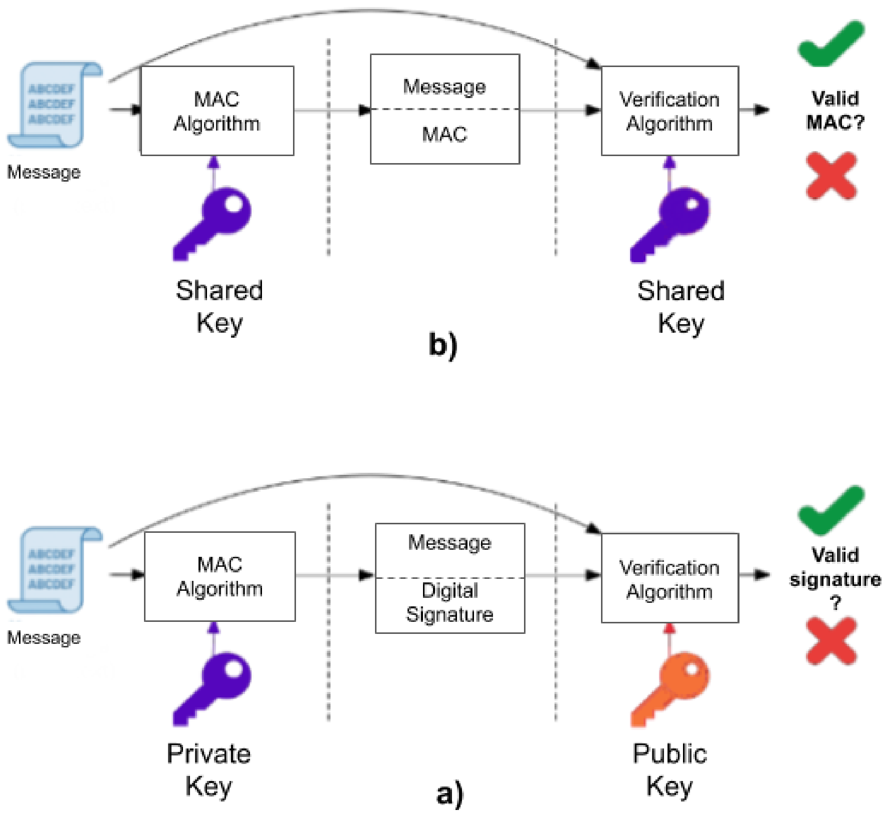


Figure 2.2: A sample usage of asymmetric and symmetric keys for the data integrity.

mission lines, and distribution. The information is conveyed to the utility center through the existing wide-area communication infrastructure of the grid that covers all the geographic areas where these Intelligent electronic devices (IED) and remote terminal units (RTU) are deployed with low density. This is typically through a Wide-Area Network (WAN) technology such as 2.5G, and other proprietary 900 MHz radio communications [GSK+11]. The typical characteristic of the wireless communication between gateway and field/IoT devices is severely limited in terms of bandwidth. Generally, it is in the order of kilo-bits, which is in line with the used technology in 2.5G or other proprietary protocols.

2.2 Key Management

Cryptography is mostly employed to ensure the security of the communications over an un-secure medium and to protect various critical applications. Cryptographic keys perform an essential role in the process of cryptography. The relation between cryptographic operations and the keys is similar to the relationship between a safe and its combination. If a thief knows the combination, even the most robust safe can not protect your valuable things. The combination/keys can be created under two different setup: asymmetric key and symmetric key setup.

Asymmetric key setup, generally known as public-key infrastructure (PKI), utilizes a key pair (e.g., a public key and private key) to perform cryptographic operations. Anyone can know the public key, but its pair, the private key, is just known by the party who creates this key pair and should be secret. With a PKI setting, the particular key of the key pair is used for different purposes to provide security, and their usage depends on the cryptographic service to be provided. However, they are mostly used to ensure the origin, identity, and integrity of the data through digital signatures. An example usage of Asymmetric keys for ensuring integrity while exchanging a message is shown in Figure 2.2(a).

Symmetric key setup is mostly utilized to scramble data, and undoing this is fundamentally difficult without knowledge of the key itself. The setup called as "symmetric" since the parties utilize the same secret key for a cryptographic operation such as scrambling/encrypting data and decrypting it. Symmetric keys are commonly used to provide data confidentiality by encryption& decryption services and like digital signatures, to ensure the origin and integrity of data through message authentication codes (MACs). Figure 2.2(b) shows the employment of the symmetric key for the message integrity.

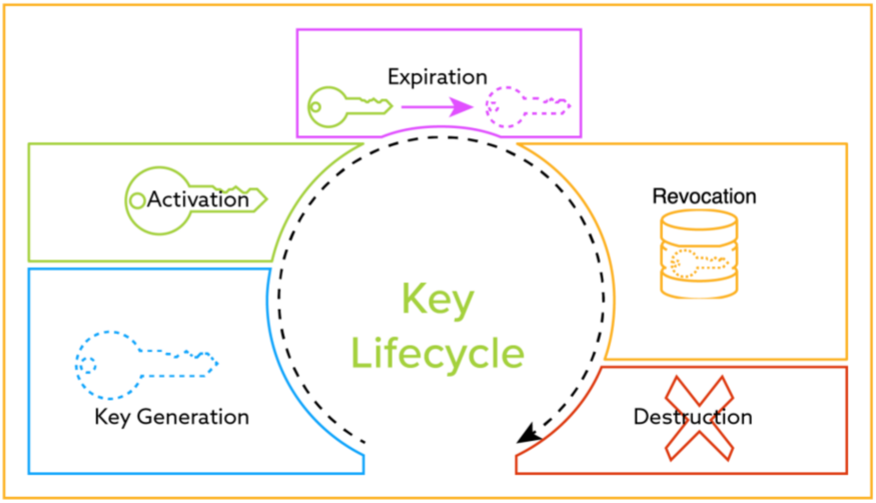


Figure 2.3: The life-cycle of cryptographic keys

Considering the security of cryptographic operations highly depends on the secrecy of the used key material, assuring the safety of the keys by a proper cryptographic key management is essential to the right use of cryptography for security. This subsection introduces key management by defining the life-cycle as shown in Figure 2.3, which contains secure generation, activation, revocation, and destruction.

2.2.1 Public-Key Infrastructure

A PKI is an infrastructure that generates and maintains key pairs in the form of digital certificates that bind the public and private keys along with the identity of the owner. Each certificate linked with digital signatures provides the availability of public keys to anyone. However, the corresponding private key is secret and accessible to the entity that owns the issued certificate. An entity can be in the form of an institution, individual, or device.

Digital Certificates

In a PKI setting, there is at least one CA with its clients/subscribers as illustrated for a very simple PKI setting in Figure 2.4. A digital certificate is issued for each subscriber that contains its public key, identity, expiration date, a serial number, which is signed by the CA.

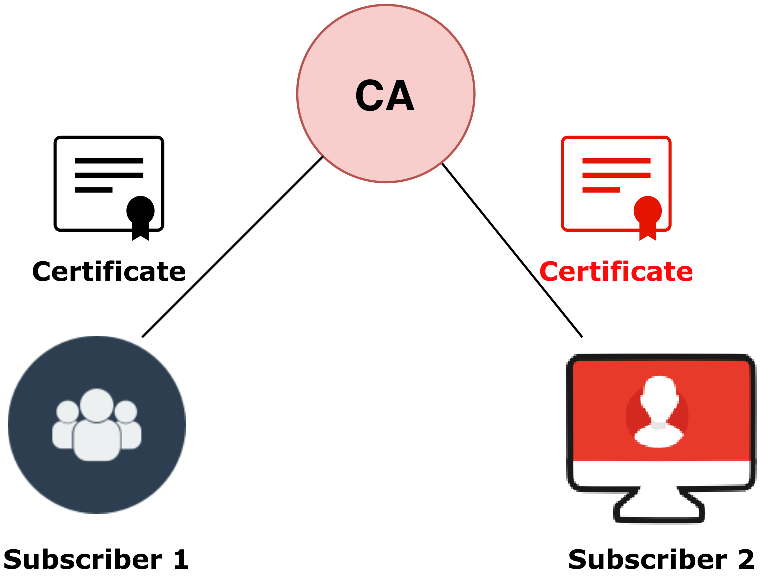


Figure 2.4: A simple PKI with a single CA

A certificate may be issued for a lifetime from 1 minute two twenty years. There are different types of certificates according to their usage. The most critical certificate is the ”root certificate” which is a self-signed certificate issued to CAs and used for signing other certificates. In addition to this, the most common type of certificate is the ”server certificate” which is issued to organizations to provide authentication about the identity of a server when a client wants to communicate with. Another type of certificate is the ”code-signing certificate” which is issued to software vendors to ensure the code has not tampered. The final type of certificate is the ”client certificate”, which is used to identify the client devices that are used on end systems. Once issued, these certificates are valid until their expiration date, and they can be used to perform various cryptographic operations.

Certificate Revocation List (CRL)

Certificate Revocation List (CRL) concept is related to the revocation of some created keys in PKI before its expiration date. The revocation of the key-pair actually means that the revocation of the issued certificate by CA. There are various reasons that cause a certificate to be revoked, such as compromising of the corresponding private key, changing of association between CA, finding a vulnerability during the key generation process, etc.

Revocation causes each CA to regularly issuing a signed list called a CRL, which is a time-stamped list consisting of serial numbers of revoked certificates and revocation dates. When a PKI enabled system uses a certificate (for example, for verifying the integrity of a message), that system should not only check the time validity of the certificate, but an additional check is required to determine a certificate's revocation status during the integrity check. To do so, CRL can be checked to determine the status of the certificate.

There are two main types of CRL: *full CRLs* and *delta CRLs*. A full CRL contains the status of all revoked certificates which are not expired yet. Delta CRLs contain only the status of newly revoked certificates that have been revoked after the issuance of the last full CRL and before the new release of it. Therefore, the most recent version of the CRL or delta CRLs is made available to all the potential nodes that will be using it. In the case of AMI, these CRLs need to be accessible to all the smart meters.

2.2.2 Symmetric Key Management

Symmetric key management requires setting up the same cryptographic key between at least two parties. Thus, symmetric keys, in other words, shared keys need a key establishment step between parties before it could be used for cryptographic operations. Scenarios for performing key establishment include: 1) generating a key from a single party and then providing it to the other one by manual ways such as using a flash drive or a printed confidential document; 2) using a key agreement protocol that incorporates a secure key-sharing scheme running by both of them.

Diffie-Hellman Key Agreement

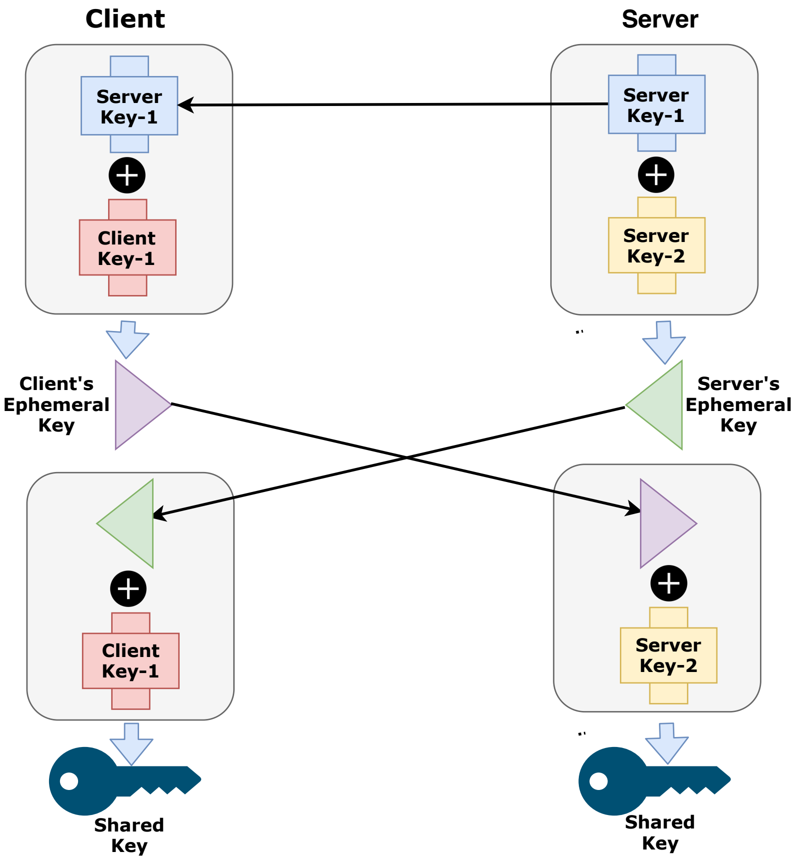


Figure 2.5: A illustration of Diffie-Hellman Key Agreement Scheme

In this subsection, we are focusing on the automatic method for symmetric key management, where both parties run a key agreement procedure in which the derived shared key is a result of a function contributed by them. That way, participants work together to determine the value of the shared key by completely depending on each other. We investigate a particular form of key agreement protocol called Diffie-Hellman (DH) key exchange. DH protocol involves two parties to derive a shared key by exchanging ephemeral (short-term) auxiliary materials.

Figure 2.5 illustrates the steps to perform a key-agreement in the DH scheme.

- **1.** The server generates two keys and shares one of them as a first step of the scheme.
- **2.** When the client receives the server's key, then it derive an ephemeral key by combining with a random key (Client Key-1), which is kept as a secret and then the ephemeral is sent to the server.
- **3.** Both parties use their secret keys, and the received ephemeral key from the other party to derive the same (identical) key as a shared secret.
- **4.** Finally, they use the derived shared secret to perform cryptographic operations to ensure the confidentiality and integrity of the messages.

CHAPTER 3

LITERATURE REVIEW

3.1 Public Key Management for Smart Grid

Due to increasing interest in Smart Grid, there has been a number of efforts to study the creation of public key infrastructure (PKI) for Smart Grid communications. For instance, Khurana et al. [KHLF10] identified public key management as a challenge due to the system scalability and complexity. Metke et al. [ME10] surveyed the existing key security technologies for extremely large, wide-area communication networks and claims that the most effective key management solution for securing the Smart Grid, in general, will be based on PKI. Mahmoud et al. [MMS13] focused on different aspects of PKI and in particular, certificate revocation problem in Smart Grid. However, these studies do not provide a specific solution for AMI networks which are mostly wireless-based.

Seo et al. [SDB13], focused on AMI and proposed a certificate-less PKI mechanism. In this scheme, smart meters hold certificate-less public/private key pairs based on their own IDs to decrease the overhead of certificate management. This approach and the likes do not require a CRL scheme due to the use of special key management mechanisms. However, such approaches are not desirable due to interoperability issues with other nodes and networks. In this chapter, beyond directly related studies on the PKI and Smart Grid relation, we also focus on studies about the distributed hash table (DHT) and cryptographic accumulators. We utilized these two different methods for solving the overhead problem of revocation management for Smart Grid. Thus, in this chapter, we examined the previous studies on DHT and accumulators and highlight major points by the aspect of revocation management.

3.1.1 Revocation Management for Smart Grid

The first study that focuses on the CRL management in Smart Grid was [MMAS15]. The authors investigated different CRL management aspects as a short-lived-certificate scheme, tamper-proof device scheme, online certificate status server scheme, CRL, and compressed CRL in various applications of Smart Grid. Later the focus shifted to AMI due to their large-scale deployment. For instance, the CRL management scheme based on Bloom Filters was proposed in [RMT$^+$15]. The size of CRLs can be reduced by Bloom Filter which is a special data structure to store the CRL information and access it quickly. However, this data structure suffers from false positives and may eventually require accessing the actual server to check the validity of a certificate. Our scheme on the other hand never requires accessing a remote server. In [ARMT14], the authors proposed a CRL management scheme based on grouping the smart meters that are within the same neighborhood and likely to communicate. In the proposed scheme, smart meters only keep the CRL of its group to minimize the communication and storage overhead of CRL. While this approach is good for a specific application, it may limit the number of applications to be run on AMI infrastructure (i.e., Demand Response applications in AMI requires communication of any smart meters). Our proposed approach does not have such a limitation and can be used for any application.

3.1.2 Distributed Hash Tables

DHTs play an important role in many applications and particularly in P2P networks [Coh06, FFM04, CSWH01, SFP10]. It provides robustness and efficiency by utilizing the properties of the hash table (e.g lookup an element with high efficiency) and enabling peers' resources to store and retrieve data. There are various types of DHT, which provide different choices for implementing DHT-based solutions.

For instance, Chord [SMK$^+$01] is one of the first distributed lookup protocol to efficiently locate which peer stores a particular data on a peer to peer network topology. It forms a virtual one-dimensional ring topology among peers to address challenges in storing data and lookup in distributed environments. For lookup operation, each node keeps a routing table which contains a total of $O(logN)$ entries. The average length of hop count is an important parameter that suggests the efficiency of the lookup operation. In Chord, the average hop count for a lookup is no more than $O(logN)$. Kademlia [MM02], Pastry [RD01] and Tapestry [GNOT92] are other forms of the DHT. They differ from the Chord because of their data model. They all form a tree structure to manage storing and lookup operations. All have the same efficiency as Chord in terms of lookup operations. CAN [RFH$^+$01] has a different data model. In CAN, each node maintains a list of 2D neighbors to form a d-dimensional hypercube. This model helps to keep the number of entries in routing table constant which is a desirable property for large scale P2P networks. However, the average lookup cost in terms of hop count is $O(N^{1/d})$.

There are also a number of studies proposing the use of DHT to manage CRL in large scale networks [YJ09] [HWQC08] [MM03] [AKDB12]. However, these studies mainly focus how to manage CRL information for P2P (e.g, BitTorrent or P2PStream) network needs which are not exactly match with AMI needs. Our method leverages some ideas from these studies and propose a new distributed infrastructure to distribute and store CRLs over AMI. We use AMI network as an infrastructure and a service provider as a P2P network. In our approach smart meters work together to provide the certificate verification service in a distributed environment. Comparing the Internet-scale size and structure of a typical P2P network with AMI networks, the used DHT schemes in the previous P2P studies are not suitable for AMI. Therefore, we propose another DHT scheme that is based on Fibonacci numbers

3.1.3 Cryptographic Accumulators

Benalog and DeMare [BDM93] first introduced cryptographic accumulators. After their first appearance, there have been studies [CL02, RY16, BCD^{+}17] offering to use them for membership testing. However, these studies solely focused on building the cryptographic fundamentals of accumulators, and thus, omit application-specific issues and security features when deploying them. Besides, these studies are offering to use accumulators for membership testing by accumulating a valid list. Considering AMI, accumulation of valid smart meter's certificates to provide a revocation mechanism would constitute a significant overhead due to the fact that revocation frequency is less than that of creating new certificates (i.e., no need to update the accumulator each time when a new smart meter is added to AMI). Furthermore, since the number of revoked certificates is also less than the number of valid certificates which affects the required computation time significantly [DKA^{+}14]. Our approach mitigates these drawbacks by addressing security and application-specific issues and offering to use CRLs instead of valid certificates.

3.2 Lightweight Symmetric Key Agrrement

In recent years, reducing the latency of the key exchange is a particular interest in the industry. As a result, a first prominent solution for minimizing the latency was introduced by Google via QUIC protocol which allows the parties to accomplish key-exchange in 0-RTT [Goo16]. Facebook also has come up with a 0-RTT protocol [Fac16] which is very similar to QUIC, except that it uses another nonce and additional encryption for the ServerHello message. Inspired from QUIC, IETF have recently updated TLS 1.2 to TLS 1.3 and DTLS 1.2 to DTLS 1.3 to 0-RTT. The new version of TLS/DTLS also enabled authentication without relying on PKI (which was the case in QUIC). However, TLS/DTLS 1.3 0-RTT was based on key resumption idea meaning that it relied on the session key created for the previous session to encrypt the first flight of data before creating a new key.

Although 0-RTT schemes are started to be used in the wild, there are some special security precautions for current 0-RTT schemes about replay attacks where data can be captured and replayed by adversaries. Neither QUIC nor TLS/DTLS 1.3 0-RTT scheme does have any replay attack protection. Due to such issue, QUIC is only used for HTTP GET requests in Google to eliminate any security impact with PUT or DELETE operations. Similarly, TLS/DTLS 1.3 standard has warnings about 0-RTT usage [RD18, RTM20]. Basically, the standard mentions that for a specific client 0-RTT should be applied only once to prevent a replay attack. The consequent key generation should be based on 1-RTT again. Furthermore, TLS 1.3 standard also urges avoiding 0-RTT use for non-idempotent operations. Specifically, it suggests sending only initial data which does not update any state in the server side. Unfortunately, both of these policies/suggestions to prevent replay attack is not applicable in power grid domain. First, the IEDs/RTUs regularly transfer data to a control center (CC). Thus, if a client device is limited to only one 0-RTT then in the next data collection round, 0-RTT could not be used and thus the key exchange will incur additional delays. Second, the data sent by client devices to CC is typically used for state estimation which is very crucial in taking control actions. Thus, since this data updates the state in the server, it is a non-idempotent operation that must be secure against any replay or injection attacks. Therefore, TLS 1.3 0-RTT or QUIC cannot be used in power grid applications. Our approach in this work fills this gap by proposing a replay-attack resistant 0-RTT key exchange.

As 0-RTT key exchange mechanisms are evolved in industry as a result of the low-latency demands, this led to some further research in academia on these protocols. For instance, the work in [FG14, HJLS17] provides a general definition of key exchange protocols to analyze the properties of QUIC. Lychev et al. [LJBNR15] analyzed the efficiency of these protocols in addition to their security. There are also two recent studies in academia which offer a 0-RTT key exchange scheme [GHJL17, DJSS18]. However, these studies do not follow the well-studied crypto frames such as RSA or Elliptic Curve to produce the key share. The first study in [GHJL17] is based on puncturable encryption (PE) while the other study in [DJSS18] is based on bloom Filter Encryption. Both of them have very large key sizes which are up to 400 MB. Therefore, they are not efficient enough to be deployed in practice yet alone for power grid domain.

CHAPTER 4

A LIGHTWEIGHT DHT-BASED REVOCATION MANAGEMENT FOR AMI AND HAN INTEGRATION

In this chapter, we propose a solution to the overhead of revocation management when HAN is integrated to AMI for a Smart City application. In a typical Smart City environment, AMI can employ PKI for security while serving as an infrastructure for other applications such as water and gas data collection. Therefore, there is a need to systematically manage the revoked keys/certificates without causing too much overhead in terms of distribution and storage. In this chapter, we aim to develop a customized solution for managing the overhead of revoked certificates by utilizing a distributed data structure called *Distributed Hash Table (DHT)* [SMK$^+$01]. DHTs which have been widely employed in P2P networks serve as a quick lookup service where the data is distributed to multiple nodes. Our study is the first to utilize DHTs to provide scalable and efficient lookup service for a revoked certificate search.

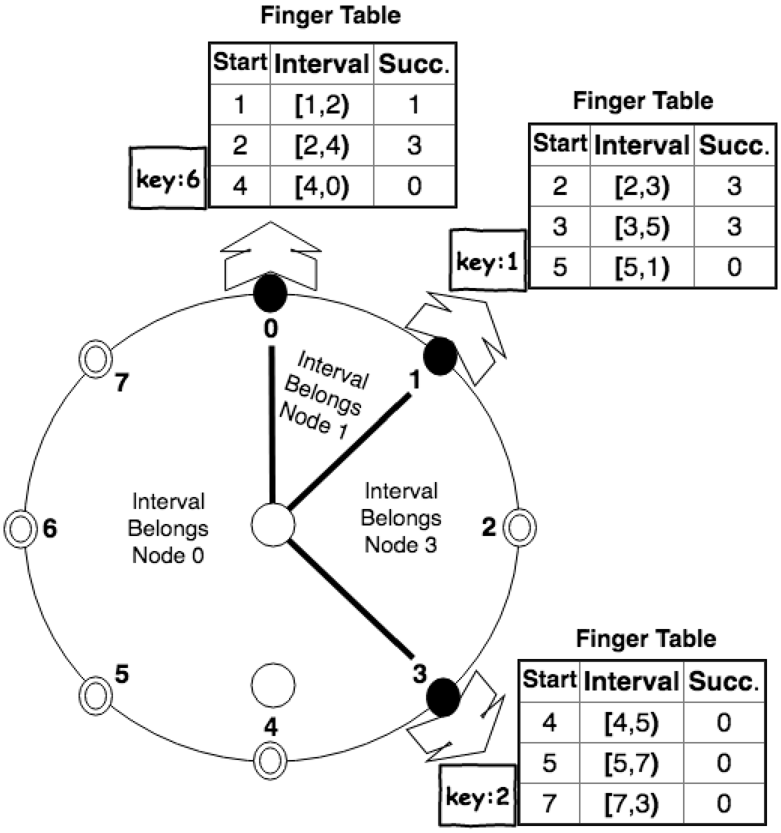
Specifically, we aim to exploit the AMI network as a baseband of the distributed infrastructure to keep the revocation information as Smart Grid AMI is envisioned to be connected two-way communication technology between customers and utility companies [Far10]. Contrary to the existing CRL approaches [RMT$^+$15] [ARMT14] [MMS13], we proposed keeping only a portion of the CRL in each smart meter by a customized DHT. Basically, the smart meters are peers within the network, which are responsible for providing and storing portions of the CRL. When a smart meter needs to access the revocation information, it gathers this information by using the DHT stored over the AMI network. The provided distributed structure significantly reduces associated overhead of revocation management since the CRL portions, which are smaller significantly smaller in size comparing to full CRL, can be updated independently. In addition, the proposed customized DHT mechanism contains several additional functionalities to ensure the security of revocation management against several threats.

The performance of the proposed approach is assessed via simulations in ns-3 network simulator and on a testbed consisting of Raspbery PI devices running IEEE 802.11s. We compared our approach with the other methods that use conventional CRL schemes [RMAT17]. The results from simulation and testbed show that the proposed DHT-based management has significantly less overhead than the other methods. It not only reduces storage requirements on the smart meters but also decreases distribution overhead with reasonable access times regardless of the AMI network size.

We organized this Chapter by first providing some background on DHT mechanism. Then, in the following Section 4.2, we give details about the Threat Model and Security Goals. The proposed approach is described in Section 4.3. Section 4.4 does the security analysis of the approach. In Section 4.6, we present and discuss the experimental results and Section 4.6 concludes the chapter.

4.1 Background On Distributed Hash Table

DHT consists of a collection of nodes and it supports a distributed data structure on an overlay network. This data structure is built by assigning each node with a portion of the key space. Node's key space determines which items should be stored at that node. Retrieving an item from DHT is accomplished by reaching the responsible node via a routing operation. To manage routing and management of organization of key space, [SMK+01] forms a DHT called *Chord* and assumes that each item to be stored in DHT is unique and builds an imaginary static ring topology to support routing to reach that item. Chord uses consistent hashing [KLL+97] mechanism to map both node IDs and keys to the same circular space. This enables a fast search method by requiring each node to keep a form of routing table which is called as finger table. The i^{th} row of finger table that belongs to node n is the successor of $n + 2^{i-1}\, mod\, 2^m$ on the ring. The first row of finger table is actually the node's immediate successor. Every time a node needs to search a key, it will pass the query to the closest successor of the key in its finger table until a node finds out the key.



Finger Table

Start	Interval	Succ.
1	[1,2)	1
2	[2,4)	3
4	[4,0)	0

Finger Table

Start	Interval	Succ.
2	[2,3)	3
3	[3,5)	3
5	[5,1)	0

Finger Table

Start	Interval	Succ.
4	[4,5)	0
5	[5,7)	0
7	[7,3)	0

Figure 4.1: Example of Chord topology

Figure 4.1 shows a Chord topology which builds an imaginary ring that has at most 2^m space where $m = 3$. The identities of each node determine the location of it on the ring topology. For this example, the ring has three nodes as 0, 1, and 3. There are 3 keys to be stored in DHT as 1, 2 and 6. These keys are stored according to node locations in the ring. Key 1 would be stored in node 1 since the successor of identifier 1 is node 1. Similarly, key 2 would be located in node 3 since the ring does not have a node at 2, and finally key 6 would be at node 0. Figure 4.1 also shows the finger table of nodes which are calculated according to $n + 2^{i-1}\, mod\, 2^m$. For node 1, the finger table points to successors information $(1\, +\, 2^0)\, mod\, 2^3\, =\, 2$, $(1\, +\, 2^1)\, mod\, 2^3\, =\, 3$, and $(1\, +\, 2^2)\, mod\, 2^3\, =\, 5$. When node 1 wants to find key 5 within Chord, it looks at its finger table and finds the successor or closest successor. In this example, the successor of key 5 is node 0 and it queries node 0 whether it has key 5 in its local list or not.

4.2 Threat Model and Security Goals

The security of the proposed approach depends on the secure implementation of DHT system. Therefore, we considered the following threats to the security of the proposed approach and identified the relevant security goals. Note that in our attack model, we assume that the adversary has limited knowledge about the AMI network topology and the compromised smart meters are a tiny fraction of the whole network. This threat model might be considered naive for AMI network because of increasing threat of state-sponsored cyber-attacks. However, it is obvious that a single counter-measure against state-sponsored cyber-attacks would not be adequate when taking into account their well-funded efforts. Thus, none of the previously mentioned revocation mechanisms would be enough by themselves to protect AMI. To ensure the security of AMI against state-sponsored attacks, the utility company must deploy intrusion prevention systems and proper attack prevention tools as well. For instance, to impede infiltration to the vast majority of smart meters, a PKI inspection platform along with Hardware Security Modules (HSMs) can be deployed that provides device-level controls to protect PKI keys and certificates. However, these efforts are beyond the scope of this dissertation.

Note that, the cases where *root certificates* are compromised is out of scope for our threat model. If a root certificate is compromised, then the PKI for the AMI becomes invalid and it should be set up from scratch by issuing all the certificates again.

Threat 1: An adversary can introduce a set of fake identifiers (i.e., sybils) to the distributed revocation management system. These sybils can be controlled by an adversary and used to attack security properties of the distributed revocation management system by altering CRL lookup forward messages or returning bogus CRL lookup response messages. Furthermore, the adversary can perform denial of service (DoS) attack by generating many CRL lookup messages to degrade the performance.

Security Goal 1: Control the participation of smart meters to the distributed revocation management system by prohibiting them from choosing their identifiers. This will prevent creating sybils.

Threat 2: An adversary can alter finger table entries of some smart meters to direct the CRL lookup messages to itself. As a result, the adversary achieves finger table poisoning attack. This allows the adversary to monitor CRL lookup messages and launching a DoS attack and hindering security of distributed revocation management.

Security Goal 2: Control the finger table updates centrally and protect integrity of update messages by cryptographic methods.

Threat 3: An honest smart meter that participates in the revocation management initially can be compromised by an attacker later. This compromised smart meter can attack the system in the following ways:

- *Not Forwarding CRL Lookup Messages:* A compromised smart meter may hinder a lookup request by refusing to forward it.

- *False CRL Lookup Message Forwarding:* A particular compromised smart meter could forward CRL lookup messages to an incorrect smart meter to degrade the performance. Since the compromised smart meter is participating in the CRL lookup system, it will appear to be alive and honest. As a result, it can continue to perform the attack gradually.

- *Falsify Retrieval Information:* The compromised smart meter could deny the existence of revocation information or give a false revocation information for a non-revoked certificate. This attack can be performed in the following ways:

    1. *False revoked response for a non-revoked certificate:* When a smart meter needs to verify validity of a certificate, it queries the certificate from the network. This query reaches the responsible smart meter. This smart meter is required to return whether the corresponding certificate is listed in its CRL portion. However, if it is compromised, it can return a revoked certificate response message for a valid certificate.

    2. *False non-revoked response for a revoked certificate:* This attack assumes a meter, say A, in the AMI which has a previously revoked certificate (i.e., it should have a certificate that is signed with trusted CA initially but revoked later). If there is another compromised smart meter and this meter is responsible to check the revoked certificate from A, it can provide a non-revoked response which is not true.

Security Goal 3: Provide data verification techniques to prevent both false CRL lookup forwarding and retrieval information.

4.3 Proposed DHT-based Approach

4.3.1 Overview

The problem of CRL management poses challenges in terms of storage overhead to smart meters with the increased network size. In addition, the large size of the CRL hinders their distribution within the AMI. Our proposed approach to this problem utilizes the concept of DHTs offered in [SMK$^+$01]. The use of DHTs not only decreases the cost of storing and distributing the CRLs but also maintains acceptable delays when checking the status of a certificate.

In a nutshell, our approach divides the entire CRL into several portions in order to avoid unmanageable CRL size for the smart meters. *Chord* builds a data structure which enables such division of CRL into several small portions. This is achieved by utilizing unique items to be stored in the *Chord* structure. Note that this is a requirement for spreading keys uniformly across the imaginary ring while mapping both node IDs (in our case IP addresses + public keys of meters) and keys (in our case certificate IDs) to the same circular space.

The uniqueness constraint of *Chord* perfectly fits to our problem since our objective is to store the revoked certificate IDs which are also unique. The CRL portions are kept and shared in a distributed way where all smart meters are considered as potential servers for CRL portions, and the gateway acts like the distributor of CRL portions. Each smart meter will just keep one portion of CRL and will use a finger table to find other CRL portions when it is required. This finger table will help smart meters work in collaboration to find out whether a certificate is revoked or not.

In our proposed scheme, the CAs are responsible for issuing the CRLs and the utility company is responsible for dividing it into smaller CRL portions. Note that when preparing the smaller CRL portions for the AMI, the threats of generation of fake CRL portions, or modification of a valid portion should be mitigated. To achieve this degree of security, the utility needs to re-sign all the produced CRL portions using their cryptographic private key in the same way as with the standard CRLs. In the balance of this section, we elaborate on CRL partitioning and CRL access when such CRL portions are distributed to smart meters. An overview of the described system is shown in Figure 4.2.

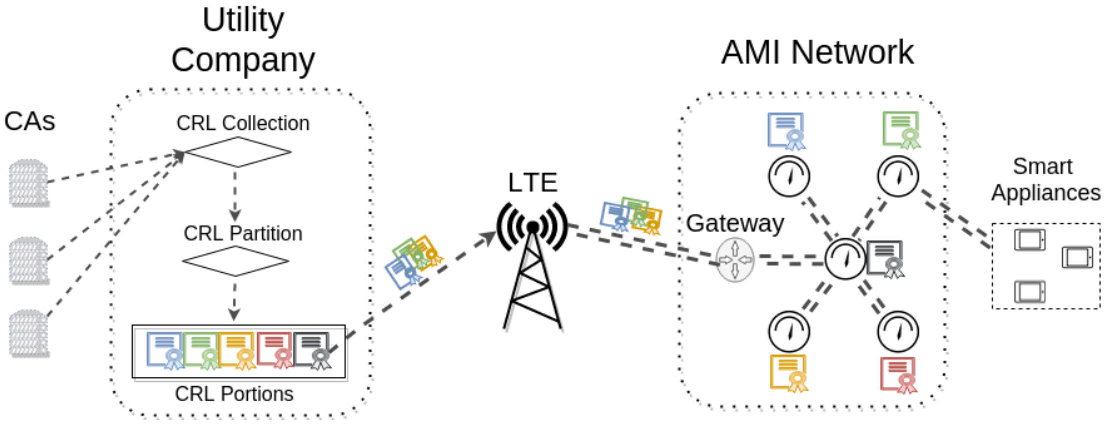


Figure 4.2: An Overview of the System

4.3.2 CRL Partitioning

CRLs are divided into N portions where N is the number of smart meters within the AMI. We utilize the IP addresses of smart meters for this purpose. Specifically, N IP addresses along with their public keys are hashed and each hash value would be the key a CRL portion. Then the question is how to decide which revoked certificate will fall into one of these portions. This is done by computing the hash of a revoked certificate and searching an appropriate portion for it based on the output of the hash (i.e., the output is compared with the hash of the IP address corresponding to a particular portion). For the hash function, CAs use a consistent hash function [KLL+97].

Definition 1: Consistent Hash Function: Consistent hashing is a special kind of hashing which ensures uniform distribution of key and value pairs on an imaginary ring. Basically, it maps each hash value of a key to a point on the circle (i.e., associates each key to an angle).

Upon completion of mapping (i.e., calculation hash of all key values), each hash value of keys represents an hash interval where the interval boundaries are determined by calculating the hash of each keys (i.e., other points on the circle). This imagery pie-shaped circle will be used to map each object to one of the portions of the circle.

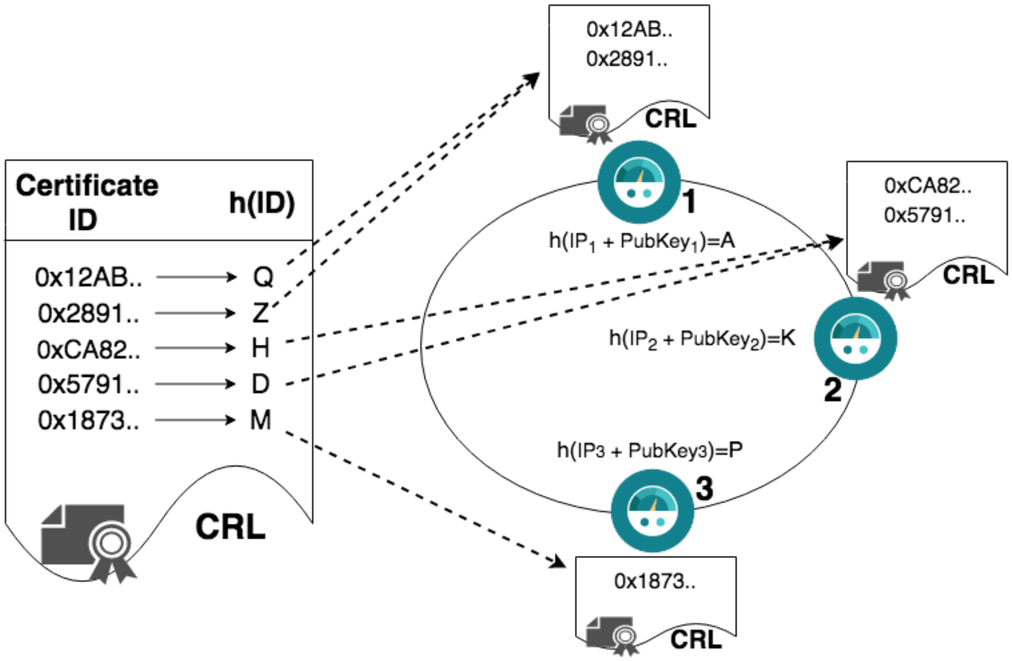


Figure 4.3: CRL Partitioning

The details for this procedure would be as follows: The CA first gets the hash of each (IP + public key) $h(ID_k)$ and uses it as an identifier of CRL portions in an imaginary ring where $k \in 1..N$ and sorts the resulting hashed values in a list

L_{total}. It then scans the provided serial numbers in CRL and gets the hash of those to create another list $L_{revoked}$(i.e., list of $h(c_i)$s, where $i \in 1..C$ and C is the number of revoked certificates). After forming these lists, our algorithm scans the $L_{revoked}$ one by one to determine the CRL portions of certificate IDs. If the result of $h(c_i)$ is between $h(ID_{k-1})$ and $h(ID_k)$, the k^{th} CRL portion will keep the corresponding certificate ID. After the scanning of $L_{revoked}$ is finished, distribution of certificate IDs into different CRL portions is completed. Eventually, each smart meter will nearly carry $1/N$ portion of the whole CRL. Algorithm 1 provides the pseudo-code for this proposed approach.

Algorithm 1: CRL Partitioning Algorithm

```
input: $L$ as sorted list of IP hashes (i.e $L_{total}$);
       $R$ as list of revoked pseudonym certificate ids (i.e $L_{revoked}$);
output: $O$ holding the set of CRL partitions;

for $i \leftarrow 1$ to $length(L)$ do
    $O_i \leftarrow \emptyset$
end
for $i \leftarrow 1$ to $length(R)$ do
    $c \leftarrow$R[i];
    $l \leftarrow$L[0];
    $k \leftarrow$0;
    while $c < l$ do
        $k \leftarrow$k+1;
        $l \leftarrow$L[k];
    end
    if $k < length(L)$ then
        $O_k \leftarrow O_k \cup c$;
    else
        $O_0 \leftarrow O_0 \cup c$;
    end
end
```

Figure 4.3 shows an example on how a CRL is partitioned into 3 CRL portions according to this scheme. Smart meters are located on an imaginary ring according to $h(ID_k)$ which maps the IP addresses + public keys to a letter. In addition, each certificate ID is mapped to a letter using $h(c_i)$. As seen in the figure, the certificate ID 23 would be kept in the CRL portion that belongs to smart meter 2 since $h(23) = H$ is in between A and K. Similarly, the certificate ID 11 would be located in CRL portion that belongs to smart meter 1 since corresponding hash value of $h(11) = Q$ is in between P and A in the ring topology and the remaining are located in the same way. In addition, each entry of CRL portions is signed by the utility using ECC-160 to mitigate the *falsify information retrieval attack* which is described in Section 4.2.

4.3.3 CRL Lookup

Chord will decrease the CRL size at the expense of additional lookup cost arising from sending lookup messages through the AMI network. Checking a certificate ID from Chord requires reaching the responsible smart meter via a routing operation. This lookup is accomplished by a *finger table* scheme. However, the finger table scheme of Chord is optimized for P2P networks which may have millions of nodes. Considering size of AMI (hundreds to thousands), which is smaller than a typical P2P network, it does not provide an optimal trade-off between resources exploited and the performance.

Thus, we use another routing scheme which is proposed in [Chi04]. This study defines an improved finger table based on Fibonacci distances where the objective is to reduce the number of hops, possibly at the expense of an increased size of the finger table. In this routing scheme, each node in the network keeps some of its successors' identities which maintain a finger table where the i^{th} entry of finger table of n^{th} node is the successor of $n + G_j(i)$ on the ring. For a given key, routing will be done comparing the entries on the finger table and the node forwards the message to the successor that is closest to the key but not greater than the key. G_j recursively defines a family of "Fibonacci" sequence members and constructs a finger table which includes the address of the peer that is located at distance $G_j(i)$ in terms of the number of hops in the ring structure from the local node as follows:

$$\forall j \in \mathbb{N}, \forall i \in \mathbb{N} : i \geq k, G_j(i+1) = G_j(i) + G_j(i-j)$$

where $\forall i \leq j,\ G_j(i) = 1$ as the initial condition. This creates "usual Fibonacci" sequence in case $j = 1$ and special Fibonacci sequences where $j > 1$. Following this idea, we can thus define a finger table, which includes the address of the smart meter that is located at distance $G_j(i)$ in terms of the hop count in the ring structure from the local node, as the i^th element of its table.

The adoption of parameter values as $j > 1$ and corresponding number of entries in the finger table for increasing AMI size is outlined in Figures 4.4 and 4.5. As can be seen in Figure 4.4, using a j value greater than 1 increases the number of entries in the finger table which will help to decrease the cost of CRL lookup. Figure 4.5 shows the method's efficiency and scalability over growing network size. As can be seen, even for AMI that has 1K smart meters, the lookup cost is around 2 hop counts with 40 entries in smart meters' finger table when $k = 15$. This lookup cost can be reduced further by storing more entries in the finger table.

Finger Table Formation

In this section, we describe how finger tables are formed according to the described Fibonacci scheme. An example of the formed finger tables are shown in Figure 4.6. In this example, smart meters are located on an imaginary ring according to $h(ID_k)$ which maps the IP and public key to a letter as described in Section 4.3.2. Each

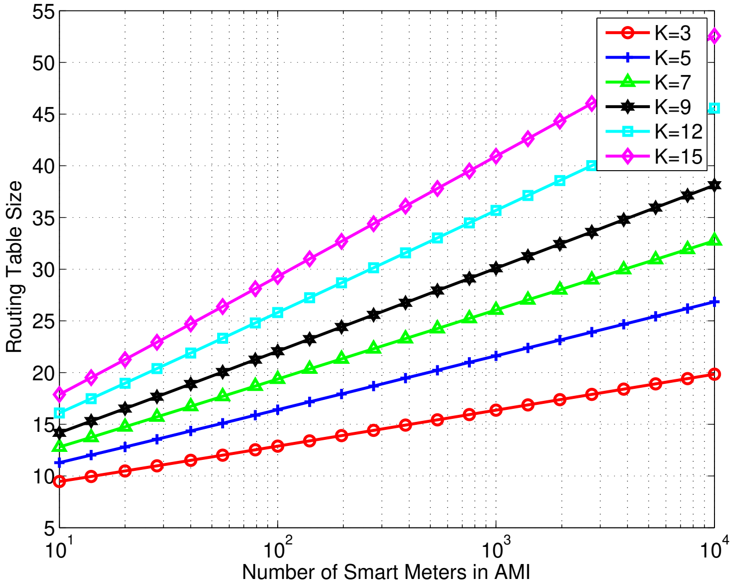


Figure 4.4: Average Number of Finger Table size with increased AMI size.

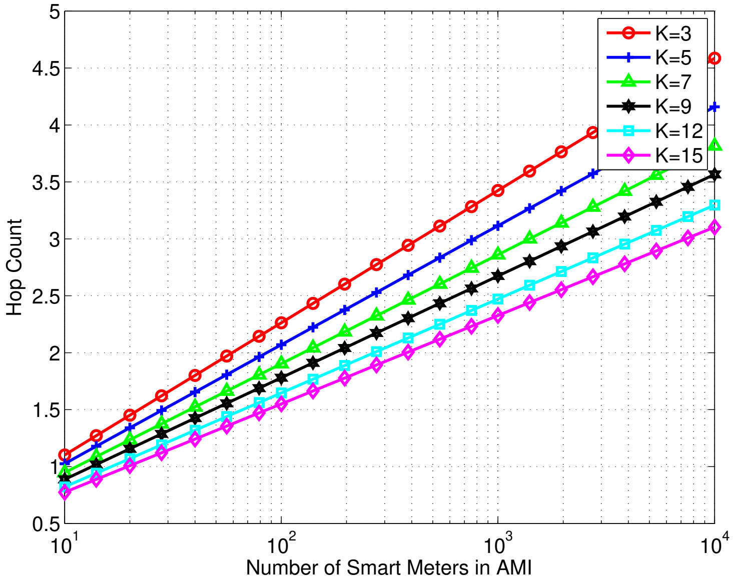
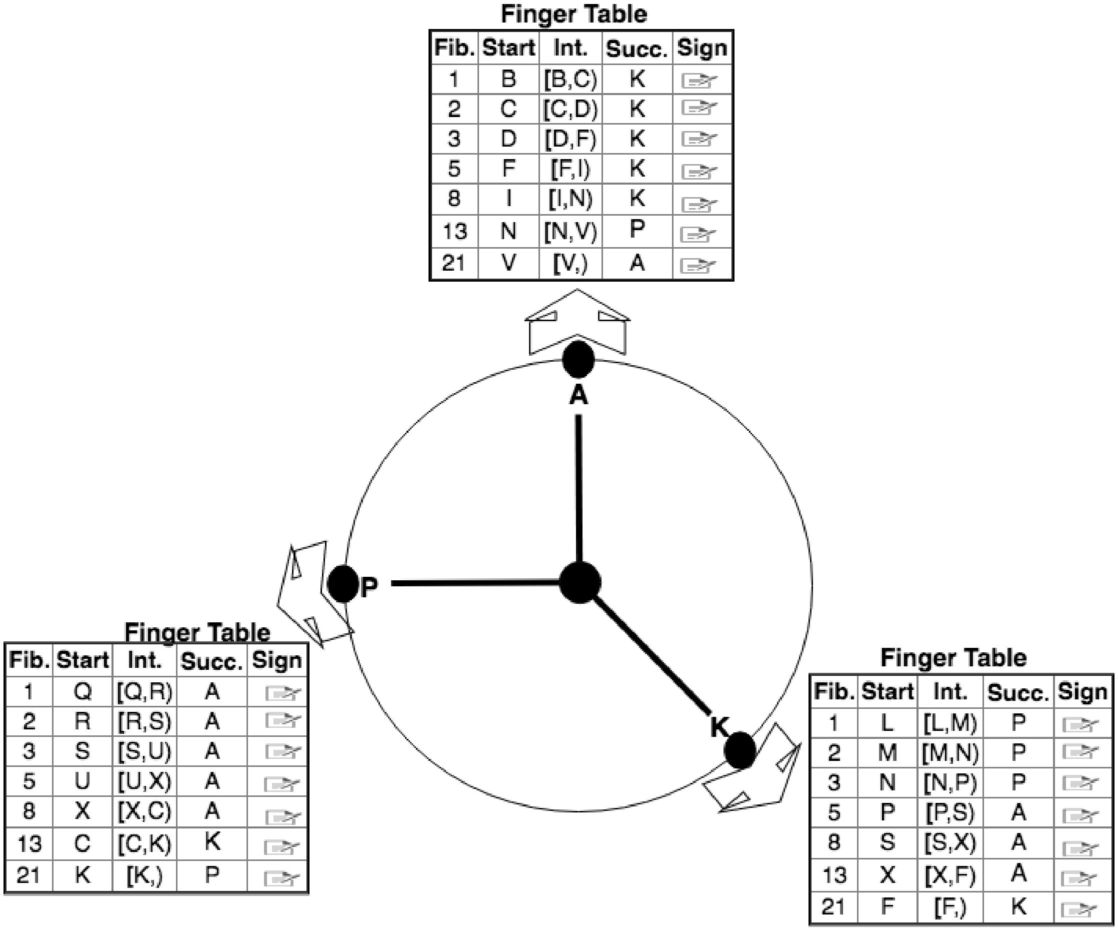


Figure 4.5: Average Number of Hop Counts against AMI size.

row of finger table shows Fibonacci neighbors of that smart meter when Fibonacci parameter $j = 1$. The i^{th} row of finger table that belongs to node n is the successor of $n + G_j(i+1)\, mod\, 26$ on the ring. For node A, the finger table points to successors information $(A + G_j(2))\, mod\, 26 = B$, $(A + G_j(3))\, mod\, 26 = C$ ...

$(A + G_j(8))\, mod\, 26 = V$. These finger tables are formed by the utility company and distributed to the related smart meters. When node A wants to find key O within DHT, it looks at its finger table and finds the successor or closest successor. In this example, the successor of key O is node P since O is between $N - V$. The revocation information of key O should be stored at node P, thus A queries node P to find out that the key O is revoked. Furthermore, each row of finger table contains a signature of routing rule value (i.e., *interval+successor*) which is signed by the utility. This signature will be used for preventing the *False CRL Lookup Message Forwarding Threat* which is described in Section 4.2. Note that, the whole finger table is signed by the utility for protecting any form of integrity attack.



Finger Table

Fib.	Start	Int.	Succ.	Sign
1	B	[B,C)	K	✍
2	C	[C,D)	K	✍
3	D	[D,F)	K	✍
5	F	[F,I)	K	✍
8	I	[I,N)	K	✍
13	N	[N,V)	P	✍
21	V	[V,)	A	✍

Finger Table

Fib.	Start	Int.	Succ.	Sign
1	Q	[Q,R)	A	✍
2	R	[R,S)	A	✍
3	S	[S,U)	A	✍
5	U	[U,X)	A	✍
8	X	[X,C)	A	✍
13	C	[C,K)	K	✍
21	K	[K,)	P	✍

Finger Table

Fib.	Start	Int.	Succ.	Sign
1	L	[L,M)	P	✍
2	M	[M,N)	P	✍
3	N	[N,P)	P	✍
5	P	[P,S)	A	✍
8	S	[S,X)	A	✍
13	X	[X,F)	A	✍
21	F	[F,)	K	✍

Figure 4.6: Fibonacci Finger Table

Lookup Operation

Figure 4.7 describes the general view of how the distributed certification verification system works. Suppose that a smart meter A wants to verify a certificate ID. As the first step, the smart meter A takes the certificate ID and its public key and calculates a hash of it as the key. In this way, A can easily detect the responsible smart meter or nearest smart meter within the AMI where corresponding key should be stored or asked by just looking at its finger table. In this specific example, we are assuming that the key should be stored in node K. However, if A only knows the nearest node J to reach K, it sends the lookup query along with the signed routing rule to node J. Then, this request reaches node J which in turn is forwarded to K which should have the corresponding certificate ID in its local CRL portion and check whether it is revoked. Before forwarding the query, J checks if the query is legitimate by checking the signature of the routing rule. Finally, K checks whether the serial number of the certificate is in its CRL portion and sends the result to A. If the serial number of certificate is stored within that portion, then it is revoked, otherwise the certificate is valid.

This look up process depends on the underlying routing protocol which is provided by HWMP that comes with IEEE 802.11s [802]. We know that HWMP already provides the routes from each smart meter to the gateway and maintains them for the lifetime. However, our DHT algorithm needs to access not only the gateway but also other neighboring smart meters listed in its DHT routing table to accomplish certificate ID lookup. Reactive routing option of HWMP helps us to determine the routes from the source smart meter to others. This will introduce additional delay for the first queries since it requires sending queries through the root node (i.e., gateway) until optimal path discovery process finishes. However, subsequent queries will be sent over the optimal path which will keep the delay

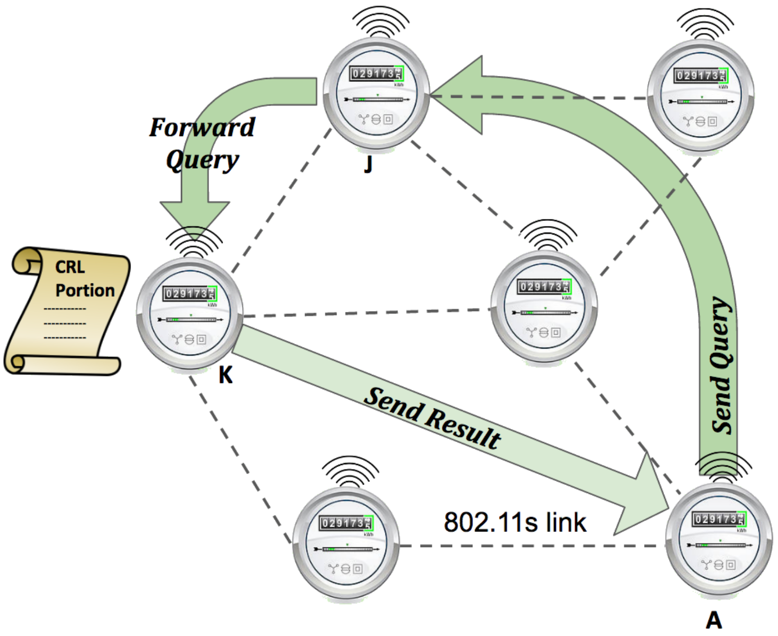


Figure 4.7: Finding a Revoked Certificate in AMI Network

constant. Since there is no mobility for AMI and the topology is mostly stable, we set the reactive route timeout interval parameter of 802.11s to a larger value to maintain the optimal paths for longer periods.

4.3.4 Updating CRLs

One of the important decisions faced in CRL management is determining the renewal schedule for CRLs. If a CA publishes a full/complete CRL frequently, this may cause significant overhead due to frequent distribution of the updated CRL to all parties within the AMI. If the updated CRL is distributed less often, this may reduce the amount of overhead, but increases the security risk.

Delta CRL concept defined in RFC 5280 [Coo08] addresses these issues by just including the newly revoked certificates information. When delta CRLs are implemented, the CA can distribute full CRLs at longer intervals and delta CRLs at shorter intervals. An important point about delta CRL concept is that it does not eliminate the requirement of full CRL distribution. The full CRL must still be redistributed when the previous full CRL expires since CRL has also a lifetime period as certificates and the lifetime period of delta CRLs are dependent on the lifetime of the previous full CRL.

As there is still an overhead associated with delta CRL distribution, our approach takes the advantage of collaborative storage and applies the same idea to delta CRL management. Specifically, it partitions delta CRL as described before and distributes it to smart meters accordingly. Therefore, our approach will send delta CRL portions to just related smart meters individually which will reduce the traffic overhead. Note that due to the nature of our approach, there will be many smart meters that do not need any information to be updated.

4.3.5 Join/Leave Operations for Smart Meters

We assumed that the utility company knows the topology of AMI and all active smart meters in the proposed revocation management system. When a new smart meter joins to the system, the utility performs following steps:

1. *Calculate the finger table of joining smart meter:* The utility calculates the finger table of the new meter according to the procedure defined in Section 4.3.3. As an example, suppose smart meter G wants to join the previous DHT shown in Figure 4.6. The ID of G is between nodes A and K. The utility acquires K as its successor and forms a finger table according to that as in Figure 4.8. This formed finger table is sent to the new smart meter.

2. *Update finger tables of existing smart meters:* After the smart meter G joins the system, the information about G will need to be entered into the finger tables of some of the existing smart meters. To do so, the utility revisits

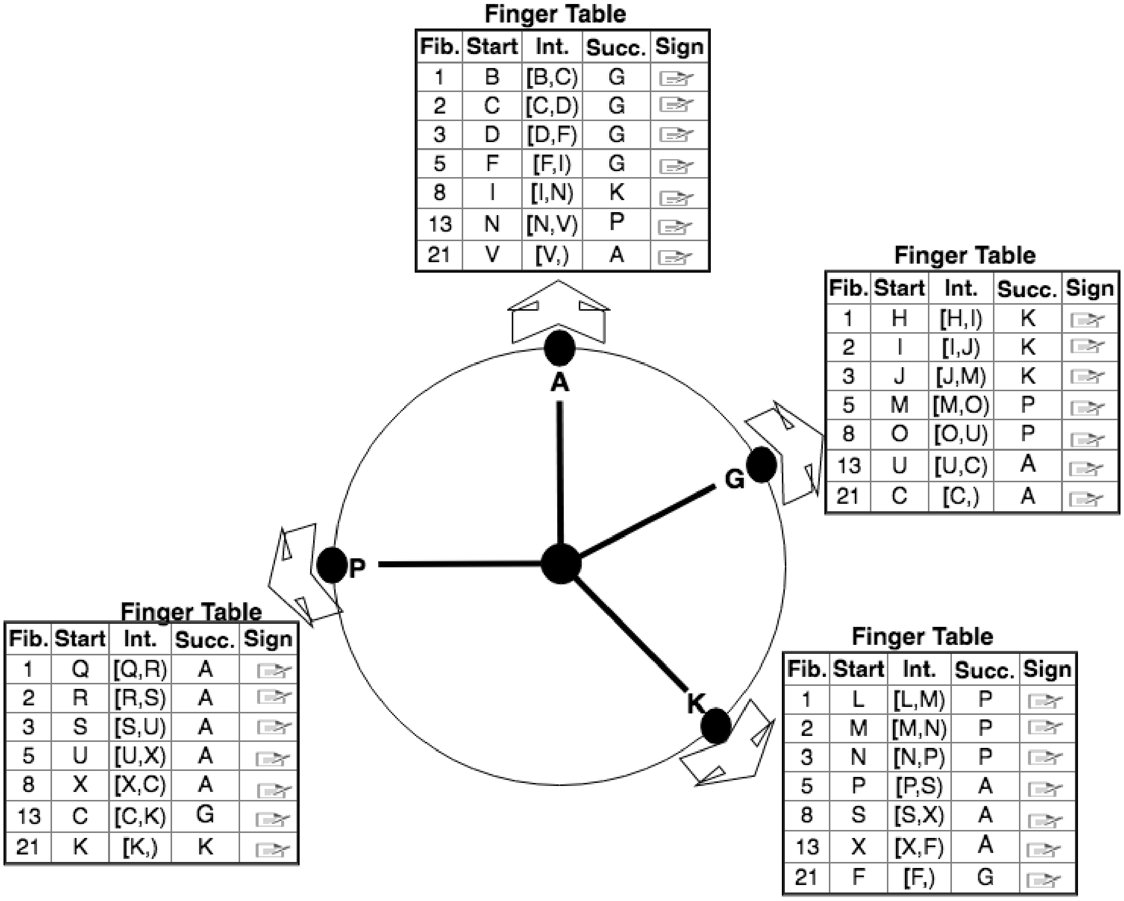
the calculation of finger table for each predecessor of G. Finger tables of the predecessors are recalculated yet again. The smart meter G will have at most $log(N)$ predecessor when the Fibonacci parameter k is 1. Thus, after this operation at most $log(N)$ number of smart meters will update their finger table. Figure 4.8 shows the updated finger table of G's predecessor after this operation.

3. *Transfer the CRL information from the successor:* The last operation that has to be performed when a smart meter joins the system is updating the CRL portion of its successor and transfer some of the revocation information from it to the new one. This typically involves moving the revocation information associated with the new key to the new smart meter. With the help of consistent hashing, smart meters join the system with minimal disruption. When a smart meters G joins the network, some certificate revocation information previously assigned to G's successor is now assigned to G. Thus, the utility only needs to update the successor and newly coming smart meters' CRL portions. The other existing smart meters are not affected and thus do not update their CRL portions.

The leave operation for a smart meter can also be performed in the similar manner by removing the information from finger tables associated with the leaving meter.

4.4 Security Analysis

In this section, we provide a security analysis of our proposed approach with respect to our threat model described in *Threat Model and Security Goals* Section 4.2.



Finger Table

Fib.	Start	Int.	Succ.	Sign
1	B	[B,C)	G	✍
2	C	[C,D)	G	✍
3	D	[D,F)	G	✍
5	F	[F,I)	G	✍
8	I	[I,N)	K	✍
13	N	[N,V)	P	✍
21	V	[V,)	A	✍

Finger Table

Fib.	Start	Int.	Succ.	Sign
1	H	[H,I)	K	✍
2	I	[I,J)	K	✍
3	J	[J,M)	K	✍
5	M	[M,O)	P	✍
8	O	[O,U)	P	✍
13	U	[U,C)	A	✍
21	C	[C,)	A	✍

Finger Table

Fib.	Start	Int.	Succ.	Sign
1	Q	[Q,R)	A	✍
2	R	[R,S)	A	✍
3	S	[S,U)	A	✍
5	U	[U,X)	A	✍
8	X	[X,C)	A	✍
13	C	[C,K)	G	✍
21	K	[K,)	K	✍

Finger Table

Fib.	Start	Int.	Succ.	Sign
1	L	[L,M)	P	✍
2	M	[M,N)	P	✍
3	N	[N,P)	P	✍
5	P	[P,S)	A	✍
8	S	[S,X)	A	✍
13	X	[X,F)	A	✍
21	F	[F,)	G	✍

Figure 4.8: Fibonacci Finger Table after Smart Meter *G* joins the network.

Threat 1: Generating Fake Identifiers/Sybil Attack): In our proposed scheme, only the utility (i.e., trusted authority) generates identifiers by concatenating the smart meters' public key and their IPs, and thus it will reject any form of fake identifier generation.

Threat 2: Finger Table Poisoning Attack: The proposed scheme is robust to this type of attack since the finger table updates are distributed by a signature of the utility to prevent any type of tampering.

Threat 3: Compromising Honest Smart Meters:

- **Not Forwarding CRL Lookup Messages:** This attack can easily be detected by reporting the non-responding smart meters to the utility.

- **False CRL Lookup Message Forwarding:** Considering the fact that the finger tables are calculated by the central authority (i.e., the utility), we mitigate this attack by signing each entry of finger table with utilities' private key. While forwarding the message, the smart meter puts the signed entry of finger table along with the original CRL lookup message. In this way, recipient can check the validity of routing (by checking validity of signature and routing rule). If the signature is not valid or the rule can not be applied for that message, the smart meter reports this bogus forwarding messages to the utility.

- **False revoked response for a non-revoked certificate:** To mitigate this attack, the utility signs each entry of CRL portions. While preparing a revoked certificate response, the responsible smart meter puts the entry of the CRL portion that contains revoked certificate ID and its signature as a proof. Thus, the responsible smart meter is only able to return revoked certificate responses when it has a proof that shows the certificate is actually revoked by the certificate authority.

- **False non-revoked response for a revoked certificate:** Our threat models excludes this attack type since it requires a complete knowledge of AMI network along with finding applicable previously revoked certificates. Note that we assumed in our threat model that only tiny fraction of honest smart meters can be compromised and the attacker does not have a complete knowledge about the AMI topology. Furthermore, the centralized identifier generation of the proposed approach will help to remove the compromised smart meters once they are detected. The utility will exclude them by updating related CRL portion of the successor smart meter and related finger table information.

Physical WiFi Settings		HWMP Protocol		802.11s Settings	
EnergyDetectionThreshold	-89 dBm	ArpMinInterval	40s	MaxBeaconLoss	20
CcaMode1Threshold	-62 dBm	HWMPnetDiameterTraversalTime	2s	MaxRetries	4
RxGain	1 dB	MaxTTL	70	MaxPacketFailure	5
TxGain	1 dB	PathTimeout	100s	**LTE Settings**	
TxPowerLevels	1	RfFlag	false	TxPower	46 dBm
TxPowerEnd	18 dBm	UnicastPreqThreshold	10	DlEarfcn&UlEarfcn	100
TxPowerStart	18 dBm	UnicastPreqThreshold	5	DlBandwidth&UlBandwidth	25
RxNoiseFigure	7 dB	DoFlag	true	IsotropicAntennaModel	true

Table 4.1: NS-3 Physical and Logical Layer Simulation Parameters

4.5 Performance Evaluation

4.5.1 Experimental Setup

To evaluate the performance, Firstly, we used network simulation to create a wireless mesh-based AMI network. The proposed approach is developed under NS-3 simulator [316] which has a built-in implementation of IEEE 802.11s. The underlying MAC protocol used was IEEE 802.11g. The gateway was integrated with the utility systems via LTE which was also implemented in NS-3 for testing the cases where there will be access to outside servers. We created two different grid topologies that consists 81 and 196 smart meters, respectively. The physical and logical layer simulation parameters for these topologies are defined in Table 4.1. Moreover, we assumed a transmission range of 120m and create grid topologies according to this range. A gateway is selected at the upper-left corner of the grid. The smart meters are assumed to generate power readings at certain intervals and create a packet size of 512bytes. The certificates are created using OpenSSL [ope]. We also prepared a DER (binary) encoded CRL list that has been digitally signed according to RFC 5280 [Coo08] which contains 90K revoked certificates.

Second, we built an IEEE 802.11s-based mesh network comprised of 8 TP-LINK TL-WN722N Wi-Fi dongles [TL] attached to Raspberry-PIs. We conducted the experiments in the same manner on the testbed as we did in simulation.

4.5.2 Baseline and Performance Metrics

We considered the following three performance metrics to assess the network performance for the CRL management:

- *Average Packet Delay*: This metrics measures the time it takes for each message to reach the intended smart meter during CRL distribution.
- *Total Time*: This metric indicates the total elapsed time to complete the CRL distribution process.
- *CRL Storage*: This metric indicates the total required storage space for each smart meters
- *CRL Lookup Time*: This is the average lookup time to check whether a certificate is revoked or not. This relates to network delay when a server is accessed. It does not consider the local search time in the CRL file as this is negligible.

We compared the performance of the proposed approach with two other cases. In the first case, we assume that each smart meters keeps the whole CRL locally. In the second case, a bloom filter is used to store revoked certificates information as reported in [RMAT17]. A number of experiment scenarios are planned for performance evaluation of CRL distribution and CRL lookup for each of these cases when applicable. We detail these scenarios below before we move on to experiment results.

Distribution of CRL

In the first scenario, we compressed the CRL file and distribute it to smart meters over the gateway. The gateway distributes the CRL by unicasting to each smart meter. However, this is a costly process in terms of the required bandwidth and time since it is literally sending the same data for over and over again.

Therefore, for the second scenario, we use broadcasting and apply random linear network coding technique [YLL09] to increase the network bandwidth efficiency. To distribute CRL by linear network coding, the compressed CRL file is divided into generations as illustrated in Fig 4.9. Each generation contains same number of k packets denoted p_i, $i = 1, 2, ..., k$, which are d bytes each. Each packet is broadcast through the gateway with an encoding vector header. When an encoded packet is received by the smart meter, the smart meter checks if the encoding header of this packet is linearly dependent with that of all previously received encoded packets. If the packet is not dependent, it is stored in a buffer, and the smart meter tries to decode all the packets stored in the buffer. If dependent, the smart meter returns an ACK message to the gateway. If all nodes finish the current generation, the next generation will be broadcast till the whole compressed CRL file is received by all smart meters.

In the third scenario, we use Bloom filter to store the revoked certificates information. To do so, we read previous CRL file and we insert each revoked ID to a Bloom filter by discarding the revocation date information. While Bloom filter can also provide advantages to reduce the overhead of distributing, storing and lookup of the CRL, it suffers from false positives, where there is a chance that a search may indicate a certificate is in the bloom filter as revoked when it actually is not. For each false positive incident, the smart meter should check the revocation information from CA with using remote point-to-point accessibility. However, Bloom filter allows to reduce false positive rates below a certain level by sacrificing its storage advantage [BMP$^+$06]. Therefore, we assumed that 1% error rate is acceptable for our scenario and built a typical Bloom filter with 1% error rate. We signed the formed Bloom filter and distribute it by using both unicast and linear network coding similar to previous scenarios.

As the final scenario, we consider our proposed case where finger table parameter k is selected as 3 and 5 for 81 node and 196 nodes AMI network cases, respectively. Since each CRL portion is different from each other, we only use unicast through the gateway to send CRL portion and finger table to the related smart meter.

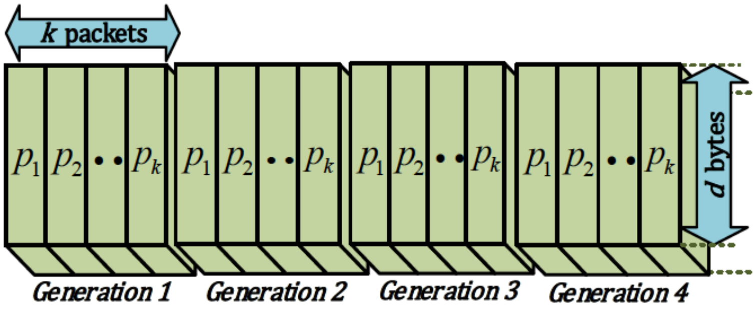


Figure 4.9: Data Partitioning for Network Coding

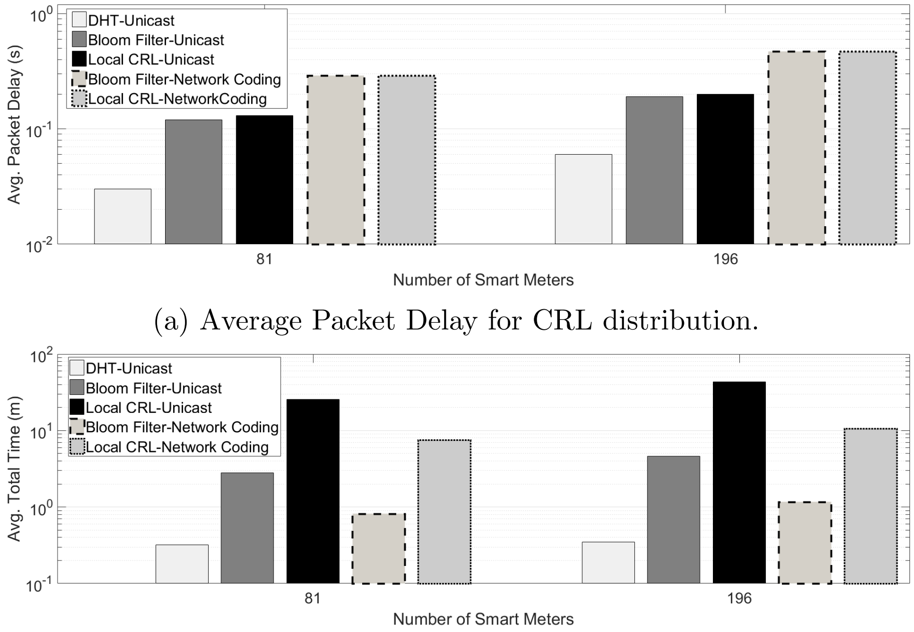
CRL Lookup

We assume that each smart meter has already got required CRL information and 10% of randomly chosen smart meters need CRL lookup at the same time. We compare our approach to the two other baselines mentioned in terms of required space to keep CRL information and the elapsed time to check whether the certificate is valid or not.

4.5.3 Experiment Results

CRL Distribution Overhead

We first conducted experiments to assess the CRL distribution overhead of the proposed approach. We set time interval 0.1 second between broadcast messages while distributing the CRL. As noted before, for DHT, we did not use broadcasting and thus only unicast results will be discussed.



(a) Average Packet Delay for CRL distribution.

(b) Average Total Time for CRL distribution.

Figure 4.10: CRL Distribution Overhead

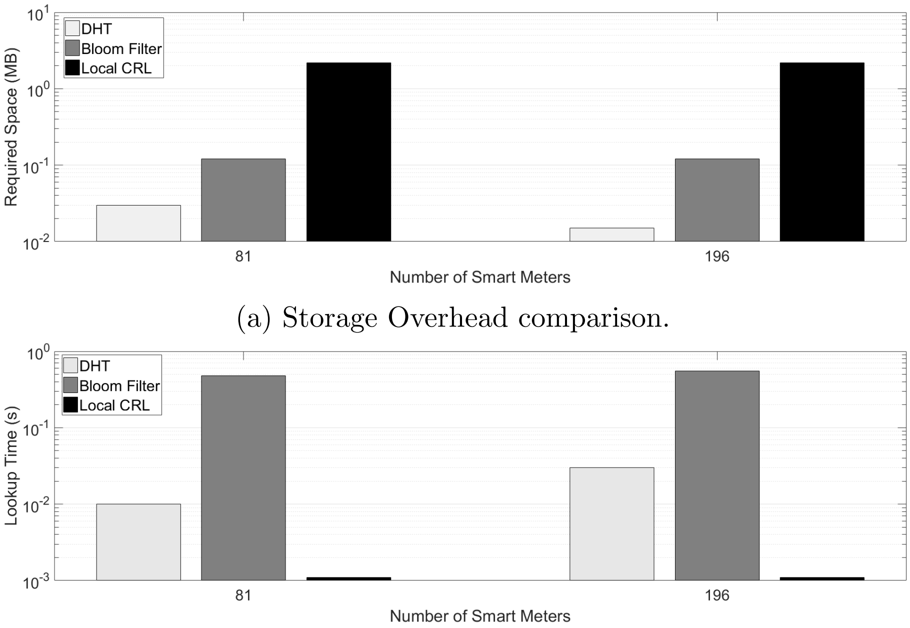
The distribution overhead for all the approaches are shown in Figure 4.10a and Figure 4.10b. These results indicate that DHT has significantly less the message delay overhead than Bloom filter and local CRL approaches due to partitioning advantage. This is also apparent when broadcasting is used in the case of Bloom filter and local CRL approaches. Another observation is that, although network coding helps to decrease the total time, it causes the average packet delay to increase. As seen from Figure 4.10a, the packet delay increases up to 0.47 seconds. This can be attributed to the fact that there will be more contention and congestion in the network and even some of the packets may be dropped and resent.

Looking at the total time results in Figure 4.10b, we see that the results are encouraging since our DHT-based approach outperforms the others and provides significant reductions in terms of total completion time of the distribution. According to these results, the average total time for the local CRL and bloom filter approaches is increasing at a faster rate than DHT which hints about the scalability of our approach. DHT approach has better scalability due to the fact that it is not affected from the network size. This makes DHT even a better candidate to be employed.

CRL Storage and Lookup Overhead

To compare the storage requirements, we identified the needed CRL size for our approach and compared with the other baselines. For our scenario, the CRL size is nearly 2MB for 90K revoked certificates. Figure 4.11a shows the comparison of these approaches. As expected, DHT needs to store a small portion of CRL since the whole CRL list is distributed to smart meters in nearly equal portions. However, Local CRL approach keeps the whole list and thus the storage requirements is much higher. DHT-based approach needs to store nearly $1/N$ of CRL size which is around 36KB and 14KB for 81 and 196 node cases respectively. Note that increased network size is an advantage and reduces storage requirement significantly. While Bloom filter's performance is also promising, it is still at least doubling the storage and additionally it may suffer from false positives which increases delay as will be described below.

Looking at the Lookup delay for the CRL, as expected, there is no delay for local CRL approach (other than the negligible search time within the file). The Bloom filter is also pretty fast but in case of false positives, it suffers from increased network delays to access a remote server which increases the average delay. Our proposed approach brings more delay since the lookup time needs to access smart meters which requires transmission of queries over the mesh network. However, this time is very reasonable and still significantly less than Bloom filter approach if the



(a) Storage Overhead comparison.

(b) Average CRL Lookup Delay comparison.

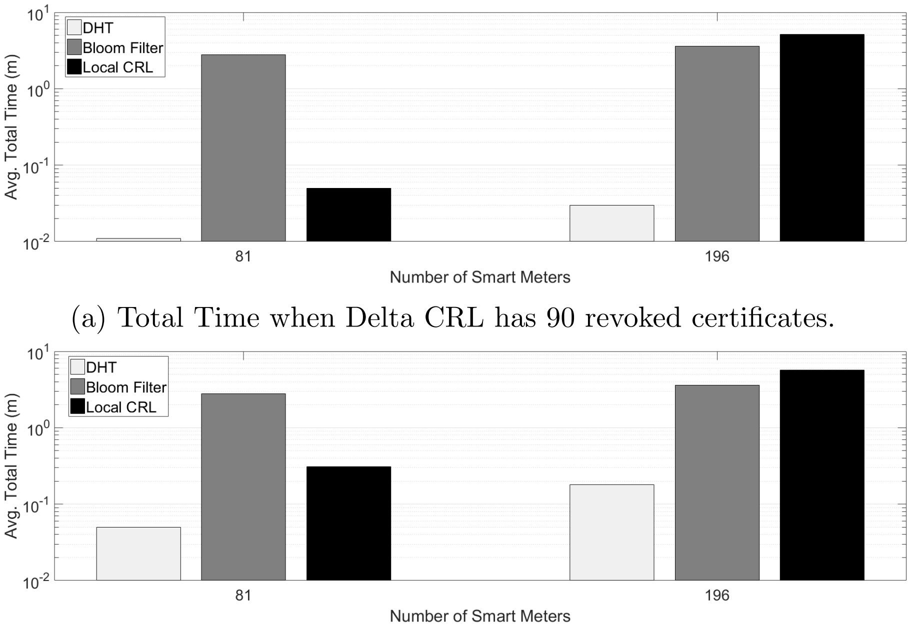
Figure 4.11: CRL Storage and Lookup Overhead

path is already known as shown in Figure 4.11b. As can be seen from the table, it is only slightly affected from the network size increase.

CRL Update Overhead

In this subsection, we conducted an experiment to asses the performance of CRL updates. We assume that the CRLs are updated regularly by using *delta CRL* concept. To assess the overhead of CRL update in two different scenario, we assume that we have two different delta CRLs which contain 90 and 900 entries in delta CRLs, respectively. Figures 4.12a and 4.12b show the distribution overhead for these two delta CRLs.

In local CRL approach, as in the case of full CRL distribution, every smart meter should store the delta CRLs as well. This requires distribution of delta CRLs to all smart meters. Therefore, the overhead of CRL distribution will be proportional



(a) Total Time when Delta CRL has 90 revoked certificates.

(b) Total Time when Delta CRL has 900 revoked certificates.

Figure 4.12: CRL Update Overhead

to the size of delta CRL. The results in Figure 4.12a and Figure 4.12b confirmed that idea. As can be seen, delta CRL reduces the distribution overhead of local CRL approach compared to the full CRL case discussed before. However, delta CRL concept does not bring any advantage to the Bloom filter approach. For each updated revocation information, the bloom filters must be created again from the scratch by using all revocation information to carry both previous and new revoked certificates. As a result, updating the CRL will have almost the same CRL distribution overhead for Bloom filter.

The only approach that brings an advantage to delta CRL concept itself is our approach. As seen, the revoked certificate IDs are almost evenly distributed to the smart meters for the delta CRL which contains 900 revoked certificates. This will result in decreasing the overhead of CRL update even further as seen in Figure 4.12b. However, the real advantage of the approach is depicted in Figure 4.12a where delta

CRL contains 90 certificates. Due to the partitioning process, most of the smart meters is not need to update its CRL portion since there is no partition that belongs to them. Therefore, partitions of delta CRL is just sent to a small set of the smart meters. As a result, the total time for updating delta CRL is around 2 seconds comparing to more than 5 and 3 minutes for local CRL and bloom filter.

4.5.4 Testbed Experiments

While building the testbed, We carefully dispersed the Raspberry PIs around aisles of the Engineering Department. The Raspberry PIs are placed not to be in a line-of-sight position between each other. There are concrete walls, doors between them as can be seen in Figure 4.13. By this positioning, we try to mimic realistic conditions that reflects the path attenuation, refraction and diffraction of the signal while it propagates through space in wild. Wireless USB Adapter TL-WN722N allows Raspberry PIs to create to a wireless network which complies with IEEE 802.11g and shows abilities of transferring data through obstacles even in a steel-and-concrete structure. In this subsection, we will describe applied test scenarios to test the proposed approach over the created testbed.

4.5.5 Distribution Overhead

We implemented a client-server application to distribute the local CRL, delta CRLs, bloom filter file and DHT CRL portions by unicasting over the defined gateway in the testbed. Looking at the total time results in Figure 4.14, we see that the DHT-based results are slightly worse than the Bloom filter results. However, the result is still encouraging since even for such a small size testbed DHT-based approach provides significant reductions in terms of total completion time. Moreover, we

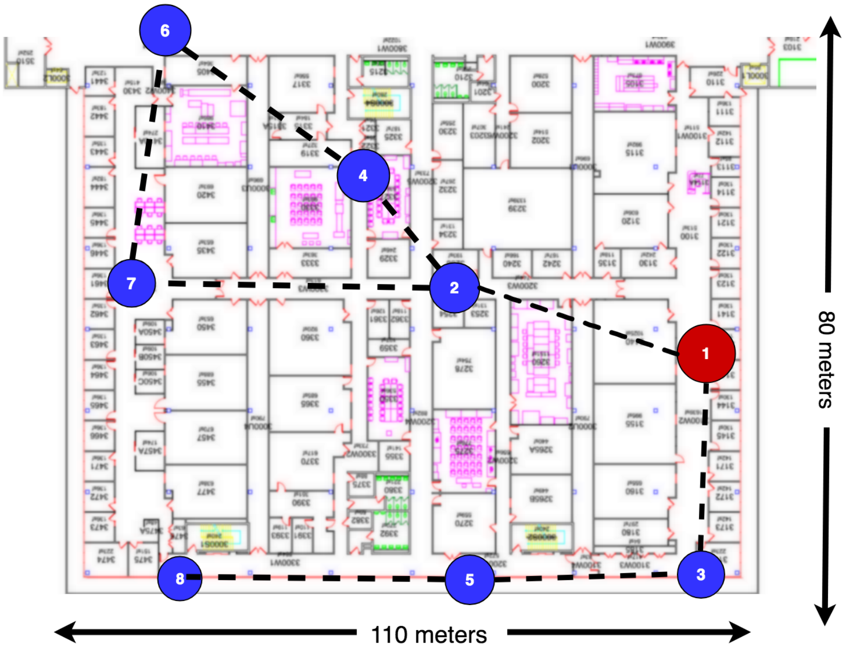


Figure 4.13: 8-node AMI Testbed created in FIU Engineering Center

observe that the distribution of local CRL takes more than 3 minutes even on such a small size network. This is mostly because of packet loss in transit due to radio frequency interference and weak radio signals because of distance or multi-path fading. This shows the importance of our approach, since it decreases the size of CRL significantly and hence reduces number of the required network packet to be sent. For delta CRL concept, our approach outperforms the others and provides significant advantage in delta CRL concept similar to the simulation results.

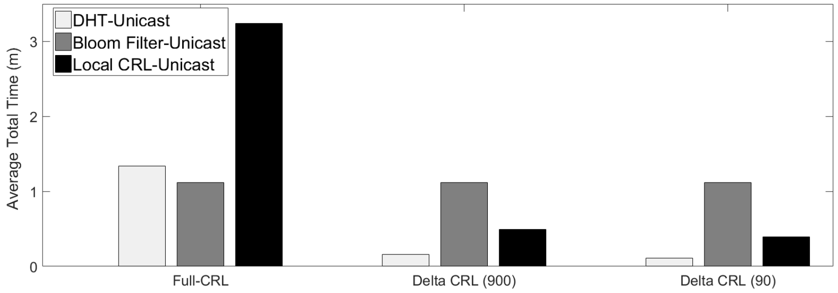


Figure 4.14: Distribution Overhead on the AMI Testbed

4.5.6 CRL Lookup Overhead

To conduct CRL Lookup experiment on testbed, we implemented our DHT lookup routing mechanism in Raspberry PIs. Similar to simulation experiment, each Raspberry PIs makes random CRL lookup requests and we measured respond time of each request. Figure 4.15 shows the results of this experiment. As in simulation environment, there is no network delay for local CRL. The Bloom filter results represent the average respond time in case of false positives. Average DHT lookup takes around 30ms which is slightly more than the results which we obtained in the simulation. This is mostly because of the channel access waiting time in CSMA/CA. Note that we observed at least 25 different access points which use the same frequency band with our testbed to serve other students and thus increased channel access time is inevitable.

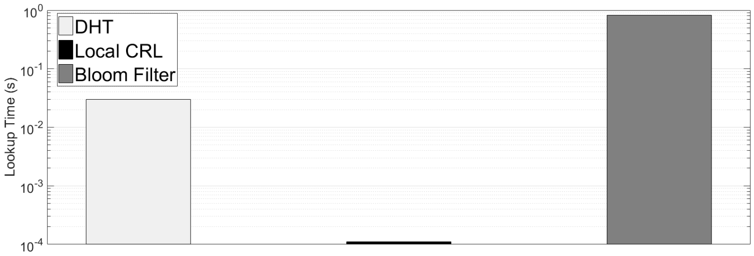


Figure 4.15: Average CRL Lookup Delay comparison on the AMI Testbed

4.6 Conclusion

Considering the overhead of certificate and CRL management in AMI networks in the context of Smart Cities, in this Chapter, we proposed a DHT-based algorithm for creating and distributing the CRLs. Our approach strives to exploit the capabilities and resources of all the smart meters so that they accomplish CRL management in a collaborative and distributed manner. Basically, DHT structure is used to access the CRLs when a certificate is to be queried. In this way, the size of CRL was significantly reduced. We presented the algorithms to create CRL portions and update them in smart meters.

We implemented the proposed DHT-based approach both in NS-3 simulator and AMI testbed that run a version of 802.11s. The experiment results indicate that the DHT-based approach can reduce the CRL size significantly which helps reducing the distribution and the storage overhead. Particularly, the AMI testbed results showed the importance of our approach for distribution overhead in a multi-hop wireless mesh network such as AMI when compared to two other existing CRL management approaches.

CHAPTER 5

A LIGHTWEIGHT CRYPTOGRAPHIC ACCUMULATOR-BASED REVOCATION MANAGEMENT FOR AMI

AMI forms a communication network for the collection of power data from smart meters in Smart Grid. As the communication between smart meters could be secured utilizing public-key cryptography, however, public-key cryptography still has certain challenges in terms of certificate revocation and management particularly related distribution and storage overhead of revoked certificates. To address this challenge, in this chapter, we propose a communication-efficient revocation or CRL mananegment scheme for AMI networks by using RSA accumulators [CL02]. RSA accumulator is a cryptographic tool which can represent a set of values with a single accumulator value (i.e., digest a set into a single value). Also, it provides a mechanism to check whether an element is in the set or not which implicitly means that cryptographic accumulators can be used for efficient membership testing. To employ accumulators for revocation management, we propose an accumulator manager within the utility company (UC) is tasked with collection of CRLs from CAs. The accumulator manager accumulates the collected CRLs (i.e., revoked certificates' serial numbers) to a single accumulator value which will then be distributed to the smart meters. In addition to that, we endorse a non-revoked proof concept to allow a smart meter to check whether another meter's certificate is revoked without a need to refer to the CRL file. We define additional entities within AMI and assign functions to them to govern an accumulator based revocation management by addressing several security threats.

The computation and communication related aspects of the proposed approach is assessed via simulations in ns3 network. In addition, we built an actual testbed using in-house smart meters to assess the performance realistically. We compared our approach with the other methods that use conventional CRL schemes and Bloom-filters [ARMT14]. The results show that the proposed approach significantly outperforms the other existing methods in terms of reducing the communication overhead that is measured with the completion time. The overhead in terms of computation is not major and can be handled in advance within the utility that will not impact the smart meters.

This chapter is organized as follows: In the next sections, we summarize the fundamentals of accumulator concept as a background. Section 5.2 introduces the threat model. Section 5.3 presents the proposed approach with its features. Section 5.4 and 5.5 are dedicated to evaluation criteria and experimental validation. Section 5.6 analyzes the security of the approach. Section 5.7 discusses the benefits and limitations. The chapter is concluded in Section 5.8.

5.1 Background on Cryptographic Accumulators

Benaloh and De Mare [BDM93] introduced the cryptographic accumulator concept which is a one-way hash function with a special property of being *quasi-commutative.* A quasi-commutative function is a special function $\mathcal{F}$ such that $y_0, y_1, y_2 \in \mathbb{Y}$:

$$\mathcal{F}(\mathcal{F}(y_0, y_1), y_2) = \mathcal{F}(\mathcal{F}(y_0, y_2), y_1) \tag{5.1}$$

The properties of this function can be summarized as follows: *1)* it is a one-way function, i.e., hard to invert; *2)* it is a hash function for obtaining a secure digest $\mathcal{A}$ (i.e., accumulator value) where $\mathcal{A} = \mathcal{F}(\mathcal{F}(\mathcal{F}(y_0, y_1), y_2), ..., y_n)$ for a set of values $\{y_0, y_1, y_2, ..., y_n\} \in \mathbb{Y}$; *3)* it is a *quasi-commutative* hash function which is different from other well-known hash functions such that the accumulator value $\mathcal{A}$ does not depend on the order of y_i accumulations.

These properties allow cryptographic accumulators to be used for a condensed representation of a set of elements. In addition, since the resulting accumulated hashes of y_i ($\mathbb{Y} = \{y_i;\ 0 < i < n\}$) stays the same even if the order of hashing is changed, it can be used for efficient membership testing by using a special value called witness value w_i. For instance, the witness w_i of corresponding y_i is calculated by accumulating all y_j except the case where $i \neq j$ (e.g., $w_i = \mathcal{F}(\mathcal{F}(\mathcal{F}(y_0, y_1), ..., y_{j-1}, y_{j+1}..., y_n))$. Then, when necessary any of the members can check whether y_i is also a member of the group by just verifying whether $\mathcal{F}(w_i, y_i) = \mathcal{A}$. Note that, because $\mathcal{F}$ is a one-way function, it would be computationally infeasible to obtain w_i from y_i and $\mathcal{A}$. However, there is a risk for collusion in this scheme when an adversary can come up with ${w_i}'$ and ${y_i}'$ pairs where ${y_i}' \notin \mathbb{Y}$ to obtain the same accumulator value: $\mathcal{F}({w_i}', {y_i}') = \mathcal{A}$. In the literature, there is already a cyrptographic accumulator, namely the RSA construction [BP97] which guarantees that finding such pairs is computationally hard by restricting the inputs to the accumulator function to be prime numbers only. This scheme is known as collision-free accumulator that enables secure membership testing (i.e., without any collision). Therefore, we chose to employ RSA construction which is elaborated next.

5.1.1 RSA Accumulator

RSA accumulator [BP97] has a RSA modulus $\mathcal{N} = pq$, where p and q are strong primes. The RSA accumulation value $\mathcal{A}$ is calculated on consecutive modular exponentiation of prime numbers set $\mathbb{Y} = \{y_1, ..., y_n\}$ and g is quadratic residue of $\mathcal{N}$ as follows:

$$\mathcal{A} = g^{y_1,...,y_n}\ (mod\ \mathcal{N}) \tag{5.2}$$

The witness w_i of corresponding y_i is calculated by accumulating all values except y_i:

$$w_i = g^{y_1,\dots,y_{i-1},y_{i+1},\dots,y_n}\ (mod\ \mathcal{N}) \tag{5.3}$$

Then, the membership testing can be done via a simple exponential operation by comparing the result with the accumulator value $\mathcal{A}$:

$$w_i^{y_i} \leftrightarrow \mathcal{A} \tag{5.4}$$

The described accumulator scheme so far basically allows generation of a "witnesses" to prove that an item is in the set. A more advanced accumulator would offer proofs of non-membership which proves that an item is **NOT** in the set [LLX07]. For this scheme, let us assume any $x \notin \mathbb{Y} = \{y_1, ..., y_n\}$. In a nutshell, the non-witness values can be computed by the following steps: Let u denote $\prod_{i=1}^{n} y_i$, the scheme finds non-witness nw_1, b value pairs of x by solving the equation of $nw_1 \times u + b \times x = 1$ using the Extended Euclidean algorithm. Then, the scheme computes an additional value nw_2 such that:

$$nw_2 = g^{-b}\ (mod\ \mathcal{N}) \tag{5.5}$$

After these steps, the item x will have cryptographic proof values nw_1 & nw_2 which can be used to ensure that the item x is **NOT** in the set $\mathbb{Y}$. Then, any third party that posses the $\mathcal{A}$ value can do the non-membership test of x via a simple exponential operation by checking whether the following equation holds:

$$\mathcal{A}^{nw_1} \leftrightarrow {nw_2}^{x} \times g\ (mod\ \mathcal{N}) \tag{5.6}$$

Besides, if a new value $y^{'}$ is added to list, the accumulator value is updated by using the previous accumulator value $\mathcal{A}$:

$$\mathcal{A}^{'} = \mathcal{A}^{y^{'}}\ (mod\ \mathcal{N}) \tag{5.7}$$

5.2 System and Threat Model

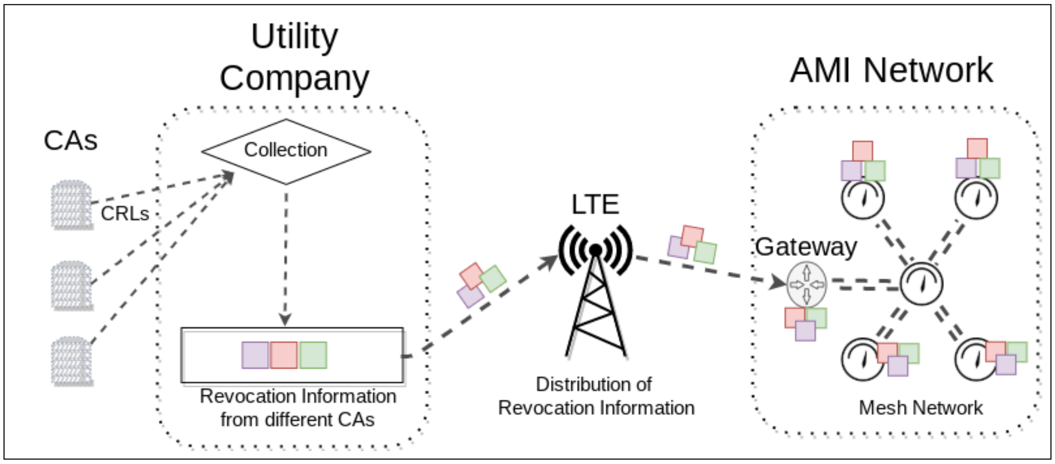

Utility Company
AMI Network
CAs
Collection
CRLs
LTE
Gateway
Revocation Information from different CAs
Distribution of Revocation Information
Mesh Network

Figure 5.1: System Model

In this Chapter, we build a revocation management scheme for a typical AMI infrastructure. Basically, revocation information is collected by the utility company in the forms of CRL files. Each CRL file contains revoked certificates IDs issued by different CAs. Then, all these revocation information are disseminated to AMI through a 4G/LTE and AMI mesh communication infrastructure. A sample system model is shown in Fig. 5.1.

The security of the proposed revocation management scheme depends on the secure implementation of the proposed accumulator-based system. Therefore, we consider the following threats to the security of the proposed approach and identified the relevant security goals. Note that in our attack model, we assume that both the accumulation process within the perimeter and smart meters (outside the perimeter) can be compromised. Besides, the communication between UC and smart meters is happening on a non-secure medium which means an adversary can eavesdrop the communication both actively and passively. This threat model is very strong and adequate to represent the increasing threats to Smart Grid. However, a single counter-measure against this threat model would not be sufficient when considering the broad and diverse attack surface of it. To ensure the security of AMI against adversaries, the utility company needs to deploy intrusion prevention systems and proper attack prevention tools as well. Thus, we assume that a PKI inspection system along with an intrusion detection system (IDS) is already deployed and provides device-level controls to protect PKI keys and informs UC in case of any infiltration.

❶ **Compromising Smart Meters:** In an attacker's perspective, the meter/gateway is the entry point to the AMI. The attacker can use a compromised smart meter or impersonate the gateway to apply various attacks.

❷ **Compromising the UC Servers:** Apparently, compromising the servers within perimeters of UC provides lots of attack opportunities to adversary. The adversary can target AMI by directly attacking revocation management through compromising servers that governs revocation operations.

❸ **Compromising the Communication:** When UCs are deploying AMI systems, they generally opt-out enabling encryption since IEEE standards does not enforce the UCs to deploy encryption due to various reasons [IEE12]. It makes AMI open to adversaries who can easily eavesdrop whole AMI traffic or a portion of it. This can also pose a threat to revocation management.

5.3 Proposed Approach

5.3.1 Overview

The proposed approach basically eliminates the need to store and distribute CRLs when the devices communicate in a secure manner. Instead of keeping a CRL file for verification of revocation status of certificates, our approach dictates to store at each device (e.g., smart meter, gateway, HES, etc.) only an accumulator value and a proof which proves the validity of the device's certificate. The accumulator value and proof can be computed at the utility company and distributed to devices in advance. Any updates regarding revoked certificates trigger re-computation of these values. Keeping just two integer values for revocation management brings a lot of efficiency in terms of storage and distribution overhead as will be shown in the Experiments section. In the next subsections, we will explain the details of our approach.

5.3.2 Adaptation of RSA Accumulator for Our Case

To apply the cryptographic accumulators for revocation management, the revocation management needs to be viewed holistically from the lens of systems thinking to ensure security. We took a bottom-up approach while adapting the accumulator scheme to our approach. First, we modified the CRL inputs to meet the requirements for constructing a secure accumulator setup. Second, we improved the performance of accumulator calculation. Third, the accumulation process was divided into different functions and their tasks were defined. Then, we introduced new entities to AMI and assigned tasks to them. Lastly, we constructed a revocation check protocol that utilizes the produced accumulator solution. This section covers how we accomplished all these steps in details.

a. **Integration of CRL and non-witness Concept:**

In the traditional CRL approach, when a smart meter presents its certificate to the recipient meter, that meter needs to verify that the presented certificate is **NOT** in the CRL. To be able to employ the accumulator approach, we generate *non-witness values* for the presenter to prove that it is not in the list. We accumulate the revocation information (stored in CRLs) into a single accumulator value and produce non-membership witnesses for the non-revoked smart meters.

b. Reducing the Complexity of Accumulator Computation:

While computing the accumulator value using Eq. 5.2, the exponent needs to be computed as $\prod_{i=1}^{n} y_i$ before doing the modular exponentiation. This becomes infeasible when the size of $\mathbb{Y}$ increases since $u = \prod_{i=1}^{n} y_i$ will be $n \times k$ bits assuming each y_i is a k-bit integer. In our approach, we decided to use Euler's Theorem [RGO05] to cope with this complexity. With access to the totient of $\mathcal{N}$ (i.e., $\phi(\mathcal{N})$), the exponent of g in accumulation computation will be $u^{'} = \prod_{i=1}^{n} y_i \ mod \ \phi(\mathcal{N})$. Thus, with the knowledge of the totient, it becomes more efficient to compute the required values via reducing the u by $\phi(\mathcal{N})$.

c. Generating Prime Inputs for the Accumulator: For accumulation, we can use the certificate IDs (c_{id}) which are generated by the CAs. However, to ensure a collision-free accumulator, we need to use only prime numbers as dictated by the RSA accumulator. Since CRLs contain arbitrary serial numbers for certificate IDs, it is necessary to compute a prime representative for each certificate ID as an input to the RSA accumulator. Thus, we used the random oracle based prime number generator described in [PTT08] to obtain prime representatives of certificates from their serial numbers (c_{id}). The scheme basically has a random oracle $\Omega()$ function which produces a random number r for an input c_{id}. We use $\Omega()$ to find a *256-bit* number, d, which causes the result of the following equation to be a prime number:

$$y = 2^{256} \times \Omega(c_{id}) + d \tag{5.8}$$

By solving this equation, we generate a prime representative y for a revoked certificate. The reader is referred to [PTT08] for security proof details of the method.

d. Defining Functions of Revocation Management:

After preparing the inputs, we compiled and modified the offered accumulator structure and proposed the following functions to construct revocation management for AMI. Our RSA accumulator uses the following input sets: $\mathbb{Y}$ *is the set of prime representatives of revoked certificates' serial numbers* and $\mathbb{X}$ *is set of prime representative of valid certificates' serial numbers* where $x \in \mathbb{X}$:

- $aux_{info}, \mathcal{N} \leftarrow Setup(k)$: This function is to setup the parameters of the accumulator. It takes k as an input which represents the length of the RSA modulus in bits (e.g., 2048, 4096, etc.) and generates modulus $\mathcal{N}$ along with aux_{info} which is basically Euler's totient $\phi(\mathcal{N})$.
- $\mathcal{A} \leftarrow ComputeAcc(\mathbb{Y}, r_k, aux_{info})$: This is the actual function which accumulates revocation information by taking prime representatives of serial numbers set $\mathbb{Y}$. While computing the accumulator value, we propose to use an initial random secret prime number r_k as a first exponent (g^{r_k}) in Eq. 5.2.
- $nr_{proof} \leftarrow ComputeNonRevokedProof(aux_{info}, \mathbb{Y}, x)$: This function first computes a pair of non-witness values represented as (nw_1, nw_2) for a valid certificate whose prime representative is x. Then, the UC concatenates the non-witness value pair with x and the serial number of the certificate creating a 4-tuple called nr_{proof}.

- $0, 1 \leftarrow RevocationCheck(\mathcal{A}, nr_{proof})$: When a smart meter which has a prime representative x wants to authenticate itself to another party, the other one uses nr_{proof} and $\mathcal{A}$ to verify that x is *not* in the accumulated revocation list by checking Eq. 5.6.
- $\mathcal{A}^t \leftarrow UpdateAcc(\mathcal{A}^{t-1}, \mathbb{Y}^t)$: This function is for updating the accumulator value $\mathcal{A}$ when the revocation information is updated via deltaCRLs. It takes a set of prime representatives of corresponding newly revoked certificates $\mathbb{Y}^t$ and latest accumulator value $\mathcal{A}^{t-1}$, and returns the new accumulator value $\mathcal{A}^t$ by utilizing Eq. 5.7.
- ${nr_{proof}}^t \leftarrow UpdateNonRevokedProof(\mathcal{A}^t, \mathbb{Y}^t, x)$: This function is for updating the non-revoked proof of corresponding valid smart meters when the revocation information is updated via deltaCRLs. It takes a set of prime representatives of corresponding newly revoked certificates $\mathbb{Y}^t$, the updated accumulator value $\mathcal{A}^t$, and the prime representative x and returns non-revoked proof ${nr_{proof}}^t$ of smart meter after some additional certificates are revoked by utilizing the process for Eq. 5.5.

Next, we define the components of the proposed framework.

5.3.3 Components of Revocation Management System

We propose the system architecture shown in Figure 5.2 to enable the proposed revocation management and to define its interaction with the deployed AMI components. In addition, the newly introduced components of this architecture and their roles in executing the above defined functions are described below:

- *Smart Meters and Gateway:* The smart meters and gateway can directly communicate with each other and with Head-end System (HES) over LTE.

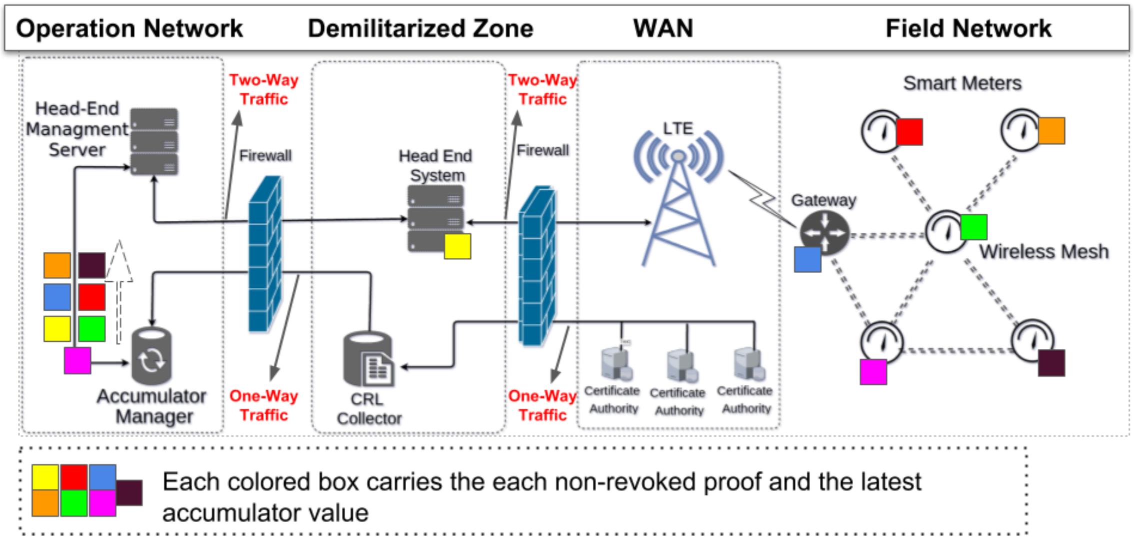


Figure 5.2: The structure of proposed revocation management.

Thus, to ensure the security of applications, these devices need to run the $RevocationCheck()$ function and carry the latest $\mathcal{A}$ and the corresponding nr_{proof}.

- *Head-End System:* HES is an interface between the utility operations center and smart meters, and it is located in a demilitarized zone (DMZ). The primary function of the HES is collecting the power data from smart meters and transfer them to head-end management servers (HMS). Since it has two-way communication with smart meters, it needs to run the $RevocationCheck()$ function and carry the latest $\mathcal{A}$ and its nr_{proof}.

- *CRL Collector:* The CRL collector plays one of the key roles in our revocation management system. It basically collects CRLs from various CAs and feeds them to the Accumulator Manager. Since it has an open interface to the outside network (communicating with other CAs), it is placed in DMZ area.

- *Accumulator Manager:* Accumulator Manager is the core of our revocation management scheme. It gets CRL information from the CRL Collector and accumulates them to obtain latest accumulator value. It implements the *Setup()*, *ComputeAcc()*, *ComputeNonRevokedProof()*, *UpdateAcc()*, and *UpdateNonRevokedProof()* functions. Whenever a new accumulator value is calculated at a time t, it sends the accumulator value $\mathcal{A}^t$ and updated ${nr_{proof}}^t$ to the HMS which then forwards them to HES for distributing to the smart meters.

- *Head End Management Server:* The collected data is managed within HMS. It basically monitors activity logs, identifies new devices and manages incident response processes. As mentioned, the HMS collects the newly generated $\mathcal{A}$ and nr_{proof} values and sends them to HES for distribution.

5.3.4 Revocation and Certificate Verification Processes

In this section, we describe the proposed revocation scheme and the protocol for certificate verification.

Accumulating the CRL

This process includes two phases namely the setup phase and the update phase which are described below.

- **The setup phase:** In this phase of our approach, the Accumulator Manager in the UC basically accumulates the revoked certificate IDs in *full CRLs*. This process works as follows: The *full CRL* files are read, and each certificate ID and its issuer's public key are concatenated to obtain a unique string that will be input to the accumulator. Note that the issuer's public key is concatenated on purpose to eliminate any duplicates in serial numbers that may come from different CAs. Then, the Accumulator Manager calculates prime representatives for each concatenated string and accumulates these prime representatives to obtain the accumulator value. Finally, the Accumulator Manager generates non-revoked proofs (i.e., the 4-tuple nr_{proof}) for each end-device (smart meter, gateway, HES, etc.) by using $ComputeNonRevokedProof()$ function.

- **The update phase:** This phase is for revocation information updates that can be done through *delta CRLs*. Due to such updates, the accumulator value $\mathcal{A}$ and nr_{proof} values should be updated. To update these values, the Accumulator Manager first prepares the prime representatives for the newly revoked certificates (i.e., the ones that are included in the *delta CRLs*) by following the same approach in the setup phase. It then updates the previously computed accumulator value, $\mathcal{A}^{t-1}$, by using the $UpdateAcc()$ function to obtain $\mathcal{A}^t$ which is then used to generate new nr_{proof} tuples for the end devices by using the $UpdateNonRevokedProof()$ function.

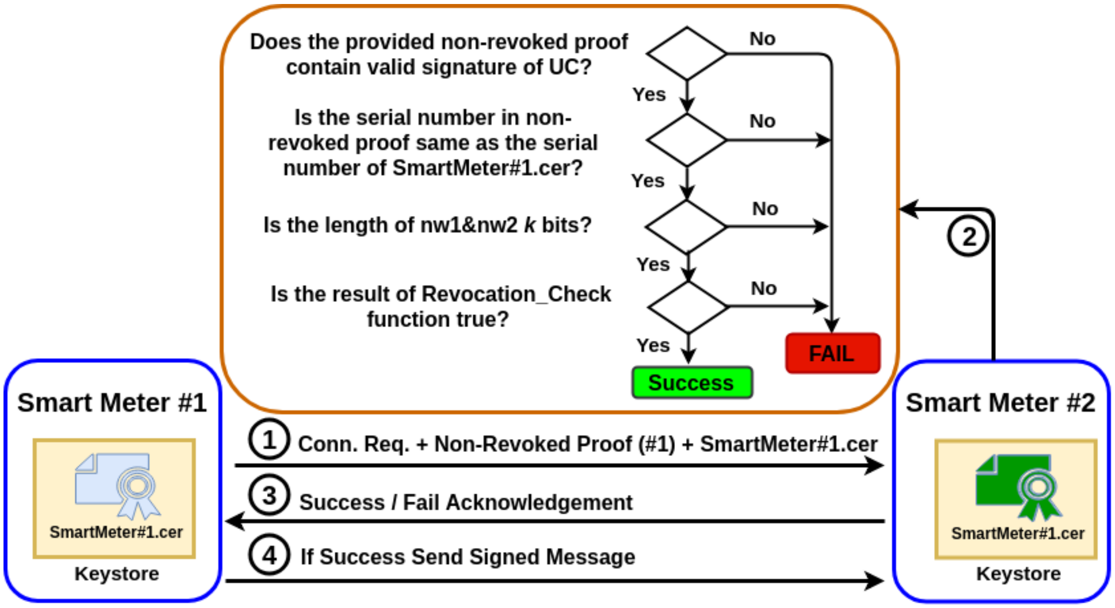

Does the provided non-revoked proof contain valid signature of UC?
No
Yes
Is the serial number in non-revoked proof same as the serial number of SmartMeter#1.cer?
No
Yes
Is the length of nw1&nw2 k bits?
No
Yes
Is the result of Revocation_Check function true?
No
Yes
FAIL
Success
2
Smart Meter #1
SmartMeter#1.cer
Keystore
Smart Meter #2
SmartMeter#1.cer
Keystore
1 Conn. Req. + Non-Revoked Proof (#1) + SmartMeter#1.cer
3 Success / Fail Acknowledgement
4 If Success Send Signed Message

Figure 5.3: Certificate Verification Protocol Scheme.

Certificate Verification Protocol

When two meters communicate by sending/receiving signed messages, the signatures in these messages need to be verified. To be able to start the verification process, a receiving device needs to use the public key (for signature verification) presented in the certificate sent to itself. To ensure that this certificate is not revoked, then it needs to initiate a process which we call as certificate verification protocol. Figure 5.3 shows an overview of this process. Basically, the receiving device checks the corresponding nr_{proof} tuple's signature to ensure that it is produced by the UC. Once the signature is verified, it then checks whether the the serial number within the tuple is same as the serial number of the provided certificate (i.e., either EndDevice#1.cer). For additional security, it also checks the length of the $nw_1 \& nw_2$ to see whether it is equal to the first accumulation setup parameter k. Next, it perfoms $RevocationCheck()$ function amd checks whether the provided nr_{proof} is correct. Finally, the signature of connection request message is checked to ensure the integrity and authenticity of the request. If all these steps are successful, the end-device has successfully complete the certificate verification protocol. Note that, without carrying the nr_{proof}, a smart meter can not be authenticated even if it has a valid certificate.

5.4 Evaluation of the Approach and its Objectives

The main objective of our work is to decrease the dissemination overhead of revocation information on AMI. However, although any reduction in this overhead is important for the general health of Smart Grid, achieving this goal without sacrificing the security is vital. Thus, we have determined the following general measures to evaluate the proposed approach in terms of security, communication, computation and storage.

- First, we will evaluate distributing process of non-revoked proofs to smart meters to assess the communication-related overhead of our approach.
- Second, since our approach requires computational resources to calculate the non-revoked proofs and accumulator value, we will evaluate computational costs on UC servers.
- Considering the limited computation resource of a typical smart meter, it is essential to evaluate computational aspects on smart meters as well. Moreover, we will evaluate storage space requirement of our approach for smart meters.
- Finally, we will assess the security of the proposed approach against threats that are defined in Section 5.2.

We evaluate the communication, computation and storage overhead of our approach by using the following metrics:

- *Completion Time*: This metric is defined for communication overhead assessment, which indicates the total elapsed time to complete the distribution of accumulator value and non-revoked proofs to the smart meters from the HES. This metric hints on the communication overhead of revocation management in terms of assessing how it keeps the communication channels busy which are critical for carrying other information.
- *Computation Time*: This is the metric to measure the total time for completing the required computations such as computation of accumulator value, prime representatives, and revocation check time, etc.

- *Storage*: This metrics indicates the amount of space for storing the CRL information in the meters.

For comparison to our approach, we used two other baselines from the literature:

- *Traditional CRL Method*: Each smart meter keeps the whole CRL [MMS13] locally which is distributed by the UC.
- *Bloom Filter Method*: A Bloom filter [ARMT14] is used to store revoked certificates information. Note that, we employed *murmur* hash function, which is a non-cryptographic hash function suitable for *fast* hash-based lookup, to build this Bloom filter. In this case, the Bloom Filter is distributed to each meter by the UC.

5.5 Performance Evaluation

5.5.1 Experimental Setup

To assess the performance of the proposed approach, we implemented it in C++ by using FLINT [HJP11], which is the fastest library for number theory and modular arithmetic operations over large integers. For the RSA modulus generation and prime representatives computation, we used Crypto++ library since it allows thread-safe operations. We prepared a binary-encoded *full CRL* and *delta CRL* that have been digitally signed according to RFC 5280 standard and contained 30,000 and 1000 revoked certificates for *full CRL* and *delta CRL* respectively. The *full CRL* was used to compute $\mathcal{A}$ and nr_{proof} tuples during the setup phase while the *delta CRL* ws used for updating both $\mathcal{A}$ and nr_{proof} tuples.

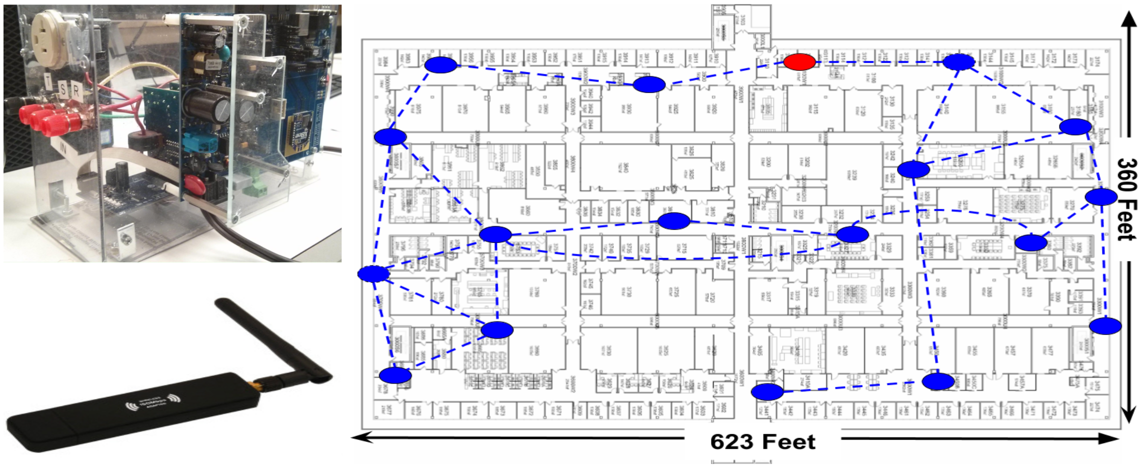
For communication overhead assessment, we used the well-known ns-3 simulator [316] which has a built-in implementation of IEEE 802.11s mesh network standard.

Physical WiFi Settings		**HWMP Protocol**		**802.11s Settings**	
EnergyDetectionThreshold	-89 dBm	ArpMinInterval	40s	MaxBeaconLoss	20
CcaMode1Threshold	-62 dBm	HWMPnetDiameterTraversalTime	2s	MaxRetries	4
RxGain	1 dB	MaxTTL	70	MaxPacketFailure	5
TxGain	1 dB	PathTimeout	100s		
TxPowerLevels	1	RfFlag	false		
TxPowerEnd	18 dBm	UnicastPreqThreshold	10		
TxPowerStart	18 dBm	UnicastPreqThreshold	5		
RxNoiseFigure	7 dB	DoFlag	true		

Table 5.1: NS-3 802.11g and 802.11s Parameters

The underlying MAC protocol used was 802.11g. We created two different AMI grid topologies that consist of 81 and 196 smart meters with 802.11g and 802.11s simulation parameters as shown in Table 5.1. Even though the number of smart meters in our simulation setup is less than a real AMI setup, it still represents a practical setup in terms of the number of hops due to limited transmission range of 802.11g which leads to multiple hops to reach a smart meter from the gateway (e.g., for 81 nodes the average hop count is 6 and for 196 setup average hop count is 9). In a typical AMI setup in the wild, utilities are able to use 900MHz frequency bands [Tin] which helps to reach thousand of smart meters through a few hops due to the extended transmission range. Unfortunately, ns-3 does not support those frequencies to build a mesh network, and thus we created a simulation environment which reflects similar number of hops as in the wild.

Although ns-3 provides very good simulation environment in terms of signal propagation, it still lacks to reflect the effects of real conditions on the signal such as path attenuation, refraction and diffraction while it propagates in wild. To see the effects of such conditions, we also built an IEEE 802.11s-based mesh network comprised of 18 Protronix Wi-Fi dongles attached to Raspberry-PIs which are integrated with the in-house meters as shown in Fig. 4a. We carefully dispersed the meters on the floor as shown in Fig. 4b and build the shown multi-hop routing structure among meters by limiting transmission range by decreasing Tx-Power up to by



(a) Smart meter

(b) Testbed topology

Figure 5.4: AMI Testbed

a factor of 16 [Tou97]. By such positioning and decreased Tx-Power, we strive to mimic realistic conditions on signal propagation and its effects on multi-hop routing in a real AMI setup.

5.5.2 Communication Overhead

Distribution Overhead

In this subsection, we report on the completion time for the non-revoked proofs distribution of our approach with respect to other baselines both in simulation and testbed environments. The results which are shown in Fig. 5.5 indicate the accumulator approach significantly reduces the completion time compared to local CRL and bloom filter approaches due to condense accumulating. Even with respect to Bloom filter, which is touted as one of the most efficient methods in the literature, our approach reduced the completion time in approximately more than 10 orders of magnitude.

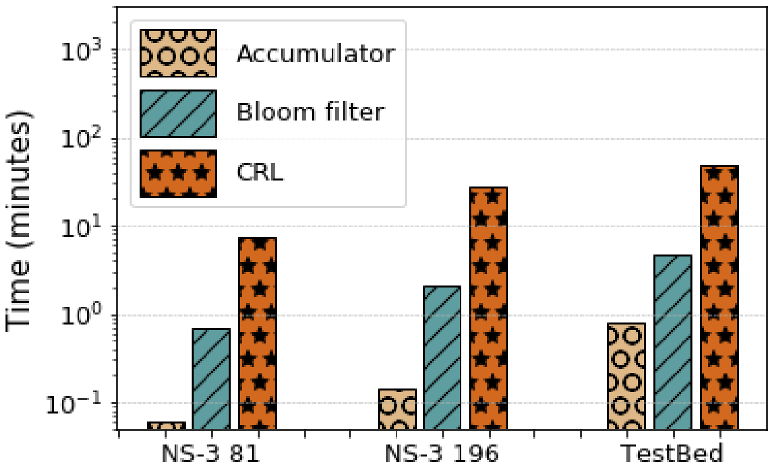


Figure 5.5: CRL distribution overhead

Another critical observation from the simulation results is the scalability capabilities of our approach. While especially for the local CRL approach, the completion time increases significantly, this is not the case for our approach. This can be attributed to the fact that the accumulator value is independent of the revoked CRL size while the overhead of other methods is proportional to the CRL size. The main overhead of our approach is directly related to the accumulator setting which was 2048 bits in our case. Therefore, even for very large-scale deployments that can have millions of meters, the overhead will not be impacted. In analyzing the experiments results for the testbed, we observe that the completion time takes more time even though the network size is much smaller. This is mainly because of the signal propagation issues such as path attenuation, refraction, interference from other devices, etc. within the building which does not exist in ns-3 simulations. Such issues cause more errors and packet loss and thus increase the re-transmissions to complete all packet distributions. In fact, the AMI infrastructure might have a similar challenge depending on the geographical location (e.g., urban vs rural environments) and thus the distribution of CRL will become even more critical. Therefore, our approach will be more suitable for such environments to reduce the impact from the wild, since condensing the revocation information will be extremely critical considering the related communication overhead.

Update Overhead

In this subsection, we conducted experiments to assess the overhead of CRL updates assuming that such updates are done regularly using the *delta CRL* concept. Fig. 5.6 shows revocation update overhead in terms of the completion time. As in the case of full CRL, our approach significantly outperforms others due to of the size of the delta CRL. However, the results for the Bloom filter approach shows a different trend this time. It performs worse than the local CRL approach. This can be explained as follows: For each updated revocation information, the Bloom filters must be created from scratch to carry both previous and newly revoked certificates. As a result, updating the CRL will take slightly more time than the whole CRL distribution for Bloom filter and thus will take more time than the local CRL approach. Note that

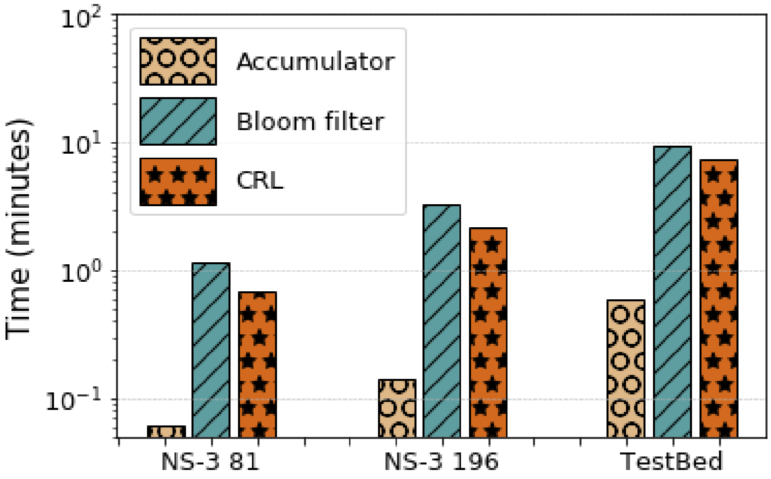


Figure 5.6: CRL update overhead

the overhead of CRL distribution is proportional to the size of the delta CRL and thus the completion time follows a similar trend with the results in Fig. 5.5.

For the testbed results, we observe a similar which consistent with the simulations. Again, the completion time is more due to signal propagation and interference issues.

5.5.3 Computation Overhead

We have demonstrated in the previous subsection that our approach significantly reduces the communication overhead. But, we need to also assess whether such a reduction introduces any major computational overhead. Thus, in this subsection, we investigated a detailed computational overhead of our approach. Specifically, we conducted two types of experiments: 1) We assessed the overhead of the computations due to the accumulation process in the Accumulator Manager. These experiments were conducted on a computer which has 64-bit 2.2GHz CPU with 10 hardware cores, and 32 GB of RAM assuming that these are reasonable assumptions for the computer that will act as the Accumulator Manager. Moreover, we also investigated whether some of these computations can be parallelized to reduce the computation times through multi-thread implementations further; and 2) We assessed the computation time for the $RevocationCheck()$ function in meters by implementing it in a Raspberry Pi (smart meter).

Overhead Results for the Accumulator Manager

In this subsection, we present and discuss the overhead at the Accumulator Manager by considering the functions below:

Computing Prime Representatives: To assess the computational overhead of prime representative generation, we computed prime representatives for different set sizes. Note that since both the valid and revoked certificates' serial numbers are used in our approach, the input size can become huge when AMI scales. Therefore, we also conducted a benchmark test by using threads to show the parallelization ability of our approach. The results are shown in Fig. 5.7. As can be seen, the computational complexity of the prime representative generation is not overwhelming. 10^5 representatives can be computed nearly in 1 minute even using a single core. Parallelization reduces the computational complexity by roughly 10 folds which allows computational times in the order of seconds.

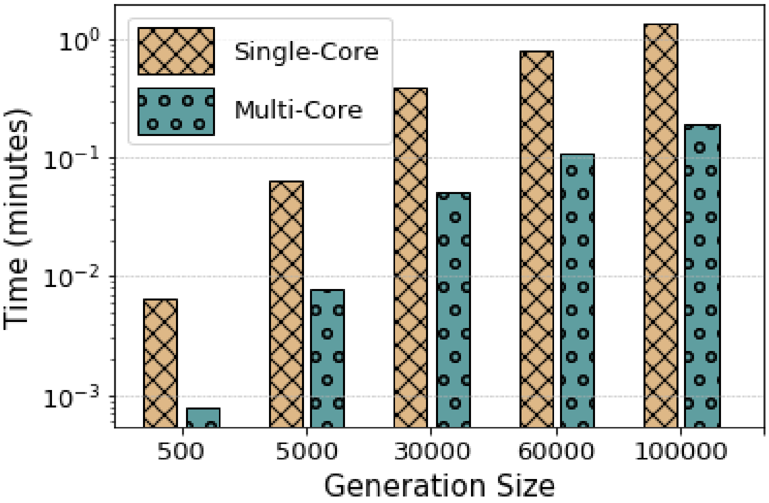


Figure 5.7: Prime representative computation

Computing the Accumulator Value: Next, we benchmark the computation cost of accumulator value according to different CRL sizes as used in the previous experiment. In addition, we also conducted tests to assess the computational difference between our setting (i.e., the Accumulator Manager has all aux_{info} information) and the case where the Accumulator Manager does not have aux_{info} as discussed in Section IV.C. Note that for the computation of the accumulator value, a parallel

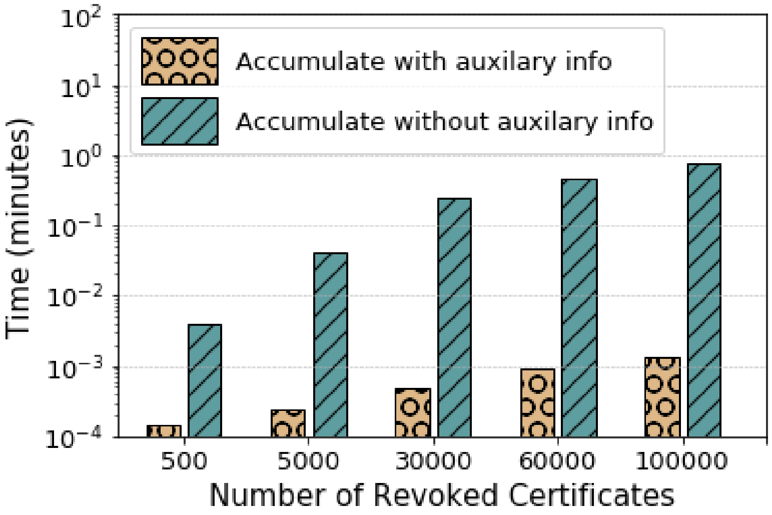


Figure 5.8: Accumulator computation

implementation was not possible since each step in the computation depends on the previous operation. As seen in Fig. 5.8, the accumulator value is calculated under a minute for 10^5 revoked certificates even without using aux_{info}. However, the availability of aux_{info} significantly reduces the computation time making it possible to finish it milliseconds regardless of the size of the CRL.

Computing Non-Revoked Values: Finally, we assessed the overhead of the computation of non-revoked proofs for both the first setup phase by using *full CRL* and the update phase by using *delta CRL*. Again, we conducted tests based on the availability/lack of aux_{info} and parallelization ability.Fig. 5.9 shows the computation overhead of this function according to different AMI sizes. As seen, aux_{info} makes a significant difference in this case. Even with parallelization, the computational times are still in the order of days which may not be acceptable in an AMI setting. The results indicate that aux_{info} needs to be available for efficient computations. We repeated the same experiment for the $UpdateNonRevokedProof()$ function and observed the same trends since the only change was the size of the CRL (i.e., delta CRL is much smaller). These results were not shown due to space constraints.

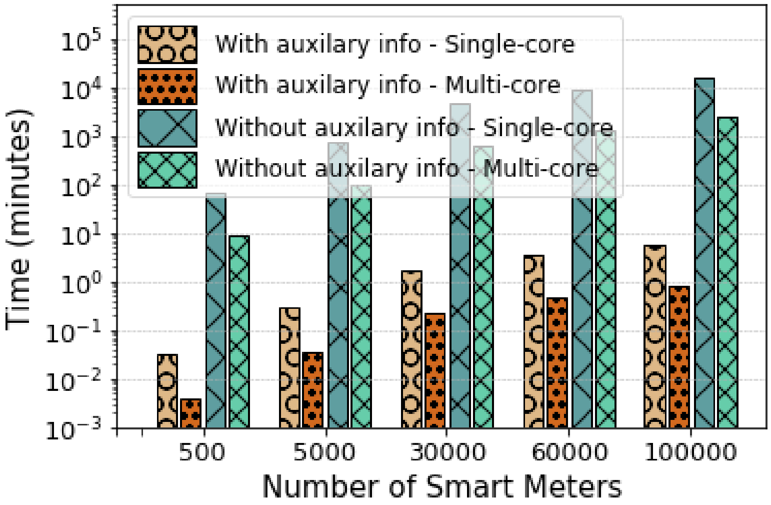


Figure 5.9: nr_{proof} computation for *full CRL*

Overhead Results for Smart Meter

Revocation Check Overhead: We looked at the computational time of revocation check operations in smart meter based on the three approaches compared. This is an important experiment to understand the computation overhead of our approach on the smart meter, considering the fact that it has limited resources. As can be seen in Table 5.2, the elapsed time for a single revocation check is around 10 milliseconds in our approach. Comparing with the other methods, the Bloom Filter has the best results as expected because it enables faster checking by efficient hash operations. However, Bloom filter suffers from false-positives which degrades its efficiency by requiring access to the server [ARMT14]. Our approach does not have such a problem. While our approach doubles the revocation check time compared to the local CRL method, the time is still pretty fast as it is in the order of milliseconds which does not impact any other operation. This is a negligible overhead given that it brings a considerable space-saving benefit which affects both distribution and storage overhead.

Table 5.2: Elapsed Revocation Check Time

	Local CRL	**Bloom Filter**	**Accumulator Approach**
Average Time (ms)	4.1	0.06	9.8

Storage Overhead: To compare the storage requirements, we identified the needed revocation information size for our approach and compared it with the other approaches, as shown in Table 5.3. As expected, accumulator has a superior advantage since smart meters just need to store a small accumulator value and non-revoked proof value. Local CRL, on the other hand, keeps the whole CRL list and depending on the number of revoked certificates, it can be huge. For our scenario, the CRL size is around 0.7MB for 30K revoked certificates. While Bloom filter's performance is also promising, it is still not better than our approach and it suffers from false positives as discussed.

Table 5.3: CRL Storage Overhead

	Local CRL	**Bloom Filter**	**Accumulator Approach**
Required Space (MB)	0.690	0.046	0.001

5.6 Security Analysis

The security of the AMI depends on the secure implementation of our approach. Therefore, we considered the threats in Section 5.2 against the security of the proposed approach and identified the relevant security goals.

❶ **Compromised smart meter attack:** An adversary can accomplish this attack in two different ways. For the first one, the adversary compromises the smart meter, and then the smart meter may be used to perform various attacks to Smart Grid. The UC is responsible for detecting malicious activity by utilizing different tools and sources. After detection, the UC puts the serial number of smart meter's certificate to the accumulation process. The updated nr_{proof} and $\mathcal{A}$ values are distributed to the other smart meters, HES and gateways. This process basically detaches the compromised smart meter from the AMI. Every other non-compromised components which may interact with this smart meter are no longer be able to interact due to the our revocation check protocol. Once the information of the compromised smart meter is accumulated to obtain a new accumulator value, the attacker can not successfully bypass the revocation check mechanism by using the compromised smart meter itself or its stolen private key and certificate in the future.

For the second one, the attacker can abuse a vulnerability in the certificate issuing process or steak smart meters' private keys from manufacturers. This time, the CA revokes certificates of the corresponding devices and publish new CRLs. Those CRLs collected by our CRL collectors. These newly revoked certificates are then accumulated, and the corresponding non-revocation proofs are disseminated to AMI. With the updated accumulator values, an adversary can not interact with any of the smart meters, HES, or gateways within AMI by using the revoked certificates.

❷ **Compromising the UC Servers:** In the event of an attack, the adversary's first target will be to compromise the accumulator manager to attack our revocation management. In our scheme, the accumulator manager plays a critical role but can be missed out easily because it is located within UC perimeters. However, the accumulator manager is the Achilles' heel of our approach and should be protected thoroughly. Thus, we investigate three possible attack scenarios and corresponding countermeasures within our revocation system.

a) First, through the architectural design in Figure 5.2 *accumulator manager* is protected from any attacks by not allowing direct communication from outside of the network through two different firewalls. For instance, the second firewall configuration just allows incoming traffic, which is directly started by the Accumulator Manager to collect CRLs. Thus, it is not easy to access to accumulator manager from outside of the perimeter.

b) Considering the ever-increasing threat environment and improved skills of adversaries, no matter what level of protection our system has, the accumulator manager can get compromised by breaking a path the proposed defense layers. In such a case, the key factor will be how quickly our approach responds to the incident. After detection of the compromise (e.g., attacker steals RSA setting parameters such as aux_info and $p\&q$), the accumulator manager can be migrated to another server, and new nr_{proof} and $\mathcal{A}$ is computed from scratch by using different RSA primes p' and q' within minutes as shown in Section 5.5.3. Then, updated proofs are distributed to smart meters to prevent any further damage. So, our approach offers a pretty easy recovery capability which is important considering critical operations in AMI.

c) Third, our scheme is also allowing computation of nr_{proof} without keeping critical security parameters of RSA accumulator settings RSA (i.e., aux_{info} and $p\&q$), since stolen aux_{info} and $p\&q$ values enable a malicious actor to prove any arbitrary statements. These parameters can be deleted once they are used in the setup phase. In such a case, compromising the accumulator manager does not give any advantage to the adversary to attack revocation management by abusing these parameters. Moreover, the computation of nr_{proof} can still be accomplished for new smart meters or/and in case of updated revocation information, but it is more computationally intensive as shown in the Experiments Section.

❸ **Compromising the Communication:** As stated before, the traffic of AMI is generally unencrypted and causes additional attack surface to our approach. In this subsection, we investigate two possible attack scenarios and countermeasures against them as follows:

a) **Accumulator freshness attack:** An eavesdropping attack of AMI traffic poses a unique threat to our approach by combining public revocation information and circulating accumulator values. An attacker may perform a targeted attack if the UC has not updated the accumulator value properly by pinpointing the smart meters that use old accumulator values. However, our approach is robust to this attack since while computing the accumulator value, we use a secret prime number r_k as a first exponent (g^{r_k}) in Eq. 5.2. This prevents inferring the freshness of accumulator value by combining publicly known revocation information and circulated values in unencrypted traffic.

b) **Stolen non-witness attack:** One possible attack can be performed by using $nw_{1\&2}$ values to masquerade a valid smart meter since it can be obtained easily by eavesdropping. Our protocol is protected from this threat, even if the corresponding non-witness values $nw_{1\&2}$ of a smart meter are tried to be abused by attacker, the authentication during revocation check will still fail due to the multi-level signature checks. Thus, we relieve the attacks by abusing of stolen $nw_{1\&2}$ values through a multi-level authentication which combines the signature check with the accumulator check.

5.7 Benefits and Limitations

There are several benefits associated with the use of the proposed approach for revocation management in AMI. Also, we highlighted some challenges and limitations of the approach.

5.7.1 Benefits

1. *Low overhead*: Our approach imposes minimal to no overhead to the smart meters deployed in the AMI and very low overhead to the central servers supporting the revocation management. In general, a revocation check contains at most one additional modular arithmetic operations if compared with the other revocation check methods (considering investigated methods realizes at least one modular arithmetic operation for signature check). Also, the overhead imposed by disseminating the revocation information to the smart meters is very low.

2. *Applicability and Security*: The steps used in our approach can be easily implemented to the current AMI infrastructure with few adjustments. We showed that which components of the current AMI setup will be affected and need to be updated with new functionality. To compare the applicability of our work with its alternatives, we determined four key benefits in total as shown in Table 3. Our approach collects the revoked certificates information without in-

Table 5.4: High-level Comparison of Revocation Management Schemes.

	Applicability	Storage Overhead Advantage	Communication Overhead Advantage	Security
OCSP	○	●	○	●
OCSP-Staple	○	●	○	●
CRL	◐	○	○	●
Delta CRL	◐	●	◐	●
Bloom Filter	◐	●	◐	●
Our Approach	●	●	●	●

● = offers the benefit; ◐ = almost offers the benefit; ○ = does not offer the benefit.

terrupting the current smart grid operational network setup. However, unlike OCSP or OCSP-stapled methods, it requires extra communication overhead to distribute revocation information. Still, AMI communication infrastructure is not natural to an off-the-shelf OCSP-based solution due to the frequent query requirement, so obviously, it does not carry any advantage for decreasing the communication overhead. On the other hand, our solution outperforms all other methods in terms of introduced distribution overhead. In brief, our conclusion from this comparative evaluation shows that our approach offers the same security benefits as other notable methods while keeping the overhead at the minimum level.

3. *A General Revocation Framework for Smart Grid*: Smart Grid is equipped with a myriad of various smart devices and sensors. This represents a new domain for security that is far beyond the traditional air-gapped operational network technology (OT) needs because of investments in distribution technologies such as renewable energy sources like rooftop solars and wind turbines. In this context, our approach emerges as the first comprehensive solution that adapts the cryptographic accumulators to instrument lightweight revocation management and can be applied different domains in smart grid beyond AMI.

5.7.2 Limitations

1. *Tight Synchronization Requirement*: Cryptographic accumulators are powerful tools for short set representation and secure non-membership proofs. However, a disadvantage of using an accumulator-based revocation scheme is that the non-revoked proof and accumulator value has to be synchronized between smart meters. This might occur in two ways. First, the accumulator value at the verifier's site is out-of-date, but the non-revoked proof of the prover is updated and vice versa. Asynchronous non-revoked proofs and accumulators between communicating smart meters may hinder the authentication operations; thus, AMI should ensure that all smart meters are updated and start to use the new proofs at the same time. Although the requirement for strict synchronization seems prohibitive, the AMI is a well-managed and synchronized network. Because of this characteristic of AMI, the synchronization requirement can be met easily.
2. *Not-allow to use Unreliable Distribution Methods*: Another limitation related to synchronization requirement is that an attacker may selectively drop the packets to cause a synchronization problem between smart meters. Thus, any unreliable method for the distribution of non-revoked proofs should be avoided and the UC should ensure that required values are completely reached to smart meters.

5.8 Conclusion

Considering the overhead of certificate and CRL management in AMI networks, in this chapter, we proposed a one-way cryptographic accumulator based approach for maintaining and distributing the revocation information. The framework condenses the CRLs into a short accumulator value and builds a *secure*, efficient and lightweight revocation mechanism in terms of communication overhead. The approach is inspired by cryptographic accumulators and adopted based on the requirements of AMI. The experiment results indicate that the proposed approach can reduce the distribution completion time significantly for compared to CRL and Bloom filter approaches while introducing only minor additional computational overhead which is handled by the UC. There is no overhead imposed to smart meters.

CHAPTER 6

A LIGHTWEIGHT SYMMETRIC KEY-MANAGEMENT SCHEME FOR IOT AND SMART GRID INTEGRATION

In this chapter, we are particularly interested in reducing the overhead of symmetric key management over legacy systems. Since reducing the latency of the key exchange is a particular interest for IoT integration, in this chapter, we tackle the potential overhead of key-management by proposing a secure and communication efficient key exchange protocol. We employ the proposed protocol to renew symmetric keys for applications that use the grid's legacy infrastructure. Thus, we propose a novel 0-RTT authentication and key agreement scheme which utilizes the *dynamic key-generation schemes* [NWL$^+$10] to achieve replay-attack resistance. Dynamic key generation is an authentication system based on *hash chains* which is specifically used for dumb terminals where one does not want to use a long-term key in the authentication. We integrate this hash-chain concept with the widely used Diffie-Hellman (DH) Key Exchange scheme [DH76]. The proposed scheme allows IoT devices to achieve lightweight mutual authentication and key exchange which means that field device can securely update the shared key in 0-RTT overhead. Since our protocol provides a key update at each data collection round, the scheme has an improved forward secrecy to enhance the security of power grid against the vulnerabilities related to state estimation.

While reducing RTT overhead of key exchange brings many advantages considering the limited bandwidth of communication links in smart grid. We further proposed an UDP adaptation for our 0-RTT protocol to reduce the over-the-wire overhead even more with the help of UDP mechanism. This adaptation provides a secure yet lightweight 0-RTT key-exchange mechanism by getting rid of related both TCP-header and 3-way handshake burden. Finally, through the use of the hash chain's elements for unique state representation at each device, our scheme is also resistant to replay attacks which is not available in the current schemes.

To assess the delay overhead of the proposed approach, we built a realistic simulation infrastructure for data collection from field devices using the LoRa communication standard that is available for wireless wide area communications [AVTP$^+$17]. We used ns-3 simulator [316] to model LoRa characteristics as a low-bandwidth communication infrastructure (i.e., in the order of kbits). The results indicated that compared to existing conventional schemes, the proposed approach provides significantly lower communication overhead while ensuring a secure and efficient key exchange.

The rest of the chapter is organized as follows: Section 6.1 provides the background on contemporary key-exchange mechanisms. Section 6.2 introduces the thread Mode. We present the proposed scheme in Section 6.3 and its UDP adaptation in Section 6.4. We offer security analysis of the protocol in Section 6.5. Section 6.6 is dedicated to experimental validation. The Chapter is concluded in Section 6.7.

6.1 Background on Key-Exchange Mechanisms

The first generation of well-known key exchange mechanisms did not consider much about the efficiency or overhead, since secure connections were considered to be the exception rather than the common. For instance, the older IPSec IKEv1 needs up to 4.5 RTT [Hof05] whereas the improved version of it (IKEv2) requires 2 RTT [KHN$^+$14].

Internet Engineering Task Force (IETF) has recently approved TLS 1.3 published as RFC 8446 in August 2018. The new standard decreases the initial key establishment between a client and a server even further to 1-RTT [RD18]. This newly introduced key exchange mechanism is one of the most significant changes in TLS 1.3. The message flows in Fig. 6.1 represent a mutual authentication handshake in PKI settings. In brief, a client first sends *Client Hello* which includes the cryptographic parameters and nonce. In addition, since TLS 1.3 utilizes DH Key exchange concept, it sends a freshly generated DH *Key Share*. The server responds with a *Server Hello* message which includes its choice of cryptographic parameters and a server nonce. The server also sends its own freshly generated DH key share *Key Share* and related *Extensions*. The server includes its *Certificate* and a signature on all messages *CertificateVerify* for authentication purposes. The server's *Finished message* is a Message Authentication Code (MAC) over the entire handshake to provide the integrity using the created shared key. Finally, the client sends its own *Certificate* and a signature, *CertificateVerify*, for client authentication. As in the server's case, the *Finished* message is a MAC over the entire handshake. This message provides both the integrity and key confirmation. Note that, TLS also provides

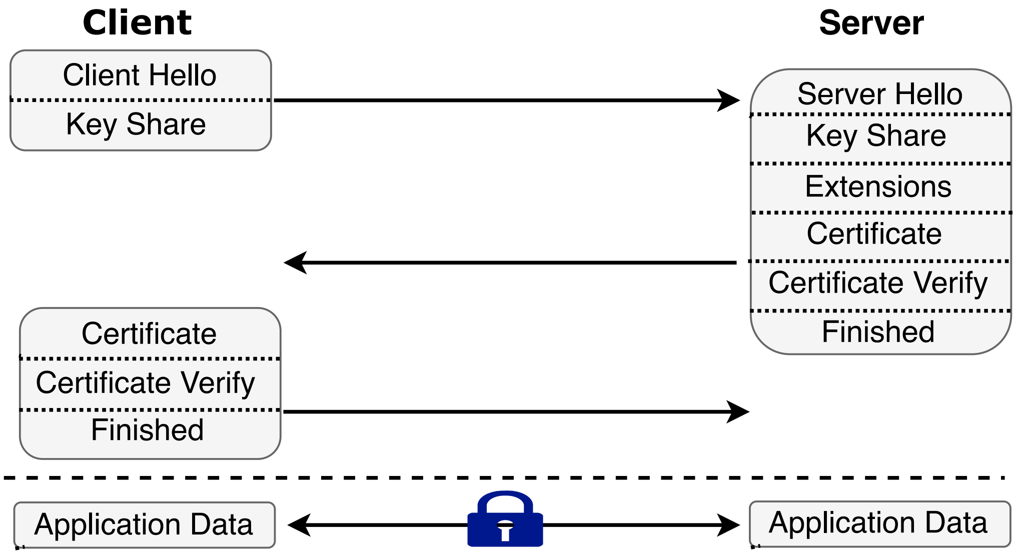


Figure 6.1: TLS 1.3 Key Exchange

a mechanism to accomplish 1-RTT key-exchange mechanism using pre-shared keys (PSK) instead of PKI.

On the other hand, for the security of datagram packets (i.e., UDP), the Data Transport Layer Security (DTLS) protocol is used with an integrated key-exchange mechanism. DTLS 1.3 is currently defined in draft RFC [RTM20] by explaining its differences from TLS 1.3. But, most of the TLS 1.3 features are reused with only modest variations. The main difference of DTLS 1.3 in number of exchanging messages is that DTLS 1.3 has an additional "HelloRetryRequest+cookie" message for replay attack protection. Since DTLS 1.3 runs over UDP, it is an easy subject for IP spoofing attack. To hinder this attack, the server sends the "HelloRetryRequest" message that contains a generated cookie as shown in Fig 6.2. Then, the client retransmits this cookie to ensure that it is able to receive the cookie and has a valid IP address.

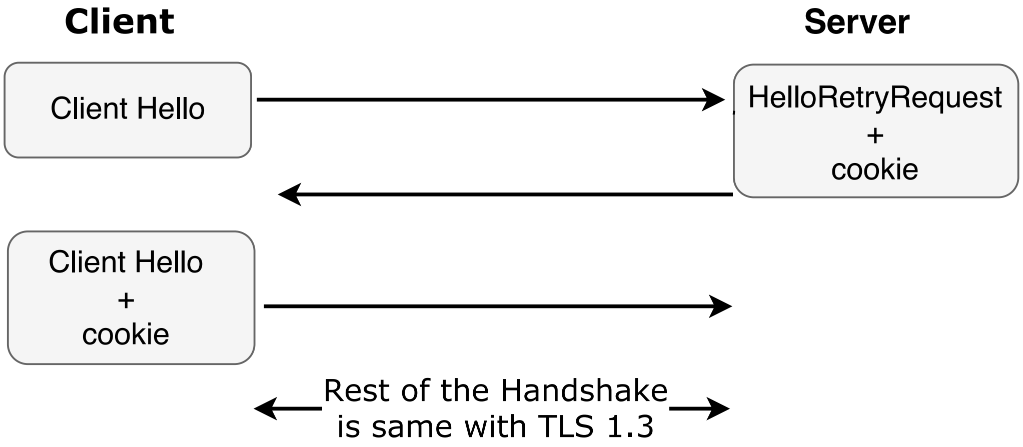

Client
Server
Client Hello
HelloRetryRequest + cookie
Client Hello + cookie
Rest of the Handshake is same with TLS 1.3

Figure 6.2: DTLS 1.3 Key-Exchange with "Cookie" mechanism

The defense mechanism in DTLS causes extra 1-RTT message exchange between client and server by exchanging a non-stateful cookie between them.

6.1.1 0-RTT Key Exchange

A further optimization to 1-RTT key-exchange scheme is the 0-RTT key-exchange. In a 0-RTT key exchange mechanism, clients can send encrypted data in their first message to the server, eliminating any additional latency due to performing the key-exchange. TLS 1.3 also describes a 0-RTT mechanism which is based on a pre-shared key (PSK) that is either obtained externally or via a previous handshake. Fig. 6.3 shows the 0-RTT mechanism described in TLS 1.3. Clients send encrypted data in their first message to the server as “early data” by encrypting it via the PSK. Although the scheme is called 0-RTT, it is actually a form of PSK resumption protocol. The main idea is that after a session is established, the client and server can derive a fresh secret by using the previous master secret. However, the first flight of data is still encrypted by the PSK rather than the newly computed fresh shared key. Compared to the 1-RTT handshake pattern, the first application data

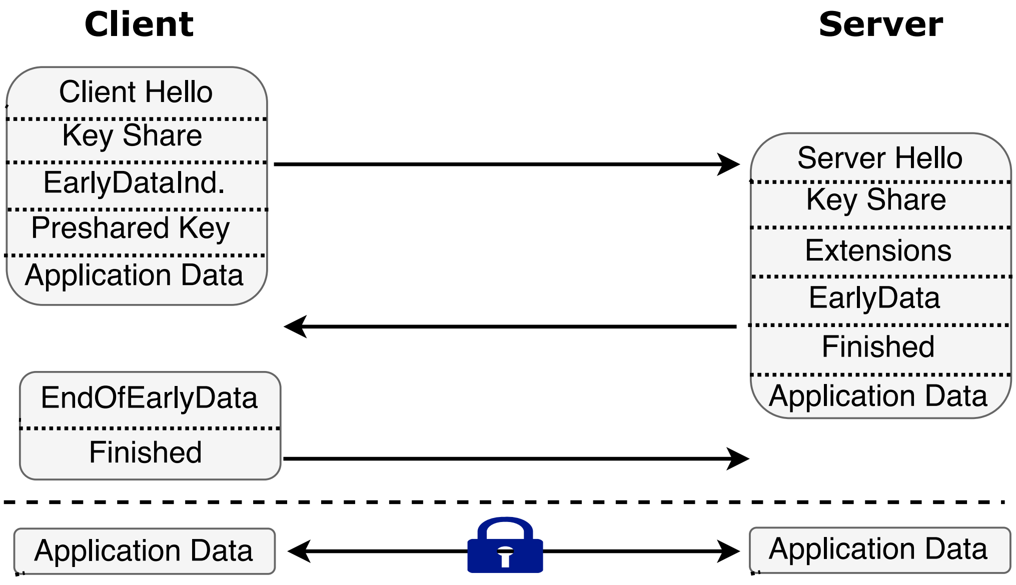


Figure 6.3: TLS 1.3 Key 0-RTT PSK Resumption

does not have forward secrecy if the server's long-term secret key is compromised. In addition, the first message from the client is also subject to replay attack where adversaries are able to update the server state by replaying the message. However, TLS 1.3 standard document concluded that this downside is of negligible importance compared to the benefits of the handshake pattern when it is used under only certain conditions.

6.2 Network and Threat Model

We assume a IoT based smart grid application where data are collected via a wide-area communication network and processed to make decisions (see Fig 6.4). The communication network is wireless that has a number of base-stations spanned through the geographical area served by the utility. The IoT devices are integrated with IEDs/RTU/DERs and communicates with the base-stations that eventually relay their data to SCADA control center using some sort of back-haul wireless or wired links. The wireless communication between base-stations and IoT devices is severely limited in terms of bandwidth. It can be in the order of kilo-bits which is in line with the used technology in 2.5G or other proprietary protocols.

For the threat model, we assume the following attacks are possible when keys are created and distributed to field devices:

- *Impersonation and Man in the Middle (MitM) Attack:* We consider the communication between two parties with the presence of a passive or active eavesdropper as the adversary. We assume that the adversary can impersonate one of the legitimate field devices or the CC. In addition, we assume that, an adversary in the middle decouples the end-to-end communication between the field device and CC by establishing independent connections with them. The

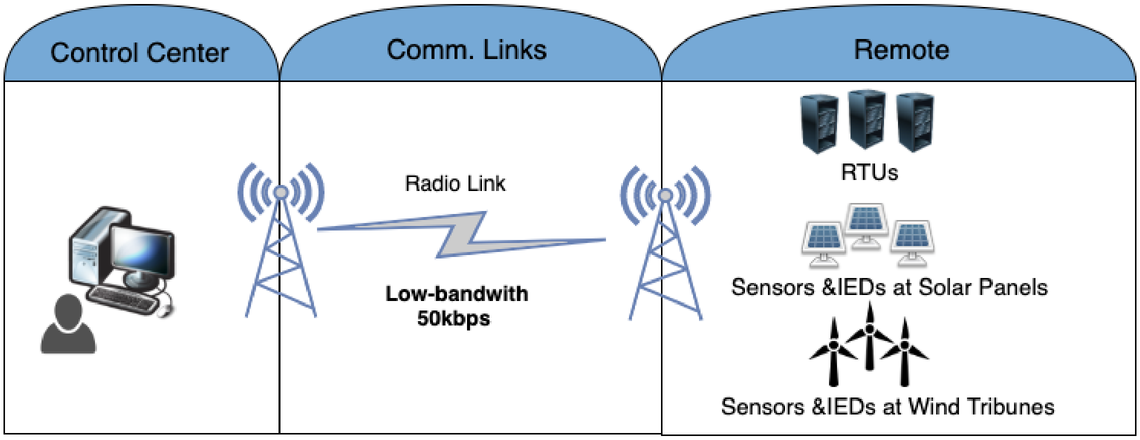


Figure 6.4: An overview of a sample legacy infrastructure.

attacker relays messages between them as if they are communicating directly with each other.

- *Ephemeral Shared Key Compromise:* We assume that either the DH Share or the shared key can be compromised by an adversary.
- *Replay attack:* We assume that an adversary can obtain valid key exchange messages from the traffic and store it. Then, it uses this message to fool either the CC or the field device into thinking that they have completed the key agreement.
- *Amplification Attack*: We assume that an adversary can send messages with spoofed IPs to accomplish DOS attack type known as the amplification attack [Ros14b].

6.3 Proposed Approach

The proposed scheme has three phases to achieve secure communication between the field devices and the CC: Setup & Configuration, Key agreement, and Key refreshment. In the first phase, when a field device joins the power grid, it completes registration with the CC. After a successful setup, the CC creates an authentication key for the field device. Then, they use this key to establish mutual authentication.

6.3.1 Setup and Configuration Phase

When a new field device joins to the power grid, it sends a DH_REQUEST message to the CC which also contains the ID of the device. Upon receiving the DH_REQUEST, the CC generates a secret key, $S_i = H(ID_i||Nonce)$ for that device. This step helps CC to identify the field device with the parameters ID_i and $Nonce$ where $H()$ is a one-way secure hash function.

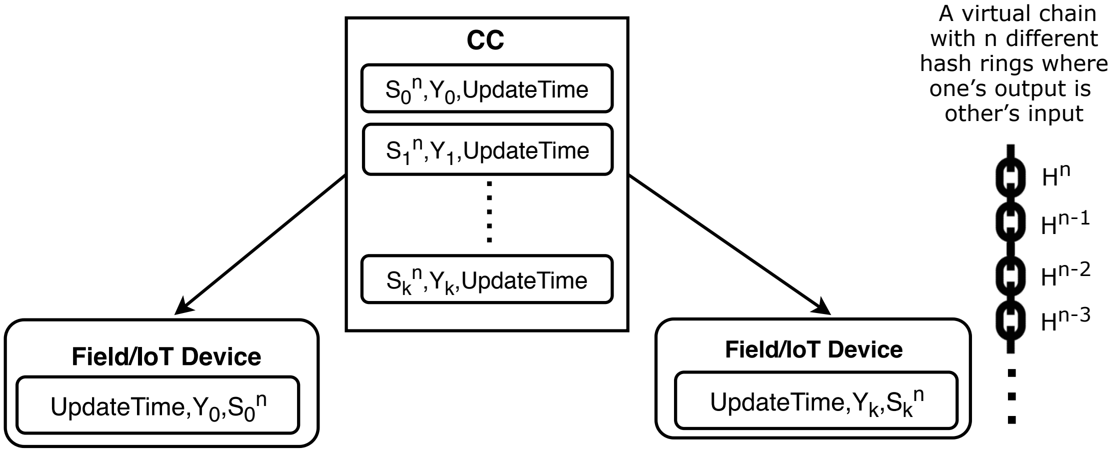


Figure 6.5: The distribution of one-time authentication keys

Afterwards, the CC computes $H^n(S_i) = S_i^n$ where n is a predefined parameter and $H^n(S_i)$ is obtained by taking n cryptographic hash of S_i: $H(H(H, ..., H(S_i))$. We note that the key part of this scheme depends on this repetitive hashing which creates a hash chain of n different one-time keys.

In the proposed scheme, the field devices will only store $H^n(S_i)$ while CC will store $H^{n-1}(S_i)$ and $H^n(S_i)$ at a given data collection period t. These keys are stored along with a deadline (e.g., $UpdateTime$ as shown in Fig. 6.5) that is used for updating the DH component value. The deadlines can be set irregularly up to several days. After this first setup, there will be no need for re-configuration until n is exhausted or until the deadlines are reached. The size of n can be adjusted to be sufficient enough for the DH period. When the deadline is reached, a new DH value must be assigned again for forward secrecy. After the creation of keys $H^n(S_i)$ for the field device, the CC sends to a device i the DH configuration which contains $< Y_i, UpdateTime, S_i{}^n) >$ where Y_i is the shared DH component with the field device and *updateTime* is the expiration time of DH component. Y_i is computed by g^c where g is the DH parameter and c is the random number picked by CC. Figure 6.5 shows this process. Y_i will be used by the field device to derive an ephemeral shared key over 0-RTT as will be explained next. The DH component will be updated when *UpdateTime* comes but this can be set in advance.

This phase of the proposed scheme is done securely right after the 1-RTT setup phase of TLS 1.3 (see Fig. 6.1 until the dashed lines). However, this is a one-time process and once it is completed, it will not be repeated until n or update time for DH values is reached.

6.3.2 0-RTT Key Agreement Phase

When a field device would like to send data to CC for the first time after setup & configuration phase, it firsts needs to compute a shared key. To this end, it picks up a random a and generates a shared secret key $k_1 = Y_0{}^a$. Using this shared key k_1, the field device can encrypt and authenticate data to be sent to the CC. It also computes a fresh ephemeral DH share $X_1 = g^a$ where g is either a prime number if RSA is used or elliptic curve parameter base if ECC is used. The data, X_1 and the stored S_i^n are sent to the CC. At the CC, the same shared key is computed

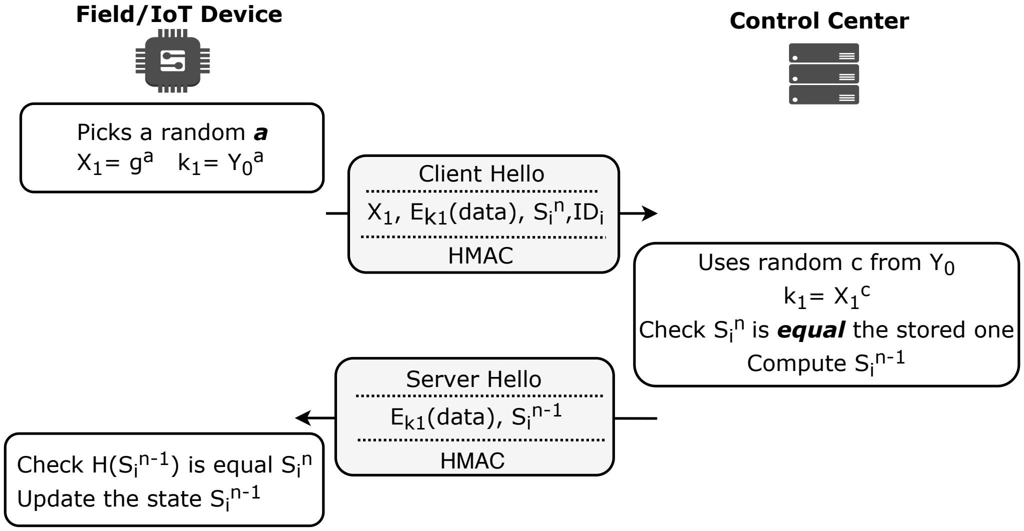


Figure 6.6: 0-RTT Key Agreement

from ${X_1}^c$. Note that, k_1 would be equal to ${X_1}^c$ which means $k_1 = g^{ac}$. The CC now ensures the freshness of the field device by checking whether the sent ${S_i}^n$ is equal to the stored one. This also ensures the authentication of the field device. Then the CC sends a fresh ${S_i}^{n-1}$ to the field device.

The field device computes the hash of $H({S_i}^{n-1})$ and checks whether it is equal to the previous ${S_i}^n$ to ensure the CC's freshness. If the result is equal to previous ${S_i}^n$, authentication of CC is done and the field device updates the old value with new ${S_i}^{n-1}$. This means it implicitly agrees on the key k_1 which is then used for all subsequent data exchanges within the same session. The entire process is shown in Fig. 6.6.

6.3.3 0-RTT Key Refreshing

Although the shared key in a previous connection can be used for "session resumption" in further connections by tying the new connection cryptographically to the previous connection without needing a handshake, we used DH key exchange in order to provide *forward secrecy* for the application data in combination with the hash chain that act as authenticators. This is one of the major novelties of our approach.

The key refreshing works as follows: As in the previous case, the field device picks up another random number, say, b and computes a fresh ephemeral DH share $X_2 = g^b$. Then, it generates another temporal shared secret key k_2 from Y^b. This temporal shared key k_2 is again used for both encryption and authentication. This time field device sends the stored ${S_i}^{n-1}$ for freshness to the CC.

At the CC, the same shared key k_2 is computed from ${X_2}^c$. The rest of the protocol uses the same computations and messages as for the previous handshake by computing ${S_i}^{n-2}$. The process is shown in Fig. 6.7. This phase can be repeated until re-configuration time when either n is exhausted or DH share expires.

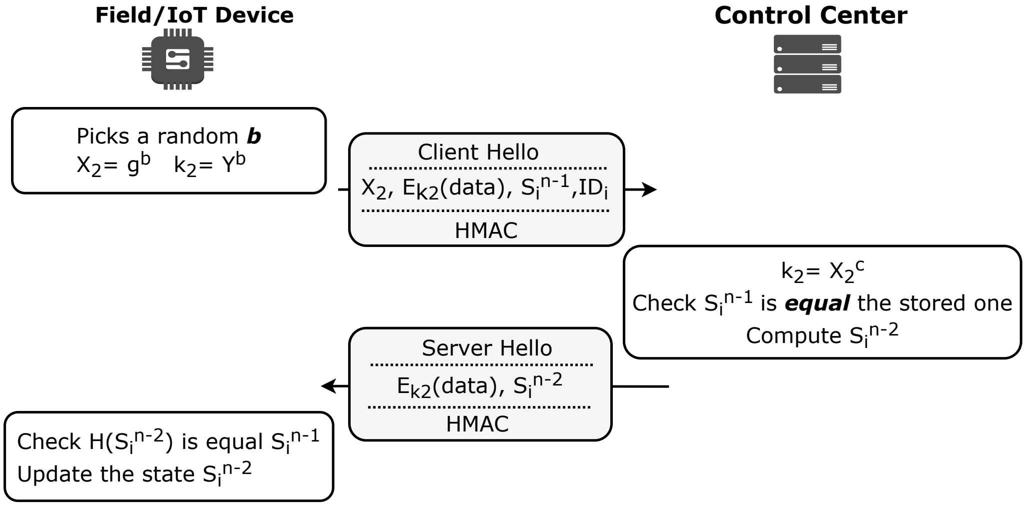


Figure 6.7: 0-RTT Key Refresh

6.4 UDP Adaptation

Our UDP adaption reuses all the protocol phases (e.g., Setup, 0-RTT Key Aggrement, 0-RTT Key Refreshing) of previous TCP based solution, with minor but essential adjustments for it to operate correctly with datagram transport. The previous one's security features, i.e., replay attack protection, depend on a subset of TCP characteristics such as reliable, in-order packet delivery, etc. But, UDP does not have these features. In this section, we describe the proposed UDP based key-exchange protocol and how it copes with the absence of these characteristics.

6.4.1 Sliding Window for Packet Re-ordering and Fast Anti-replay Checking

Our TCP based protocol implicitly employs TCP header sequence numbers for handling lost/redelivered messages. We require similar functionality in our UDP adaption, but at this time, we have to define it explicitly in headers (Client & Server Hello) since packets can drop or reach out of order. Unlike TCP, our sequence number is incremented by 1 for and is reset to 0 whenever the state is reset. Considering the characteristics of the smart grid, we do not need a sequence number which can carry gigabytes of streaming data as in TCP (e.g., via 2 different 32 bits field). The back-and-forth messages between field devices and Master are at most in tens of bytes. Thus, one or a few datagrams can carry the whole message between them. As a result, the sequence numbers (#Sequence field in Fig. 6.8) are set as 8 bits to reduce the associated overhead. Note that our sequence number does not represent the number of bytes unlike TCP; instead, it represents the number of datagram packets. Morever, we used the most significant bit (MSB) as representing the last datagram. Thus, there can be 128 different datagrams which are used to send at most 128∗MTU(Maximum Transport Unit) size application data. Moreover, the datagram contains one byte re-transmission field to track the re-transmitted datagrams which is explained in the next subsection.

Algorithm 2: Receive Data with Sliding Window

```
; // Input: datagram, Context with timeout, slots&window, Output: Decrypted payload
Function ProcessDatagram(datagram,cntxList,window,slots,plainData):
    header, port, ip = extractInfo(datagram);
    cntxt=null;
    if header.LI<header.seqNo or header.seqNo>window.HI then
        return -1; // Out-of-window-boundaries;
    if window[header.seqNo]=0 then
        if header.seqNo=0 then // The first datagram
            // create context and extract keys;
            id, S = extractIDandState(header);
            if !isCorrectState(S) then
                return -2; // '$S_i$'& #Retrans is not the expected one;
            dh_X, sharedKey = extractDHShareandKey(header,id);
            if sharedKey=-1 then
                return -3; // "Key Error";
            payload,HMAC = extractPayload(header,datagram);
            if chckHMAC(payload,HMAC,sharedKey)=-1 then
                return -4; // "HMAC Error";
            cntxt = getAndAddContext(port,ip,cntxList,sharedKey); // AddCntxt
            slots[header.seqNo]=decrypt(datagram,sharedKey);
            window[header.seqNo]=1;
            if header.seqNo=window.LI then
                slideWindow(window) // Sliding Window;
            if header.MSB==1 then // the last datagram
                window.HI=header.seqNo;
        else // Already has context, extract context first
            cntxt,sharedKey = getContext(port,ip,cntxList);
            payload,HMAC = extractPayload(header,datagram);
            if chckHMAC(payload,MAC,sharedKey)=-1 then
                return -4; // "HMAC Error";
            slots[header.seqNo]=decrypt(datagram,sharedKey);
            window[header.seqNo]=1;
            if header.MSB==1 then // the last
                window.HI=header.seqNo;
    if window.LI=window.HI then // Check window is full
        plainData = constructData(slots);
    return 1;
```

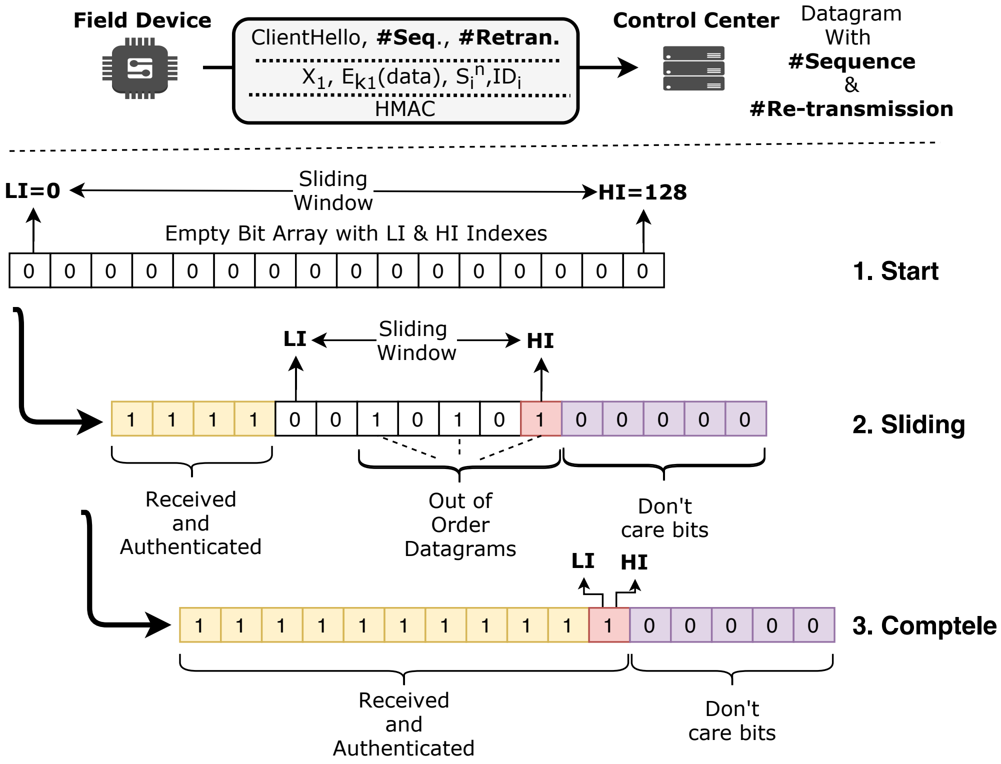


Figure 6.8: Sliding Window Mechanism for Out-of-Order Datagrams

We employ a simplified form of window mechanism inspired from IPSEC [KA] to detect out of order delivery packets. We implemented a simple bit-array window where the set bits represent received datagrams. To track datagrams, the party prepares an array of 128 bits (due to at most 2^7 number of sequences) and check the sequence number of datagram against the bitarray as shown in Fig. 6.8. As a complementary mechanism to the bitarray, we define a sliding window protocol that provides both packet re-ordering and the quick anti-replay mechanism, while the server is managing the bit array. The value of each bit shows whether or not datagram with that order has been received and verified. Moreover, the protocol maintains two index values for the low and high-end index (LI, HI) of the window to reduce the time for checking the datagram validity. If the server receives a datagram with the sequence number t and t is inside the window ($LI \leq t \leq HI$), then checks the corresponding bit to detect if this datagram has already received. If the bit has already been set or not within the window, it simply discards that datagram. The window slides when t is equal to LI, or the most significant bit (MSB) of t is set. In the first case the LI becomes $LI = LI + 1$, in the second case the HI is updated as $HI = t$. This mechanism limits the number of expected out of order datagrams and helps the receiver discarding the out-of-window datagrams without checking it so that a replay datagram can be identified and discarded easily. When the LI is equal to HI, the receiver just understand that it gets all required datagrams without a gap in the window, then it reconstruct the message. The algorithm 2 shows that how sliding window mechanism is integrated along with verification of datagrams and extracting key operations.

6.4.2 Reliable Delivery Adaptation

Because UDP packets may be lost, our protocol needs a mechanism for retransmission. We implemented retransmission mechanism using a single timer at the initiator side (i.e., client). The client's side keeps retransmitting its current message until a response is obtained. The state diagram of timer that implements resulting retransmissions is shown in Fig. 6.9.

Timer for Reliability Management

Once a client has data to send, it moves to the transmission state, a timer is triggered and starts ticking (State1 & State2 in Fig. 6.9). When there is no more diagrams to send, client makes a transition to the "Wait ServerHello" state (State3) to receive the "ServerHello" response from the server. If the client receives a datagram that is the expected (i.e., within the window) and validated then it checks whether LI and

HI equals to reconstruct the whole message. If it is equal, it makes transition to "Finish" state (State 5) and the transmission mechanism is canceled and the current state of the client is halted. Otherwise, the timer continues ticking. When the timer expires (State4), the client restarts the process (State0) and begins transmitting the same datagrams again.

In the server-side (i.e, master), we did not implement a timer mechanism sonce its process is highly coupled with the client. The server is always in waiting state and accepts datagrams from the client (State 1). If datagram is the expected (i.e., within the window) and authenticated, then server checks whether or not the message is complete (State 2). If so, it sends the "ServerHello" to the client (State3). If the server continues to receive message after sending "ServerHello", it implicitly means that the "ServerHello" message did not reach the client-side. Thus, the server re-transmits "ServerHello" message again for each repeated received message. But the critical part of this design is distinguishing the replayed datagrams from an attacker and re-transmitted datagrams from the client. Otherwise, the server continues to send "Server Hello" to the client in case an attacker replays the datagrams. The retransmission number field in Fig. 6.8 solves this problem and helps to identify whether a datagram has been replayed or re-transmitted. If the client does not get "ServerHello" from server, it simply increase the re-tranmission field and restart the transmission again. In this way, the server distinguishes replayed datagrams and the ones re-transmitted since the re-transmitted ones have a incremented value in their re-transmission field. So, the server will just ignore those replayed datagrams and does not send "Server Hello" although the datagrams are valid and authenticated. On the other hand, if the server receives new datagram from the client with updated key materials since our protocol updates key materials for each session (see subsection 6.3.3), the previously used datagrams for replay attack become useless because the server has now a different S_i in its state tracking. Thus, it simply ignores those replayed datagrams. (See line 9 in Algorithm 2 for corresponding check mechanism.)

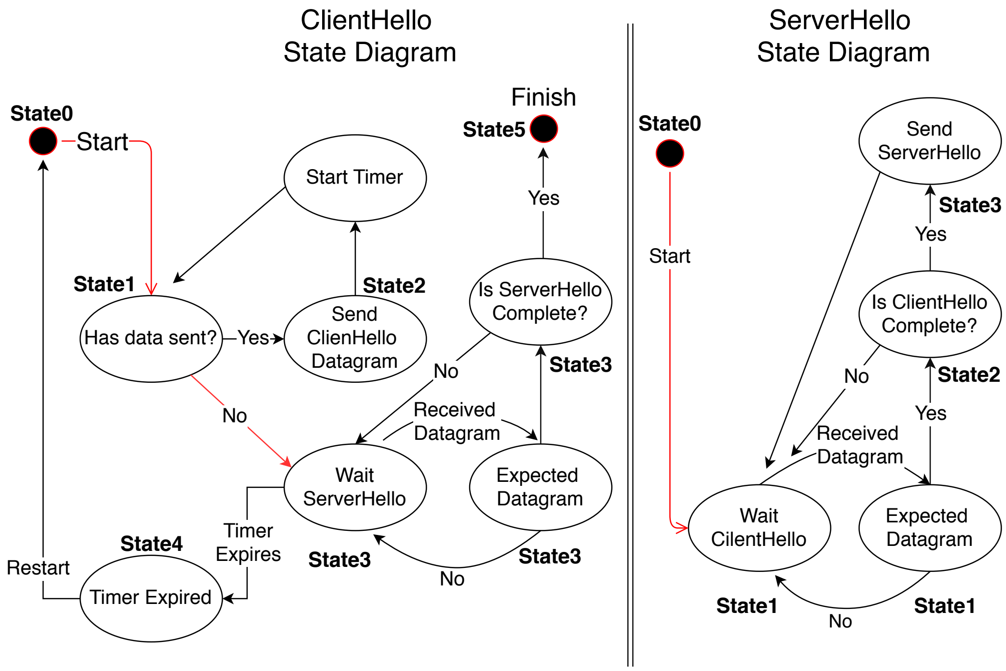


Figure 6.9: State diagram of ClientHello & ServerHello Messages

RTT Computation and Adjusting Timer

Considering the communication characteristic of a smart grid, picking appropriate expiration time for retransmission is a difficult problem due to the lossy nature of the communication and the high variance in round trip times (RTT). While measuring RTT would provide an idea about timer value, requiring an estimate from our protocol is an unnecessary burden. Since deciding on the exact expiration time is very tricky, we chose a conservative 2 seconds expiration time to avoid unnecessary retransmissions and increase the expiration value with a simple exponential back-off mechanism (2^n) with 2 minutes upperbound. When a retransmission timer expires, the entire flight of ClientHello message is retransmitted from scratch. As a matter of fact, the typical ClientHello message with associated application data is not large (at most a few datagrams), like reading from a smart meter, a tiny amount of network bandwidth is wasted in retransmission. An alternative approach would be to let the receiver to send an ACK message to indicate the message is received. This would prevent retransmitting a couple of unnecessary datagrams again. Considering the nature of the problem and typical message size in the smart grid, we decided not to add an ACK feature to our protocol, since ACKs would not provide enough improvement to be beneficial for the smart grid case and will cause additional overhead on communication.

6.5 Security Analysis

6.5.1 MitM prevention and Mutual Authentication

The proposed scheme mitigates this attack by providing mutual authentication between the field device and CC. The mutual authentication is achieved through the computed S_i values which uniquely identifies a client by hashing the ID of it with a nonce $H(ID_i, Nonce)$. In addition, the chain of the hash also provides an implicit authentication for the CC itself since the field device can authenticate the CC by computing the $H(S_i^{n-1})$. For any message altering between the field device and CC, the HMAC mechanism provides tamper detection. The adversary cannot calculate a legitimate HMAC without forging the shared key.

6.5.2 Forward Secrecy

In the case of an attack where ephemeral DH share Y is stolen, the attacker will be able to drive k_1. However, since Y needs to be updated in a medium-term, this will enable computing new session keys from the new DH share and thus providing forward secrecy. Our scheme allows power grid operators to arrange the level of forward secrecy by managing the DH update time policy. Note that in case of the compromise of a shared key k_1, the key refresh phase requires picking up a new random number b to obtain another shared key, k_2, and thus the adversary will not be able to obtain k_2. This ensures perfect forward secrecy.

6.5.3 Replay Attacks

Replay attacks are well studied and there can be different solutions to address them. However, the challenge in our case is to address them in a severely restricted environment. For instance, in TLS 1.3 1-RTT, this is achieved through *nonce* values. Each side generates a unique nonce value which is employed to guarantee that the other party is fresh by forcing them to include the nonce in the key derivation. However, when you are restricted to use 0-RTT, only one side will be able to add nonce not both. Another widely used option might be applying timestamps in preventing a replay attack. When a field device wants to send a message to the CC, it includes its measure of the time in the message. Thus, the CC only accepts messages for which the timestamp is within a reasonable time-limit. Although the timestamping solution works with accurate synchronization and reliable communication environment where the packet latency is negligible, it might cause problems when the communication medium is unreliable and has high data latency which is the case for our applications. Moreover, even though secure time synchronization is achieved, an adversary can still accomplish replay attacks, if s/he performs it fast enough within the needed time-limit.

Our approach for replay attack mitigation is based on state changes for the CC or field devices. Whenever, a new key is created, the device or CC will move to a new state. Therefore, if there is a replay message, it will be ignored as the state of the device or CC is changed. These states are created based on ${S_i}^n$ values. Specifically, in our protocol, the field device sends ${S_i}^n$ to CC and this is considered as the state of the CC. The CC first confirms that the received state information matches with its own state (i.e., ${S_i}^n$) and then computes ${S_i}^{n-1}$ to update its state. This process ensures the freshness of the message coming from the field device when encrypting the first data message. Thus, even though an adversary performs a replay of this first message, the CC will not validate ${S_i}^n$ as it already moved to a new state (i.e., ${S_i}^{n-1}$). After the CC receives the first message from the field device, it now sends ${S_i}^{n-1}$ to it. The field device validates $H({S_i}^{n-1})$ and update its own local state to ${S_i}^{n-1}$. Thus, a replay message to the field device for this message would not be accepted as its state is changed. Therefore, our protocol is resistant to replay attack yet performing a 0-RTT key exchange.

6.5.4 Amplification Attack

Amplification attack is a particular type of DoS attack that is categorized as distributed reflective denial-of-service (DRDoS) attacks [Ros14b]. The aim of the adversary is exhausting the bandwidth via IP spoofing where internet packets have fake source addresses. Considering limited bandwidth of the smart grid, this attack might be very harmful even by an attacker with limited amount of resources. A particular from of this attack to a key-exchange protocol can easily be accomplished by sending fake “ClientHello” message with spoofed IPs which seem to originating from the legit field devices in smart grid. The server then response with a “Server Hello” message (which has a significant size) to a non-requesting field device. The field device respond the “Server Hello” with an ”alert message” indicates that he is not the requester. This message-exchange might waste a significant size of bandwidth due to the relatively big size of “Server Hello” messages.

In the case of where TCP used for key-exchange (e.g, TLS and our TCP based approach), a “Client Hello” is sent only after the three-way handshake of TCP is done, which means that the field device with that IP actually have initiated the communication. However, if the key-exchange mechanism is realized over a connectionless setting, the key-exchange becomes vulnerable to the amplification attack. To do so, the attacker just sends “Client Hello” message directly to CC server with spoofed IP. The server then responds with a “Server Hello” message. To prevent this attack, DTLS has an configuration option to send additional *HelloRetryRequest* message (which has smaller size than “ServerHello”) to the client before replying with “Server Hello” message to ensure the client’s existence. This mechanism is really an imitation of the TCP three-way handshake. This countermeasure costs one extra roundtrip which brings DTLS back to TCP like three-way handshake.

Our approach hinders this particular amplification attack and does not do any computation or send “ServerHello” message before verifying the client. In order to do this, our protocol just checks S to determine that the client is capable of sending packets. If the CC does not confirm the incoming S, it just aborts any further messaging and computation. This mechanism also has indirect effects on the security of smart grid communication. For instance, network operators of the smart grid might disable the related amplification attack prevention option of DTLS [RTM20] to gain overhead advantage since the option requires an additional *HelloRetryRequest* message, However, this makes smart grid open to this vulnerability which can be employed anytime. Our protocol provides DoS protection by design without creating an extra overhead to the communication.

6.6 Evaluations

6.6.1 Evaluation of the Approach

The main objective of our work is to create a 0-RTT key-exchange scheme for Smart Grid to decrease the associated overhead, which is vital for improving the general health of Smart Grid. For comparison to our approach, we used four other baselines from the literature that utilize UDP and TCP. The compared key-exchange protocols are DTLS 1.2 [RM12], DTLS 1.3 [RTM20], TLS 1.2 [DR08], TLS 1.3 [RD18]. We evaluate protocols under two different security settings:

- *Public Key Infrastructure(PKI) Setting:* When the security infrastructure of smart grid uses PKI setting, the parties authenticate each other by utilizing Elliptic Curve Digital Signature Algorithm (ECDSA) and their certificates during key-exchange. Note that, the public keys are generated Elliptic-curve with curve definition secp256r1 [Bro09] for reduced certificate overhead.
- *Pre-shared Key(PSK) Setting:* If the infrastructure uses PSK setting, the parties identify each other via a PSK and does not send certificates. The server and client uses the PSK to authenticate the messages while negotiating for key-exchange.

For both settings, we used SHA256 when ensuring message integrity and Elliptic-curve Diffie–Hellman (ECDH) for establishing a shared key. To compare methods, we consider if a new key is to be established between the CC and field device, the data to be sent needs to wait until the key is created since this is similar to establishing a session in TCP. Therefore, in our experiments, we used the *average elapsed time* metric that measures the first encrypted payload data to reach the CC from a device. This metric assesses the impact of key exchange on transfer delay of application data.

6.6.2 Experimental Setup

Testbed Setup

To assess the performance of the proposed key-exchange mechanism, we created a testbed environment by LoRa communication technology [SLE+15]. LoRa utilizes 915 MHz ISM Band which is a band that exactly represents a typical severely bandwidth-constrained environment of Smart Grid. The testbed contains Lora modules attached to Raspberry-PIs which are registered to Lora gateway as shown in Fig. 6.10 which is then connected to AWS-IOT cloud service acted as a control center (CC) in smart grid. In the figure, Raspberry PIs represents IoT/field devices that communicate to CC server. To achieve this communication setup LoRaWAN gateway/switch forwards LoRa packets to The Things Network (TTN) which acts as a router. A message coming from any LoRa module integrated raspberry PIs is collected by the TTN and forwarded to the CC server and vice versa. This setup enables to mimic commonly used propriety 900Mhz radio technologies within smart

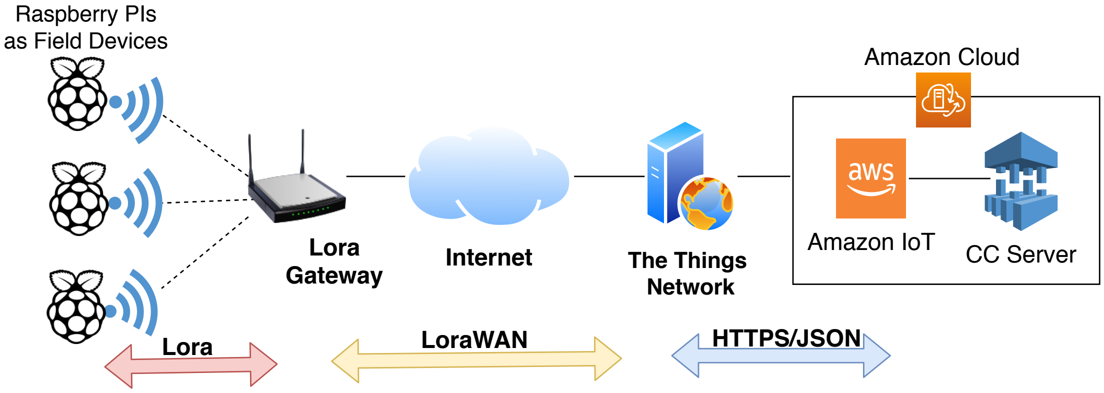


Figure 6.10: Testbed utilizing Lora and AWS

Physical Channel Settings	
PropagationDelay	SpeedDelayModel
Signal Loss Exponent	3.76
ReferenceDistance	1
ReferenceLoss	7.2
MaxPayload	61/133 bytes
DataRate	1760/3125 bytes

Table 6.1: Lora Physical Channel Parameters

grid, which has bandwidths in the order of kilobits. Therefore, the testbed which is taking this restriction into account can be considered a realistic test environment to assess different key-exchange mechanisms on smart grid.

Simulation Setup

Although testbed provides very good environment for mimicking real conditions that reflects the bandwidth and propagation characteristics of a 900Mhz signal in wild, it still lacks to represent a large-scale setup which contains hundreds of IoT devices. To assess the performance of the proposed key-exchange mechanism in such a large-scale setting, we created a severely bandwidth-constrained environment in ns-3 network simulator by mimicking LoRa. To apply this setup, we virtualized the network, field devices and CC using docker containers [Ros14a] as shown in Fig. 6.11 and integrate them with ns-3 through ns-3 tap bridge mechanism. We conduct key-exchange experiments using this setting. The physical communication channel of simulation set according to LoRa v1.1 [Spe18] specifications as shown in Table 6.1.

6.6.3 Testbed Results

The evaluation of key-exchange methods utilizing the LoRa wireless technology was performed in the MMC Campus of FIU, an environment with a mix of many concrete

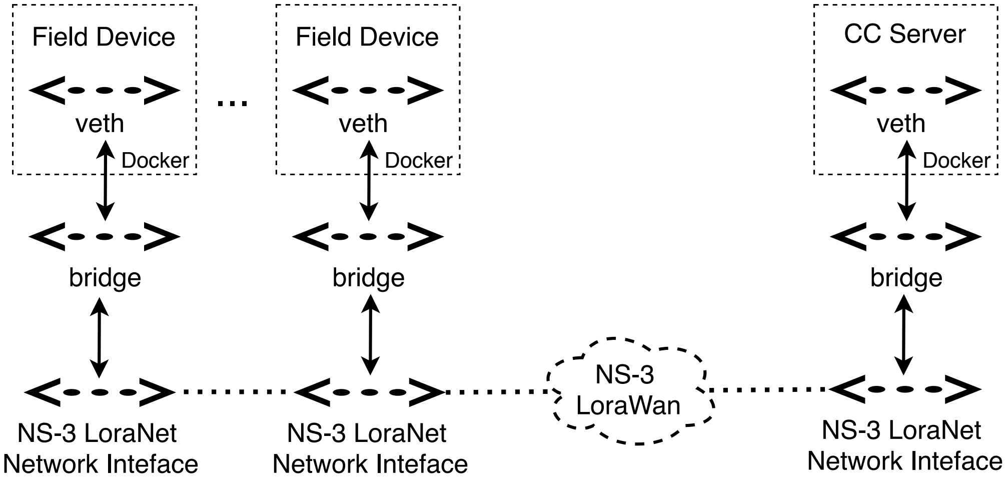


Figure 6.11: NS-3 Simulation Setup

and glass buildings up to 15 stories. The gateway was placed above a parking garage on the top of a seven-story building on the FIU campus. The LoRa module installed Raspberry PIs that mimic the field devices performed key-exchange operations at different locations within the campus which covers 2000 m^2 area. We collected the experiments under two categories by arranging the Spreading Factor of LoRa signal and distance of the Raspberry PIs from the gateway to represent the urban and rural settings. We report on the elapsed time for the key-exchange with respect to other baselines under these settings. During the tests, we also logged the received signal strength to be able to show distance impact on the received signal quality.

The required time to accomplish a key-exchange for all the approaches are shown in Figure 6.12 for PSK settings. These results indicate that our approach has significantly less message delay than TLS and DTLS approaches due to its RTT advantage. Another observation is that, DTLS is slightly worse than TLS even though it utilizes UDP. The UDP-based key-exchange in DTLS does not decrease the elapsed time, in fact it causes an increase because of the three-way handshake through *HelloVerifyRequest* is slightly heavier than the TCP handshake of TLS. As seen from Figure 6.12, the gap between DTLS and TLS results grows even more for rural settings. This

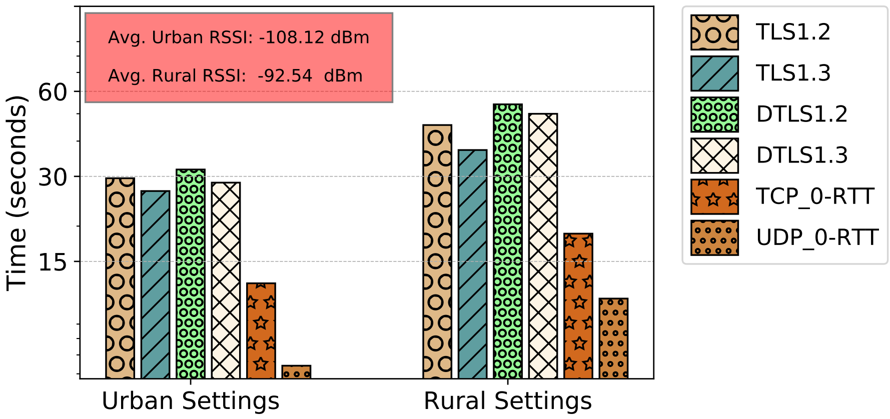


Figure 6.12: Testbed Results under PSK Settings

can be attributed to the fact that the bandwidth is decreasing in rural settings because the signal quality is close to the minimum supported Received Signal Strength Indication (RSSI) value of −120 dBm of LoRa. The RSSI indicates that how well the gateway is hearing the signal from the Raspberry PIs which affects directly the bandwidth of the communication.

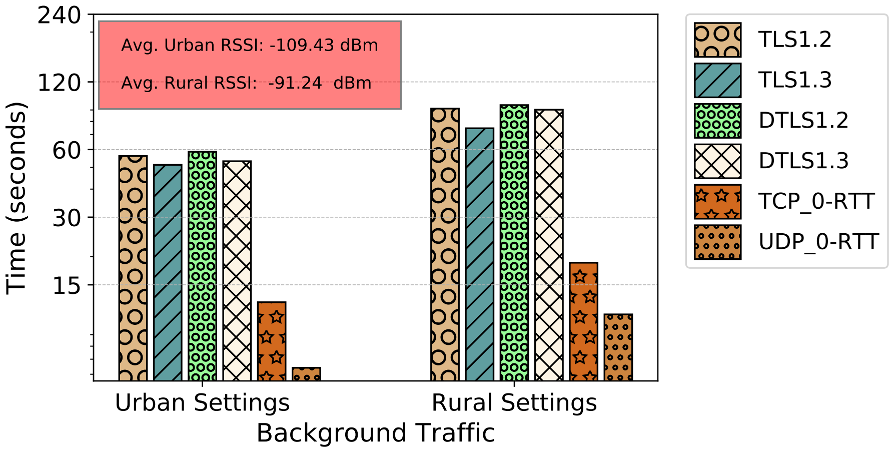


Figure 6.13: Testbed Results under PKI Settings

Looking at the elapsed time results in Figure 6.13 under PKI settings, we see that our results are encouraging since our approach outperforms the others and provides significant reductions in terms of delay. According to these results, the average time for the TLS and DTLS approaches is close to 2 minutes for rural settings which hints about the applicability of our approach since our UDP based approach is under 10 seconds which makes it even a better candidate to be employed.

6.6.4 Simulation Results

In the simulation, we created two settings where there is a single gateway surrounded by 200 IoT devices. The first setting mimics the LoRa in urban environment and the second setting mimics in rural. To see the scaling effects, we generated background data using these devices that produces 4, 8, 16 and 32bps bandwidth consumption in total.

Urban Environment Results

Fig. 6.14 shows the obtained results under PKI settings. The results indicate that the key-exchange is a costly operation especially under low-bandwidth. On the other hand, our simulation setup very is successful to represent the testbed if we check the results at “4kbps” background data traffic. The results are very similar to the results at urban setting in testbed which is shown in Fig. 6.13.

As expected, the elapsed time required to complete the key-exchange and transmit the first encrypted data increases as background traffic grows. For instance, even in ideal simulation environment, the key-exchange by utilizing TLS and DTLS versions takes nearly 2 minutes which significantly hinders actual data transmission and thus may put the Smart Grid operations at risk.

PSK on the other hand does not carry the overhead of certificate and signature exchange to establish secure communication. As such, the results for PSK is much better than the PKI results as seen in Fig. 6.15. For example, the DTLS1.2 takes

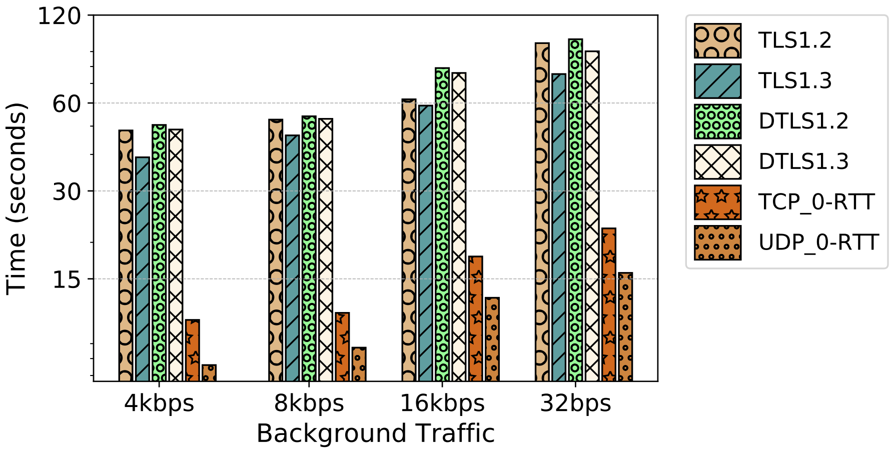


Figure 6.14: Urban-PKI Settings

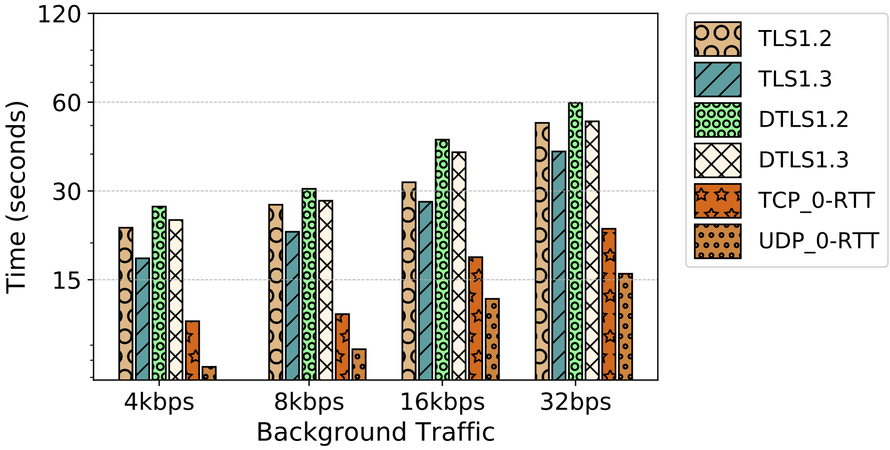


Figure 6.15: Urban-PSK Settings

around nearly 1 minute instead of 2 minutes as in the previous case. One the other hand, as expected both of the TLS1.3 version perform better than the other DTLS and TLS versions due to 1-RTT reduction.

However, for both cases, our approach significantly reduces the elapsed time compared to the other standards due to its 0-RTT nature while establishing the handshake. Even with respect to TLS 1.3, which is touted as one of the most efficient methods for the secure transport layer, our UDP_0-RTT approach reduced the data submission latency approximately 10 orders of magnitude for PKI and 5 orders of magnitude for PSK. It shows our UDP adaptation helps significantly to decrease the overhead in further.

Rural Environment Results

Fig. 6.16 shows the elapsed time results of all approaches in PKI settings for the rural environment. The latency becomes even worse in rural settings. Even with TLS 1.3, the key-exchange takes nearly more than 2 minutes. This is due to even reduced bandwidth with increased distances. It is also interesting to observe that even though we are using ECC PKI setup which has a decent public key and signature size compared to RSA PKI, the required certificate and signature exchange for mutual authentication affects the latency adversely. Since the additional overhead to carry them causes significant latency in a severely low-bandwidth communication environment.

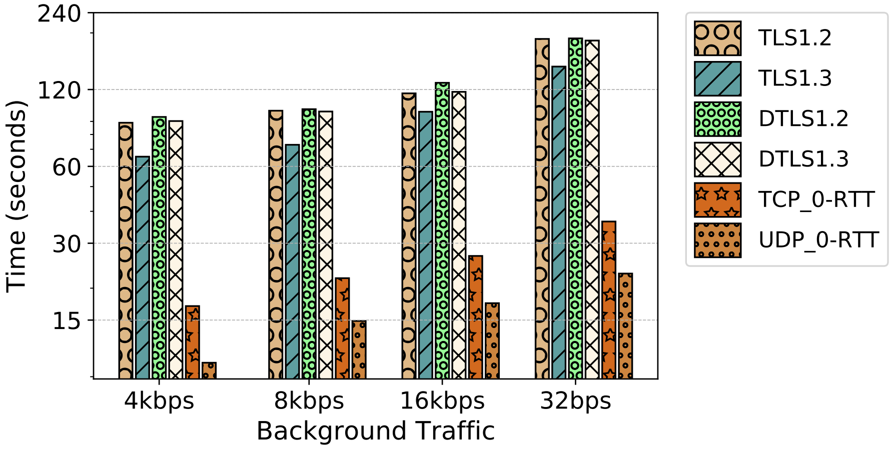


Figure 6.16: Rural-PKI Settings

As in the PKI-settings, our approach significantly reduces the elapsed time under PSK as well. For instance, it takes around 80 second to transport the payload to the CC even using TLS 1.3, However, our UDP based approach significantly reduces the time and complete the same task in around 20 seconds. Overall, we can confidently

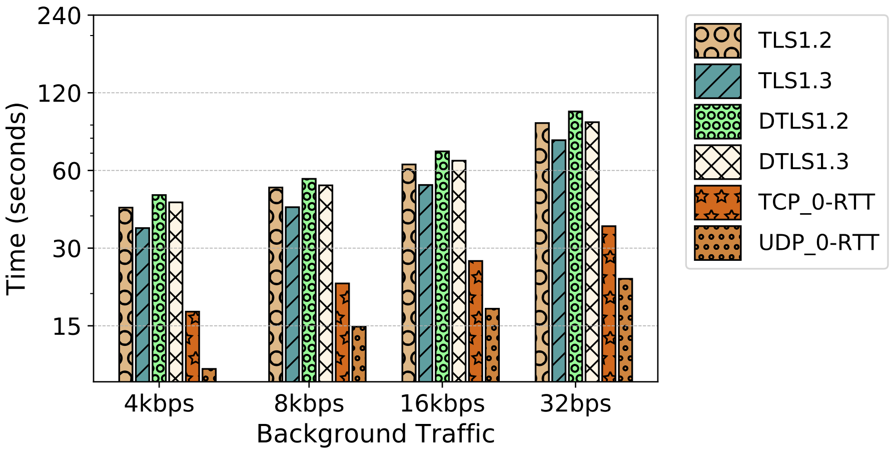


Figure 6.17: Rural-PSK Settings

claim that the rural environment stresses the importance of our approach in a much better way as it significantly outperforms all versions of TLS and DTLS.

6.7 Conclusion

The overall purpose of this Chapter was reducing the overhead of key-exchange mechanisms in the low-bandwidth communication environments for Smart Grid. To do so, we presented two different replay-attack resistant 0-RTT key-exchange mechanism based on TCP and UDP. For evaluation, we built a simulation environment by using ns-3 to mimic the communication characteristics of LoRa. The results showed the superior performance of our approach compared to the other standard key-exchange mechanisms and made a strong case that low-bandwidth links may significantly hinder the applicability of security protocols for smart grid applications.

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this dissertation, we tackled the overhead of key management for different Smart Grid use cases. To prevent the possible effects of the overhead on general health and security of Smart Grid, we proposed three distinct approaches that employ different techniques to relieve the overhead and boost the security and the efficiency of Smart Grid.

First, we introduced a DHT-based approach for maintaining the revoked keys. We showed the approach's ability to mitigate various attacks by cleverly integrating the Elliptic Curve Digital Signature Algorithm (ECDSA) signatures and the conventional DHT lookup algorithm. The experiment results indicate that our approach decreases the revocation management significantly. Particularly, the results in the delta CRL case showed the efficiency of our approach by consuming at least ten times less resources compared to the traditional CRL method.

Second, we proposed a novel revocation management scheme for AMI by utilizing the cryptographic accumulators. The scheme has different aspects for improving overhead issues and the security of AMI. In this regard, we introduced a non-revoked proof concept that was not used before in any of the revocation works. Then, this concept is included in a certificate verification protocol with various countermeasures against possible threats. We demonstrated the superior efficiency of the approach in terms of storage and bandwidth usage. The results indicated that, for a decent AMI network which contains nearly 200 meters, our approach distributes the revocation information to all smart meters under a minute as opposed to several ten minutes when the traditional CRL method used.

Our third solution is related to relieving the associated over-the-wire overhead of the key-exchange in symmetric key cryptography settings. We proposed a novel 0-

RTT key-exchange mechanism to significantly decrease messaging cost particularly for low-bandwidth communication setup of Smart Grid. In addition to TCP-based 0-RTT key-exchange that naturally provides ordered and error-checked stream of information, we also introduced a UDP based 0-RTT key-exchange. We showed UDP based key-exchange is a more proper way to deliver messages in such a low-bandwidth environment to avoid unnecessary header overhead of TCP due to flow-control, error-check, and connection setup. We developed a significantly efficient key-exchange mechanism over the UDP channel while introducing solutions to the unique challenges because of its connectionless nature. Our UDP-based key-exchange ensures the reliability of establishing keying material by introducing a lightweight re-delivery mechanism. We examined potential security threats and showed the robustness of our approach against them. Finally, the experiments showed the superiority of the proposed solution. For instance, by our approach, the key-exchange is completed at least five times faster than TLS1.3 which is the current state-of-art method for key-exchange.

Hereafter, we present potential future research directions related to expanding the proposed methods in this dissertation.

- One drawback of our DHT based approach is that the proposed approach only considers the small and medium-sized AMI networks while deploying the distributed revocation management. However, some AMI can contain thousands of smart meters that may hinder the efficiency of our solution. One potential future research direction to tackle this problem can be introducing a multi-level distributed revocation management by creating hierarchically organized DHT stores. This may help create a mechanism to efficiently reach the revocation information by limiting the perimeter of the individual DHTs.

- As a future work of our accumulator-based solution, one can aim to incorporate an improved accumulator scheme to relax the tight synchronization requirement. Since different smart meters is now able to use asynchronous proofs and accumulator values to check the validity of the certificate, the proposed method may require some architectural modifications to enable a relaxed revocation check but still ensure security.

- A further study for the accumulator approach can be accomplished by utilizing Bilinear accumulators [DT08] instead of RSA accumulators. Bilinear accumulators have a significant disadvantage for requiring a linear proof and accumulator size depending on the number of elements committed to the accumulator. Nevertheless, if utility company has an upper bound on the number of revoked items, it may bring some advantages in terms of computation and security. The ensemble of RSA and Bilinear accumulators for different use cases may open several improvement opportunities by combining their strengths.

- In our 0-RTT key-exchange scheme, the parties are required to set up a DH-share value and use this value while determining different shared keys. As remarked, the DH-share value should have an expiration time and be updated regularly for security purposes. In the dissertation, we did not define a specific lifetime for it and used a static DH-share key size. However, these parameters may affect the performance of the key-exchange. As future work, the best lifetime and DH-share size can be investigated according to the specific requirements of Smart Grid while considering its limited communication infrastructure. Thus, developing a secure key-exchange policy that adapts according to the features of the deployed network (e.g., according to number of IoT devices, the capability of these devices, etc.) can be pursued to determine the best values to ensure security and performance.

- In the dissertation, we showed that the deployed SCADA systems have significant restrictions in terms of communication bandwidth. One potential direction to improve the security of Smart Grid can be enhancing the security of the Distributed Network Protocol (DNP3), which is a de facto protocol used by legacy devices. We believe that the employed dynamic chaining mechanism in our approach can bring advantages to improve the Secure Authentication (SA) module of DNP3 which is used to manage old field-devices.

REFERENCES

[316] ns 3. ns-3: network simulator 3. Release 3.24.1, 2016.

[802] The status of ieee 802.11s standard.

[AKDB12] Agapios Avramidis, Panayiotis Kotzanikolaou, Christos Douligeris, and Mike Burmester. Chord-pki: A distributed trust infrastructure based on p2p networks. *Computer Networks*, 56(1):378–398, 2012.

[ARMT14] Kemal Akkaya, Khaled Rabieh, Mohamed Mahmoud, and Samet Tonyali. Efficient generation and distribution of crls for ieee 802.11 s-based smart grid ami networks. In *Smart Grid Communications (SmartGridComm), 2014 IEEE International Conference on*, pages 982–988. IEEE, 2014.

[AVTP^{+}17] Ferran Adelantado, Xavier Vilajosana, Pere Tuset-Peiro, Borja Martinez, Joan Melia-Segui, and Thomas Watteyne. Understanding the limits of lorawan. *IEEE Communications magazine*, 55(9):34–40, 2017.

[BCD^{+}17] Foteini Baldimtsi, Jan Camenisch, Maria Dubovitskaya, Anna Lysyanskaya, Leonid Reyzin, Kai Samelin, and Sophia Yakoubov. Accumulators with applications to anonymity-preserving revocation. *IACR Cryptology ePrint Archive*, 2017:43, 2017.

[BDM93] Josh Benaloh and Michael De Mare. One-way accumulators: A decentralized alternative to digital signatures. In *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1993.

[BMP^{+}06] Flavio Bonomi, Michael Mitzenmacher, Rina Panigrahy, Sushil Singh, and George Varghese. An improved construction for counting bloom filters. In *European Symposium on Algorithms*, pages 684–695. Springer, 2006.

[BP97] Niko Barić and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology—EUROCRYPT'97*, pages 480–494. Springer, 1997.

[Bro09] Daniel R. L. Brown. Sec 1: Elliptic curve cryptography, 2009.

[Chi04] Giovanni Chiola. Extended fibonacci distances for fault-tolerant routing in chord-like dhts. In *Peer-to-Peer Systems, 2004. International Workshop on Hot Topics in*, pages 10–15. IEEE, 2004.

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Crypto*, volume 2442, pages 61–76. Springer, 2002.

[Coh06] Bram Cohen. Bittorent protocol, 2006.

[Coo08] Dave Cooper. Internet x. 509 public key infrastructure certificate and certificate revocation list (crl) profile, 2008.

[CSWH01] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Designing privacy enhancing technologies*, pages 46–66. Springer, 2001.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *Transactions on Information Theory*, 1976.

[DJSS18] David Derler, Tibor Jager, Daniel Slamanig, and Christoph Striecks. Bloom filter encryption and applications to efficient forward-secret 0-rtt key exchange. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 425–455. Springer, 2018.

[DKA$^+$14] Zakir Durumeric, James Kasten, David Adrian, J Alex Halderman, Michael Bailey, Frank Li, Nicolas Weaver, Johanna Amann, Jethro Beekman, Mathias Payer, et al. The matter of heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 475–488. ACM, 2014.

[DR08] Tim Dierks and Eric Rescorla. The transport layer security (tls) protocol version 1.2, 2008.

[DT08] Ivan Damgård and Nikos Triandopoulos. Supporting non-membership proofs with bilinear-map accumulators. *IACR Cryptology ePrint Archive*, 2008:538, 2008.

[Fac16] Facebook. Noise protocol framework. Release, 2016.

[Far10] Hassan Farhangi. The path of the smart grid. *IEEE power and energy magazine*, 8(1), 2010.

[FFM04] Michael J Freedman, Eric Freudenthal, and David Mazieres. Democratizing content publication with coral. In *NSDI*, volume 4, pages 18–18, 2004.

[FG14] Marc Fischlin and Felix Günther. Multi-stage key exchange and the case of google's quic protocol. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014.

[GDGV13] Parimala Garnepudi, Tipura Damarla, Jyotshna Gaddipati, and D Veeraiah. Proactive, reactive and hybrid multicast routing protocols for wireless mesh networks. In *Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on*, pages 1–7. IEEE, 2013.

[GH15] Joaquin Garcia-Hernandez. Recent progress in the implementation of ami projects: Standards and communications technologies. In *2015 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)*, pages 251–256. IEEE, 2015.

[GHJL17] Felix Günther, Britta Hale, Tibor Jager, and Sebastian Lauer. 0-rtt key exchange with full forward secrecy. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2017.

[GNOT92] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[Goo16] Google. Quic proocol. Release 3.24.1, 2016.

[GSK$^+$11] Vehbi C Gungor, Dilan Sahin, Taskin Kocak, Salih Ergut, Concettina Buccella, Carlo Cecati, and Gerhard P Hancke. Smart grid technologies: Communication technologies and standards. *IEEE transactions on Industrial informatics*, 2011.

[GSM$^+$13] Slava Galperin, Stefan Santesson, Michael Myers, Ambarish Malpani, and Carlisle Adams. X. 509 internet public key infrastructure online certificate status protocol-ocsp, 2013.

[HJLS17] Britta Hale, Tibor Jager, Sebastian Lauer, and Jörg Schwenk. Simple security definitions for and constructions of 0-rtt key exchange. In *International Conference on Applied Cryptography and Network Security*, 2017.

[HJP11] William Hart, Fredrik Johansson, and Sebastian Pancratz. Flint–fast library for number theory, 2011.

[Hof05] Paul Hoffman. Algorithms for internet key exchange version 1, 2005.

[HWQC08] Jun Huang, Zhao Wang, Zhao Qiu, and Mingrui Chen. Theoretical analysis of issuing mechanism in distributive digital certificate revocation list. In *Computer and Electrical Engineering, 2008. ICCEE 2008. International Conference on*, pages 199–203. IEEE, 2008.

[IEE12] IEEE. Ieee standard for electric power systems communications-distributed network protocol (dnp3),. *https://standards. ieee. org/standard/1815-2012. html*, 2012.

[KA] Stephen Kent and Randall Atkinson. Rfc 2401: Security architecture for the internet protocol, november 1998.

[KGM+19] Shreyas Kulkarni, Qinchen Gu, Eric Myers, Lalith Polepeddi, Szilárd Lipták, Raheem Beyah, and Deepak Divan. Enabling a decentralized smart grid using autonomous edge control devices. *IEEE Internet of Things Journal*, 6(5):7406–7419, 2019.

[KHLF10] Himanshu Khurana, Mark Hadley, Ning Lu, and Deborah A Frincke. Smart-grid security issues. *IEEE Security & Privacy*, 8(1), 2010.

[KHN+14] Charlie Kaufman, Paul Hoffman, Yoav Nir, Pasi Eronen, and Tero Kivinen. Internet key exchange protocol version 2 (ikev2), 2014.

[KLL+97] David Karger, Eric Lehman, Tom Leighton, Rina Panigrahy, Matthew Levine, and Daniel Lewin. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 654–663. ACM, 1997.

[lan15] landisgyr. Grid stream solutions overview, 2015.

[LJBNR15] Robert Lychev, Samuel Jero, Alexandra Boldyreva, and Cristina Nita-Rotaru. How secure and quick is quic? provable security and performance analyses. In *Symposium on Security and Privacy*. IEEE, 2015.

[LLX07] Jiangtao Li, Ninghui Li, and Rui Xue. Universal accumulators with efficient nonmembership proofs. In *ACNS*. Springer, 2007.

[LTCP16] Jaime Lloret, Jesus Tomas, Alejandro Canovas, and Lorena Parra. An integrated iot architecture for smart metering. *IEEE Communications Magazine*, 54(12):50–57, 2016.

[LTQ13] Xuelian Long, David Tipper, and Yi Qian. An advanced key management scheme for secure smart grid communications. In *International Conference on Smart Grid Communications*. IEEE, 2013.

[ME10] Anthony R Metke and Randy L Ekl. Security technology for smart grid networks. *IEEE Transactions on Smart Grid*, 1(1):99–107, 2010.

[MM02] Petar Maymounkov and David Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *International Workshop on Peer-to-Peer Systems*, pages 53–65. Springer, 2002.

[MM03] Matei Ciobanu Morogan and Sead Muftic. Certificate revocation system based on peer-to-peer crl distribution. In *Proc. of the DMS'03 Conference*, 2003.

[MMAS15] Mohamed MEA Mahmoud, Jelena Mišić, Kemal Akkaya, and Xuemin Shen. Investigating public-key certificate revocation in smart grid. *IEEE Internet of Things Journal*, 2(6):490–503, 2015.

[MMS13] Mohamed MEA Mahmoud, Jelena Misic, and Xuemin Shen. Efficient public-key certificate revocation schemes for smart grid. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 778–783. IEEE, 2013.

[NIS14] NIST. Guidelines for smart grid cybersecurity. NISTIR 7628 Rev. 1, 2014.

[NSS$^+$17] Matus Nemec, Marek Sys, Petr Svenda, Dusan Klinec, and Vashek Matyas. The return of coppersmith's attack: Practical factorization of widely used rsa moduli. In *to appear at 24th ACM Conference on Computer and Communications Security (CCS'2017)*. ACM, 2017.

[NWL$^+$10] Huy Hoang Ngo, Xianping Wu, Phu Dung Le, Campbell Wilson, and Balasubramaniam Srinivasan. Dynamic key cryptography and applications. *IJ Network Security*, 10, 2010.

[ope] openssl. Open ssl.

[Pet13] Yngve Pettersen. The transport layer security (tls) multiple certificate status request extension, 2013.

[PTT08] Charalampos Papamanthou, Roberto Tamassia, and Nikos Triandopoulos. Authenticated hash tables. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 437–448. ACM, 2008.

[Raz13] Shahid Raza. *Lightweight security solutions for the internet of things.* PhD thesis, Mälardalen University, Västerås, Sweden, 2013.

[RD01] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing*, pages 329–350. Springer, 2001.

[RD18] Eric Rescorla and Tim Dierks. The transport layer security (tls) protocol version 1.3, 2018.

[Res] E Rescorla. The transport layer security (tls) protocol version 1.3 (august 2018).

[RFH$^+$01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network, 2001.

[RGO05] Kenneth H Rosen, Bart Goddard, and Kevin O'Bryant. *Elementary number theory and its applications.* Pearson/Addison Wesley, 2005.

[RM12] Eric Rescorla and Nagendra Modadugu. Rfc 6347: Datagram transport layer security version 1.2. *Internet Engineering Task Force*, 13:101, 2012.

[RMAT17] Khaled Rabieh, Mohamed MEA Mahmoud, Kemal Akkaya, and Samet Tonyali. Scalable certificate revocation schemes for smart grid ami networks using bloom filters. *IEEE Transactions on Dependable and Secure Computing*, 14(4):420–432, 2017.

[RMT$^+$15] Khaled Rabieh, Mohamed Mahmoud, Samet Tonyali, et al. Scalable certificate revocation schemes for smart grid ami networks using bloom filters. *IEEE Transactions on Dependable and Secure Computing*, 2015.

[Ros14a] Rami Rosen. Linux containers and the future cloud. *Linux J*, 2014.

[Ros14b] Christian Rossow. Amplification hell: Revisiting network protocols for ddos abuse. In *NDSS*, 2014.

[RTM20] TLSE Rescorla, H Tschofenig, and N Modadugu. The datagram transport layer security (dtls) protocol version 1.3 draft-ietf-tls-dtls13-37. Technical report, Thttps://tools. ietf. org/html/draft-ietf-tls-dtls13-37, 2020.

[RY16] Leonid Reyzin and Sophia Yakoubov. Efficient asynchronous accumulators for distributed pki. In *International Conference on Security and Cryptography for Networks*, pages 292–309. Springer, 2016.

[SAU12] Nico Saputro, Kemal Akkaya, and Suleyman Uludag. A survey of routing protocols for smart grid communications. *Computer Networks*, 56(11):2742 – 2771, 2012.

[SCRC19] Yasir Saleem, Noel Crespi, Mubashir Husain Rehmani, and Rebecca Copeland. Internet of things-aided smart grid: technologies, architectures, applications, prototypes, and future research directions. *IEEE Access*, 7:62962–63003, 2019.

[SDB13] Seung-Hyun Seo, Xiaoyu Ding, and Elisa Bertino. Encryption key management for secure communication in smart advanced metering infrastructures. In *Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on*, pages 498–503. IEEE, 2013.

[SFP10] Aleksandar Seovic, Mark Falco, and Patrick Peralta. *Oracle Coherence 3.5*. Packt Publishing Ltd, 2010.

[SLE$^+$15] Nicolas Sornin, Miguel Luis, Thomas Eirich, Thorsten Kramp, and Olivier Hersent. Lorawan specification. *LoRa alliance*, 2015.

[SMK$^+$01] Ion Stoica, Robert Morris, David Karger, M Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review*, 31(4):149–160, 2001.

[Spe18] LoRaWAN Specification. v1. 1 available at: https://loraalliance.org/resource-hub/lorawantm-specification-v11. *Online Accessed*, 10, 2018.

[Tin] TinyMesh. Radio frequency (rf) datasheet.

[TL] TP-Link. Tp-link wireless adapter.

[Tou97] Jean Tourrilhes. Wireless extensions for linux. *Linux*, 1997.

[UIA12] Suleyman Uludag, Tom Imboden, and Kemal Akkaya. A taxonomy and evaluation for developing 802.11-based wireless mesh network testbeds. *International Journal of Communication Systems*, 25(8):963–990, 2012.

[YJ09] Gao Ying and Zhan Jiang. Research on crl distribution in p2p systems. In *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on*, pages 574–577. IEEE, 2009.

[YLL09] Zhenyu Yang, Ming Li, and Wenjing Lou. A network coding approach to reliable broadcast in wireless mesh networks. In *International Conference on Wireless Algorithms, Systems, and Applications*, pages 234–243. Springer, 2009.

VITA

MUMIN CEBE

2005	B.S., Computer Science Ege University Izmir, Turkey
2008	M.Sc., Computer Science Bilkent University Ankara, Turkey
2020	Ph.D., Electrical and Computer Engineering Florida International University (FIU) Miami, Florida

PUBLICATIONS

- "Communication-efficient Certificate Revocation Management for Advanced Metering Infrastructure", Mumin Cebe, Kemal Akkaya. *Elsevier Future Generation Computer System, 2020 (Accepted).*
- "Efficient certificate revocation management schemes for IoT-based advanced metering infrastructures in smart cities" , Mumin Cebe, Kemal Akkaya. *Journal of AdHoc Networks, Volume:92, p.101801-101841, 2018.*
- "A Replay Attack-Resistant 0-RTT Key Management Scheme for Low-Bandwidth Smart Grid Communications", Mumin Cebe, Kemal Akkaya. *IEEE Global Communications Conference (GLOBECOM), pp. 1-6. IEEE, 2019.*
- "Performance evaluation of key management schemes for wireless legacy smart grid environments: poster", Mumin Cebe, Kemal Akkaya. *ACM Conference on Security and Privacy in Wireless and Mobile Networks, pp. 334-335. ACM, 2019.*
- "Efficient Public-key Revocation Management for Secure Smart Meter Communications using One-way Cryptographic Accumulators", Mumin Cebe, Kemal Akkaya. *IEEE International Conference on Communications, pp. 1-6. IEEE, 2018.*
- "Efficient certificate verification for vehicle-to-grid communications", Nico Saputro, Samet Tonyali, Kemal Akkaya, Mumin Cebe and Mahmoud M., *In International Conference on Future Network Systems and Security, pp. 3-18. Springer, 2017.*

- “Efficient management of certificate revocation lists in smart grid advanced metering infrastructure”, Mumin Cebe , Kemal Akkaya *IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems, pp. 313-317. IEEE, 2017.*
- 2017 , “Utilizing Advanced Metering Infrastructure to Build a Public Key Infrastructure for Electric Vehicles”, Mumin Cebe, Kemal Akkaya. *ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications, pp. 91-98. ACM, 2017.*