Security and Privacy in Smart Grid Context: Problems and Solutions

by

Hasen Nicanfar

B.Sc., Sharif University of Technology, 1993
M.Sc., Ryerson University, 2011

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate and Postdoctoral Studies

(Electrical and Computer Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

October 2015

Abstract

In order to improve the power grid and provision the *Smart Grid* concept, one well-defined approach would be to utilize new information and communication technology. Live power consumption data in addition to the time base power consumption rate are essential requirements in this context. These communications are supposed to be bi-directional between consumers, providers and smart grid administrations (market, operators, etc.). However, one of the most essential requirements that should be preserved is to address communication security and privacy. There are many opportunities for adversaries to attack the smart grid system, even remotely anywhere in the world, that could result in costly issues and damages in the system, even jeopardize user privacy.

In the first part of this thesis, we concentrate on improving the efficiency of security mechanism and present our tailored authentication and key management mechanisms. We propose two solutions, one for communications between home appliances and a home gateway (smart meter), while the second solution aims at communications between the home smart meter and an appropriate server located in the smart grid utility network.

We then propose enhancements on key management by developing two key construction mechanisms based on the Password Authentication Key Exchange (PAKE) protocol. The first is a cluster-based group key mechanism between smart grid entities, e.g. consumers in a neighbourhood area network. The second enhancement is a multi-layer key mechanism motivated by controlling the home smart appliances using different smart grid controllers located in different layers of the controlling hierarchy network.

The second part of the thesis concentrates on Privacy. In this part, we present a privacy mechanism based on enhanced network coding for communications between smart meters and utility servers via a mesh topology. Finally, we propose a privacy-aware security solution for mobile devices. For example, to support electric vehicles in buying and selling the power from and to the grid, or in case of the smart phones in the heterogeneous network (4G and/or 5G), to support handover between the access points.
Hasen Nicanfar

Preface

Mainly this research was conducted in the WiNMoS laboratory, department of Electrical and Computer Engineering at the University of British Columbia, under the supervision of Professor Victor C.M. Leung. All of the chapters in this thesis are based on work conducted in UBCs WiNMoS lab, and the results are published as follows:

Chapter 2

[1] H. Nicanfar, P. Jokar, K. Beznosov and V.C.M. Leung, "Efficient Authentication and Key Management Mechanisms for Smart Grid Communications", *IEEE Systems Journal, Special Issue on Smart Grid Communications Systems*, vol. 8, no. 2, pp. 629-640, June 2014

This manuscript and its proposed solution was mainly designed by myself under my supervisor direction. P. Jokar reviewed the writing and discussed the solution, and Dr. K. Beznosov reviewed the solution, as well as directed the evaluation, especially adding the adversary model. Final review and correction are done by my supervisor.

[2] H. Nicanfar, P. Jokar and V.C.M. Leung, "Efficient Authentication and Key Management for the Home Area Network", in Proc. IEEE ICC, Ottawa, ON, June 2012

[3] H. Nicanfar, P. Jokar and V.C.M. Leung, "Smart Grid Authentication and Key Management for Unicast and Multicast Communications", in Proc. IEEE PES ISGT, Perth, Australia, Nov. 2011

Last two manuscripts ([2] and [3]) and their proposed solutions were mainly designed by myself under my supervisor direction. P. Jokar helped review the writing of the manuscript as well as discuss the solution. Final review and corrections are done by my supervisor.

Chapter 3

[**4**] H. Nicanfar and V.C.M. Leung, "Password Authenticated Cluster-Based Group-Key Agreement for Smart Grid Communication", *Security and Communication Networks*, Special Issue on Smart Grid Communication Systems: Reliability, Dependability & Performance, vol. 9, no. 1, pp 221-233, Jan. 2014

The proposed solution was designed by myself under my supervisor direction. The manuscripts were written by myself and reviewed and modified/corrected by my supervisor.

Chapter 4

[**5**] H. Nicanfar and V.C.M. Leung, "Multilayer Consensus ECC-Based Password Authenticated Key-Exchange (MCEPAK) Protocol for Smart Grid System", *IEEE Transaction on Smart Grid*, Special Issue on Security in Smart Grid, vol. 4, no. 1, pp 253-264, Mar. 2013

[**6**] H. Nicanfar and V.C.M. Leung, "Smart Grid Multilayer Consensus Password-Authenticated Key Exchange Protocol", in Proc. IEEE ICC-WS SFCS, Ottawa, ON, June 2012

This chapters manuscripts and their proposed solutions were mainly designed by myself under my supervisor direction; he also reviewed and corrected the manuscripts.

Chapter 5

[**7**] H. Nicanfar, P. TalebiFard, A. Alasaad and V.C.M. Leung, "Enhanced Network Coding to Maintain Privacy in Smart Grid Communication", *IEEE Transaction on Emerging Topics in Computing*, Special Issue on Cyber-Physical Systems (CPS), vol.1, no.2, pp.286-296, Dec. 2013

[**8**] H. Nicanfar, P. TalebiFard, A. Alasaad and V.C.M. Leung, "Privacy-Preserving Scheme in Smart Grid Communication Using Enhanced Network Coding", in Proc. IEEE ICC, Budapest, Hungary, June 2013

In developing the proposed solutions, I was in-charge of the privacy side and Smart Grid network, and P. TalebiFard brought the technical information about the enhance network coding. The design was from a brain storming discussion under my supervisor direction. I also wrote the most part of

the manuscrpit. P. TalebiFard wrote the abstract and conclusion, as well as rewrote the network coding subsection of the background. The "Communication and network performance analysis" was a shared task of the A. Alasaad and P. TalebiFard. A. Alasaad improved the writing and provided the figures. My supervisor also reviewed and corrected the manuscripts at the end.

Chapter 6

[**9**] H. Nicanfar, J. Hajipour, F. Agharebparast, P. TalebiFard and V.C.M. Leung, "Privacy-Preserving Handover Mechanism in 4G", in Proc. IEEE CNS, Washington, DC, Oct. 2013

In developing the proposed solution, I was in charge of the privacy side, and J. Hajipour and F. Agharebparast brought the heterogeneous network knowledge. In developing the solution, they looked at it from the HetNet point of view. The design was done in a team base (including P. TalebiFard), under my supervisor direction. I also wrote most of the manuscript. J. Hajipour, F. Agharebparast and P. TalebiFard helped review literature, background, and figures. They also revised and corrected the writing. My supervisor also reviewed and corrected the manuscripts at the end.

[**10**] H. Nicanfar, P. TalebiFard, S. Hosseininezhad, V.C.M. Leung and M. Damm, "Security and Privacy of Electric Vehicles in the Smart Grid Context: Problem and Solution", in Proc. ACM DIVANet, Barcelona Spain, Nov. 2013

[**11**] H. Nicanfar, S. Hosseininezhad, P. TalebiFard and V.C.M. Leung, "Robust Privacy-Preserving Authentication Scheme for Communication Between Electric Vehicle as Power Energy Storage and Power Stations", in Proc. IEEE INFOCOM-WS CCSES, Turin, Italy, Apr. 2013

In developing the proposed solution, I was in-charge of the privacy side and Smart Grid network, and S. Hosseininezhad brought the vehicular network knowledge. In a brain storming session with P. TalebiFard we developed the solution, under my supervisors direction. I also wrote the most part of the manuscript. P. TalebiFard wrote the abstract and conclusion, and S. Hosseininezhad rewrote and modified the literature review and background. M. Damm reviewed the final version from a market point of view, and my supervisor corrected and finalized the last version.

Table of Contents

List of Tables

List of Figures

List of Acronyms

4G Forth Generation

5G Fifth Generation

AAA Authorization, Availability, Accountability

AAM Advanced Asset Management

ADO Advanced Distribution Operations

ADR Automated Demand Response

AKE Asymmetric Key Exchange

AMI Advanced Metering Infrastructure

AMR Automated Meter Reader

A_N Home Appliance

AP Access Point

ASER Average Symbol Error Rate

ATO Advanced Transmission Operations

AVISPA Automated Validation of Internet Security Protocols and Application

BACnet Building and Automation Control Networking

BAN Building Area Network

B_C Controller of a building area network

BD Burmester and Desmedt (protocol)

BEMS Building Energy Management System

BW BandWidth

CA Certificate Authority

C_C Smart Grid Central Controller

CDR Cloud-based Demand Response

CIA Confidentiality, Integrity and Availability

Cl-AtSe Constraint Logic-based Attack Searcher

CP Charging Point

CP-ABE Ciphertext-Policy Attribute-Based Encryption

CPS Cyber-Physical System

D-H Diffie and Hellman

DIEMS Distributed Intelligent Energy Management System

DLP Discrete Logarithm Problem

DOE Department of Energy

DoS Denial of Service (attack)

DRM Demand Response Management

DSL Digital Subscriber Lines

EAP Extensible Authentication Protocol

EAP-TLS Extensible Authentication Protocol Transport Layer Security

EBS Exclusion Basis Systems

EC Elliptic Curve

ECC Elliptic Curve Cryptography

ECC-CDH Elliptic Curve Cryptography Cofactor Diffie-Hellman

ECDH Elliptic Curve Diffie-Hellman

EIBC Enhance Identity-Based Cryptography

EMS Energy Management System

ENC Enhanced Network Coding

EPAK ECC-based Password Authenticated Key-Exchange

ES Estate Estimator

EV Electric Vehicle

FA Feeder Automation

GEV Global Encoding Vector

GK Group Key

GKA Group Key Agreement

GKR Group Key Reconstruction

HAN Home Area Network

H_C Controller of a home area network

HetNet Heterogeneous Network

HLPSL High Level Protocol Specifications Language

HPEV hybrid Plug in Electric Vehicle

HVAC Heating, Ventilation and Air Condition

IAN Industry Area Network

IBC Identity-Based Cryptography

ICT Information and Communication Technology

ID Identity

IDS Intrusion Detection System

IPv6 Internet Protocol version 6

IT Information Technology

LEV Local Encoding Vector

LTR Long Term Refreshment

MBWA Mobile Broadband Wireless Access

MCEPAK Multilayer Consensus ECC-based Password Authenticated Key-exchange

MDMS Meter Data Management System

MID Multicast Group ID

MITM Man-in-the-Middle (attack)

MGR Multicast Group Receiver

MGS Multicast Group Source

MMS Microgrid Management System

MTR Medium Term Refreshment

MWM Mobile Work-flow Management

NAN Neighbourhood Area Network

N_C Controller of a neighbourhood area network

NC Network Coding

NIST National Institute of Standards and Technology

NV End Value

OFMC On-the-Fly Model-Checker

OPF Optimal Power Flow

PACGKA Password Authenticated Cluster-based Group Key Agreement

PAKE Password Authentication Key Exchange

PbKE Public Key (or Asymmetric) Encryption (system)

PEV Plug-in Electric Vehicle

PKG Private Key Generator

PKI Public Key Infrastructure

PLC Power Line Communication

PrKE Private Key (or Symmetric) Encryption (system)

PRNG Pseudo Random Number Generator

PV Photovoltaic

QoS Quality of Service

ROI Return On Investment

RSA Rivest, Shamir and Adleman

RSU Road Side Unit

RT Real Time

RTU Remote Terminal Unit

SAS Security and Authentication Server

SC Smart Card

SCADA Supervisory Control And Data Acquisition

SG Smart Grid

SGCC Smart Grid Central Controller

SGKM Smart Grid Key Management

SGMA Smart Grid Mutual Authentication

SGS Smart Grid Server

SM Smart Meter

SMK Source Multicast Key

SN Serial Number

SRP Secure Remote Password (protocol)

STR Short Term Refreshment

SV Seed Value

TOU Time Of Use

TRNG True Random Number Generator

TS Time Stamp

V23PPA Vehicle To Third Party Privacy-preserved Authentication

V2G Vehicle To Grid)

V2GA Vehicle To Grid Authentication

V2R Vehicle to Roadside

VANET Vehicular Ad-Hoc Network

VIN Vehicle Identification Number

VT Valid Time

W2V2G Wind To Vehicle To Grid

Acknowledgements

I would like to acknowledge my appreciation for my supervisor, Professor Victor C.M. Leung for his technical guidance, his support, and advice during my PhD program. Without his style of supervising the candidates, I could not be where I am today.

In addition, I would like to thank the committee members and examiners in the qualifying and final exams, for taking their time and for their constructive comments: Professor Ian Blake, Professor Konstantin Beznosov, Professor Karthik Pattabiraman, Professor Jose Marti, Professor Juri Jatskevich, Professor Norm Hutchinson, Professor Vijay Bhargava, and Professor Ashraf Matrawy.

Also, I would like to extend my appreciation to my colleagues in the WiNMoS lab, and co-authors of my publications, as well as my family and my friends for their help and their always support during this program.

Dedication

- To Ava and Hana

Chapter 1

Introduction and Background

The rapid development of Information and Communication Technology (ICT) provides opportunities for a more pleasurable life style and more efficient systems. A good portion of these systems, like Smart Grid (SG), or other Cyber- Physical Systems (CPSs), such as, electronic health-care system, are multi-entity systems which consist of some sub-systems working together as systems of systems. In addition, new concepts and opportunities, such as cloud computing, provide better and more efficient service delivery for the ICT department to manage and run their applications and systems more efficiently and with a less investment and cost. Moreover, current enhancements in mobile devices contributes to smart applications running on smart devices, which in some cases need to be run on the cloud as well. Today mobile cloud computing has its own area in the ICT field is growing daily.

Having these benefits makes ICT an interesting subject; however, it does cause many security and privacy issues as well. The users' information is on the fly, or in some cases, the users do not have much control where their information is being saved and backed up. As a result, the security and privacy solutions and mechanisms should be revisited ensuring the new requirements are fully addressed. From a business point of view, the investors mainly concentrate on efficiency and improving their profit margins. However, they also need to pay enough attention to the security (and privacy). It is obvious that mostly nobody profits from security; and security (and privacy) mechanisms are part of the system requirements and generally considered as cost of a system. Therefore, mostly the intention is to decrease the cost by designing a security mechanism that addresses the minimum requirements, or covers coming short (reasonable) future. For instance, even the governments data is being released after a period of time (e.g. 30-40 years). The security mechanisms should be able to keep the data secure for that specific period of time and duration. From this point of view, the aim in the security field is to make the provided mechanism as efficient and as light as possible, and at the same time, meet the demands and requirements

of the system and application.

Another related area in the security field, which is its own area and filed of research now-a-day, is privacy. Considering above discussion, these days because of the complexity design of the new systems, and utilizing different new technologies, the users' privacy gain more attention. This is users concern to know and sometimes decide about the location, and even country, that their data and/or information is being kept, and who has access to them. For instance in an old fashion ICT environment, a limited number of people could have access to the system and its data, and mostly it was doable to track the accesses. However, when the users data, like photos, are being uploaded to the Cloud, the users cannot trace and/or monitor the data movements like old days anymore.

If we want to present an overall discussion, from a communication point of view, when a user/entity wants to communicate with another user/entity/system via a platform/media or even a third party, the following steps can be considered although it is not the only perspective:

- Authentication: Firstly, an entity or a user should be introduced to the one that wants to communicate (one-way authentication), or the entities need to recognize each other (mutual authentication).
- Security key or key management (e.g. for encryption and/or signing): Normally most of the authentication protocols are tailored and ended with an appropriate key construction mechanism. In fact, after the authentication and in order to set up a secure channel, entities require a key, for instance, to encrypt their data in communication and protect it from unauthorized access. Managing the key, between two entities or a group of entities that may require to communicate with each other, is our next step.
- Maintaining user's Privacy: Maintaining the encryption can protect the data; however, cannot fully preserve the privacy of the users. Therefore, the mechanism and system should be designed somehow to preserve the privacy of the users as well, which is our final steps in this thesis.

In this thesis and following to the above steps, first of all we concentrate on increasing the efficiency of authentication scheme (Chapter 2) and key management protocol. Then we study and provide two multi-entity key management mechanisms (Chapter 3 and Chapter 4), and then finally move to the privacy solutions (Chapter 5 and Chapter 6).

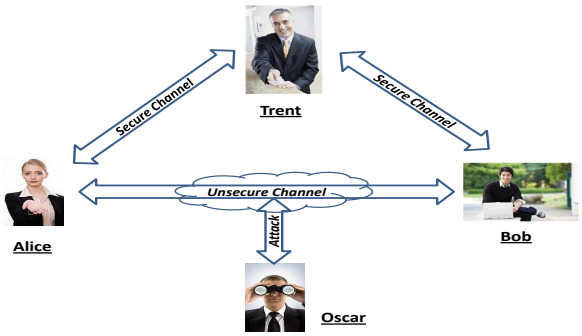


Figure 1.1: PbKE Main Parties

In the following sections of this chapter, some of the standards and base concepts in each of the above mentioned areas are being discussed and briefly reviewed.

1.1 Security Background

Following the above discussion, most of this thesis in the Security field concentrates on efficiency of the authentication and key management. Security, which is "*safety, or freedom from worry,*" can be introduced as a risk management topic where "`Risk = Asset * Vulnerability * Threat`". In the ICT world, three policies are identified for the security such as Confidentiality, Integrity, and Availability (CIA). Precisely, designing a security system in ICT means providing required, e.g., mechanism to address any or all of these policies. Communication security can be divided to`channel security` and `data security`; however, "*authentication and key management mechanisms are the main parts of any security system in the ICT*". The fundamental technique used in the security mechanism is applied mathematics, or cryptography.

1.1.1 Private and Public Key Encryption Systems

There are two systems that make a message confidential between two parties, such as Private Key (or Symmetric) Encryption (PrKE) system, and Public Key (or Asymmetric) Encryption (PbKE) system. The PrKE system requires only one key (called private key) to be shared by sender and receiver. In fact, the sender encrypts the message with the shared key and

the receiver decrypts the encrypted message with the same key. One of the well-known solution to construct a symmetric key is proposed by Diffie and Hellman (D-H) [2].

On the other hand and in the PbKE system (Figure 1.1), two keys, a public key and a private key are provided for each entity. The entity keeps the private key in private and secure; however, the public key is defined to be accessible publicly. One of the well-known solutions to construct an asymmetric key is proposed by Rivest, Shamir, and Adleman (RSA)[3].

Normally, in a practical PbKE system, a third party acts as a Private Key Generator (PKG) or Certificate Authority (CA) (Trent), e.g., issues an individual certificate for each entity which includes private key of the entity, and is in charge of the entire key management process including the key refreshment. When Bob wants to send a secure message to Alice to avoid intrusion from an intruder (Oscar), he encrypts the message using Alices public key. On the other end, Alice decrypts the received encrypted message using her own private key. Furthermore, to protect the message integrity and origin, Bob signs the message using his own private key. Alice refers to the Bobs public key for verifying the signature.

1.1.2 Identity-Based Cryptography

As per above discussion, PbKE requires Alice and Bob to have access to each others public keys. To overcome this essential and primary communication, the IBC system (invented by Shamir [4]) distributes a unique function $F(.)$ to all parties (i.e. a one-way hash function). As shown in (1.1), this function can be applied to each partys identity (ID) to obtain the partys public key. A party/entity ID can be e.g. party's email address, phone number, IP address, or a combination of them. PKG selects a random number s and calculates each party's private key, using (1.2), and provides it to the party via a secure channel.

$$PubK(ID) = F(ID) \tag{1.1}$$

$$PrvK(ID) = s \times PubK(ID) = s \times F(ID) \tag{1.2}$$

IBC from the weil pairing

Three well-known pairing characteristics are *bilinear*, *non-degenerate*, and *computable*. Let $\mathbb{G}_1$ be an additive group, $\mathbb{G}_2$ be a multiplicative group of a prime order q, and p be the group generator of $\mathbb{G}_1$. The discrete logarithm problem (DLP) for $\mathbb{G}_1$ and $\mathbb{G}_2$ is assumed to be enough hard. The

bilinear pairing map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ takes into account the following characteristics:

- *Bilinear:* $\hat{e}$ should have the following properties:

$$\hat{e}(q_1Q_1, q_2Q_2) = \hat{e}(Q_1, Q_2)^{q_1 q_2} \tag{1.3}$$
$$\hat{e}(Q_1 + Q_2, Q_3) = \hat{e}(Q_1, Q_3) \ . \ \hat{e}(Q_2, Q_3) \tag{1.4}$$
$$\hat{e}(Q_1, Q_2 + Q_3) = \hat{e}(Q_1, Q_2) \ . \ \hat{e}(Q_1, Q_3) \tag{1.5}$$
$$\forall \ Q_1, \ Q_2 \ \& \ Q_3 \in \mathbb{G}_1 \ \ \& \ \forall q_1 \ \& \ q_2 \ \in \ \mathbb{Z}_q^* \tag{1.6}$$

- *Non-degenerate:* $\hat{e}(P, P) \neq 1$, therefore P is a $\mathbb{G}_2$ generator.

- *Computable:* There is a competent algorithm to compute:

$$\hat{e}(Q1, Q2) \ subject \ to \ \forall \ Q_1 \ \& \ Q_2 \ \in \ \mathbb{G}_1$$

As an IBC solution, the followings are steps of developed system by Boneh and Franklin [5]:
Setup: Trent (PKG) chooses a secret value $s \in \mathbb{Z}_q^*$, calculates its own public key as $P_0 = s.P$, and makes P_0 accessible by public (including Alice and Bob). Subsequently, Trent picks three hash functions H_1, H_2 and H_3, and then forms system parameters set $\widehat{Parm}$:

$$H_1 \ \{0,1\}^* \rightarrow \mathbb{G}_1 \tag{1.7}$$
$$H_2 : \ \mathbb{G}_2 \rightarrow \{0,1\}^l, \ l = max(plain \ text) \tag{1.8}$$
$$H_3 : \ \mathbb{G}_2 \rightarrow \mathbb{Z}_q^* \tag{1.9}$$
$$\widehat{Parm} \ = \ \{\hat{e}, P, H_1, H_2, H_3, \mathbb{G}_1, \mathbb{G}_2\} \tag{1.10}$$

Note: The hash functions that we are referring to in our designs and proposed mechanisms can be any of the hash functions, e.g. SHA-1 [5]. Indeed, we only call hash functions in proposing our mechanism and use the characteristics of a hash function. Choosing the right and appropriate hash function depends on the application and system or environment that the mechanism is being implemented and used.

Alice and Bob will have access to set $\widehat{Parm}$, and at the same time, they are capable of obtaining the Trents public key (P_0). Also, for example, Alice applies H_1 to ID of Bob (ID_B) and extracts Bob's public key $PbK_B = H_1(ID_B)$).

Private key extraction: Trent calculates, for example, Alice's private key as $PrK_A = s.H_1(ID_A)$ and provides it only to Alice via a secure channel. Alice verifies her own private key as follows:

$$\begin{aligned} \hat{e}(PrK_A, P) &= \hat{e}(s.H_1(ID_A), P) \\ &= \hat{e}(H_1(ID_A), P)^s \\ &= \hat{e}(H_1(ID_A), s.P) \\ &= \hat{e}(H_1(ID_A), P_0) \end{aligned} \tag{1.11}$$

Encryption and Decryption: Let us consider the situation that Bob finds it necessary to sent message $M \in \{0,1\}^l$ to Alice. He calculates Alice's public key ($PbK_A = H_1(ID_A)$), chooses a random variable $r \in \mathbb{Z}_q^*$, and then calculates $U = r.P$, and $V = M \oplus H_2(\hat{e}(PbK_A, P_0)^r)$. Finally, he forwards $C = (U, V)$ to Alice as an encrypted message of M. Alice employs her own private key and decrypts the encrypted message M:

$$\begin{aligned} V \oplus H_2(\hat{e}(PrK_A, U)) &= V \oplus H_2(\hat{e}(s.PbK_A, r.P)) \\ &= V \oplus H_2(\hat{e}(PbK_A, P)^{rs}) \\ &= V \oplus H_2(\hat{e}(PbK_A, s.P)^r) \\ &= V \oplus H_2(\hat{e}(PbK_A, P_0)^r) \\ &= M \oplus H_2(\hat{e}(PbK_A, P_0)^r) \oplus H_2(\hat{e}(PbK_A, P_0)^r) \\ &= M \end{aligned} \tag{1.12}$$

Signature and Verification: Bob utilizes H_3 and computes $\sigma = H_3(M) \times PrK_B$ to be his signature for message M, and consigns σ along with the message M to Alice. To verify signature σ, Alice checks if $\hat{e}(H_3(M).P_0, H_1(ID_B)) = \hat{e}(P, \sigma)$ holds, where she follows:

$$\begin{aligned} \hat{e}(P, \sigma) &= \hat{e}(P, (H_3(M).PrK_B)) \\ &= \hat{e}(P, s.PbK_B)^{H_3(M)} \\ &= \hat{e}(P, PbK_B)^{s.H_3(M)} \\ &= \hat{e}(H_3(M).s.P, PbK_B) \\ &= \hat{e}(H_3(M).P_0, PbK_B) \\ &= \hat{e}(H_3(M).P_0, H_1(ID_B)) \end{aligned} \tag{1.13}$$

Key Refreshment: Trent selects a new secret value s, and also recalculates his own public key P_0 as well as the entire parties' private keys. He provides each entity's private key to the entity via the secure channel.

1.1.3 Elliptic Curve Cryptography

Due to the many benefits of elliptic curve cryptography (ECC) [6], it has been used in various environments [1], especially where there are resource constraints [7–11]. One of the most important benefits of the ECC is providing the same level of the security with a smaller key size. For instance, ECC with 160 and 512 bit keys provide the same level of security as D-H, RSA or ElGamal [12] cryptography with 1024 and 15360 bit keys, respectively, which is much closer to the PrKE system key size. Table 1.1 presents more values for comparison, in key sizes as well as certificate sizes (bits). In this table, we show a base for comparison, which the symmetric key size, which can be names as a target for the security level.

In addition to addressing the resource constraint issue, ECC is also beneficial in enabling an efficient protocol that supports current and future devices with various levels of technology, which is important in emerging SG systems. Generally, ECC is presented as an Elliptic Curve (EC) nodes/points (x, y) over $\mathbb{Z}_p$, via the following definition:

$$\begin{cases} y^2 \equiv x^3 + ax + b \bmod p \\ where: \quad (x, y) \in \mathbb{Z}_p \\ s.t.: \quad p > 3 \ (A\ large\ prime\ number) \\ \quad \& \quad a,\ b \in \mathbb{Z}_p \quad \& \quad 4a^3 + 27b^2 \not\equiv 0 \bmod p \end{cases}$$

The National Institute of Standards and Technology (NIST) issued an implementer's guide that specifies the EC Diffie-Hellman (ECDH) key-agreement schemes from NIST SP 800-56A, which aims at pair-wise key establishment schemes using discrete logarithm cryptography. The document specifies the ECs and domain parameters, key generation methods, the ECDH primitive, key derivation function, and other auxiliary functions that are necessary for ECDH scheme implementations to be in compliance with SP 800-56A and Suite B [1].

Table 1.1: NIST Guideline for Key and Certificate Size (bits) [1]

Symmetric Key Size (Security Level)	RSA and D-H Key Size	ECC Key Size	RSA Certificate Size	ECC Certificate Size
80	1024	160	2048	193
112	2048	224	4096	225
128	3072	256	6144	257
192	7680	384	15360	385
256	15360	521	30720	522

1.1.4 Review Some of the Definitions and Attacks

Although various attacks identified already against the communication, we only review some of them that are more relevant to the authentication and/or key construction, along with the two properties, especially in the group key management, that are considered in this thesis.

Social engineering attack: The attacker gains access to the system secrecy and confidential information, such as the server administrator password, by somehow manipulating people who have access to that information.

Brute-force attack: Brute-force attack, or exhaustive key search, can be used against any encrypted data by systematically checking all possible keys until the correct key is found which may involve traversing the entire search space.

Replay attack: A valid data transmission is maliciously or fraudulently repeated or delayed, which can be carried out either by the originator or by an adversary who intercepts the data and retransmits it.

Denial of Service (DoS) attack: An adversary fires several request for a service in the system/network to overwhelm the service provider. Even though the requests may not be qualified to be delivered, receiving and performing the initial request can cause the entity to be over loaded.

Man-in-the-Middle (MITM) attack: For example, in an open key construction session, an adversary makes individual connections with the victims and then relays and controls messages between them, where they believe that they are communicating directly to each other over a private connection. In this case, the adversary can intercept all messages and inject new ones.

Dictionary attack: Dictionary attack is similar to the brute-force attack. By preparing a list of possible values (for the key), by guessing or analyzing the information that the victim may refer to for choosing their key/password. In fact, this attack aims at making the search space smaller than the brute-force one. There are two models of the dictionary attack, based on the way an adversary performs it, such as on-line and off-line dictionary attacks.

Unknown key share attack: When a key K in constructed between two parties, Alice and Bob, Alice believes K is shared by Bob; however, Bob believes K is shared between Bob and somebody else.

Denning-Sacco attack: If an intruder somehow finds a symmetric key used in the authentication scheme, the intruder can find the origin data that the found symmetric key is made by, such as an initial shared password between the parties.

Key privacy and insider attack: For example, in a key construction protocol, the middle nodes/parties that were in charge of relaying the messages between two parties will gain access to the private key of the parties by performing this attack.

Ephemeral key compromise impersonation attack: In apposite site of the "Unknown key share attack", if an adversary performs an off-line dictionary attack, brute-force, or even social engineering attack and obtains the initial password between parties, by performing this attack, the adversary will find the final constructed key, even after the fact.

Forward and backward secrecies: Forward secrecy refers to means a new entity that joins a group should not gain access to the past information. On the other hand, if a member of the group leaves, they should not gain access to the future information, which is called backwards secrecy.

1.1.5 Authentication and Key Management

By definition, authentication means binding an ID to a subject or principal. This can be accomplished by showing what the subject:

(i) is capable of doing, e.g., performing a digital signature, or

(ii) knows, e.g., a password, or

(iii) possesses, e.g., a smart card, or

(iv) has biometrically, e.g., fingerprints

Moreover, in a networking environment, nodes should follow a mutual authentication to establish a certain level of trust [1]. Then, parties need to set-up a secure communication channel, normally by employing a security key, to protect their data from accessing by unauthorized parties. Hence, the proposed mechanisms in this area normally come as a tailored solution that authenticates the parties followed by constructing a key and required key management.

In 2009, the IEEE *1363.2* standard [13] for password based public key cryptographic techniques was released. The standard specifies primitives and schemes designed to utilize passwords and other low-grade secrets as a basis for securing electronic transactions. To be more precise, the standard specifies the schemes for password-authenticated key agreement and password-authenticated key retrieval.

Following are three well-known mechanisms that are treated in the literature as the main references in the authentication and key management.

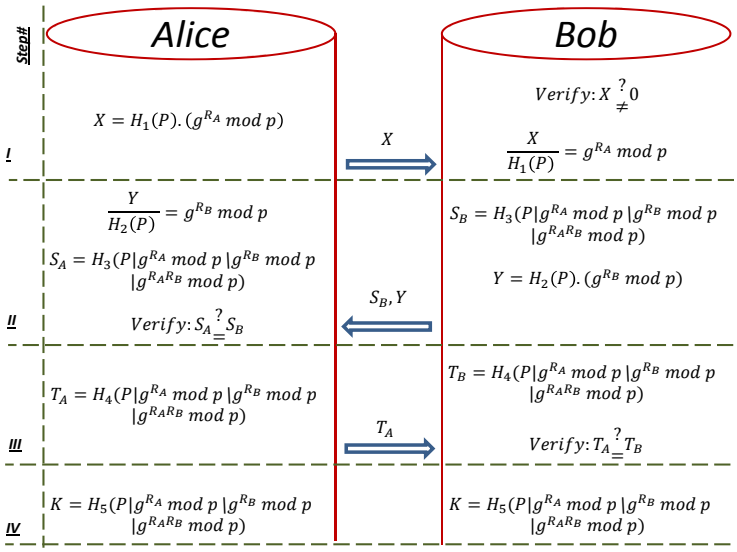


Figure 1.2: PAKE Protocol: X.1035 Standard

X.1035 standard (password authenticated key exchange - PAKE)

The PAKE protocol presented in the *X.1035* standard [14] presumes that two parties share a password pw. The four-phase mutual authentication protocol defined in the *X.1035* standard constructs a symmetric cryptographic key via D-H exchange by employing D-H values g *and* p and five shared hash functions $H_1 - H_5$. Depicted in Figure 1.2, in the following phases, ID_A *and* ID_B are the IDs of two parties named Alice and Bob, respectively, $P = (ID_A|ID_B|pw)$, and R_A & R_B are the respective random numbers chosen by them:

Step I Alice obtains X via (1.14) and forwards it to Bob:

$$X = H_1(P).(g^{R_A} \bmod p) \tag{1.14}$$

On the other side, Bob extracts "$g^{R_A} \bmod p$" from X by (1.15):

$$\frac{X}{H_1(P)} = \frac{H_1(P).(g^{R_A} \bmod p)}{H_1(P)} = g^{R_A} \bmod p \tag{1.15}$$

Step II Bob computes Y *and* S_B following (1.16) and (1.17), and sends them to Alice.

$$Y = H_2(P).(g^{R_B} \bmod p) \tag{1.16}$$

$$S_B = H_3(P|g^{R_A} \bmod p|g^{R_B} \bmod p|g^{R_A R_B} \bmod p) \tag{1.17}$$

Alice also similarly obtains "$g^{R_B} \bmod p$" from Y per (1.18) and then calculates S_A per (1.19) for the verification.

$$\frac{Y}{H_2(P)} = \frac{H_2(P).(g^{R_B} \bmod p)}{H_2(P)} = g^{R_B} \bmod p \tag{1.18}$$

$$S_A = H_3(P|g^{R_A} \bmod p|g^{R_B} \bmod p|g^{R_A R_B} \bmod p) \tag{1.19}$$

Step III Alice computes T_A via (1.20) and sends it to Bob.

$$T_A = H_4(P|g^{R_A} \bmod p|g^{R_B} \bmod p|g^{R_A R_B} \bmod p) \tag{1.20}$$

Then, Bob calculates T_B via (1.21) for the verification:

$$T_B = H_4(P|g^{R_A} \bmod p|g^{R_B} \bmod p|g^{R_A R_B} \bmod p) \tag{1.21}$$

Step IV The verification of S_A *and* S_B and T_A *and* T_B by Alice and Bob means a mutual authentication derived by pw. Using the above values, Alice and Bob can obtain the symmetric key K through (1.22):

$$K = H_5(P|g^{R_A} \bmod p|g^{R_B} \bmod p|g^{R_A R_B} \bmod p) \tag{1.22}$$

Secure remote password protocol

The Secure Remote Password (SRP) protocol [15] utilizes a predefined password and the verifier to construct a key, which delivers most of the characteristics that are expected from an authentication scheme. SRP is a fast mutual authentication scheme that uses the session key in the mechanism and resists the dictionary attacks. Furthermore, in the SRP protocol, compromising the server does not make it easy to find the password, compromising the password does not lead to revealing the past session keys (forward secrecy); and finally, compromising the session key does not lead to compromising of the password.

In SRP, depicted by Figure 1.3, the client initially enters a password and then the server computes a verifier from the password using a randomly

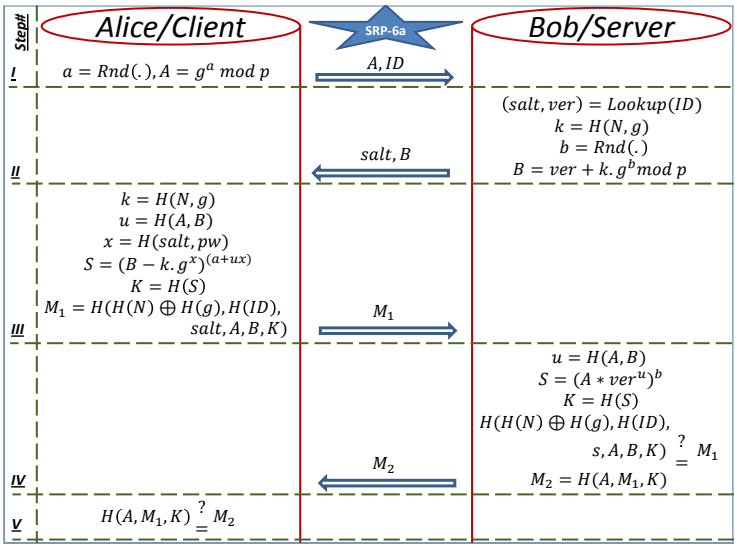


Figure 1.3: Secure Remote Password Protocol

generated salt and then stores the client's ID, salt and verifier in the server database. Subsequently, the client is authenticated to the server by providing the password to the server, which computes the verifier again using the salt stored against the client's ID and checking it against the one stored in its database. Furthermore, each party generates a random number, then calculates the session key based on the password, verifier and random numbers as well as verifies the key utilizing a one-way hash function.

SRP [15] (latest version 6a [16]), is an authentication and key-exchange protocol for secure password verification and session key generation over an insecure communication channel. SRP utilizes Asymmetric Key Exchange (AKE) [15], and stores verifiers instead of the passwords. AKE uses a one-way (hash) function to compute the verifier and stores it in the server. Therefore, compromising the server and finding the verifier is not enough to obtain the key, since the password is still required.

Burmester-Desmedt protocol

The "*conference key system*" proposed by Burmester and Desmedt [17], known as the BD protocol, is a protocol that addresses the symmetric key construction for a group of users. This protocol consists of three steps.

Consider n parties:

$$U_i, \qquad i = 1, 2, ..., n$$

forming a cyclic group such that:

$$U_{n+1} = U_1$$

I. Each member U_i generates a random value r_i, computes X_i via (1.23) and broadcasts it.

$$X_i \equiv g^{r_i} \bmod p \tag{1.23}$$

II. After receiving the broadcast values by others in the previous step, each member (U_i) calculates Y_i via (1.24), and broadcasts it:

$$Y_i \equiv (\frac{X_{i+1}}{X_{i-1}})^{r_i} \bmod p \tag{1.24}$$

III. Then assuming the values of the previous steps are received by all the members, each member (U_i) calculates the shared key (K_i) via:

$$K_i \equiv (X_{i-1})^{n.r_i}.Y_i^{n-1}.Y_{i+1}^{n-2}...Y_{i-2} \mod p \tag{1.25}$$

As can be seen from step III, the K_is of the nodes are the same, which is called the shared (group) key K.

1.2 Privacy Background

Due to the broadcast nature of wireless transmissions, an attacker can overhear the communication and detect valuable information that can compromise the privacy of the clients. Even if an attacker cannot decode packets or senders addresses due to packet encryption, they can correlate different amounts of traffic transmitted by a user at different times using a model of the users behaviour. Consequently, a well-defined privacy protection system is a preliminary demand in order to make SG ready for implementation [1, 18].

Steganography: Steganography, started in 15^{th} century, is a sub-division of the cryptography that deals with the privacy. It is a technique of communication that transfers the message embedded in a different object. For instance, during the cold-war, information was being transferred inside a person's eye in a picture, or inside the musical fonts, where nobody noticed

them except the sender and receiver. This technique is being used more in the intelligent services although the concept is deeply about hiding the information, and can be used in communication, particularly in VoIP [19].

Random path: Random path greatly reduces the chance of sources being identified. Even if an eavesdropper detects one packet of a sender, the next packet is unlikely to follow the same path, thus rendering the previous observation useless. Although the message delivery time using the random path is a longer than the minimum-hop approach, it is still acceptable if the enhanced privacy preserving capability of the random path approach is considered. To implement the scheme, a technique called phantom routing has emerged [20]. Although the scheme is robust, it involves a large overhead and may not withstand attacks under a collaborative adversary model. Privacy-aware parallel routing scheme is used to maximize the source traceback time [21]. Packets from the same source are routed over different paths to the destination, beside a weighted random stride routing to break the entire routing into strides. However, this scheme will not be effective in protecting source privacy in case of a global eavesdropping adversary.

1.2.1 Definition of Privacy

One of the most famous and original definitions of the privacy that has also been adopted by NIST [18] is "*The right to be left alone*". Bob Blakley defined it as "*The ability to lie about yourself and get away with it*" [22]. In this regard, Pfitzmann and Hansen provided six definitions in the privacy context [23]:

Anonymity: "*Anonymity of a subject means that the subject is not identifiable within a set of subjects, the anonymity set*". Anonymity, the most popular used term in the literature, aimed at making a party anonymous from others, even a peer, which can be defined as *Sender Anonymity* and *Receiver Anonymity*.

Unlinkability: Unlinkability means not being able to distinguish relationship between two items in a system. An item for instance can be a Smart Meter (SM), controller of a Home Area Network (HAN), Building Area Network (BAN) or Neighborhood Area Network (NAN), or aggregator.

Undetectability: Undetectability refers to the existence of an entity, application or process being agnostic to the eyes of an observer.

Unobservability: Unobservability means having characteristics of both anonymity and undetectability. Unobservability is applicable when there is a relationship among the players, e.g., sending and receiving.

Pseudonymity: "*A pseudonym is an identifier of a subject which is*

different from the subject's real name". Pseudonym can be defined as person pseudonym, role pseudonym, relationship pseudonym, role-relationship pseudonym, transaction pseudonym, respecting the relationship the holder.

Identity Management: Entities that follows pseudonymity approach have multiple identities, based on one or some attributes of the entity. Managing the identities in terms of assigning and controlling them in a way that makes the item unidentifiable by any unauthorized party, is the task of identity management.

1.2.2 Smart Grid Privacy Challenges

In 2010, NIST released the guideline to cyber security of SG. Volume 2 of the guideline addresses the privacy, and identifies four categories of the privacy:

Privacy of personal information

Personal information of a customer implies to name, address, phone number, and similar attributes that yield identifying the customer, directly or indirectly, and with or without combination by other attributes of the customer that are publicly available (e.g. roughly age or origin of nationality). It is a right of the customer to decide how and up to what extend her/his personal information is allowed to be shared by others, fully or partially [18].

Privacy of the person

Privacy of the person infers to the persons body situation and requirements, which yields health and physical aspects of the body. The health issues and body physical information as well as required treatments are part of the privacy of the person. Some of the smart medical devices are used in home that their existence and operation pattern can yield health status of the customer [18].

Privacy of personal behaviour

This category indicates a person right about his activities choices and keeps them in private. For instance, a customer may have multiple vendors to receive a specific service. So, it is the customer's right to keep this information and asks SG not to share them with others [18].

Privacy of personal communication

This category deals with control-free communication of a person with others. The control-free applies to undue surveillance, monitoring, or censorship. Since the communication between the customer and service provider is part of the SG context, it is the customers right to identify her/his level of communication, without any controls of SG [18].

1.3 Security Analysis

In most of our work, we follow one, or both of the following approaches.

1.3.1 Automated Validation of Internet Security Protocols and Application - AVISPA

In this thesis, Automated Validation of Internet Security Protocols and Application (AVISPA) [24] software is used to simulate and analyze the proposed mechanisms. AVISPA is a software tool for the automatic verification and analysis of Internet security protocols that is currently considered by the research community as one of the most trusted evaluation tools to analyze the ability of a scheme or protocol to withstand different attacks.

AVISPA integrates automatic security analysis and verification back-end servers like On-the-Fly Model-Checker (OFMC) and Constraint-Logic-based Attack Searcher (Cl-AtSe). First of all, the mechanisms under examination by AVISPA must be coded in the High Level Protocol Specifications Language (HLPSL) to be evaluated by the back-end servers. HLPSL is an expressive, role-based formal language used to describe the details of the protocol in question. Our HLPSL codes (see Appendix A) includes different sections used to model the roles of entities in a proposed solution, as well as the role of the environment and the security goals that have to be achieved. We normally start with the original model already existing in the AVISPA library, and then developed our HLPSL codes based on the proposed mechanism.

1.3.2 Adversary Model

The next tool to evaluate our mechanism is *Adversary model*. In our works, we consider Dolev-Yao model [25], which has different assumptions. The adversary controls the network completely, or even the adversary is the network itself. Therefore an adversary can record, delete, replay, reroute, reorder,

and completely control the scheduling of messages. Furthermore, the honest parties receive/send the packets/messages only from/to the adversary. In addition the adversary is capable of selecting the required additional information and receiver of the packets. Normally it is assumed that the adversary has information about other parties (their IDs) and also has access to their e.g. public keys. In some cases, the adversary has access to private keys of a few of parties, also and if required, is capable to generate a valid nonce. In case of the attacks like reply attack, the adversary can resend many times the same message. The adversary is capable to try decrypting the encrypted messages to perform attacks such as brute-force or dictionary attacks.

Our adversary models consists of four parts. (i) The first one shows the OBJECTIVE of the adversary in which we clearly mention in each case specifically what is/are the objectives. Since we want to analyze our proposed mechanisms, normally we set the model to attack the mechanism from different angles. (ii) Then we describe INITIAL CAPABILITIES of the adversary, for instance, having knowledge about the topology, or public keys, or our proposed mechanism. (iii) Then, we discuss CAPABILITIES DURING THE ATTACK of the adversary. For instance, and as per aforementioned Dolev-Yao model, the adversary can receive all messages and tries to decrypt them. (iv) Finally, we present the DISCUSSION part, to see if the adversary is capable of breaking our mechanism, and what the limits are.

In some cases we provide two adversary models, including *internal adversary* and *external adversary*. In case of external adversary, normally we assume that our adversary is not one of the system or network parties, and does not have any valid key. In fact, the adversary is attacking the system from outside. On the other side and in case of the internal adversary, we assume our adversary for instance has a full control on at least one of the system malicious parties. The malicious party has a valid e.g. private key to be able to communicate securely with other parties including the secure server (if there is one).

1.4 A Brief Introduction to Smart Grid

Achieving a successful implementation of a *High Tech Smart Grid (SG)* will provide benefits like improvement in asset management and planning in production and distribution side, enhancement in managing risk of outage, and improve cost efficiency in electricity consumption side. Providing a high level of security is one of the most important and challenging topics

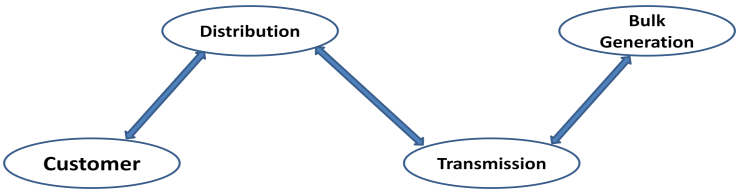


Figure 1.4: SG Power Bidirectional Flows

in the SG design, which has gained substantial attention in the research community [1].SG is a combination of different systems and sub-systems, and is vulnerable to various attacks that may cause different level of harms to the devices and even society-at-large [26]. Since SG is moving the power grid from a closed control system to one employing open IP networks [27], a variety of threats have been identified in the SG context, e.g., MITM, DoS, impersonation, which can affect the data integrity and authentication of users and devices. Moreover, different viruses or attacks such as brute-force and dictionary attacks can target the data security and confidentiality. The Stauxnet worm is another example that can cause a significant impact on even national security [27]. Once an entry point is found, an intruder or a malicious node may perform different action to compromise the whole system. Since millions of homes are connected to an SG, the impact of such

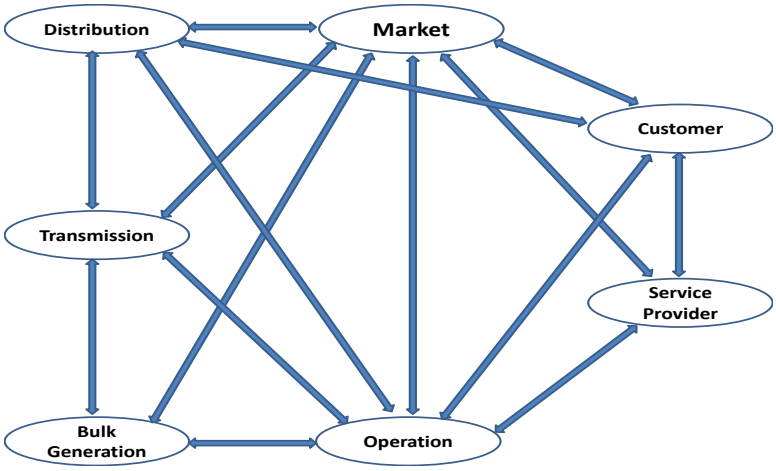


Figure 1.5: SG Data Bidirectional Flows

attacks can cause a significant loss or harm on society, e.g., by causing a blackout, changing the customer billing information, or changing the pricing information sent to the customers [26–28].

US Department of Energy (DOE) in Energy Data Book 2009 reported that residential, commercial and federal buildings use about 39% of U.S. primary energy consumption in 2006. Electricity is the main and fastest growing source of building energy consumed by about 74%. During last few years, attention to developing a new enhanced grid (SG) has been increased, to improve the power system efficiency, such as generation, transmission, distribution, consumption and billing. In a traditional model, there are two main flows: (i) Power flow from a provider to the customer, (ii) Data flow about the metering and for billing purposes which is from a customer to the provider. This data follow normally was/is in a long time bucket (monthly), and almost over an off-line communication. Furthermore, both mentioned flows are just unidirectional. Over years this model has changed and improved and then, Automated Meter Reader (AMR) system was developed which is mainly used in the developed areas.

In an AMR system, which its name shows as well, metering data communication from the customers are automated. Also, some off-line or even on-line information about the electricity price per week-day, and time of the day, is provided to the customers to manually make their decision about managing power usage in a better and more efficient way. However, some of the customers that may have power generation facility like Photovoltaic (PV), may have extra electricity and are willing to return (and sell) it back to the grid. In short, using the Advanced Metering Infrastructure (AMI) instead of the AMR system, target is providing a bidirectional flow of power (Figure 1.4) and data (Figure 1.5) [1].

SG has gained attention from different parties comprises Governments, Market and Energy Providers, Society and Research Community (Figure 1.6).

Government: SG for government is currently a national [1, 26, 29] and even an international project which requires all resources collaboration, especially in regards to standardizing [1, 26]. Most of the power energy providers use national resources like fossil fuel-powered generating plants [1, 29], which are hard or impossible to be replaced. Moreover, it is an interest for governments because of the global warming impact and emissions control [26] as well as society (businesses and individual) Security and Privacy.

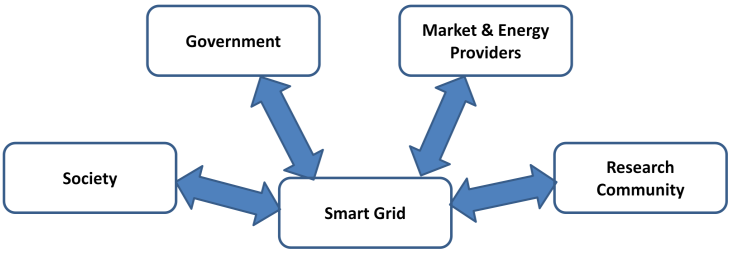


Figure 1.6: SG Involved Parties

Market and energy providers: They are involved to generate and transmit/distribute the energy in an efficient manner as well as manage/prevent the blackout risk. For instance, they need to use the consumers power usage and demand information in order to manage the system in operation side. So, they rely more on the customer data most likely in a live and on-time manner, and make the data security more critical for them [30].

Society: First of all, they have concern about national resources consumption. Secondly, they expect and require service delivery improvement on receiving it on-time and with a low cost [26]. To address the security, they want to make sure the new system is reliable enough and secure to be always available. Since personal and business information can be discovered from the electricity usage, they need their privacy to be fully maintained.

Research community: The whole SG project is still new and has many different sections that require research. One of the approaches is increasing use of ICT and new technology in SG to improve the power system from all aspects [1, 26]. As McDonal et al. mentioned "*it (Smart Grid) is a network of computers and power infrastructure that monitor and manage energy usage*". Having more information technology (IT) in serving power system demands more research [26]. Furthermore, it is the research community duty to design a secured and privacy-aware system to address other parties concerns.

1.4.1 SG System Structure

SG system is supposed to provide appropriate and on-time information. This information plays the main role in improving live planning and scheduling,

cost efficient production and transmission and distribution as well as asset management, for provider and distributor end; also using electricity in a cost efficient fashion for customer end. In this part, we review some of the SG topologies suggested by research community.

H. Gharavi et al. provided a Mesh Network Architecture model for the last mile SG. This architecture has two mesh based domain. First domain covers HAN including appliances and at least one Mesh-station with access point (MSAP). One of the MSAP (can be Smart Meter or SM) duties is to act as an Access Point (AP) for HAN mesh network. Second domain handles NAN mesh network that connects HANs domains to the AMI head-end via data aggregation point (DAP) and Mesh-Relay-Station if needed. As they mentioned, the role of the second mesh network is to expand the coverage area of the network by using multiple hops connection. Then, they proposed multiple paths connection between each home SM and DAP. This is one of the latest proposal in this area that we found it more practical solution. Using multi-hops model in NAN (from a meter to the aggregator) is reasonable; otherwise we need to have several collectors to cover our Metropolitan Area Network (MAN). Furthermore, they used AODV for NAN routing protocol and provided a solution for path selection in a NAN domain [31]. Some of the papers provided sub-sections per physical location e.g. HAN, BAN, and HAN. M. M. Fouda et al. proposed model for HAN is almost a star topology, and their NAN topology is a Mesh based structure. Their HAN model uses ZigBee and NAN uses WiMax connections including a base station to connect BAN/HAN gateways to the NAN gateway. Their next step is communication between the NAN gateway and Control Centres in Transmission Centres via local Distribution Centres [32].

NIST is in developing process of the SG required standards and guidelines. NIST followed two approaches of top-down and bottom-up, and developed required standards for interfaces between domains (coming from top-down view). They identified seven domains (Figure 1.5), 46 actors, 130 possible logical interfaces in 22 categories. They also mentioned 180 high-level security requirements in 19 groups.

- *Bulk Generation Domain:* This domain developed energy from distributed resources, which are usually connected to their local electrical loads. After answering local demand, extra energy flows into the grid through routers.

- *Transmission Domain:* It is responsible to transmit the energy from the generation sources to the consumers.

- *Distribution Domain:* Routers track the demands changes to adapt the energy distribution dynamically.
- *Operation Domain:* It collects grid status such as the current resources energy capacities and the current customers energy demands, in order to optimize grid operations.
- *Market Domain:* It gathers energy supply and demand information from grid to balance supply and demand.
- *Customer Domain:* Customers buys, and in case of generating energy from renewable resources, sells extra via grid to the service providers.
- *Service Provider Domain:* This domain roll is buy and sell the energy. They deals with customers as well as energy provider sources.

So far, NIST has covered cyber security strategy, logical architecture and interfaces, high-level security requirements, cryptography and key management, privacy, vulnerability classes, bottom-up security analysis, R&D themes for cyber security, overview of standards, key power system use cases for security requirements. Encryption of critical security parameters are under developments by NIST [1]. Some of the organizations that develop related standards are: IEC Technical Committee 57 WG 15; ISO/IEC 15408; ITU X.805; ANSI/ISA-99.00.01-2007; NIST 800 series (SP 800-82, SP 800-53); ANSI C12, IEEE 1402; NERC Critical Infrastructure Protection (CIP); European Energy Regulations CEER & ERGEG; Roadmap 2010-18 by EEGI; IEEE P2030; IEC Global Standards and several IETF request for comments (RFCs) [29, 32].

1.4.2 Review Systems and Applications

Like any other system, SG currently uses, and potentially will use, some applications and systems and subsystems. For instance, Supervisory Control And Data Acquisition (SCADA), Energy Management Systems (EMS), Building Energy Management System (BEMS), Micro-grid Management System (MMS), Distributed Intelligent Energy Management System (DIMES), Vehicle To Grid (V2G), Wind To Vehicle To Grid (W2V2G), Demand Side Management (DSM), Estate Estimator (ES), Automated Meter Reading (AMR), Advance Metering Infrastructure (AMI), Demand Response Management (DRM), Meter Data Management (MDM), Cloud-based Demand Response (CDR) [33], Automated Demand Response (ADR) [34], Feeder

Automation (FA) [34], Electric Vehicle (EV) [34], Mobile Work-flow Management (MWM) [34], Notification [35], Analytics [35] and Photovoltaic (PV). In all of these systems and application, data security and customers privacy should be addressed.

1.4.3 Pricing

One of the main advantage of using SG is giving opportunity to the customers to consume energy costs efficiently. There are two types of pricing techniques called Time of Use (TOU) and Real Time (RT) [36]. In TOU, the price is set in a long forecast fashion before the time of use, like monthly or even annually. Mainly, the historical data delivers a suggestion of the same demand, and future known demand and development project as well as other related data are utilized to have TOU. On the other hand, RT is an improved and more efficient technique for price delivery, an hour before usage for example. RT can only be delivered based on AMI technology.

Furthermore, supporting "what-if" simulations can be performed only in new SG and AMI. In a "what-if" simulation, a customer may need to see the effect of their decision before finalizing it. "What-if" simulation techniques can be used by market and service providers to efficiently invest or run the systems [35]. Price list and appropriate load balancing is one of the main subjects in the research community, in which the optimization technique plays a main role.

1.4.4 Review Smart Grid Communications for Outside Home (Except Customer Domain)

In this part, we review some of the wireless and wired technology mainly for outside of a home communications.

Power line communication (PLC)

In this technique, current power line infrastructure is used for communication as well, which supports wide access and low costs. It can be used to transfer metering information from SM to the concentrators in SG infrastructure. It is one of the initial solutions; there is quite a bit of literature in this area, and some implementation. However, this solution has some disadvantages, which mostly come from the nature of power lines. For example, a line is too noisy and bandwidth is low enough to increase the concern of some applications that required high rate [37]. Standard IEC 61334 has been developed to cover this communication [38].

Digital subscriber lines (DSL)

Using DSL in most of the areas has been experienced. It is an already implemented infrastructure that uses wired phone network for communication. This solution provides a low cost and high bandwidth in a wide area that makes it an interesting solution for most of the current project. Although its disadvantage is the potential line down time and lake of required standard and distance dependency. It also requires installing the wired network in rural area, which forces line maintenance that increase the cost of solution [37].

Fiber optic

Using fiber optic as the main network back bone has been around for many years, which provides a very high bandwidth and reliable communication that addresses most of the application requirement. However, it requires installation in rural areas and maintenance, which makes it a costly solution and techniques for SG [37]. It can be used in part of the SG network like inside bulk generator domain, but may not be a good solution for all of the sections.

Wireless lan

Wireless LAN has been used for a long time now and IEEE 802.11 standard based model are studied, improved and developed during last decades. NIST also recognized IEC 61850 for SG, which proposed Ethernet based communication. Currently, both can provide a good and reliable basis for communication in SG as per application requirement. Wireless can be used in SG and is able to provide different specifications: (i) Enhanced transformer differential protection, (ii) Redundant link for distribution automation system, (iii) Communication aided line protection, and (iv) Control and monitoring of remote DERs [37, 39].

WiMAX

WiMAX technology as part of IEEE 802.16 has been developed for MAN, which delivers a high bandwidth high distance coverage. The drawback of using WiMAX is its high implementation cost, since it requires WiMAX tower infrastructure. Literature recommends using this technology for communication between smart meters and the utility network. It can support

real time pricing since, automated meter reading, and outage detection and responses since has a high speed communication [39].

Cellular

The 3G/4G technology is the next suggestion for the SG communication, mainly for outside HAN up to utility station. The required infrastructure for cellular network is already implemented and can cover most the areas; however, bandwidth and channel speed, channel security, call drop, and connection cost are concerning. For instance, this technology has been suggested to be used for SCADA interference for remote distribution substation and monitoring and metering of remote DERs [39].

MBWA

Mobile Broadband Wireless Access (MBWA) or MobileFi based on IEEE 802.20, is a new technology that provides high speed bandwidth as well as supporting high mobility. For some of the SG application, such as plug-in electric vehicles, wireless backhaul for SG monitoring, and SCADA systems, this technology has been suggested by literature [39].

Digital microwave

This technology can support point to point communication for SG applications with a very long distance coverage (up to 60KM). For instance, it can be used for transfer trips between DER and distribution substation feeder protection relay. It is capable of receiving data from Ethernet or ATM and transmitting it as microwave radio, although it is vulnerable to interference and signal fading [39].

1.4.5 Review Smart Grid Communications for Inside Home (Customer Domain)

As one of the main seven domains of the SG system is the customer domain, we review some of the technologies that are proposed for communication inside a HAN (or customer domain).

Bluetooth

Bluetooth is part of IEEE 802.15.1 that is being used in HAN devices. It can cover from 1m and up to 100m distance communication. Bluetooth has

all seven layers of OSI communication stack, although it does have a strong security since has been designed to be a light weight technology [39].

INSTEON

INSTEON can cover up to 45m and up to 4 hops in a mesh based model. It uses the time slot synchronization scheme concept and nodes are allowed to transmit in certain time slots to avoid the collision. Devices can send or receive, and they relay a received packet as long as not being the destination. It can handle unicast, multicast, and broadcast communications. To handle end-to-end reliability, INSTEON uses acknowledge (ACK) and NAK and for security, can encrypt the messages. Although INSTEON specification is not publicly available, it is an easy technology to be implemented [40].

Wavenis

Wavenis, which has physical, link and network layers, is designed to provide indoor and outdoor services. It covers up to 200m for indoor usages and up to 1000m for outdoors needs. It uses a TDMA mechanism for synchronized communication combining with carrier sense algorithm; it also uses CSMA/CA for non-synchronized scheme. Device connection is based on required Quality of Service (QoS) defined by the node in time of connecting to the network. Nodes do not relay the packets and only communicate with the root. Also, Wavenis uses several algorithm such as 3DES and 128 bits AES encryptions for security. Currently millions of devices are produced using this technology, although its specification are not publicly accessible [40].

Z-Wave

Z-Wave is a light weight technology designed for HAN and offices, has five layers such as physical, MAC, transfer, routing and application. It can cover up to 30m for indoor and up to 100m for outdoor communications. It is mainly designed to handle controlling command, and devices can play two rolls of slave or controller. It can support up to 4 hops in a source routing based, and can handle unicast, multicast, and broadcast communications. The mechanism used in MAC is CSMA/CA, ACK is used to provide end-to-end reliability, and for security, it uses 128 bits AES encryption [40].

ZigBee

The best known technology in this area is ZigBee. It was designed for short range and low data rate application. It covers from 10-100m distance, and based on topology, can handles 5, 10, or 30 hops. ZigBee follows IEEE 802.15.4 standard and includes physical, MAC, network, and application layers. Nodes in a ZigBee can be coordinator, router and end device, and covered communications are unicast, multicast (application and network layers), unicast and indirect accessing. MAC covers two access model such as beacon-enabled assuming existence of a coordinator which generates beacon, and beacon-less that utilizes CSMA/CA. ZigBee technical information is publicly accessible, and millions of devices are being produced based on ZigBee or are planned to be produced for HAN (plan is about 30 millions in north America). ZigBee handles up to 127 bytes packet size and, in terms of reliability, ZigBee takes advantage of ACK and duplicate packet control. For security, ZigBee supports integrity, confidentiality, access control and key managemen [40].

6LoWPAN

6LoWPAN mainly takes advantages of ZigBee and IEEE 802.15.4 plus uses IPv6 technology. It is a light weight design for HAN compared to the original IPv6 and covers IP, TCP and application layers as well. The beauty of it is its compatibility with internet commutation compared to ZigBee. Similar to ZigBee, 6LoWPAN specification is publicly accessible and in most of the features are similar or an improved version over ZiGbee. For instance, it can cover between 10-100m (same as ZigBee), and supports unicast, multicast, broadcast, and IPv6 anycast. Devices can be edge router, mesh node, router and host, with a maximum of 255 hops. The packet sizes can be up to 127 bytes and uses TCP or UDP to provide reliability. In terms of security, it handles integrity, confidentiality, access control; however, key management is not yet supported [40].

According to our study and the amount of efforts that are putting in place to develop 6LoWPAN, there is a high chance for the adoption of this protocol, although current Smart Objects constraints may not allow the use of IPv6 at least yet.

Others

Some of the other HAN technologies are HomePlug [38], HomePlug Green PHY, PRIME and G3-PLC [37].

1.4.6 Standards

In this part, we briefly review some the standards discussed by literatures [37, 38].

IEC 61850 and UCA 2.0

This standard is initially developed for intra-substation communication and can also be used for metering application, as well as based on IEC 62445, for the communication between central controllers and substations.

IEC 62056-21 / IEC 61107

It has been developed to describe software protocols and hardware suitable for data exchange with utility meters, which is widely used today.

IEC 62056-31

This standard is designed for remote and local meter reading, which support of nearly 10 million already implemented devices in Europe.

SML

It is a communication protocol for data acquisition, which was designed to be a simple and suitable for low power embedded devices.

BACnet

Building and Automation Control Networking (BACnet) is a system for HAN applications such as HVAC, security and lighting, as well as for communication of external application to the HAN system.

1.4.7 Smart Grid Security and Privacy

In genera, and like any other system, the main challenges of security including availability, integrity and confidentiality should be addressed by SG. Furthermore, privacy is the fourth SG security concern [26]. In this part, we study only IT domain security, includes IT-networks, IT-infrastructures, computers, applications and related peripherals. Here we study some of the security and privacy related works in the literature. More is provided in each chapter accordingly to the subject of the chapter.

SG security conceptual design

AMI needs two way communications versus one way communication of AMR system, although security and privacy should be addressed in all of them [29]. T. Zhen et al. presented a framework for information security based on national (China) and international (Asian, US and European) standards. Their model is a closed loop includes three layers of Strategy, Management and Technology [41]. Being a closed loop model expected to make the system active and dynamic and improving the security system. In overall design, this model has six blocks including Security Business, Security Management, Application Security, Data Security, Infrastructure Security and Security Technical Measures.

W. Yan-liang et al. proposed an architecture security model based on the cloud computing and cloud security. Main assumption in this model is using cloud concept model to maintain power system information security as the main target. This model architecture has two sections of service and client [42]. R. Zhung et al. proposed three layers structuring of Device Layer, Network Layer and Service Layer for the SG. In service layer, they defined a metric named service security to evaluate potential network failure on the power system. This metric is based on risk factors in availability, integrity and confidentiality for security (Privacy in not addressed) [43].

Another conceptual framework is proposed by D. Wei et al. that is based on having three layers system structure including power, automation and control and security. This model has three main components such as security agent, managed security switch and security manager, and has been designed to be a multi-layers IDS. They also mentioned that patches regarding new detections should be transferred to the system via the public communication, which could be unsafe/unsecured for the security related information [44].

R. Zerbst et al. proposed a zone principal based on Defense-in-Depth approach, which is a computer standard, for security control principle. This model has six zones: process, critical automation/basis control, critical operation control/supervisory control, operation support/management, business automation/logistics and external partner/connections. This model emphasized on addressing international, national, regional, and other related standard based on these zones [45].

SG security and privacy detail design

The main concern of M. M. Fouda et al. is attack in link-layer in ZigBee for HAN, which covered the HANId conflict. In fact, HAN coordinator uses

message coming from the appliance that has received two messages from two sources with the same HANId, and wants to detect the attack. In such a case, the coordinator selects a new HANId and sends it to all of its own nodes [32]. Finding the issue by appliances may not be possible, because appliances receive the packet and HANId that are coming from the sources, and are not able to recognize they are from more than one source. Secondly, if HANId is incremental, two HAN coordinators that their HANId were the same may choose the same HANId again, unless they select it in a pure random model.

Z. Lu et al. described Denial-of-Service (DoS) attack as one of the attacks in the SG system. They introduced an index based on dividing *traffic flooded by attacker* to the *total channel bandwidth* to study system behaviour. Study shows: Increasing *intensity index* affects delay performance very much unless it gets close to one. In such a case, detection becomes a high risk. Decreasing packet size in Distributed Network Protocol (DNP3) causes this protocol more robust to the DoS attack. Making smaller packet size will cause more overhead, which is a delay root cause [46].

Y. Wang et al. mentioned the most of the SG characteristics (in HAN domain) would be similar to the Wireless Sensor Network (WSN). They suggested most of the developments such as design in security can be transferred to the SG. They prepared a list of differences including: deployment topology, data processing, energy less sensitive, remote maintenance and configuration, harsh environment conditions, reliability and latency Quality-of-Service (QoS) requirements, and high security requirement. Then, they defined required security features of the WSN for the SG [47].

The next idea uses certificate to secure data transmission between parties. It is to maintain the data communication Security and Privacy from an SM to the utility in high rate (every few minute) and low rate (every week or month). Based on high-frequency ID and low frequency ID of each SM, two parts of ID profiles named personally identifiable SM and anonymous SM are introduced in this model, which are used to create Client Data Profile and Anonymous Data Profile. This model focused on implementation of such a service and used nonce, shared certification authority), electrical signature and time stamp procedure [48]. This is a costly solution and may need modification per data required security.

E. Aydey et al. proposed an authentication mechanism for the SG HAN section that covers three communication sections such as gateway-SM, smart appliances-HAN, and transient devices-HAN. They showed that their mechanism has a low overhead and is good for the SG devices with resource constraint. However, they assumed all of the devices have pair-wise key

with a trust center. After sending authentication request to the trust center, this center creates and sends to the entire nodes one key per every two devices communication link. To be more precise, one key between the SM and the gateway, and one key per appliance per connection to the SM and the gateway. Also, cloud should keep all of these keys. This study requires each node to have multiple keys only for authentication purposes. If a Transit Device temporary visits a HAN other than its own HAN, new keys are required as well[49].

1.5 Our Contribution

The above discussion is presented as an initial introduction to the security and privacy challenges and research in this area. More discussion is delivered at the beginning of each chapter to study about focus of the chapter. However, and as part of introduction, as it is shown in Figure 1.5, there are different communications in the SG system, and at the different levels and sub-systems. There are enough study in the literature about the requirements, and in this thesis, we only touched a few problems aligned with the NSERC research project that has funded this thesis.

One of the main ingredients that the SG system works based on is live data about the power consumptions, actual and/or planned, that needs to be collected. One side of these communications can be smart appliances inside the homes, and other side can be up to the server located in the utility network. We address the security of these communications by providing our efficient authentication and key management mechanism.

In the SG system, collaboration between customers (or between the small power providers) in efficient electric power consumption (or provisioning) are required. They need to securely communicate to each other as part of a group. We developed an efficient group key management that address the required secure group communications. In some situations, for instance in case of an emergency, or any other similar situation, smart appliances inside a home needs to be controlled by SG controllers. These controllers can be located at the HAN or SG central controller unit, or in between. The security of these controlling commands, and addressing the hierarchy authority of the controllers is our next contribution.

Preserving consumers' privacy in the SG data communication is one of the key point to have the SG system ready and being accepted by the customers. Referring to our previous discussion, the users privacy has different aspects and points of view. Electric Vehicles are one of the SG system

components, which use as well as can carry the electric power and acts as a mobile storage in the SG context. However, one of the users concern is their privacy that can be jeopardize by tracing of contact points of the users electric vehicles to the grid.

1.6 Road Map

The first part of the thesis, including the first three chapter, deals with the security mechanism. Chapter 2 presents our authentication and key management mechanisms for smart appliances and home gateway, e.g. smart meter, as well as from the smart meter to the SG server located in the utility network, via NAN aggregators. We propose our efficient group key mechanism in Chapter 3. Our cluster based group key management can be used by a group of consumers on efficient power consumptions; or a group of small supplier on efficient power provisioning. Our multilayer consensus password authenticated key exchange mechanism, which is based on elliptic curve cryptography, is presented in Chapter 4.

Then in the second part of the thesis, including five and six chapters, we focus on users' privacy. In Chapter 5, we present our privacy preserving mechanism for data communication in the SG network, which utilizes enhanced network coding. Then, we concentrate in the electric vehicle communicating with the smart grid, via third party entities, such as power stations. We provide a privacy-aware security solution in Chapter 6 as our last part of the thesis.

As above steps shows, we concentrate more on customer domain of the seven SG domains (Figure 1.5), and customer relevant communication (from and to a HAN). Although our group key management presented in Chapter 3 can be implemented for, and used by, a group of providers too, again its main focus is customers. We try to look at the SG system from a customer point of view, and even overs EV as a customer domain element.

Chapter 2

Efficient Authentication Schema and Key Management Protocol

In this chapter, concentration is on efficiency of authentication schema and key management protocol, which are normally tailored together. As our case study, we presents our solution in the SG context. An efficient scheme is proposed that mutually authenticates an SM of a HAN and an authentication server in SG by utilizing an initial password, by decreasing the number of steps in the SRP protocol from five to three, and number of exchanged packets from four to three. Furthermore, we propose an efficient key management protocol based on our Enhanced Identity Based Cryptography (EIBC) [1] for secure SG communications using Public Key Infrastructure (PKI). Our proposed mechanisms are capable of preventing various attacks while reducing the management overhead. The improved efficiency for key management is realized by periodically refreshing all public/private key pairs as well as any multicast keys in all the nodes using only one newly generated function broadcast by the key generator entity.

2.1 Introduction

NIST suggests using PKI to secure SG communications [1]. PKI [50] (and PKE) is briefly reviewed in Chapter 1. Our proposal facilitates secure and efficient authentication and key management on top of PKI.

The customer's side of an SG consists of HANs in customer premises where smart appliances and controllers are connected to SMs, which form the end-points of the AMI that provides two-way data communications between SMs and the utility's Meter Data Management Center. This work is focused on authentication and key management over the AMI. The AMI

[1] H. Nicanfar and V.C.M. Leung, "EIBC: Enhanced Identity-Based Cryptography, a Conceptual Design", in Proc. IEEE SysCon, Vancouver, BC, Mar. 2012

will likely employ Internet Protocol version 6 (IPv6) technology in a mesh based topology [28]. Although PLC has gained much attention in Europe, in North America WMN is a more popular and dominant solution for the AMI [27].

In [51], a verifier is utilized for key establishment, with the support of a server as a trusted third party. Each party has an individual password and the server holds the appropriate verifier. The entities establish temporary session keys used to construct the final symmetric key in a protocol with four phases.

Contributions: In this chapter we propose a secure and efficient SG Mutual Authentication (SGMA) scheme and SG Key Management (SGKM) protocol. SGMA provides efficient mutual authentication between SMs and the security and authentication server (SAS) in the SG using passwords; it reduces the number of steps in SRP from five to three and the number of exchanged packets from four to three. SGKM provides an efficient key management protocol for SG communications using PKI as specified by NIST [1]; it employs our proposed EIBC scheme to substantially reduce the overhead of key renewals.

Security analysis shows that these schemes are capable of preventing various well-known attacks such as Brute-force, Replay, MITM and DoS. Furthermore, we reduce the network overhead caused by the control packets for key management. The improved efficiency results from our key refreshment protocol in which the SAS periodically broadcasts a new key generation to refresh the public/private key pairs of all the nodes as well as any required multicast security keys.

2.2 Related Works

2.2.1 EIBC: Enhanced Identity-Based Cryptography

Our proposed EIBC [52] enhances IBC by making the private key refreshment more efficient and accommodating distribution and refreshment of any multicast key needed in the network. The modifications to IBC are described as follows.

One-way/Hash function $F(.)$

The static function $F(.)$ in IBC is made dynamic in EIBC as function $F_i(.)$. Precisely, PKG periodically generates and broadcasts function $f_i(.)$ that is applied to $F_i(.)$ to obtain $F_{i+1}(.)$, which is the new one-way function of

the system. In this case, all of the public keys and private keys are being updated. Each party updates the public key of any other party by applying $f_i(.)$ to the current public key of that party. Also, each party uses $f_i(.)$ in the private key refreshment algorithm that will be explained shortly. The index "i" represents the current state (called *live* in this chapter) of the system.

$$\begin{cases} F_{i+1}(.) = f_{i+1}(F_i(.)) & (2.1a) \\ PubK_i(ID) = F_i(ID) & (2.1b) \end{cases}$$

System secret value s

In IBC, s is the product of a True Random Number Generator (TRNG) managed and kept secret by PKG. In EIBC, s is replaced by two values: s_i from (2.2a) is a non-shared TRNG value kept by PKG, and $\widetilde{s}_i$ is obtained from (2.2b) using a Pseudo Random Number Generator (PRNG) with parameters a, b and modulus q, shared by all entities.

$$\begin{cases} s_{i+1} = f_{i+1}(s_i) & (2.2a) \\ \widetilde{s}_{i+1} \equiv (a * \widetilde{s}_i + b) \bmod q & (2.2b) \\ s.t. : \quad i, a, b, q \in \mathbb{Z} \ \& \ \widetilde{s}_i \in \mathbb{Z}_q^* \end{cases}$$

Seed and end values

In EIBC, some of the parameters have a Seed Value (SV) as well as an End Value (NV). For instance, PKG has "public key SV" ($\widetilde{PubK}^i_{PKG}$) and "public key NV" ($PubK^i_{PKG}$). Moreover, each entity has a private key SV ($\widetilde{PrvK}^i_A$) and a private key NV ($PrvK^i_A$). PKG produces SVs of the keys via (2.3a) and via (2.3b), and all entities perform (2.4a) and (2.4b) to obtain the live NVs:

$$Seed\ Values : \begin{cases} \widetilde{PubK}^i_{PKG} = s_i.\breve{P}^i_{PKG} & (2.3a) \\ \widetilde{PrvK}^i_A = s_i.F_i(ID_A) & (2.3b) \end{cases}$$

$$End\ Values : \begin{cases} PubK^i_{PKG} = f_i(\widetilde{s}_i).\widetilde{PubK}^i_{PKG} & (2.4a) \\ PrvK^i_A = f_i(\widetilde{s}_i).\widetilde{PrvK}^i_A & (2.4b) \end{cases}$$

Key refreshment periods

In EIBC, there are different values that need to be updated or refreshed from time to time, including $f_i(.)$, s_i, $\widetilde{s}_i$, and the PRNG parameters "a & b". EIBC employs three timers for Short, Medium and Long Term Refreshments (STR, MTR and LTR) for the refreshment of these parameters.

STR process PKG generates a new function $f_{i+1}(.)$ and makes it publicly accessible, along with a VT, which is the start time of moving to a new live ($i \rightarrow i+1$). At the time of VT, each party refreshes $\widetilde{s}_i$ following (2.2b), updates $F_i(.)$ via (2.1a) in order to have refreshed public keys of others. Also, the party refreshes the public key of PKG as per (2.5a) and (2.5b), as well as its own private key based on (2.5c) and (2.5d), utilizing the updated values of $\widetilde{s}_{i+1}$ and $F_{i+1}(.)$:

$$\begin{cases} \widetilde{PubK}_{PKG}^{i+1} = f_{i+1}(\widetilde{PubK}_{PKG}^{i}) & \text{(2.5a)} \\ PubK_{PKG}^{i+1} = f_{i+1}(\widetilde{s}_{i+1}).\widetilde{PubK}_{PKG}^{i+1} & \text{(2.5b)} \\ \widetilde{PrvK}_{A}^{i+1} = f_{i+1}(\widetilde{PrK}_{A}^{i}) & \text{(2.5c)} \\ PrvK_{A}^{i+1} = f_{i+1}(\widetilde{s}_{i+1}).\widetilde{PrvK}_{A}^{i+1} & \text{(2.5d)} \end{cases}$$

MTR process PKG renews the PRNG parameters "a & b" along with the required VT, and shares them with all the parties to be used starting at VT.

LTR process PKG reselects the system non-shared secret values, along with the system shared secret values, and updates one-way function $F_i(.)$, in order to refresh all the keys, i.e., public and private keys of all parties. PKG also updates the private key of each party, and informs the party along with a VT via the secure channel.

Note that the LTR process is similar to the IBC key refreshment process. As it has been analyzed in the [52], EIBC simultaneously improves key management process overhead cost and system security level.

Multicast group key support

To support secure multicasting, EIBC incorporates two mechanisms to manage the *multicast group source/receiver key pair*. Each multicast group is identified by a Multicast Group ID (MID), which is used similar to ID of an entity, to obtain Source Multicast Key (SMK) of the group via (1.1). At

the same time each group has a Receiver Multicast Key (RMK) managed by SAS and obtained via (2.3b) and (2.4b). Each Multicast Group Source (MGS) entity receives the group's SMK and RMK, and grants membership to a Multicast Group Receiver (MGR) entity by sending RMK to the new MGR. So, MGS encrypts the messages by SMK, and a MGR uses RMK to decrypt the messages. In order to authenticate the source of a multicast packet and because a SMK can be compromised, MGS signs the messages using its own entity (original) private key ($PrvK^i_{ID}$).

Furthermore, EIBC generates $\widetilde{m}_i$, similar to $\widetilde{s}_i$, using a Muticast Group Pseudo Random Number Generator with its own setup values "c & d" and initial value $\widetilde{m}_0$. Receivers use $\widetilde{m}_i$ to refresh RMK.

2.2.2 SG Security Schemes in the Literature

The security scheme in [53] is aimed at data transfer via the PLC technology for SG communications. In this mechanism, the manufacturer of any device, e.g., meter, modem or aggregator, should obtain a certificate for the device from the SG security server following the PKI approach, and embeds it in the device. Then, each node/device utilizes its own public/private key pair to construct a shared symmetric key with the next node. In this system, the SG security server is involved in authentications of all the nodes in each stage of the mechanism, which can be a heavy workload in the SG environment. Another concern about this proposal is the assumption that all the manufacturers of the devices are fully trusted parties. Also, the shared symmetric key is chosen by one node and transferred to the peer encrypted with the public key of the peer. Therefore, the proposed mechanism is vulnerable to attacks, e.g., by malicious nodes that have obtained a certificate illegally, or devices from a rogue manufacturer.

The use of symmetric keys for SG security is proposed in [54, 55], the former based on the D-H algorithm, and the latter based on the elliptic curve approach of the D-H algorithm; both adds a key verification step to the pairwise key construction. Use of symmetric keys is vulnerable to MITM attacks, despite the verification phase. Furthermore, using symmetric keys for communications over the entire SG system is not scalable due to the large number of devices and nodes. Consequently, PKI is recommended in [1] to secure SG communications.

In order to decrease the cost of key distribution, the proposal in [56] requires all packets to be transferred through a server. Each source encrypts its packet with the public key of the server and sends it to the server. Then, the server uses its private key to decrypt the packet, and uses the public

key of the destination to re-encrypt the packet and sends it to the destination, e.g., a service provider. In an SG, this mechanism causes a very high demand on the server to handle the decryption and re-encryption of packets and on the network to route each packet via the server. Thus, the cost of key distribution is lowered at the cost of a highly loaded server and increased data packet communication load. Furthermore, this method does not preserve confidentiality of the packets since all packets are decrypted by the server, which is not the end receiver. The mechanism presented in [57] is also vulnerable to the MITM attack, although the authors mentioned that it is safe against this attack. For instance an authenticated malicious node can perform the MITM attack. This scheme requires two hash functions, and needs a third party in the key construction process, in initializing the key construction as well as the key verification.

Using IBC to secure vehicle-to-grid communications over SG is proposed in [58]. The authors mainly focused on the key management, and they provide a one-way authentication for authenticating the vehicles to the grid. Using biometrics is proposed in [59] for the authentication of users in SG. The author suggested that their proposal addresses the user privacy issue in SG communications [59], although the need to collect users' fingerprint information can raise overall user privacy concerns.

Authors of [60] studied the approaches of having a Unified Key Management Function (UKMF) and Dedicated Key Management Functions (DKMF) or a hybrid of the two for different applications in SG. They showed that using UKMF is more efficient, and furthermore, they suggested an Extensible Authentication Protocol based mechanism to be used in SG.

Our work is built on top of PKI, the preferred method to secure SG communications, and provides secure and efficient mechanisms for initial authentication and key generations and updates.

2.3 Smart Grid Mutual Authentication

2.3.1 System Setup

We concentrate on data communications over the AMI outside of the HAN domain, which includes an SAS that is charged with supporting the required authentication and key management mechanisms. We also cover the key management for unicast, multicast and broadcast communications that may be needed to support any application over SG. Our assumptions are as follows:

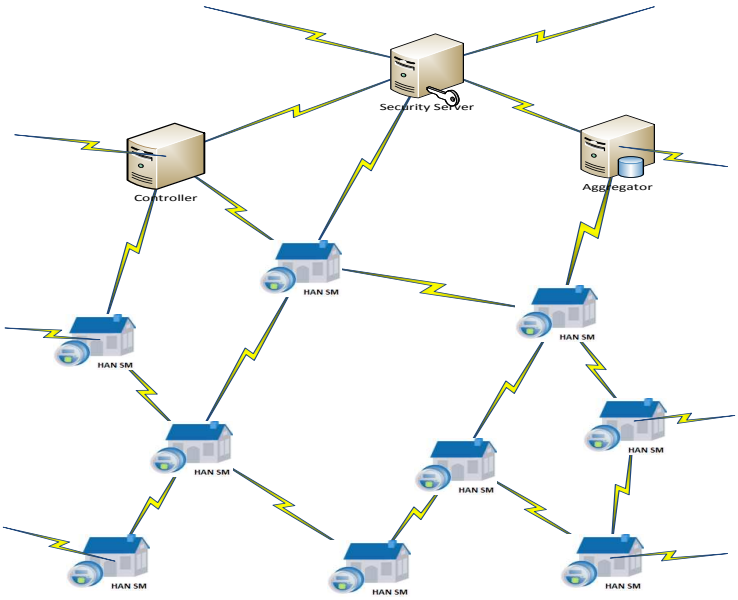


Figure 2.1: Smart Grid Topology for AMI

- Nodes are connected in a WMN, with requires unicast technology support for the multi-hop communications.
- Each node has a unique ID (most likely an IPv6 address), which may be manually assigned to the node by a technician at set up time.
- Each SM has a unique serial number SN embedded by the manufacturer, and an initial secret password pw loaded by the installing technician, for authentication purposes. On the other hand, SAS holds the appropriate verifier ver and $salt$ for the SM, in support of the SRP algorithm.
- Each node is initially loaded with the $H(.)$ function, and values "g & p" to be used in the SRP algorithm, which can be loaded by the technician at set up time, or at manufacturing.
- Nodes are all synchronized in time, and the newly installed SM would be able to synchronize itself with others using a suitable synchronization system, which design is outside of the scope of this chapter.

- SAS is responsible for the authentication as well as the key management mechanisms.

The system topology is depicted by Figure 2.1, which is based on [31]. Referring to our discussion in Section I, when a new SM is installed, it mutually authenticates itself with the SAS, and receives its private key from the SAS as well.

Definition:

Let us define system state (i, j):

- *Dimension i:* Represents the index, also referred as live, of system functions $f_i(.)$ and $F_i(.)$ as well as random values s_i and $\widetilde{s}_i$.
- *Dimension "j":* Represents index of PRNG set up values "a_j & b_j" used in (2.2b), which are shown only by "a & b" for simplicity.

2.3.2 Mutual Authentication Scheme

Depicted by Figure 2.2, our SRP-6a based mutual authentication scheme consists of following three steps:

Step I

New SM, sm, selects a random value R_{sm} and calculates:

$$G_{sm} = g^{R_{sm}} \bmod p$$

Then, SM sends G_{sm} along with its own SN_{sm} and ID_{sm} to the SAS.

Step II

SAS performs the following steps upon receiving the packet from SM in Step I:

- SAS lookups values "*ver* & *salt*" associated with SN_{sm}.
- SAS computes
$$k = H(N, g)$$
, and picks random values R_{sas}.

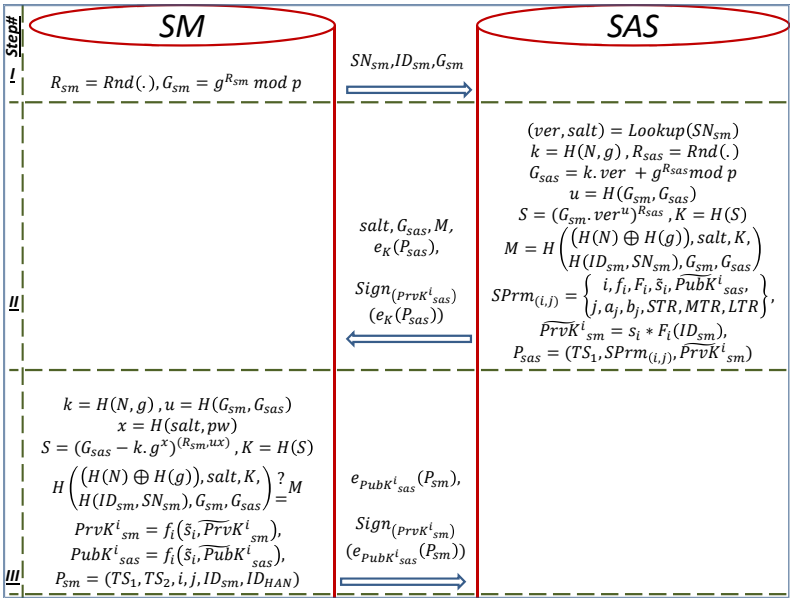


Figure 2.2: Smart Grid Mutual Authentication (SGMA)

- SAS calculates

$$\begin{aligned} G_{sas} &= k.ver + g^{R_{sas}} \bmod p \\ u &= H(G_{sm}, G_{sas}) \end{aligned}$$

- SAS computes

$$\begin{aligned} S &= (A * ver^{u})^{R_{sas}} \\ K &= H(S) \end{aligned}$$

  and verifier value "M" as:

$$M = H((H(N) \oplus H(g)), H(ID_{sm}, SN_{sm}), salt, G_{sm}, G_{sas}, K)$$

- Furthermore, SAS computes the private key SV of SM, $\widetilde{PrvK}^{i}_{SM}$, and forms the system parameter set for SM.
- Finally, SAS sends values "$salt$, G_{sas} & M" along with the encrypted and signed parameters set of the system to SM.

Step III

SM performs the following steps when it receives the packet sent by SAS in Step II:

- SM calculates:

$$\begin{aligned} k &= H(N, g) \\ u &= H(G_{sm}, G_{sas}) \end{aligned}$$

- SM computes:

$$\begin{aligned} x &= H(salt, pw) \\ S &= (G_{sas} - k.g^x \bmod p)^{(R_{sm}+u.x)} \end{aligned}$$

- Then, SM calculates K:

$$K = H(S)$$

  , and then verifies K based on the received M by comparing it with:

$$H((H(N) \oplus H(g), H(ID_{sm}, SN_{sm}), salt, G_{sm}, G_{sas}, K)$$

- If the verification condition holds, SM is assured that the symmetric key K shared with the server is valid. So, SM is able to decrypt received values, as well as is capable of checking the signature.
- Finally, SM obtains its own private key and sends an encrypted and signed acknowledgement to the SAS.

Note that by this point, SM and SAS are mutually authenticated to each other, and SM has received system parameters as well as its own private key.

2.4 Smart Grid Key Management Protocol

Our proposed SGKM is based on EIBC. Thus far, nodes have the appropriate private-public keys to be used for unicast and node-to-node secure communications based on PKI. In this section, we introduce our key refreshment mechanism as well as solutions for the required multicast and broadcast keys.

2.4.1 Key Refreshment

Referring to the EIBC mechanism presented in Section 2.2.1 and [52], the system needs to set the values of three timers STR, MTR and LTR. Values of these timers are transferred as parts of the system parameter in Step II of the authentication process described above.

Short term refreshment process

As depicted by Figure 2.3, the system regularly runs this process to move the system state from (i, j) to $(i + 1, j)$ based on the value of STR.

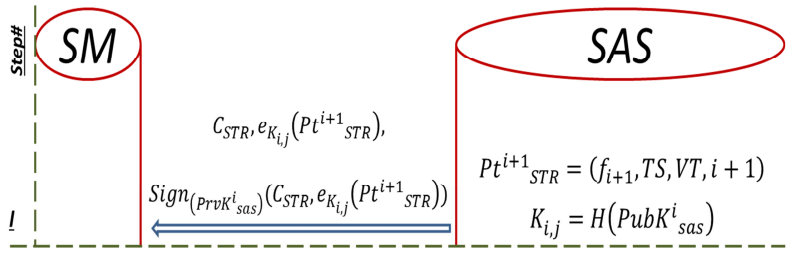


Figure 2.3: Broadcasting an Encrypted and Signed Packet in STR

SAS duties SAS first generates a new function $f_{i+1}(.)$ according to the new system state "$i + 1$". Then, SAS prepares a packet Pt^{i+1}_{STR} containing the $f_{i+1}(.)$ function, Time Stamp (TS) of producing the $f_{i+1}(.)$, Valid Time (VT) of the current system state dimension i and its new value "$i + 1$". Then, SAS applies the original $H(.)$ function to its own live public key to obtain a symmetric key $K_{i,j}$ via (2.6):

$$K_{i,j} = H(PubK^i_{SAS}) \tag{2.6}$$

Note: We describe more about $K_{i,j}$ at the end of this section, since we use this technique to handle the broadcasting key in the broadcast key management part.

Finally, SAS encrypts the Pt^{i+1}_{STR} packet utilizing the $K_{i,j}$ key, and broadcasts it along with the STR control command C_{STR}. SAS also signs these values with its own live private key in order to provide source authentication.

SMs duties As soon as any of the SMs receives the broadcast information identified by C_{STR}, uses the live public key of SAS to verify the signature. If the signature is valid, SM calculates the symmetric key $K_{i,j}$ following (2.6) and decrypts the received packet Pt^{i+1}_{STR}. Then, SM verifies the received system state "$i+1$" as part of the packet to make sure it is one after the current state. Furthermore, to prevent the replay attack, SM checks that TS is more than the VT received in the previous STR refreshment command. Finally, prior to VT, SM utilizes $f_{i+1}(.)$ to refresh the appropriate keys using (2.5a)-(2.5d) by following the steps in the short period refreshment process of EIBC, and starts using them by VT.

Medium term refreshment process

The system runs the medium term refreshment process presented in Figure 2.4 to change the system state from (i, j) to $(i, j+1)$ based on the value of MTR.

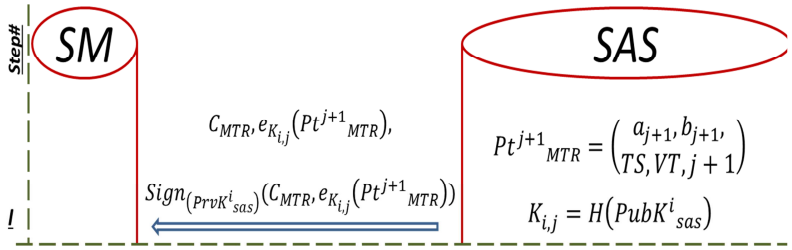


Figure 2.4: Broadcasting an Encrypted and Signed Packet in MTR

SAS duties Referring to EIBC, SAS first generates a new pair of PRNG parameters "a_{j+1} & b_{j+1}" for the new system state $(i, j+1)$. Then, SAS prepares a packet Pt^{j+1}_{MTR} containing the "a_{j+1} & b_{j+1}" values, Time Stamp TS of the packet, Valid Time VT of the new setup values plus the new system state "$j+1$". Then, SAS applies the original $H(.)$ function to its own live public key to obtain a symmetric key $K_{i,j}$ (2.6). Finally, SAS broadcasts the encrypted packet Pt^{j+1}_{STR} utilizing the $K_{i,j}$ key, along with the MTR control command C_{MTR}. SAS also signs this packet with its own live private key in order to maintain source authentication.

SMs duties When a SM receives the broadcast information identified by C_{MTR}, it obtains the live public key of SAS to verify the signature. If the signature is valid, SM calculates the symmetric key $K_{i,j}$ following (2.6), and decrypts the received packet Pt_{MTR}^{j+1}. Then, SM makes sure the system state "$j+1$" is one after the current one (j), and checks TS to prevent a replay attack. Finally, starting by VT, SM updates its $\widetilde{s}_i$ parameters according to (2.2b).

Long term refreshment process

Based on the value of LTR, the system runs the long term refreshment process as shown in Figure 2.5 to go from the (i,j) state to the $(0,0)$ state. SAS needs to regenerates the system parameters as well as the private key of each node and inform them one by one.

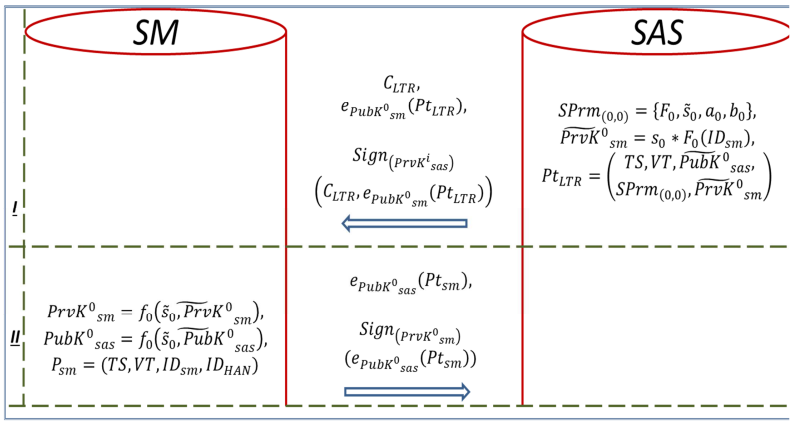


Figure 2.5: Unicasting an Encrypted and Signed Packet in LTR

2.4.2 Multicast Key Mechanism

SMK is used by a group source to encrypt the multicast packets. Furthermore, RMK is used by all group receivers to decrypt the messages that are encrypted by SMK. Our assumptions are:

- The multicast group is source based, and joining is initiated by the receiver.

- Each group is identified by a unique MID.
- SAS is in-charge of the multicast group key management.

Beside the SMK and RMK keys, each group also has a public/private key pair that is used in the multicast join algorithm. Similarly and by utilizing MID, system manages this key pair based on (2.3a), (2.3b), (2.4a) and (2.4b).

For the SMK and RMK keys, we define multicast group state (k & l) in a manner similar to the (i & j) state. Furthermore, "$g_k(.)$ & $G_k(.)$" similar to the "$f_i(.)$ & $F_i(.)$" functions, and finally "m_k & $\widetilde{m}_k$" along with "c_l & d_l" are similar to the "s_i & $\widetilde{s}_i$" and "a_j & b_j" items in our original system design for the unicast communication.

$$\begin{cases} G_{k+1}(.) = g_{k+1}(G_k(.)) & \text{(2.7a)} \\ m_{k+1} = g_{k+1}(m_k) & \text{(2.7b)} \\ \widetilde{m}_{k+1} = c_l * \widetilde{m}_k + b_l & \text{(2.7c)} \\ SMK_k = G_k(MID) & \text{(2.7d)} \\ RMK_k = (\widetilde{m}_k, (m_k * G_k(MID)) & \text{(2.7e)} \end{cases}$$

Establishing a multicast group

(i) An MGS that wants to form a multicast group sends a request to SAS. (ii) SAS provides MGS with the group initial parameters set consisting of $\{MID,\ \widetilde{m}_0,\ RMK_0$ & $G_0\}$ along with the private key SV of the group per (2.3b) and (2.4b) based on MID. (iii) MGS picks "$c_0,\ d_0$ & $g_0(.)$" and completes the group parameter set for the multicast group $(0, 0)$ state. Once the multicast group is established by MGS, MID is made publicly accessible by the parties that want to join. Note that MGS is in-charge of generating the $g_k(.)$ function in each state.

Joining multicast group

The join algorithm, as presented in Figure 2.6, consists of the following steps:

Join request (Step I) The new MGR applies the current system state function $F_i(.)$ to MID to obtain the public key via (2.1b). Then, MGR broadcasts its join request encrypted by the public key of the group, including its own ID.

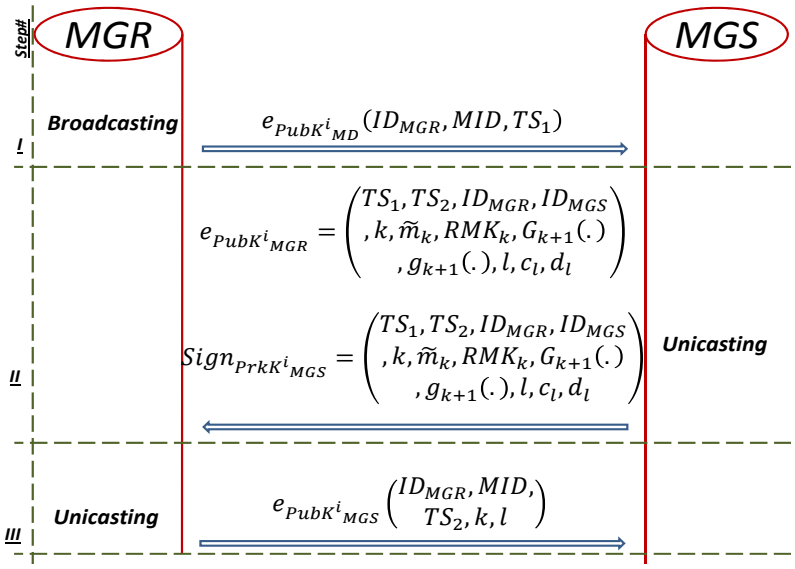


Figure 2.6: Joining a Multicast Group

Grant membership (Step II) Since only MGS has private key of the group, only MGS can decrypt the packet and replies with the membership grant, which consists of the group parameter set "$\widetilde{m}_k$, RMK_k, G_{k+1}, g_{k+1}, c_l, d_l", and at the same time, sends the $g_{k+1}(.)$ to the entire (current) group members to support forward secrecy. For the source authentication purposes, MGS signs this packet with its own private key.

Acknowledgement of membership (Step III) Firstly, MGR verifies the signature, and then accepts the information and joins the group if it is a valid one. Then, MGR sends an acknowledgement to the source notifying the source that MGR has successfully joined the group.

Key refreshment process

The reasons for the key refreshments in case of multicasting situation are different than the aforementioned unicast situation and consist of two cases: (i) a member joining or leaving causes the system to refresh the keys in order to maintain forward and backward secrecy, and (ii) providing overall

multicast key secrecy. However, we define and use a similar algorithm in both cases. To be more precise, each multicast group has timers similar to the unicast case, which are set by the system administrator as per group establishment purposes and application requirements. Referring to the unicast timer refreshment processes, we only describe relevant points of the multicast timers refreshment.

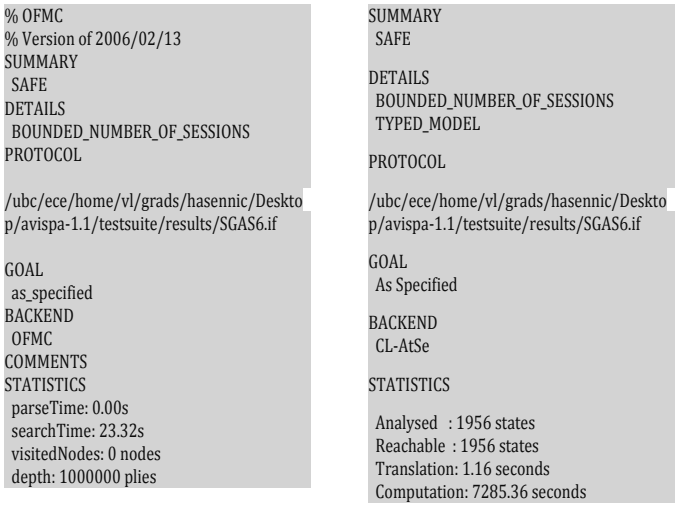
- For multicasting forward and backward secrecy concerning the receivers join/leave situation, we follow the short term refreshment process.
- MGS is in-charge of generating and distributing the new $g_{k+1}(.)$ in a manner similar to the short term key refreshment, and proceeding from the (k, l) to $(k + 1, l)$ state.
- MGS is in-charge of distributing the $\widetilde{m}_k$ set up values "c_{l+1} & d_{l+1}" addressing in a manner similar to the medium term key refreshment, moving from the (k, l) state to the $(k, l + 1)$ state.
- SAS is in charge of the long term key refreshment process, moving from the (k, l) state to the $(0, 0)$ state. SAS provides appropriate parameters including keys to the MGS, and then MGS unicasts them to the members utilizing their unicast public/private pair key.

2.4.3 Broadcast Key Mechanism

Referring to our unicast medium term key refreshment process, we apply the system original $H(.)$ function to the public key of SAS to obtain a symmetric key. Since the public key of SAS is dynamic and changes periodically according to the $f_i(.)$ function and state of the system, only the parties authenticated by the SAS, who receive their key management service from the SAS, have the live public key of SAS.

2.5 Security and Performance Analysis

In this section, we evaluate the security of our proposed SGMA and SGKM mechanisms using the AVISPA security analyzer. Furthermore, we review the adversary models including adversary interests and capabilities to attack the system. Then, we review the system security against attacks. At the end of this section, we verify the overhead cost reduction of our proposal.



(a) OFMC (b) ATSE

Figure 2.7: AVISPA Results

2.5.1 Formal Validation Using Software Tool: AVISPA

The results of the evaluation presented in Figure. 2.7a and 2.7b show that our proposed mechanism is secure and safe from attacks. To be more precise, the symmetric key that we prepare at the end of our authentication to be used by SAS to send the system parameters to SM is a valid and safe key. The system parameters consists of the PRNG and its setup values "a & b", as well as the private key SV of SM. Furthermore, SM is capable of finding the public key of SAS, and sends acknowledgement back to SAS, which is secure as well.

2.5.2 Adversary Models

Since we may have different situations for an adversary, we describe two scenarios addressing the adversary's different objectives and initial knowledge. In the first scenario, the adversary does not have control on any party; however in the second scenario, the adversary has full control on one of the SMs (i.e., there exists a malicious SM).

First scenario

Objectives The adversary wants to gain access to the system resources, like SAS or any of the SMs, and wants to be able to decrypt and encrypt the messages. Other possible objectives of the adversary are performing a DoS attack against SAS, or compromising the SAS.

Initial capabilities The adversary knows the IDs of all of the parties, as well as initial $H(.)$ function and "g & p" values in SGMA. Also, the adversary knows in detail the design of SGMA, and can make or have a valid serial number.

Capabilities during the attack During the attack, the adversary is able to receive the entire SMs and SAS communications, encrypted and unencrypted packets. If the adversary is able to steal the private key of any victim SM, it will be able to decrypt the encrypted packets sent to the SM, and impersonate the SM in sending packets with forged signatures of the SM. Therefore, the adversary will be able to send incorrect pricing information to the SM, take control of the smart appliances attached to the SM, modify billing information, etc. The adversary will also be able to mount a DoS attack by sending multiple authentication requests to the SAS.

Discussion: An adversary forging an SM's signature to mount a DoS attack on the SAS by sending multiple authentication requests (Step III in Figure 2.2) to SAS. As soon as SAS receives the requests, it checks its database for the $(ver, salt)$ pair associated with each request. Incorrect or missing values of $(ver, salt)$ cause the SAS to drop the request and ignore subsequent requests from the SM once a number of requests have been dropped.

If the adversary initiates the request with valid ID & SN that have been stolen from a SM, SAS may find the $(ver, salt)$ values and process the request by sending the response back to SM, and goes to the next step of SGMA. Since the adversary does not have the appropriate password, s/he is not able to obtain the key and decrypt the packets. However, SAS will leave the session open. Note that SAS sends a time stamp (TS_1) among other information in Step II of SGMA. SAS can close the session if the appropriate acknowledge is not being received within a certain time period (e.g. session expiry time). Furthermore, to prevent DoS attack in Step I, SAS can limit the number of the authentication requests it process within a given time frame. So, sending a large number of requests does not harm the SAS.

The adversary may try to perform a replay attack by forwarding a previous acknowledgement from the SM to the server. This solution does not help the adversary since the acknowledgement should be encrypted and signed utilizing the valid and appropriate system public and private keys. Also, the acknowledgement consists of the time stamp and ID of SM, which is not the valid one for the authentication session of the adversary.

The next option for the adversary is performing a brute-force attack and obtaining access to the encrypted packets. Normally, brute-force attack is time consuming, based on size of the key that packets are encrypted with. If the attacking time takes more than the session expiry time, the attack will not cause any issue. In the worst situation, the adversary can move to the on-line dictionary attack to speed up, or performs an off-line dictionary attack and find the session key, and finally obtain an expired private key for a not valid SM. However, the adversary would gain access to the system parameters, and if SAS has not run the key refreshment process yet, the adversary can keep going making the system parameters valid and fresh. In summary, by using any of the aforementioned attacks, the adversary is not able to compromise the server, since the adversary can only communicate with others, and only if the other parties send information to the malicious node, the adversary would be able to decrypt the packets. Furthermore, since SGMA uses a hash function, our authentication provides forward secrecy, and the adversary is not able to find out the original password.

To perform a MITM attack as another option for the adversary, the adversary may receive the first packet generated by a victim SM and change the value of G_{sm}. However, the adversary is not able to decrypt the second packet coming from the server, because the adversary needs the password of the victim to obtain the symmetric key K.

The other option is compromising the server by social engineering. Compromising the server does not give the adversary access to the passwords of SMs since SAS only keeps the verifier (and salt). However, if SAS records and keeps the private keys of the nodes (to be more precise, the private key SVs), the adversary will have private keys of the entire SMs. This attack is costly and unfortunately works in almost most of the situations. If SAS only generates the private keys and does not log them, to some extent this will prevent the attack from harming the previous generated keys. However, the adversary will be able to attack the new SMs. The best solution to prevent this attack is improving the server security well enough, for instance by changing the server password more often.

Second scenario

Objectives Similar to the previous scenario, the adversary wants to gain access to the system resources, like SAS or any of the SMs. The adversary would like to decrypt and encrypt the messages. Other objectives of the adversary may include performing a DoS attack against the SAS, or compromising the server or any of SMs.

Initial capabilities Similar to the previous scenario, the adversary knows the IDs of all the parties, the system parameter $H(.)$ function and "g & p" values regarding SGMA, as well as the detail design of the SGMA protocol. Furthermore, the adversary has a valid password to start SGMA, and by proceeding with the SGMA protocol, the adversary has a valid private key and all of the system parameters like $F_i(.)$.

Capabilities during the attack During the attack, the adversary is able to receive the entire SMs and SAS communications, encrypting and decrypting packets. Since the adversary has a valid private key of a SM, the adversary is able to decrypt and encrypt packets to and from the SM. For instance, the adversary can change the HAN commands, price list, or meter/billing information.

Discussion: In this situation, the adversary has full control of a malicious SM, in other words the adversary is a valid SM. Therefore, the adversary can rerun SGMA to be authenticated, and some-how perform a DoS attack. However, the adversary has only one password, and can resend the same ID and SN of victim SM to initiate a session, and in the worst case causes one open session.

The previous discussion about analyzing the adversary behaviour is valid in this scenario as well. The only differences are having valid system parameters like PRNG. Generally speaking, being in this scenario does not help an adversary to improve the chance of a successful attack. For instance, the adversary can run a brute-force attack by having a valid private key and communicate with others to obtain their private keys by brute-force. In this case, off-line dictionary can work because the adversary has the system parameters, like $f_i(.)$ and PRNG, and can find the live private key. However, just by performing one LTR process by SAS, the system can prevent the adversary from continuing the successful attack.

2.5.3 Other Security Characteristics

As per our design, a mutual authentication is performed since SAS needs to know the password verifier, and on the other side, SM needs to know the password. Both ends require one of these values to calculate the session key. In terms of attacks resilience, we refer to the discussion in the previous subsection, about the most well-known attacks such as brute-force, DoS, replay, on-line and off-line dictionary and MITM attack, which cover parts of the attacks resilient summary as presented by Table 2.1. We also refer to the above section about the social engineering attack that may work partially on the server; however, compromising one SM does not help the adversary to attack the whole system. In Table 2.1 we compare our mechanism with five of the schemes described in literature review section, which include mechanisms for authentication and/or key construction. Since author of [56] proposed using PKI and aimed at reducing the number of certificates (or issued private keys), the proposed mechanism in [59] suggests using users' biometric parameter (fingerprint) for authentication and presented research in [60] does not have detail design of the authentication and/or key construction, therefore we did not include them in this table.

Table 2.1: Summary of Resilience to the Attacks

Attack	[53]	[54]	[55]	[57]	[58]	Ours
Social engineering	✖	✖	✖	✖	✖	✔ & ✖
Brute-force	✖	✔	✔	✔	✖	✔
Replay	✔	✔	✔	✔	✔	✔
DoS	✖	✖	✖	✖	✖	✔
MITM	✔	✖	✖	✖	✔	✔
On-line dictionary	✔	✔	✖	✖	✖	✔
Off-line dictionary	✖	✔	✔	✔	✖	✔
Unknown key share	✔	✔	✔	✔	✔	✔
Compromised impression	✔	✔	✔	✔ & ✖	✔	✔
Denning-Sacco	✖	✔	✔	✔	✖	✔
Key privacy & insider	✔	✖	✔	✔	✖	✔
Ephemeral key compromise impersonation	✖	✔	✖	✖	✖	✔

Unknown key-share attack The second packet of the authentication scheme presented in Figure 2.2 is encrypted by symmetric key K. Encryption of this packet by SAS shows SAS has the key, and decryption the packet by SM and acknowledging the SAS proves that SM has the key as well.

Compromised impression resilience Referring to our analysis at the beginning of this section, finding the private key of any SM does not help an intruder to obtain the private key of any other node or SAS.

Denning-Sacco attack resilience If an intruder somehow finds a symmetric key used in the authentication scheme, since the key is the product of a hash function, which is a one-way function, the intruder would not be able to find the original password or the verifier. Furthermore, finding a private key does not help the adversary to find a symmetric key of the authentication session.

Privacy and insider attack resilience Since our scheme is based on PKI, each private key is known only by the owner (and maybe the server). Other nodes know only the public keys of all the nodes, which in fact is required by them to communicate with each other. Even if other nodes in between relay the packets, since the packets are encrypted and signed, they cannot have access to the private key of the source or destination nodes.

Ephemeral key compromise impersonation Suppose an adversary performs an off-line dictionary attack or brute-force or even social engineering attack and obtains the password of a SM. Because the password is only one of the values required for the session key construction, the adversary still is not able to find the session key, or the private key.

2.5.4 Performance Analysis

Consider the topology shown by Figure 2.1. Suppose SAS wants to refresh the keys of all the SMs. Compared to the original PKI, the IBC approach yields a better performance in the overhead cost, as we have discussed in previous sections. Therefore, we only compare our proposal with an SG that uses the IBC approach to secure data exchanges.

We assume that on average, each SM is connected to "$H_{sm} > 1$" neighbours (dimension of SM), and the average hop counts between SAS and any SM is equal to L_{sas} (Length of SAS network). Moreover, we define bw_l as the bandwidth (BW) of each link required per key distribution while the total network BW to refresh all the keys is BW_{net}. To compare the delay, we define d_h as the delay/time required by each hop (or link) to deliver/process a packet, and D_{net} to be the total system delay/time to refresh all the keys. For simplicity, we assume SAS generates same packet sizes in STR, MTR and LTR. Since the LTR process is similar to the key refreshment process

in the original IBC, we use it as our bench mark in this study. In order to show the improvement of SGKM employing EIBC, we assume the following relations exists between values of the timers:

$$\begin{cases} MTR = ms * STR \ , \ ms > 1 & (2.8a) \\ LTR = lm * MTR \ , \ lm > 1 & (2.8b) \\ LTR = ls * STR \ , \ ls > 1 & (2.8c) \\ ls = lm * ms & (2.8d) \end{cases}$$

The total network required BW and applicable delay by each key refreshment process are as follow:

$$\begin{cases} D_{net}(LTR) = d_h.(H_{sm} + \sum_{v=2}^{L_{sas}} v.H_{sm}^{v-1}) & (2.9a) \\ BW_{net}(LTR) = bw_l.\sum_{v=1}^{L_{sas}}(v.H_{sm} + v - 1).H_{sm}^{v} & (2.9b) \\ D_{net}(STR) = d_h.(1 + 2.d_h) & (2.9c) \\ BW_{net}(STR) = 2.bw_l.H_{sm}.\dfrac{H_{sm}^{L_{sas}} - 1}{H_{sm} - 1} & (2.9d) \end{cases}$$

In (2.9a)-(2.9d), we assume that in each STR (and MTR) process, 50% of the nodes broadcast concurrently, and in the LTR process, SAS processes H_{sm} SMs at the same time.

By a reasonable estimation, we have:

$$F_D(L_{sas}, H_{sm}) = \frac{D_{net}(LTR)}{D_{net}(STR)} \approx \frac{\sum_{v=2}^{L_{sas}} v.H_{sm}^{v-1}}{2.L_{sas}} \quad (2.10)$$

$$F_{BW}(L_{sas}, H_{sm}) = \frac{BW_{net}(LTR)}{BW_{net}(STR)} \approx \frac{\sum_{v=1}^{L_{sas}}(v.H_{sm}^{v+1})}{2.H_{sm}^{L_{sas}}} \quad (2.11)$$

F_D in (2.10) represents the relationship between the delays of the key refreshment processes, while F_{BW} in (2.10) demonstrates their required network bandwidth. Although these two quantities depend on the network topology, they are always greater than one.

Table 2.2 illustrates a few examples of F_D and F_{BM} based on H_{sm} and L_{sas}. As the table shows, the values increase with H_{sm} and L_{sas}. Note

Table 2.2: F_D and F_{BM} Based on H_{sm} and L_{sas}

H_{sm}	L_{sas}	$F_D(L_{sas}, H_{sm})$	$F_{BM}(L_{sas}, H_{sm})$
3	5	54.6	10.13
3	10	14024	21.37
3	20	8.45E+08	43.875
3	40	3.00E+08	88.87
4	5	159.2	12.45
4	10	1.69E+05	25.78
4	20	1.50E+11	52.44
4	40	2.00E+23	105.78
5	5	371	14.84
5	10	1.19E+06	30.47
5	20	1.18E+13	61.72
5	40	1.13E+27	124.22

that STR (and MTR) processes are run more frequently in our mechanism compared to LTR, whereas in the original IBC (and PKI), the key renewal (similar to LTR) process are run at almost the same rate as STR in our mechanism. For example if "$H_{sm} = 4$" and "$L_{sas} = 40$", the system requires less than 1% bandwidth to distribute the private keys following SGKM, compared with IBC/PKI. The time required for key distribution is reduced to "$5E - 24$" of the LTR delay. The data in Table 2.2 along with the above examples clearly shows that the proposed mechanism is much more efficient and greatly reduces the key refreshment delays compared to the original IBC or PKI mechanisms.

Overall analysis

In our design, we take advantage of the SRP, PKI and IBC approaches. Each one brings some benefits to our proposed mechanisms. Besides, our enhancement of each mechanism has improved the overall benefits to the system.

Firstly, we have reduced the required number of packets in our authentication scheme. To be more precise, we reduced the number of packets needed for mutual authentication from four to three. Furthermore, in the three packets, the entire set of system parameters are delivered as well as the private key of the new SM. Our analysis shows that SGMA is fast and robust and secure.

Secondly, implementing the private key cryptography system in a distributed environment causes providing a symmetric key between every two nodes that need to communicate to each other. Moreover, increasing the number of nodes that want to communicate with a single node requires that

the node keeps and manages a large number of keys (one per peer node), which is the case in the SG context. However, PKI requires only one key pair per entity in spite of a larger key size. In fact, while a node has its own private/public key pair, it is sufficient for the node and others to exchange secure communications.

Also, since IBC reduces the public key distribution overhead in PKI, we take advantage of this technique in our design. Furthermore, we have designed EIBC, an improved version of the IBC, and utilized it in SGKM. The most important benefit of using EIBC in this design is reduction of the private key distribution and refreshment overhead. In EIBC, most of the key refreshments are accomplished by the PKG broadcasting a packet to all nodes instead of unicasting one packet to each node, which yields substantial reduction in the system overhead cost. Indeed, broadcasting is used in two out of three key refreshment processes (STR and MTR), while unicasting is used in the LTR refreshment process, which is run much less frequently than the STR and MTR processes.

Cost

In order to have the above mentioned benefits of our solution:

1. the server (PKG) needs to generate a hash function $f_i(.)$ periodically
2. the entire parties need to be synchronized
3. at each point of time and in order to have new public keys of the parties, each entity need to apply the new hash function $f_i(.)$ to the previous hash function $F_i(.)$ to obtain $F_{i+1}(.)$, and then calculate the new public keys
4. each entity needs to run the PRNG and also using $f_i(.)$ to obtain its own new private key

However, considering the benefits of our proposal, the above costs are acceptable, especially for a dense network with so many entities. Also, as per our proposal and role of function $f_i(.)$ that is mainly used to obtain $F_{i+1}(.)$ out of $F_i(.)$, this function ($f_i(.)$) can be in any even simple format to improve the cost of our proposal.

Chapter 3

Password Authenticated Cluster-Based Group-Key Agreement

Several multi-party systems supporting group- and cloud-based applications have been proposed, e.g. in the context of SG. An important requirement of these systems is that the devices/parties need to communicate with each other as members of a group. In this chapter, we present an efficient group-key (GK) management scheme aimed at securing the group communications, for instance, from the utility to appliances and smart meters located in different homes. Our scheme is based on the X.1035 PAKE protocol standard, and also follows the cluster-based approach to reduce the costs of the GK construction and maintenance for large groups. Our protocol enables secure communications utilizing any communication technology. The proposed scheme supports forward and backward secrecy, and is more efficient in comparison with other GK mechanisms in the literature.

3.1 Introduction

A key motivation of the SG is that ICT technologies can support the use of dynamic pricing to counteract the inefficiency of engineering and operating a power grid based on the peak demand of consumers. A price increase in the peak-hours of power demand is one of the tools that providers can use to encourage consumers to shift their demand to the off-peak hours [61]. Therefore, different applications and ICT systems are emerging to support the consumers' needs to manage their energy demand in a smart way and even in real-time. Also, SG will integrate small power producers, which highlights the need for multi-party communications over the SG [62]. Different applications that require multi-party interactions in the SG context to address a variety of the customer needs have been reviewed in [63].

As shown by Figure 3.1, a typical SG links a group of consumers to a

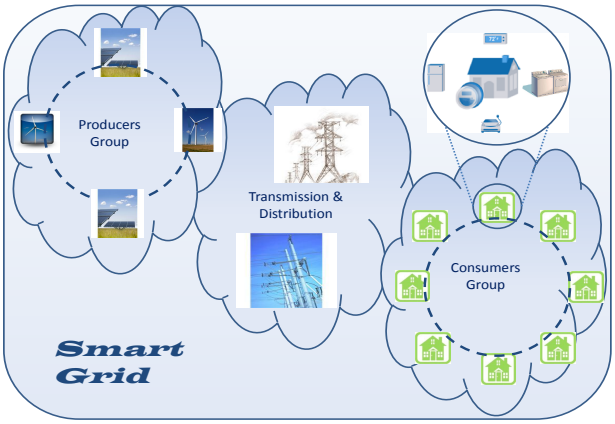


Figure 3.1: Consumers Group and Producers Group in Different Smart Grid Domains

group of producers. The power generated by the producers is sent to the SG to be delivered via the transmission and distribution domains of the SG to the customers. The producers need to communicate with each other as a group, to balance their power generation in order to reach a better Return On Investment (ROI) for their assets. In the customers' domain, the devices such as smart appliances or SMs need to communicate with each other as part of a group, to balance their demands in order to take advantage of the best/lowest price. For instance, the plug-in electric vehicles in different homes in a NAN can schedule their charging time to achieve a flat power demand.

Generally speaking, in order to have the benefits of the smart consumption and/or generation, devices/parties are required to communicate with each other as part of a group, to balance their resources and/or demands. While these group-based applications can be centralized or distributed, distributed ones are more efficient since the parties can locally make decisions. Most of these communications are many-to-many, e.g., in [64] and [33], and without any doubt, having a symmetric GK is the best solution to secure the communications.

Contribution: In this chapter, we propose the Password Authenticated Cluster-based Group-Key Agreement (PACGKA) protocols to manage the security of group communications in SG to support multi-party applications. PACGKA extends the PAKE protocol to construct and manage a GK among a cluster of devices, utilizing a pre-shared password for authentication.

3.2 Literature Review

In most of the existing solutions for construction of symmetric keys between two or more parties, the D-H protocol [2], or a D-H based protocol is used. Research and proposals on GK construction/management are mainly in two categories. In the first category, the GK is generated and managed by a central authority/controller. In the second category, the GK is constructed by participation of all the group members.

The first category is mainly motivated by multicast communications, which may have one source or one core node that handles data distribution to all the other nodes. The central controller generates and distributes the key between the receivers. The main problem in this category is in the efficiency and robustness of the key distribution and refreshment, along with the handling of membership changes (join and leave). There are different solutions in the literature for this category, most of which use the concept of structuring and forming the group in a tree topology [65] [66]. Since these systems use a central entity for the key management, they are vulnerable to a single point of failure. Although they are efficient in managing the join and leave of nodes/members, mostly the data needs to be partially decrypted by each node before being forwarded to the downstream nodes.

On the other hand, the second category is mainly motivated by many-to-many communications. They try to address the key construction in a distributed fashion by having participations of the entire membership. They are based either on the D-H or the BD protocol, with different techniques added to improve the key construction from the security and/or efficiency points of view.

The group PAKE protocol [67] assumes that each user has an individual password shared with the server. This design dictates having multiple passwords saved in a server, which decreases the efficiency of the system. The BD protocol is extended in [68] to address the failure of the group members as well as the size of the messages that are transferred between the members. The proposal assumes having authenticated links between the members, and constructs the key during two rounds in a ring-based group. The GK construction in [69] is aimed at small groups of entities. It assumes that each user has a workstation as well as a mobile device. The users meet each other while they carry their mobile devices. The mobile devices setup an initial shared value that they use in the workstations to communicate with each other.

The protocol presented in [70] is an extension of the existing protocol called S-3PAKE, both of which construct the GK assuming the existence

of a server. The protocol of [70] increases the number of members of the group from three to n. In both protocols, the server plays the main role by receiving messages from all members and then responds to the members. Since the server needs to provide services to the entire membership and is involved in all the steps in the interaction, the protocol is vulnerable to the single point of failure.

By utilizing Exclusion Basis Systems (EBS) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE) techniques in [71], a GK management for large scale systems is proposed. It provides an EBS-Based protocol that supports forward/backward secrecy relative to the join/leave process, and resilience to collusion attacks. Instead of using a clustering approach, it uses CP-ABE to handle large groups, which is more useful for the multicast communications.

IBC is used in [72] to design a GK agreement for multicast communications. The design maintains forward secrecy and integrity, and is developed for a dynamic environment. The system requires a group leader with whom each member communicates to prepare the shared values for the key construction. Although the process consists of two rounds, in each round communication with the leader is required. The protocol proposed in [73] is based on identities and do not require certificates. The protocol starts by each member choosing a random number and sending it to other members. Then, the results of the second round calculations are broadcast to all the members. The members are able to compute the GK after the second round. Similar to many other proposals, this protocol relies on broadcasting data/messages to others, which may not be robust for large groups.

Several IBC-based GK agreement protocols are evaluated in [74]. Moreover, a survey on security of group communications is presented in [75]. A brief survey on cluster-based GK Agreement (GKA) protocols for wireless sensor networks is presented in [76], differentiated into infrastructure-based and infrastructure-less networks. The infrastructure-based protocols studied include the Hierarchical Key Agreement Protocol, GKA protocol for Circular Hierarchical Group, Password-Based GKA protocol for Hierarchical Group and AP-1 which is a cluster-based GKA protocol based on the constant round multi-party dynamic key agreement protocol. The survey shows that the best performance is delivered in a system with equal cluster size and a small number of layers.

The proposal in [77] provides a GK management for the advanced distribution automation system of SG, which is based on a three-tier tree structure and decentralized architecture. In [54], firstly a SG gateway constructs a symmetric key with each SM based on a D-H algorithm. Then, the gateway multiplies the symmetric keys to form a GK, and finally, sends the GK

using the symmetric keys to the SMs.

The tree concept is used in the key management proposal in [78] to cover unicast, multicast and broadcasting keys for the SG, in which the multicast key is close to our design. The design is based on a binary tree in which each node uses two hash functions to calculate secret values of the tree nodes, which requires knowing the entire tree construction. Due to the high resource consumption and overhead cost, it may not be suitable for the SG with many nodes.

Discussion: Generally speaking, the mechanisms that are based on the BD protocol may suffer from the following weaknesses: (i) Some of them rely on a server, which makes them vulnerable to a single point of failure. (ii) Mostly they use broadcasting to distribute the key construction messages, which lack robustness as the messages may not be received by all the members. Even if they include a verification step to address this issue, it makes the algorithm time consuming and increases the system overhead. The problem is worsened in a large group with a long distance between nodes, or if the Internet is used for the communications.

Thus, to overcome the aforementioned issues, especially the second (ii) problem, we propose to unicast the messages in the PACGKA protocol presented in this chapter, which is based on the PAKE protocol in the X.1035 standard. As we will show in Section 3.5, an approach based on the BD protocol would be less efficient as it requires a larger number of messages in the protocol operation.

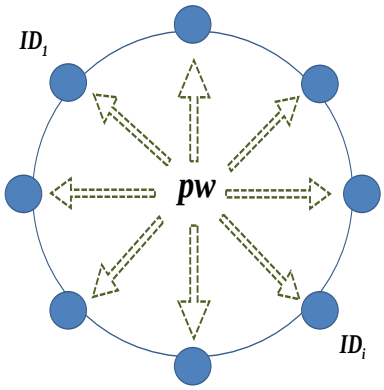


Figure 3.2: Single Cluster (Ring-Based) Structure

3.3 PACGKA-I Protocol for Single Cluster

The PACGKA protocol for a single cluster (PACGKA-I) is presented in Algorithm 1 for constructing and verifying a shared value, and calculating the GK. The mechanism constructs the shared value in two rounds involving $2 \times n - 1$ messages. PACGKA-I consists of the protocol for forming the GK and the auxiliary protocol for key maintenance. As shown in Figure 3.2, we assume the members' IDs form a cyclic group.

To describe the protocol based on Figure 3.2, consider a group with four parties ID_1, ID_2, ID_3 & ID_4, which are preloaded with the g, p & $H(.)$ parameters. They also receive a shared password pw from the system along with the required system parameters such as number of entities ($n = 4$) in the group (ring/cluster) plus IDs of the neighbours (prior & next). The protocol use a message vector M that has $(n-1)$ fields (three in this example).

3.3.1 Group Key Construction

First round

We run the protocol starting from ID_1.
Note: For encryption of the message vector M, the parties simply multiply each field of the M to the forward session key P_{k+}. Thus for decryption, the parties only need to divide the fields of the received vector M to the backward session key P_{k-}.

ID_1: First, ID_1 generates random value r_1, computes initial value and loads the $M.[1]$ to begin with. Then, ID_1 calculates the backward and forward session keys with ID_4 and ID_2, which are given by P_{1-} and P_{1+}, encrypts M with P_{1+}, and sends it to ID_2.

$$\begin{cases} r_1 = Rand(.)\ ,\ M.[1] \equiv g^{r_1} \bmod p \\ P_{1-} = H(ID_4|pw|ID_1) \\ P_{1+} = H(ID_1|pw|ID_2) \end{cases}$$

ID_2: ID_2 generates random value r_2 and also computes the backward and forward session keys P_{2-} and P_{2+}. Then, ID_2 receives M and decrypts it with P_{2-}. Note that $P_{2-} = H(ID_1|pw|ID_2) = P_{1+}$. Then, ID_2 updates M

Algorithm 1 PACGKA-I: Group-key Formation for a Single-cluster Group

Define:
n : Total number of members, where "$n+1 \equiv 1$" and "$-1 \equiv n$".
g & p : D-H algorithm parameters.
$H(.)$ & pw : Shared Hash function, & shared password.
M : An $n-1$ element message vector; $M.[k]$ is k^{th} field of the M.
ID_k : ID of the k^{th} party.
$Rand(.)$: Random number generator function.
r_k & SV_k : Random value and final shared value of ID_k.
P_{k+} & P_{k-} : Forward & backward session keys of the k^{th} party.
$E_K(X)$ & $D_K(X)$: Encryption & decryption of X with the K key.
V_{i-} & V_{i+} : Verifier for the previous and next parties.
$F_S(.)$ & $F_R(.)$: Background functions to send and receive messages.
Algorithm:
First round: ID_1 ***to*** ID_{n-1}
$r_1 \leftarrow Rand(.)$
$M.[1] \leftarrow g^{r_1} \bmod p$
$P_{1-} = H(ID_n|pw|ID_1)$
$P_{1+} = H(ID_1|pw|ID_2)$
$MyEncM \leftarrow E_{P_{1+}}(M)$
$F_S(MyEncM \rightarrow ID_2)$
for $i = 2 \rightarrow n-1$ **do**
  $r_i \leftarrow Rand(.)$
  $P_{i-} = H(ID_{i-1}|pw|ID_i)$
  $P_{i+} = H(ID_i|pw|ID_{i+1})$
  $F_R(MyEncM \leftarrow ID_{i-1})$
  $M \leftarrow D_{P_{i-}}(MyEncM)$
  for $j = i \rightarrow 2$ **do**
    $M.[j] \leftarrow M.[j-1]^{r_i} \bmod p$
  end for
  $M.[1] \leftarrow g^{r_i} \bmod p$
  $MySnd \leftarrow E_{P_{i+}}(M)$
  $F_S(MySnd \rightarrow ID_{i+1})$
end for
Second round: ID_n ***and*** ID_1 ***to*** ID_{n-1}
$F_R(MyEncM \leftarrow ID_{n-1})$
$M \leftarrow D_{P_{n-}}(MyEncM)$
$SV_n \leftarrow M.[n-1]^{r_n} \bmod p$
for $j = n-1 \rightarrow 1$ **do**
  $M.[j] \leftarrow M.[j-1]^{r_n} \bmod p$
end for
$V_{n+} \leftarrow H(pw|M.[n-1]|SV_n)$ {Verifier for the next party}
$MyEncM \leftarrow E_{P_{n+}}(M)$ $\{n+1 \equiv 1\}$
$F_S((MyEncM, V_{n+}) \rightarrow ID_1)$
for $i = 1 \rightarrow n-1$ **do**
  $F_R((MyEncM, V_{i-}) \leftarrow ID_{i-1})$
  $M \leftarrow D_{P_{i-}}(MyEncM)$
  $SV_i \leftarrow M.[n-1]^{r_i} \bmod p$
  if $V_{i-} == H(pw|M.[n-1]|SV_i)$ **then**
    $GK_i \leftarrow H(pw|SV_i)$
  else
    return $Error: Verification\ failed$
  end if
  for $j = n-1 \rightarrow i+1$ **do**
    $M.[j] \leftarrow M.[j-1]^{r_i} \bmod p$
  end for
  $V_{i+} \leftarrow H(pw|M.[n-1]|SV_i)$
  $MyEncM \leftarrow E_{P_{i+}}(M)$
  $F_S((MyEncM, V_{i+}) \rightarrow ID_{i+1})$
end for

and finally, encrypts it with P_{2+} and sends it to ID_3.

$$\begin{cases} r_2 = Rand(.) \\ P_{2-} = H(ID_1|pw|ID_2) \\ P_{2+} = H(ID_2|pw|ID_3) \\ M.[2] \equiv M.[1]^{r_2} \bmod p \equiv g^{r_2 r_1} \bmod p \\ M.[1] \equiv g^{r_2} \bmod p \end{cases}$$

ID_3: Similarly, ID_3 generates random value r_3 and also computes the backward and forward session keys P_{3-} and P_{3+}. Then, ID_3 receives M and decrypts it with P_{3-}. Then, ID_3 updates the vector M and finally encrypts it with the forward key P_{3+} and sends it to ID_4.

$$\begin{cases} r_3 = Rand(.) \\ P_{3-} = H(ID_2|pw|ID_3) \\ P_{3+} = H(ID_3|pw|ID_4) \\ M.[3] \equiv M.[2]^{r_3} \bmod p \equiv g^{r_3 r_2 r_1} \bmod p \\ M.[2] \equiv M.[1]^{r_3} \bmod p \equiv g^{r_3 r_2} \bmod p \\ M.[1] \equiv g^{r_3} \bmod p \end{cases}$$

Second round

This round starts with ID_4.
ID_4: Similar to ID_3, firstly ID_4 generates random value r_4 and computes the backward and forward session keys P_{4-} and P_{4+}. Then, ID_4 receives M and decrypts it with P_{4-}. ID_4 (last member of the cyclic group) now is able to calculate its shared value SV_4. Then, ID_4 updates M and computes the GK as well as a verifier for the next party (ID_1). Finally, ID_4 encrypts M with P_{4+} and sends it along with the verifier to ID_1.

$$\begin{cases} r_4 = Rand(.) \\ P_{4-} = H(ID_3|pw|ID_4) \\ P_{4+} = H(ID_4|pw|ID_1) \\ SV_4 \equiv M.[3]^{r_4} \bmod p \equiv g^{r_4 r_3 r_2 r_1} \bmod p \quad (3.1) \\ M.[3] \equiv M.[2]^{r_4} \bmod p \equiv g^{r_4 r_3 r_2} \bmod p \\ M.[2] \equiv M.[1]^{r_4} \bmod p \equiv g^{r_4 r_3} \bmod p \\ M.[1] \equiv g^{r_4} \bmod p \\ V_{4+} = H(pw|M.[3]|SV_4) \\ GK_4 \leftarrow H(pw|SV_4) \quad (3.2) \end{cases}$$

ID_1: First of all, ID_1 receives M and decrypts it with P_{1-}. Then, ID_1 calculates its shared value ($SV_1 = SV_4$) and then verify it versus the received verifier VF_{1-} ($= VF_{4+}$) from ID_4. Assuming the verification holds positive, ID_1 is assured that its shared value is the same as the one that ID_4 has. Then, ID_1 updates M and also calculates a verifier VF_{1+} for the next party. ID_1 finally encrypts M with P_{1+} and sends it along with the verifier to ID_2.

$$\begin{cases} SV_1 \equiv M.[3]^{r_1} \bmod p \equiv g^{r_4 r_3 r_2 r_1} \bmod p & (3.3) \\ V_{1-} \stackrel{?}{\Longleftrightarrow} H(pw|M.[3]|SV_1) \\ M.[3] \equiv M.[2]^{r_1} \bmod p \equiv g^{r_4 r_3 r_1} \bmod p \\ M.[2] \equiv M.[1]^{r_1} \bmod p \equiv g^{r_4 r_1} \bmod p \\ V_{1+} = H(pw|M.[3]|SV_1) \\ GK_1 \leftarrow H(pw|SV_1) & (3.4) \end{cases}$$

ID_2: Similarly, ID_2 receives M and decrypts it with backward session key P_{2-}. Then, ID_2 calculates its shared value ($SV_2 = SV_1$) and then verifies it versus the received verifier VF_{2-} ($= VF_{1+}$) from ID_1. If the verification holds positive, ID_2 is assured that its shared value is the same as the one that ID_1 has, which is the same as the shared value of ID_4. Then, ID_2 updates M and also calculates a verifier VF_{2+} for the next party. ID_2 finally encrypts M with P_{2+} and sends it along with the verifier to ID_3.

$$\begin{cases} SV_2 \equiv M.[3]^{r_2} \bmod p \equiv g^{r_4 r_3 r_2 r_1} \bmod p & (3.5) \\ V_{2-} \stackrel{?}{\Longleftrightarrow} H(pw|M.[3]|SV_2) \\ M.[3] \equiv M.[2]^{r_2} \bmod p \equiv g^{r_4 r_2 r_1} \bmod p \\ V_{2+} = H(pw|M.[3]|SV_2) \\ GK_2 \leftarrow H(pw|SV_2) & (3.6) \end{cases}$$

ID_3: ID_3 receives M and decrypts it with P_{3-}. Then, ID_3 calculates its shared value and then verifies it versus the received verifier VF_{3-} ($= VF_{2+}$) from ID_2. Assuming the verification holds positive, ID_3 is assured that its shared value is the same as the one that ID_2 has, which is the same as the shared value of ID_1 and ID_4. ID_3 is the last party that was supposed to calculate the shared value. The only step left is verifying it for the ID_4. Therefore, ID_3 calculates a verifier VF_{3+} and sends it to ID_4.

$$\begin{cases} SV_3 \equiv M.[3]^{r_2} \bmod p \equiv g^{r_4 r_3 r_2 r_1} \bmod p & (3.7) \\ V_{3-} \stackrel{?}{\Longleftrightarrow} H(pw|M.[3]|SV_3) \\ V_{3+} = H(pw|M.[3]|SV_3) \\ GK_3 \leftarrow H(pw|SV_3) & (3.8) \end{cases}$$

ID_4: Finally, ID_4 only needs to check the verification value. The positive verification result assures that ID_4 has the same shared value that ID_3 has.

$$V_{4-} \overset{?}{\Longleftrightarrow} H(pw|M.[3]||SV_4)$$

Note: Note that the group members have the same shared value and can be seen by (3.1), (3.3), (3.5) and (3.7). Therefore, the GK_is are the same, which are shown by (3.2), (3.4), (3.6) and (3.8).

3.3.2 Key Maintenance

Key refreshment

To improve and guarantee/increase the secrecy of the GK, PACGKA-I refreshes the key periodically. In order to do this, we propose setting up a timer to initiate and trigger the refreshment process. Note that the timer value that determines how often the key is refreshed depends on the application as well as the size of the group. Therefore, we propose the following Group Key Reconstruction (GKR) process for PACGKA-I: the system controller distributes a new password along with the start and expiry times to the entire group members to construct the new GK.

Join and leave process

In the case of a new node joining the existing group, or an existing node leaving the group, the controller performs GKR to support the forward and backward secrecies.

Malicious behaviour of a node

In case one of the group members begins behaving maliciously, the malicious node is removed from the group. In PACGKA-I, the system controller relies on both peer neighbours of the malicious node to vote jointly to identify the misbehaving member. In this case, they directly send a unicast message via the secure channel to the system controller. Subsequently, the system controller invokes the GKR algorithm for the group while excluding the malicious one. If a group of nodes decide to behave maliciously, they most probably will be able to attack the system; however, it will be a costly attack since a group of nodes are required.

3.4 Cluster-based Mechanism: PACGKA-II Protocol

In this section, we present an efficient multi-cluster GK management mechanism, PACGKA-II, which is based on the single cluster algorithm presented in Section 3.3. This scheme is motivated by the fact that in the case of a large group, the PACGKA-I protocol becomes time consuming as the nodes should perform many polynomial and arithmetic operations. Although SG systems are mostly static with low occurrences of node joining or leaving, security considerations dictate running GKR every so often for key refreshment. To overcome the latency issue, we propose using a clustering approach.

3.4.1 Clustering Scheme

We define our clustering scheme following the presentations in Section 3.2 and [76]. Consider a group with N members. We divide the group into n clusters with no more than m members in each cluster, where:

$$N \leq m \times n \tag{3.9}$$

An example of the clustering scheme is depicted by Figure 3.3. We identify each cluster by:

$$Clstr_u \ , \quad u = 1, ..., n$$

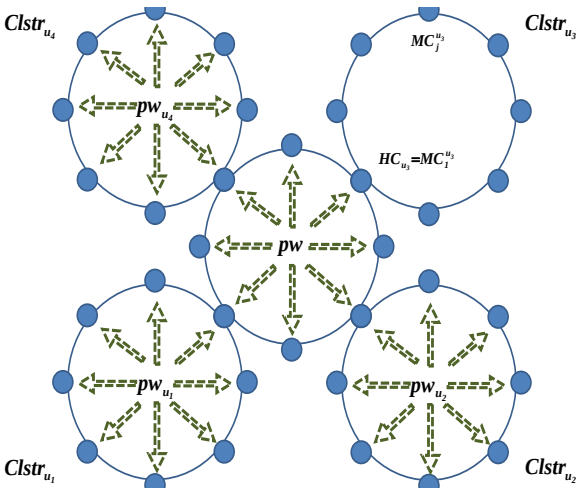


Figure 3.3: Multi Cluster Ring-Based Structure

Furthermore, members of the u^{th} cluster are denoted by:

$$MC_k^u \ , \quad k = 1, ..., m$$

One of the cluster members acts as the cluster representative, or cluster head, and is denoted as HC_u for the u^{th} cluster. The cluster heads/representatives form a core ring/sub-group consisting of all the cluster heads, given by $C - Clstr$.

Note that finding the right value for m (or n) is an optimization problem and can be formed based on the criteria that are important for the system and the application, and we leave this task to the administrator of the system. For instance, the problem can be formed to minimize the number of operations, or the delay of the key formation process, or any other system parameter or security measures. Indeed, the problem should address the application, system, resources and security aspects. After presenting our protocol, we will give an example of this problem to find the optimum values of n and m.

3.4.2 The Logic of the Multi-cluster Group Key Mechanism

Overall, PACGKA-II follows a similar concept as PACGKA-I. In fact, PACGKA-II can be considered as an extended version of PACGKA-I. The main steps of the PACGKA-II protocol are as follows:

I. Dividing the main group to clusters $Clstr_u$.

II. Nominating one party per cluster as the cluster head HC_u to represent the cluster.

III. Forming the core cluster consisting of all the cluster heads. Note that each cluster head is a member of two sub-groups: the cluster it is representing and the core cluster.

IV. The protocol starts by sending a password pw to the core cluster/sub-group.

V. Each cluster head picks a password pw_u, sends it to its own cluster members to construct a GK using PACGKA-I within the cluster.

   Note that, since the cluster head is a member of the cluster it is representing, at the conclusion of PACGKA-I, it has the GK of the sub-group (sub-GK) SG_u. Also, the cluster head has the shared value that

the members of the sub-group used to obtain sub-GK. Let us call this shared value the "sub-group shared value" G_u.

$$\begin{cases} G_u \equiv g^{\prod_{j=1}^{m} r_j^u} \bmod p & (3.10) \\ SG_u = H(G_u | pw_u) & (3.11) \end{cases}$$

VI. Using the password received in Step IV, the members of the core cluster run PACGKA-I to construct the shared value HSV_u and GK K_{Group}, taking the sub-group shared value (from Step V) as the random value of the cluster head (G_u).

$$\begin{cases} HSV_u \equiv g^{\prod_{i=1}^{n} G_i} \bmod p & (3.12) \\ K_{Group} = H(HSV_u | pw) & (3.13) \end{cases}$$

VII. The cluster head distributes the GK K_{Group} to the cluster utilizing sub-GK SG_u for encrypting the GK.

3.4.3 Key Maintenance

All of the situations that require key maintenance as explained in Section 3.3 regarding single cluster GK formation are applicable to the multi-cluster GK formation as well. To handle the key refreshment, we need to rerun the complete PACGKA-II protocol. However, for situations such as a member joining or leaving, and detection of a malicious node, we propose a different solution. If a member joins the group, the new member should join one of the clusters, so it can be considered as a *sub-group event*. If one of the cluster members becomes malicious or leaves the cluster, again it can be considered as a *sub-group event*, unless the malicious node, or the node that is leaving the group is a cluster head, in which case we call it a *cluster head event*.

Sub-group event

Let us assume that the event occurs inside the u^{th} cluster. In this case, the cluster head HC_u reselects a password pw_u and shares it with its cluster members. Then, the cluster members of the $Clstr_u$ performs PACGKA-I to form sub-GK. Then, the cluster heads perform Steps VI and VII of PACGKA-II. In fact, the other sub-groups do not need to reconstruct their sub-GK, and the cluster heads can still use the prior values. Finally, the cluster heads inform their cluster members about the new GK.

Cluster head event

In this case, either a cluster head is malicious or it leaves the group. Firstly, a new cluster head for that cluster needs to be chosen, and secondly, the GK should be constructed by performing PACGKA-II completely.

3.4.4 Size of the Clusters

As shown above, PACGKA-II involves running PACGKA-I in two rounds, once around each cluster and then around the core cluster. Here we illustrate the optimization of the size of the clusters with respect to the delay, by formulation the delay expression and then minimizing it. Table 3.1 shows our parameters.

Table 3.1: Parameters of the Cluster Size Problem

Parameter	Description
N	Total number of the members in the group
m	Number of the members per cluster
n	Number of the clusters (sub-groups)
$\hat{d}$	Party processing time including message delivery
$\hat{D}$	Delay of the GK construction

We assume equal "party processing time including message delivery" values ($\hat{d}$) for each party, and equal size of the clusters. Our problem is minimizing the "delay of the GK construction" ($\hat{D}$), which can be formulated as follows:

$$\begin{cases} Min \quad \hat{D} = \hat{d} + \hat{d} + (2m-1)\hat{d} + (2n-1)\hat{d} + \hat{d} \\ S.t: \qquad m \times n \geq N \end{cases}$$

In the above problem formulation, each term in the right hand side of the delay equation respectively represents the delay of:

- Distributing password within the core cluster.
- Distributing password within each sub-group.
- Sub-GK construction.
- GK construction within the core cluster.
- Distributing the GK inside the clusters.

To solve this problem, we simplify it and rewrite as follows:

$$\begin{cases} Min \quad \hat{D} = (2m + 2n + 1)\hat{d} \\ S.t: \qquad m \times n \geq N \end{cases}$$

$\hat{D}$ is a convex function, which has a minimum. We solve the problem in the border of $m \times n = N$, and calculate the first derivative respect to n in order to find the optimal value:

$$\begin{cases} m \times n = N \rightarrow m = \dfrac{N}{n} \\ \hat{D} = (2n + \dfrac{2N}{n} + 1)\hat{d} \\ \dfrac{\partial \hat{D}}{\partial n} = (2 - \dfrac{2N}{n^2})\hat{d} = 0 \rightarrow m = n = \sqrt{N} \end{cases}$$

Therefore, the best performance of the protocol and the minimum delay happens when $m = n = \sqrt{N}$.

3.5 Security and Performance Analysis

To analyze and evaluate the security of the PACGKA protocols, we consider the Dolev-Yao approach [79].

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
/ubc/ece/./hasennic/Desktop/avispa
-1.1/testsuite/results/SGGK4.if

GOAL
as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.06s
  visitedNodes: 12 nodes
  depth: 11 plies
```

(a) OFMC

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  UNTYPED_MODEL

PROTOCOL
/ubc/ece/./hasennic/Desktop/avispa
1.1/testsuite/results/SGGK4.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 69 states
  Reachable  : 14 states
  Translation: 0.02 seconds
  Computation: 0.00 seconds
```

(b) ATSE

Figure 3.4: AVISPA Results

3.5.1 Formal Validation using Software Tool

We develop our analysis for a group consisting of four members, corresponding to the example in Section 3.3. The simulation results presented by Figure 3.4a and Figure 3.4b show that the GK constructed by the PACGKA mechanism is secure and safe to be used by the members of the group. Although the system controller has provided the shared password, it does not have access to the GK. We assume this entity is trusted and does not perform any attacks like MITM.

3.5.2 Adversary Model

Objective

- Gaining access to the system resources, like a SM or an appliance.
- Performing a MITM attack to gain access to the GK, or a sub-GK.

Initial capabilities

- The adversary has complete knowledge about the topology and the exact address/ID of each party.
- The adversary has access to the system hash function $H(.)$ and g & p used in our protocol.
- The adversary knows the detail design of the PACGKA mechanism (PACGKA-I and PACGKA-II protocols).

Capabilities during the attack

- The adversary receives the entire encrypted and unencrypted (plain) data in different stages of the key formation, or later on and during the using of the GK.
- If the adversary gains access to any password (core cluster, or any other sub-group cluster), she/he will attempt to perform a MITM attack.
- If the adversary gains control to a malicious node, she/he can perform DoS by joining and leaving continuously.

Discussion

We assume cluster heads and cluster members receive the appropriate password via a secure channel. Therefore, if the adversary finds out the password of a cluster after completion of the initial sub-GK formation (PACGKA-I protocol), the adversary cannot gain any further information since the password is not being used any more. Similarly, if the adversary by performing any attack like brute-force or off-line dictionary obtains the shared password of the core cluster after the GK construction, this information is useless for the adversary since the key is formed and the password is like a one-time password. Thus there is resilience against Ephemeral key compromise impersonation. However, if the adversary finds/steals the password before the key formation process starts in any level such as in a cluster (PACGKA-I & PACGKA-II) or in the core cluster (PACGKA-II), she/he can take advantage of this password by performing a MITM attack. As long as the GK is valid without any changes, the adversary can use it. However, the GKR process changes the key completely. Thus, key refreshment by GKR periodically should be considered as a requirement for the system.

Moreover, our adversary can compromise the server by for instance social engineering attack. Consequently, the adversary can send the new password to the cluster head and dictates them to re-construct the GK. Although we improved the process of the key formation by using the clustering approach, it can harm the system resources. On the other hand, the adversary can participate in the key formation and gain access to the GK easily. Performing social engineering attack against the server is possible in any system and environment. The only solution to prevent this attack is having a strong system security management procedure. Generally speaking, although technically feasible, the social engineering attack should be a very expensive attack. Therefore, the best solution is increasing the cost of the attack, in order to make it unattractive for the adversary.

3.5.3 Attack Analysis

Based on aforementioned discussion about the adversary, plus the PACGKA assumptions in Section 3.3 (i.e., parties are already authenticated to the system and have valid security system and key management to be able to have a secure communication), Table 3.2 analyzes the resilience of PACGKA against different well-known attacks.

Unknown key-share attack In our proposed mechanism, all of the parties should participate in the key formation and the verification steps. Indeed, the key is formed in a consensus manner with commitments of the entire membership. Thus, our protocol guarantees that if one of the members has the key, its neighbours have it as well.

Denning-Sacco attack resilience Due to using hash functions in the final key calculation steps in the sub-GK or in the final step of the GK (and verification steps), finding a sub-GK or a GK does not help adversary to gain access to the cluster or cluster head initial passwords.

Table 3.2: PACGKA Attacks Resilience Summary

Attack	Resilience
Social engineering attack	✔ & ✖
Brute-force attack	✔
Replay attack	✔
DoS attack	✔
MITM attack	✔
On-line dictionary attack	✔
Off-line dictionary attack	✔
Unknown key share attack	✔
Denning-Sacco attack	✔
Ephemeral key compromise impersonation	✔

3.5.4 Overhead Analysis

Following our discussions in Section 3.2, in a BD-based mechanism, the messages are supposed to be distributed to the entire membership. The original concept is to broadcast the messages to the group members, although it may not be possible in all cases. One may consider broadcasting the message in the overlay layer; however, in the lower layer the messages are transferred by unicast communications. Moreover, it may be possible to broadcast the messages only to a small group within a short distance. Thus, making sure that the messages reach the destinations can cause extra overheads. Missing any message by any member causes failure on the algorithm.

Let us assume that we have a group with n members, all in one cluster. We assume the following scenarios:

1. BD protocol based model: The messages of each member in the first round should be delivered to two members ($2 \times n$ messages), and in the second round, to all the members ($n \times (n-1)$ messages), which totally is $n \times (n+1)$ messages.

2. PACGKA: We require $2 \times n - 1$ message delivery (including the verification).

Regardless of the n value, the second scenario has a smaller number of message deliveries. If we increase the n value to a high value, the second scenario requires about $2/n$ times the number of message deliveries in the first scenario.

3.5.5 Implementation Considerations

Any application that requires a GK can use the PACGKA protocol. Since we use the clustering, the method is scalable and can be easily implemented based on different system specifications. Same as any other security system, the strength of the key required to achieve certain the security/confidentiality level depends on its size. We do not specify the key size or the time period of the key refreshment process, and leave them to be defined by the system administrator. Furthermore, while we propose that a group can be divided to the clusters, the number of clusters and size of each cluster are also parameters to be determined by the system administrator. For instance, the administrator may define each NAN as a cluster, and choose the NAN controller to act as the cluster head. Indeed, these set up values are driven by the application and system conditions. The detail analysis of the application and system resources helps the administrator of the system to identify the key size, as well as the size of the g & p parameters used in the PACGKA key construction mechanism.

Cost

In our proposal, leaving a non-head cluster entity from a cluster has a low cost, since only the shared value of that cluster needs to be recalculated. However, if a cluster head leaves, the entire algorithm needs to be rerun, and a new group key should be calculated. Even if we don't cluster the entities, and mostly in a group key mechanism that the group key is being calculated by participation and coordination of the entities, a similar things needs to be done. Moreover, if we use the central model, in which an entity acts as group head to generate the group key, the group head needs to regenerate the group key and share it with the remaining of the entities.

Furthermore, the other cost of our proposal is calculating the size of the clusters, as well as forming the clusters. Therefore, entities need to know where they are standing, and what are the previous and next entities. In

addition, each entity needs to know for how long needs to wait for the next value to be received.

Considering the benefits of our proposal, which are described in above mentioned sub-sections, the costs of using the mechanism worth it, since in any other group key mechanism, especially if the forward and backward secrecies need to be maintained, a similar cost in applied.

Chapter 4

Multilayer Consensus ECC-Based Password Authenticated Key-Exchange Protocol

This chapter aims at providing a key agreement protocol for SG to cope with access control of appliances/devices located inside a HAN by a set of controllers outside the HAN. The commands/packets initiated by the controllers in crisis cases should be delivered fast and immune from any interruption. The HAN controller, which acts as a gateway, should not cause any delay by decrypting and re-encrypting the packets, nor should it has any chance to modify them. Considering the required level of security and quality of service, we design our protocol with an ECC approach. We improve and implement the PAKE protocol in two steps. First, we propose an auxiliary mechanism that is an ECC version of PAKE, and then extend it to a multilayer consensus model. We reduce the number of hash functions to one, and utilize a primitive password shared between an appliance and HAN controller to construct four valid individual consensus and authenticated symmetric keys between the appliance and upstream controllers by exchanging only 12 packets.

4.1 Introduction

Our proposal is a key agreement protocol for secured access control in a hierarchical architecture for the SG communication infrastructure with different layers between smart appliances in users' premises and upstream controllers of the HANs, BANs, NANs and SG Central Controllers (SGCC), which are located in distribution networks or substations [80]. Typically, the HAN controller is a SM that serves as the gateway to the user's premise. Such a protocol provides a secured means for controllers upstream of the HAN

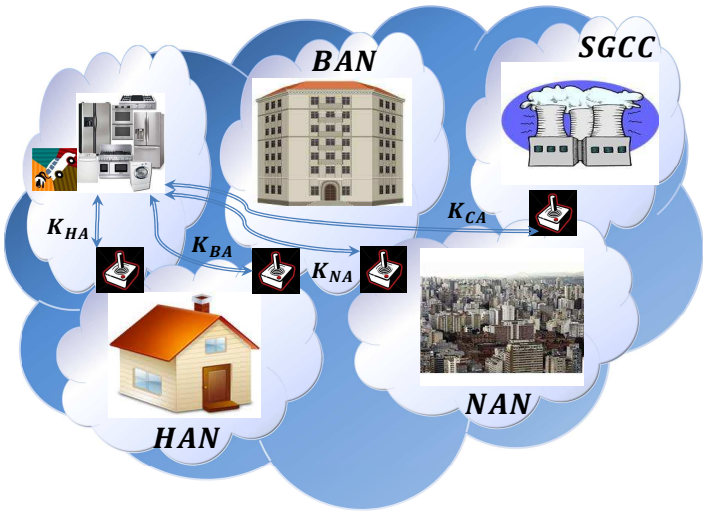


Figure 4.1: Required Symmetric Keys

controllers to access and control the smart appliances in users' premises, e.g., to modify the thermostat setting of the homes heating, ventilation and air condition (HVAC) systems when a brown-out is impending. This study is independent of the technologies used for the SG communications; i.e., our work is equally applicable whether PLC or wireless technologies are used in any of the layers.

Various existing controlling commands that may be sent to a smart appliance from outside the HAN have been considered in [81]. For instance, a NAN controller (located outside a HAN) may supervise electric charging of a plug-in electric vehicle (PEV) located inside the HAN. Also, in the case of a disaster or an emergency, SGCC may need to remotely turn off low-priority high-demand appliances. In such operations, the HAN controllers should not interfere with and delay such commands by decrypting and re-encrypting the corresponding packets. Therefore, we need to address the appropriate secrecy level in the SG control system design while providing the quality of service (QoS) required in terms of keeping the command-response delay within an acceptable limit.

Contributions: In this chapter, we present two protocols. The first one is an auxiliary model of an ECC based PAKE protocol (EPAK protocol) that can be used in any environment and application. The second one, which

is our main work, is a Multilayer Consensus EPAK (MCEPAK) protocol developed for communications in the SG control system.

The scope of the work is shown in Figure 4.1, which addresses communications in up to four layers between a home appliance A_N, HAN controller H_C, BAN controller B_C, NAN controller N_C and SGCC C_C. We consider the SG controllers with the hierarchical architecture share common secrets and are designed to be trust-worthy to each other. Precisely, we assume that controllers have a pre-established trust relationship; i.e., they are already authenticated to the upstream and downstream controllers if any, and are able to communicate with the neighbours in a secure fashion. When a smart appliance joins a HAN, it also needs to share a common secret (assumed to be a simple password) with the HAN controller for it to be trusted in the HAN. The question is how to extend this trust to multiple controllers in a secure and efficient manner. Moreover, our proposal addresses the requirement that each controller needs to set up a secure and private communication channel with A_N, with any controllers in between simply acting as a part of the communication connection without participating in the security operations. Based on assumption of sharing a primitive password by A_N and H_C, we derive four individual consensus password-authenticated symmetric keys between A_N and the upstream controllers.

4.2 Literature Review

In [82], a light-weight and robust PAKE for smart card (SC) is presented, which identifies and delivers an entity-server mutual authentication. The scheme supports only one SC per device and requires SC management. Furthermore, each SC is only utilized for two-party authentication, which limits its usefulness in SG. In [83], a three steps PAKE protocol is presented to resist dictionary, password compromise impersonation and ephemeral key compromise impersonation attacks, and to supply forward secrecy. The mechanism presented in [84] reduces the number of required hash functions while changing the parameters accordingly, which is a concept that we use in our design.

The ECKE-1 protocol [7] improves on the previous proposals in the construction of a mutual authenticated shared symmetric key between two parties, utilizing EC techniques. The ECKE-1 mechanism uses three point multiplications and two field multiplications along with twice applications of a hash function. Author of [8] provided a password based remote authentication scheme for SC based on ECDH. The model aims at providing

a lightweight solution for devices with limited resources, which eliminated needs to keep the password table in server side in order to reduce the side effect of compromising the server. Also, the mechanism presented in [9] concentrated on two-party key designed for sensor networks that contain low-resource devices with a heterogeneous large-scale deployment, based on EC and key management based on AVL tree. The next two-party ECDH based mechanism is ECKE-1N presented in [10], which improves ECKE-1 by constructing the key via 2.5 point multiplications, one field multiplication and twice hash function applications. Later on, EECKE-1N [11] further improves ECKE-1N by constructing the key via only one point multiplication and one field multiplication to construct the two-party key.

Although the above mechanisms are designed efficiently and follow ECDH, mostly they use a predefined private and public key, which requires the support of certificates issued by a certificate authority. In this work, we design an ECC-based model of the PAKE protocol (as in X.1035 standard) called EPAK, which uses only a predefined password and delivers an improvement on ECKE-1N and EECKE-1N mechanisms. We utilize EPAK as our auxiliary protocol for our main MCEPAK protocol.

4.3 EPAK: ECC-Based Password Authenticated Key-exchange Protocol

In this section, we present the EPAK protocol, which is designed as an ECC version of the PAKE protocol presented in the *X.1035* standard.

Table 4.1: EPAK Parameters

Parameter	Description
a and b	Two field elements that define the equation of EC.
p	The field size.
G	An ECC point that generates the subgroup of order n.
n	The order of the point G.
h	The order of EC divided by n.
x_W and y_W	Two elements of the finite field of size p (in the range of $[0, p-1]$), which are the x and y coordinators of point W.
d_W	Private key of party W, which are integers in range $[2, n-1]$.
Q_W	Public key of party W.
S_W and T_W	Verifiers generated by party W.
$U = E_{k_e}(V)$	U is encryption of V using key k_e.
$V = D_{k_d}(U)$	V is decryption of U using key k_d.

Let us consider key agreement between Alice and Bob utilizing a pre-shared password pw. Similar to the *X.1035* standard, we define $P = (ID_A|ID_B|pw)$. Furthermore, we assume that both parties have knowledge of the EC parameters set $\{a, b, p, G, n, h\}$ and hash function $\widetilde{H}$. Table 4.1 presents the list of parameters and their definitions used in our design.

4.3.1 Description of EPAK Protocol

Shown by Figure 4.2, the EPAK protocol has the following steps:

Step I

Alice Let us assume that Alice is the initiator. She picks a random number $d_A \in [2, n-1]$ (as her private key) and multiply it to the group generator G to obtain her public key Q_A via (4.1) and an appropriate EC point (x_a, y_a) via (4.2). Finally, she computes $\widetilde{H}(P)$ to obtain a symmetric key with which

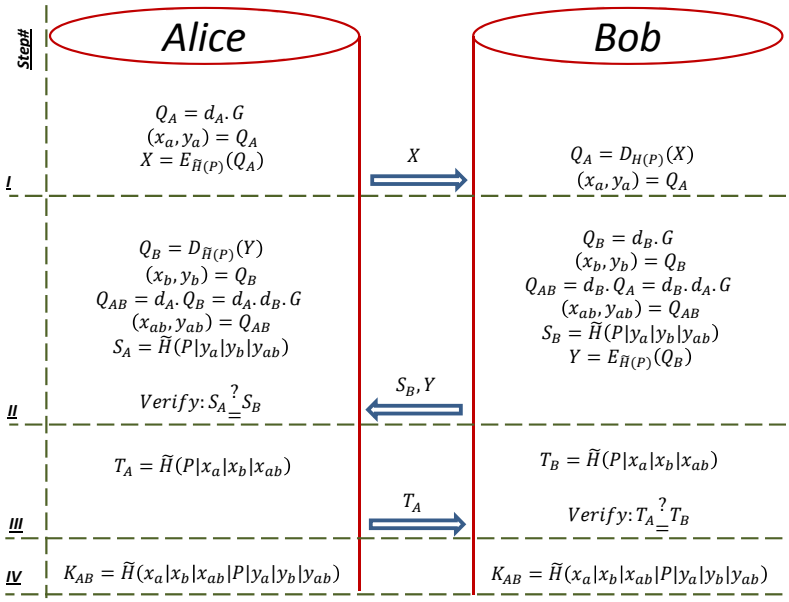


Figure 4.2: ECC-Based PAKE (EPAK) Protocol

she encrypts Q_A as X via (4.3) and sends it to Bob.

$$Q_A = d_A.G \tag{4.1}$$
$$(x_a, y_a) = Q_A \tag{4.2}$$
$$X = E_{\widetilde{H}(P)}(Q_A) \tag{4.3}$$

Bob Upon receiving packet X from Alice, Bob uses $\widetilde{H}(P)$ to decrypt X and obtain Q_A following (4.4), and the appropriate EC point (x_a, y_a) aligned with the Q_A value shown by (4.2).

$$Q_A = D_{\widetilde{H}(P)}(X) \tag{4.4}$$

Step II

Bob Bob picks a random number $d_B \in [2, n-1]$ (as his private key) and multiplies it to the group generator G to obtain his public key Q_B via (4.5). He also calculates the appropriate EC point (x_b, y_b) aligned with the Q_B value based upon (4.6):

$$Q_B = d_B.G \tag{4.5}$$
$$(x_b, y_b) = Q_B \tag{4.6}$$

Then, he multiplies his private key to the Alice's public key to obtain shared value Q_{AB} through (4.7), and finds appropriate EC points (x_{ab}, y_{ab}) as per (4.8). Then, he computes S_B for the verification of having the values Q_A, Q_B & Q_{AB} through (4.9), and finally, uses $\widetilde{H}(P)$ to encrypt Q_B via (4.10), and sends it to Alice.

$$Q_{AB} = d_B.Q_A = d_B.d_A.G \tag{4.7}$$
$$(x_{ab}, y_{ab}) = Q_{AB} \tag{4.8}$$
$$S_B = \widetilde{H}(P|y_a|y_b|y_{ab}) \tag{4.9}$$
$$Y = E_{\widetilde{H}(P)}(Q_B) \tag{4.10}$$

Alice Alice uses $\widetilde{H}(P)$ to decrypt Y and obtains Q_B through (4.11), and also computes the appropriate EC point (x_b, y_b) aligned with the Q_B given by (4.6). Then, she multiplies her private key to Bob's public key (Q_B) to obtain shared value Q_{AB} via (4.12), followed by (x_{ab}, y_{ab}) given by (4.8). Finally, she computes S_A for verification of having the values of

Q_A, Q_B & Q_{AB} through (4.13). If the verification holds, she can be sure that Bob has the required values.

$$Q_B = D_{\widetilde{H}(P)}(Y) \tag{4.11}$$

$$Q_{AB} = d_A.Q_B = d_A.d_B.G \tag{4.12}$$

$$S_A = \widetilde{H}(P|y_a|y_b|y_{ab}) \tag{4.13}$$

Step III

Alice Alice needs to make Bob assure that she has the values as well. Therefore, she performs (4.14) to calculate T_A out of Q_A, Q_B & Q_{AB}, and sends it to Bob.

$$T_A = \widetilde{H}(P|x_a|x_b|x_{ab}) \tag{4.14}$$

Bob On the other side, Bob calculates T_B via (4.15) and compares it with T_A. If the verification holds, Bob is assured that Alice has the required values as well.

$$T_B = \widetilde{H}(P|x_a|x_b|x_{ab}) \tag{4.15}$$

Step IV

So far, both parties have the required parameters and have verified each other. Finally, they perform (4.16) to calculate the shared symmetric key.

$$K_{AB} = \widetilde{H}(x_a|x_b|x_{ab}|P|y_a|y_b|y_{ab}) \tag{4.16}$$

4.3.2 A few Comments About the EPAK Protocol

In the first verification initiated by Bob (S_B & S_A), we use only x coordinates (and P). Furthermore, in the second verification initiated by Alice (T_A & T_B), only y coordinates are used (and P). However, in the final step and for calculating the symmetric key K, we have a combination of all values consisting of x and y coordinates as well as the initial password and identities of the parties (as part of the P).

Thus far, we have defined neither the mechanism of $U = E_{\widetilde{H}(P)}(V)$ nor $V = D_{\widetilde{H}(P)}(U)$ in the aforementioned design. Since V is a point in the form of (x_v, y_v), for instance to encrypt this pair, we can multiply each coordinate by $\widetilde{H}(P)$:

$$C_T : \begin{cases} x_u = \widetilde{H}(P).x_v \\ y_u = \widetilde{H}(P).y_v \end{cases}$$

Consequently, on the other side we need to divide them by the $\widetilde{H}(P)$ and obtain the original values:

$$P_T : \begin{cases} x_v = \dfrac{x_u}{\widetilde{H}(P)} = \dfrac{\widetilde{H}(P).x_v}{\widetilde{H}(P)} = x_v \\ y_v = \dfrac{y_u}{\widetilde{H}(P)} = \dfrac{\widetilde{H}(P).y_v}{\widetilde{H}(P)} = y_v \end{cases}$$

4.3.3 Brief Analysis of the EPAK Protocol

Comparing to the previous models, we eliminate the fixed initial private key. To be more precise, each party chooses a random number to be the private key per session. Also, our mechanism constructs the key only via one multiplication given by (4.7)/(4.12), and one hash function for the key in (4.16). In fact, other times that hash function is utilized are for verification purposes; this is an add-on to the ECKE-1N and EECKE-1N protocols. Furthermore, other multiplications in (4.10) and (4.11) are for encryption, whereas exchanged packets are not encrypted at all in ECKE-1N and EECKE-1N.

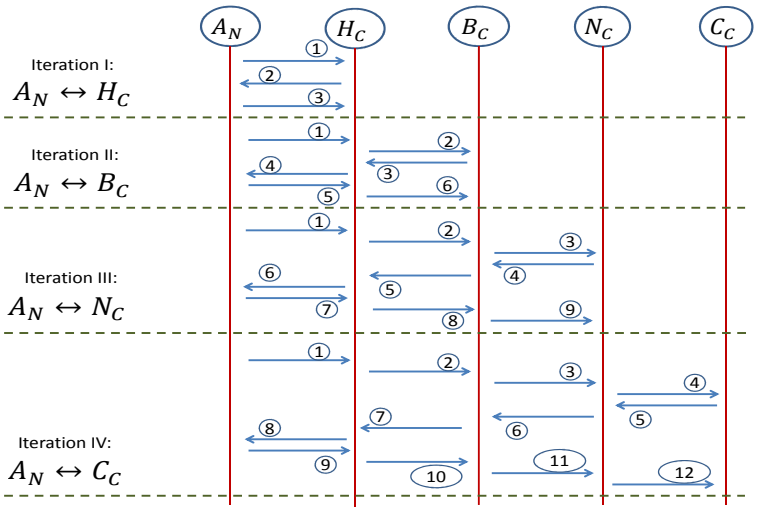


Figure 4.3: Four Keys Construction Based on PAKE or EPAK

4.4 Multilayer Consensus ECC-Based Password Authenticated Key-exchange Protocol

Referring to our previous discussion, our objective is a mutual password-based authenticated key agreement between an appliance A_N and the controllers H_C, B_C, N_C and C_C, resulting in an individual key between the appliance and each one of the upper layer controllers. Based on Figure 4.1, we need four symmetric keys. Our model based on PAKE protocol presented in X.1035 standard (or EPAK in Section 4.3) is shown in Figure 4.3. In this abstract model and for each key, we need to have a predefined shared password between the two involved parties (appliance and one of the controllers). We will show that our approach decreases the number of packets and improves the security of the design. In this section, we extend our EPAK protocol in order to address the SG requirements, based on the EPAK protocol, using the same notations as presented in the previous sections. The appliance A_N knows at least the ID of the HAN and can obtain ID of the H_C. Also via our four-phase mechanism, A_N gains access to information of the other controllers. We assume:

- A_N and H_C share a predefined secret password pw.
- The ECC parameter set $\{a, b, p, G, n, h\}$ and $\widetilde{H}(.)$ are known and shared by all parties.
- Controllers H_C, B_C, N_C & C_C have already been authenticated to the upstream and downstream controllers, if any, and can have secure communications with them.
- Controllers are trusted parties that form parts of and are controlled by the grid domain; the appliance belongs to the customer domain, and is controlled by the customer.
- The following symmetric keys already exist:
  - k_{hb}: Shared between H_C and B_C.
  - k_{bn}: Shared between B_C and N_C.
  - k_{nc}: Shared between N_C and C_C.

Note: Although the secure channels that we assume to already exist between the controllers may not be certain, we need to trust it, otherwise *"we would be unable to ever get any work done"* [85]. Furthermore, the

trust assumption is especially feasible in the multilayer architecture of SG controllers [80] considered in this chapter.

Furthermore, we introduce a new vector $\widehat{V}$ (entities identifications set), which carries the IDs of the entities involved in our protocol as a part of the information exchanged between them. Our four-phase MCEPAK protocol depicted in Figure 4.4 consists of the following steps.

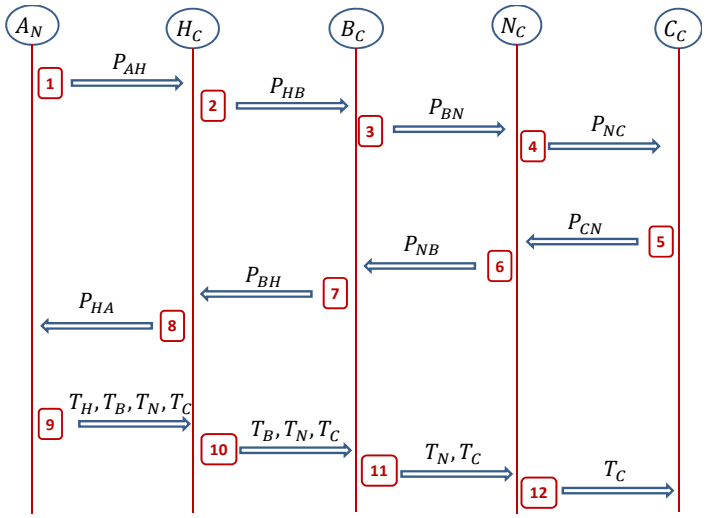


Figure 4.4: MCEPAK Protocol Phases and Packets

Phase I: Initial Flow

In MCEPAK, A_N initiates the keys establishment process:

First packet

Firstly, A_N follows (4.17) to utilize the initial password pw shared by H_C to calculate temporary key k_{ah}^t.

$$k_{ah}^t = \widetilde{H}(ID_A|pw|ID_H) \tag{4.17}$$

A_N also picks a random number $d_A \in [2, n-1]$, then computes Q_{AH} via (4.18) and appropriate coordinates (x_a, y_a) given by (4.19).

$$Q_{AH} = d_A.G \tag{4.18}$$

$$(x_a, y_a) = Q_{AH} \tag{4.19}$$

Then, A_N puts its own ID in field A of $\widehat{V}$ given by (4.20). Finally, A_N forms packet P_{AH} by Q_{AH} and $\widehat{V}$ all encrypted by the k_{ah}^t key as per (4.21), and then sends the packet to H_C.

$$\widehat{V}.[A] \leftarrow ID_A \tag{4.20}$$

$$P_{AH} = E_{k_{ah}^t}(Q_{AH}, \widehat{V}) \tag{4.21}$$

Second packet

First, H_C calculates temporary key k_{ah}^t by performing (4.17), and decrypts received packet from A_N by way of (4.22) to obtain Q_{AH} and $\widehat{V}$.

$$(Q_{AH}, \widehat{V}) = D_{k_{ah}^t}(P_{AH}) \tag{4.22}$$

Then, H_C picks a random number $d_H \in [2, n-1]$ and computes Q_{HB} through (4.23).

$$Q_{HB} = (Q_{AH}).d_H = (d_A.G).d_H = d_A.d_H.G \tag{4.23}$$

$$(x_{hb}, y_{hb}) = Q_{HB} \tag{4.24}$$

Then, H_C puts its own ID into field H of $\widehat{V}$ by way of (4.25), and also computes pw_b via (4.26).

$$\widehat{V}.[H] \leftarrow ID_H \tag{4.25}$$

$$pw_b = \widetilde{H}(k_{ah}^t | ID_B) \tag{4.26}$$

Finally, H_C dispatches $\widehat{V}$ along with Q_{HB} and pw_b to B_C, all encrypted with the k_{hb} shared key following (4.27).

$$P_{HB} = E_{k_{hb}}(Q_{HB}, \widehat{V}, pw_a) \tag{4.27}$$

Third packet

First, B_C obtains Q_{HB}, $\widehat{V}$ and pw_b by decryption of the received packet P_{HB} from H_C via (4.28):

$$(Q_{HB}, \widehat{V}, pw_b) = D_{k_{hb}}(P_{HB}) \tag{4.28}$$

Then, B_C chooses random number $d_B \in [2, n-1]$ and computes Q_{BN} through (4.29):

$$Q_{BN} = (Q_{HB}).d_B = (d_A.d_H.G).d_B = d_A.d_H.d_B.G \tag{4.29}$$

$$(x_{bn}, y_{bn}) = Q_{BN} \tag{4.30}$$

Then, B_C copies its own ID into the $\widehat{V}$ field B in (4.31), and computes pw_n via (4.32). Finally, B_C forwards $\widehat{V}$, Q_{BN} and pw_n to N_C, all encrypted with the predefined shared key of k_{bn} through (4.33).

$$\widehat{V}.[B] \leftarrow ID_B \tag{4.31}$$
$$pw_n = \widetilde{H}(pw_b|ID_N) \tag{4.32}$$
$$P_{BN} = E_{k_{bn}}(Q_{BN}, \widehat{V}, pw_n) \tag{4.33}$$

Fourth packet

Firstly, N_C follows (4.34) to obtain Q_{BN}, $\widehat{V}$ and pw_n from the packet P_{BN} received from B_C:

$$(Q_{BN}, \widehat{V}, pw_n) = D_{k_{bn}}(P_{BN}) \tag{4.34}$$

Then, N_C chooses random number $d_N \in [2, n-1]$ to obtain Q_{NC} via (4.35).

$$Q_{NC} = (Q_{BN}).d_N = (d_A.d_H.d_B.G).d_N = d_A.d_H.d_B.d_N.G \tag{4.35}$$
$$(x_{nc}, y_{nc}) = Q_{NC} \tag{4.36}$$

Then, N_C updates $\widehat{V}$ field N with its own ID as depicted by (4.37), also computes pw_c through (4.38).

$$\widehat{V}.[N] \leftarrow ID_N \tag{4.37}$$
$$pw_c = \widetilde{H}(pw_{an}|ID_C) \tag{4.38}$$

Finally, N_C forms packet P_{NC} out of $\widehat{V}$, Q_{NC} and pw_c as shown by (4.39), encrypts it by k_{nc}, and forwards it to C_C.

$$P_{NC} = E_{k_{nc}}(Q_{NC}, \widehat{V}, pw_c) \tag{4.39}$$

Phase II: Response Flow

This flow starts with C_C replying to the fourth packet above.

Fifth packet

First, C_C obtains the Q_{NC}, $\widehat{V}$ and pw_c values by decryption of the packet P_{NC} received from the N_C following (4.40).

$$(Q_{NC}, \widehat{V}, pw_c) = D_{k_{nc}}(P_{NC}) \tag{4.40}$$

Then, C_C extracts ID of any of the controllers (if needed) as well as ID of the appliance ID_A from $\widehat{V}$ ($\widehat{V}.[A]$), and also calculates k_{ca}^t through (4.41). Beside, C_C inserts its own ID into field C of $\widehat{V}$ as presented by (4.42).

$$k_{ca}^t = \widetilde{H}(ID_C|pw_{ac}|ID_A) \tag{4.41}$$

$$\widehat{V}.[C] \leftarrow ID_C \tag{4.42}$$

Then, C_C picks a random number $d_C \in [2, n-1]$ to obtain Q_C and Q_{CC} following (4.43) and (4.44) respectively.

$$Q_C = d_C.G \tag{4.43}$$

$$Q_{CC} = (Q_{NC}).d_C = (d_A.d_H.d_B.d_N.G).d_C = d_A.d_H.d_B.d_N.d_C.G \tag{4.44}$$

$$(x_c, y_c) = Q_{CC} \tag{4.45}$$

Then, C_C obtains coordinates (x_c, y_c) as shown by (4.45) and (x_{nc}, y_{nc}) as depicted by (4.36), and then computes S_{CN} via (4.46) for verification purpose.

$$S_{CN} = \widetilde{H}(k_{ca}^t|y_{nc}|y_c) \tag{4.46}$$

Finally, C_C follows (4.47) to form P_{CN} from S_{CN}, Q_C and $\widehat{V}$, in which C_C encrypts the packet by k_{nc} as shown in (4.47).

$$P_{CN} = E_{k_{nc}}(S_{CN}, Q_C, \widehat{V}) \tag{4.47}$$

Sixth packet

First, N_C decrypts the packet received from C_C to obtain the S_{CN}, Q_C and $\widehat{V}$ values following (4.48). Then, N_C calculates k_{na}^t through (4.49).

$$(S_{CN}, Q_C, \widehat{V}) = D_{k_{nc}}(P_{CN}) \tag{4.48}$$

$$k_{na}^t = \widetilde{H}(ID_N|pw_{an}|ID_A) \tag{4.49}$$

Then, N_C utilizes its own random number d_N (fourth step) to calculate Q_N via (4.50), and Q_{NC} via (4.51). Then, N_C follows (4.52) to calculate S_{NB} for the verification purpose.

$$Q_N = d_N.G \tag{4.50}$$

$$Q_{NC} = (Q_C).d_N = (d_C.G).d_N = d_N.d_C.G \tag{4.51}$$

$$S_{NB} = S_{CN} \oplus \widetilde{H}(k_{na}^t|y_{bn}|y_{nc}) \tag{4.52}$$

Finally, N_C forms P_{NB} out of S_{NB}, Q_N, Q_{NC} and $\widehat{V}$, and encrypts the packet by k_{bn} as shown in (4.53) to be sent to the BAN controller (B_C).

$$P_{NB} = E_{k_{bn}}(S_{NB}, Q_N, Q_{NC}, \widehat{V}) \tag{4.53}$$

Seventh packet

First, B_C obtains the parameters S_{NB}, Q_N, Q_{NC} and $\widehat{V}$ as presented by (4.54) by decrypting packet received from N_C. Then, B_C calculates the k_{ba}^t key via (4.55).

$$(S_{NB}, Q_N, Q_{NC}, \widehat{V}) = D_{k_{bn}}(P_{NB}) \tag{4.54}$$

$$k_{ba}^t = \widetilde{H}(ID_B|pw_{ab}|ID_A) \tag{4.55}$$

Then, B_C uses its own random number d_B (third step) to obtain the Q_B via (4.56), Q_{BN} through (4.57) and Q_{BNC} via (4.58).

$$Q_B = d_B.G \tag{4.56}$$

$$Q_{BN} = (Q_N).d_B = (d_N.G).d_B = d_B.d_N.G \tag{4.57}$$

$$Q_{BNC} = (Q_{NC}).d_B = (d_N.d_C.G).d_B = d_B.d_N.d_C.G \tag{4.58}$$

Then, B_C obtains coordinates (x_{nc}, y_{nc}) and (x_{bn}, y_{bn}) as shown by (4.36) (4.30) respectively, and calculates S_{BH} through (4.59) for verification.

$$S_{BH} = S_{NB} \oplus \widetilde{H}(k_{ba}^t|y_{hb}|y_{bn}) \tag{4.59}$$

Finally, B_C forms P_{BH} packet by S_{BH}, Q_B, Q_{BN}, Q_{BNC} and $\widehat{V}$, encrypted by k_{nc} as shown in (4.60), and sends the packet to H_C.

$$P_{BH} = E_{k_{hb}}(S_{BH}, Q_B, Q_{BN}, Q_{BNC}, \widehat{V}) \tag{4.60}$$

Eighth packet

First, H_C decrypts the packet received from B_C and obtains S_{BH}, Q_B, Q_{BN}, Q_{BNC} and $\widehat{V}$ as depicted by (4.61). Then, H_C calculates k_{ha}^t through (4.62).

$$(S_{BH}, Q_B, Q_{BN}, Q_{BNC}, \widehat{V}) = D_{k_{hb}}(P_{BH}) \tag{4.61}$$

$$k_{ha}^t = \widetilde{H}(ID_H|pw_{ah}|ID_A) \tag{4.62}$$

Then, H_C utilizes its own random number d_H (second step) to compute Q_H via (4.63), Q_{HB} through (4.64), Q_{HBN} via (4.65) and Q_{HBNC} through (4.66).

$$Q_H = d_H.G \tag{4.63}$$

$$Q_{HB} = (Q_B).d_H = (d_B.G).d_H = d_H.d_B.G \tag{4.64}$$

$$Q_{HBN} = (Q_{BN}).d_H = (d_B.d_N.G).d_H = d_H.d_B.d_N.G \tag{4.65}$$

$$Q_{HBNC} = (Q_{BNC}).d_H = (d_B.d_N.d_C.G).d_H = d_H.d_B.d_N.d_C.G \tag{4.66}$$

H_C obtains coordinates (x_{bn}, y_{bn}) and (x_{hb}, y_{hb}) as depicted by (4.30) and (4.24), respectively, and then computes S_{HA} using (4.67) for verification.

$$S_{HA} = S_{BH} \oplus \widetilde{H}(k_{ha}^t | y_a | y_{hb}) \tag{4.67}$$

Finally, H_C forms P_{HA} packet out of S_{HA}, Q_H, Q_{HB}, Q_{HBN}, Q_{HBNC} and $\widehat{V}$, encrypted by k_{ha}^t as shown by (4.68), and sends the packet to A_N.

$$P_{HA} = E_{k_{ha}^t}(S_{HA}, Q_H, Q_{HB}, Q_{HBN}, Q_{HBNC}, \widehat{V}) \tag{4.68}$$

Phase III: Verification

Appliance (ninth packet)

In this phase, A_N verifies the received values and dispatches the confirmations to the upstream controllers. First, A_N computes the k_{ha}^t temporary key via (4.62), to decrypt the received packet P_{HA} from H_C in order to obtain S_{HA}, Q_H, Q_{HB}, Q_{HBN}, Q_{HBNC} and $\widehat{V}$ following (4.69).

$$(S_{HA}, Q_H, Q_{HB}, Q_{HBN}, Q_{HBNC}, \widehat{V}) = D_{k_{ha}^t}(P_{HA}) \tag{4.69}$$

Then, A_N utilizes its own random number d_A (first step) to calculate Q_{HB} via (4.70), Q_{BN} through (4.71), Q_{NC} via (4.72) and Q_{CC} through (4.73), which are shared by H_C, B_C, N_C and C_C, respectively.

$$(Q_H).d_A = (d_H.G).d_A = d_A.d_H.G = Q_{HB} \tag{4.70}$$
$$(Q_{HB}).d_A = (d_H.d_B.G).d_A = d_A.d_H.d_B.G = Q_{BN} \tag{4.71}$$
$$(Q_{HBN}).d_A = (d_H.d_B.d_N.G).d_A = d_A.d_H.d_B.d_N.G = Q_{NC} \tag{4.72}$$
$$(Q_{HBNC}).d_A = (d_H.d_B.d_N.d_C.G).d_A = d_A.d_H.d_B.d_N.d_C.G = Q_{CC} \tag{4.73}$$

Then, A_N uses the above shared values to obtain coordinates (x_c, y_c), (x_{nc}, y_{nc}), (x_{bn}, y_{bn}) and (x_{hb}, y_{hb}) as shown in (4.45), (4.36), (4.30) and (4.24), respectively. Then, A_N utilizes the coordinates and performs (4.46), (4.52), (4.59) and (4.67) to substantiate S_{HA}. If the verification holds, A_N proceeds to the next step. Note that, since A_N has pw, it is able to obtain pw_b, pw_n & pw_c based upon (4.26), (4.32) & (4.38). Finally, A_N generates four values T_{AH} via (4.74) for H_C, T_{AB} through (4.76) for B_C, T_{AN} via (4.78) for N_C and T_{AC} through (4.80) for C_C, as verifiers of the shared values, and forwards

them to H_C.

$$
\begin{aligned}
T_{AH} &= \widetilde{H}(k_{ah}^t|x_a|x_{hb}) && (4.74)\\
k_{ab}^t &= \widetilde{H}(ID_A|pw_{ab}|ID_B) && (4.75)\\
T_{AB} &= \widetilde{H}(k_{ab}^t|x_{hb}|x_{bn}) && (4.76)\\
k_{an}^t &= \widetilde{H}(ID_A|pw_{an}|ID_N) && (4.77)\\
T_{AN} &= \widetilde{H}(k_{an}^t|x_{bn}|x_{nc}) && (4.78)\\
k_{ac}^t &= \widetilde{H}(ID_A|pw_{ac}|ID_C) && (4.79)\\
T_{AC} &= \widetilde{H}(k_{ac}^t|x_{nc}|x_c) && (4.80)
\end{aligned}
$$

HAN controller (tenth packet)

H_C receives the above substantiation values and then verifies T_{AH} based upon (4.74). If the verification holds, H_C relays the other values to B_C.

BAN controller (eleventh packets)

B_C receives the above values and then verifies T_{AB} following (4.76). If the verification holds, B_C relays the other values to N_C.

NAN controller (twelfth packets)

N_C receives the eleventh packet and then verifies T_{AN} through (4.78). If the verification holds, N_C relays the other values to C_C.

SGCC controller

C_C receives the twelfth packet and then verifies T_{AC} via (4.80).

Phase IV: Keys Calculation

Thus far, all parties have their verified shared values. Finally, they can generate their appropriate symmetric keys per (4.81), (4.82), (4.83) and (4.84).

$$
\begin{aligned}
A_N \;\&\; H_C &: K_{HA} = \widetilde{H}(x_a|x_{hb}|k_{ha}^t|k_{ah}^t|y_a|y_{hb}) && (4.81)\\
A_N \;\&\; B_C &: K_{BA} = \widetilde{H}(x_{hb}|x_{bn}|k_{ba}^t|k_{ab}^t|y_{hb}|y_{bn}) && (4.82)\\
A_N \;\&\; N_C &: K_{NA} = \widetilde{H}(x_{bn}|x_{nc}|k_{na}^t|k_{an}^t|y_{bn}|y_{nc}) && (4.83)\\
A_N \;\&\; C_C &: K_{CA} = \widetilde{H}(x_{nc}|x_c|k_{ca}^t|k_{ac}^t|y_{nc}|y_c) && (4.84)
\end{aligned}
$$

4.5 Analysis

Since the proposed MCEPAK protocol is based on ECC, X.1035 standard and D-H algorithm, it inherits most of their benefits. In this section, we study and model the adversary, analyze the security of the system mainly in terms of different attacks, and evaluate the security of the keys. To analyze and evaluate the security of our proposed protocol, we follow the Dolev-Yao approach [25]. In the Dolev-Yao model, the adversary is capable of recording, deleting, re-playing, re-routing, re-ordering and re-scheduling the messages. All of the messages generated by the honest parties are sent to the adversary, and the honest nodes receive the messages only from the adversary. Also, we analyse the protocol mechanism from the system and network overhead point of views.

4.5.1 Adversary Models

We consider two models for internal and external adversaries.

Internal adversary

In this model, our adversary is one of the trusted parties that has become malicious.

Objective The objectives of the adversary are (i) Gaining access to the system resources such as the appliance or any of the controllers, (ii) Performing a MITM attack to gain access to any of the keys.

Initial capabilities The adversary has complete knowledge about the topology and the exact address/ID of each party. Furthermore, the adversary knows the detail design of the key agreement mechanism and has access to the system parameters required for the key agreement. Depending on which one of the involved parties is the malicious one, the adversary's knowledge in each case is listed in Table 4.2.

Capabilities during the attack The adversary receives the encrypted and unencrypted (plain) data in different stages of the keys agreement, or later on during the using of the key. In case of having control on a controller, the adversary will attempt to perform a MITM attack. She/He can destroy the packets and cause failure in one of the verification phases that yields re-initiation of the key agreement protocol, which is essentially a DoS attack.

Table 4.2: Internal Adversary Knowledge

Party	Knowledge
A_N	The initial shared password pw.
H_C	pw and shared symmetric key with B_C
B_C	Shared symmetric keys with H_C and with N_C
N_C	Shared symmetric keys with B_C and with C_C
C_C	Shared symmetric key with N_C

Furthermore in case of a malicious A_N, the adversary can perform DoS attack by initiating the key agreement protocol continuously.

Discussion Referring to the assumptions of the MCEPAK protocol, controllers are fully trusted parties as parts of the SG domain, and they are controlled/setup/managed by the grid administrators. Even a HAN controller, which is usually a SM that also acts as a gateway, is not under customer control. Therefore, initially they follow the steps of the algorithm and do not show any misbehaving action. Also, the administrator of the SG monitors them to protect the SG from any malicious controller. So, dealing with malicious controllers is beyond the scope of this chapter.

However, a malicious A_N can perform a DoS attack easily, for instance by failing the verification phase. To prevent it, the system can define a limit of the key agreement sessions per appliance in each period of time. If after a number of tries, still the appliance could not finish the process, it means that either the node is malicious or the initial password between A_N and H_C does not match. Therefore, the system stops the appliance and cancels its future tries. Having said this, the attack can be detected at, e.g., H_C level as long as A_N uses the same ID. Initiating the key construction sessions by different IDs is another option for A_N to perform DoS attack against the HAN controller. In this case, H_C can limit the number of open sessions in the HAN domain to prevent such an attack.

External adversary

In this model, the adversary is not any of the involved parties, and performs attacks from outside of the controllers and appliance set.

Objective (i) Gaining access to the system resources, like any of the controllers or A_N. (ii) Performing a MITM attack to gain access to any of the

symmetric keys. (iii) Performing a DoS attack to overload the system (any of the controllers).

Initial capabilities Similar to the internal model, the external adversary has complete knowledge about the topology and the exact address/ID of the parties. The adversary also knows the detail design of our mechanisms.

Capabilities during the attack The adversary receives the encrypted and unencrypted (plain) packets during and after the key agreement process.

Discussion Since our adversary receives all the packets, in order to gain access to the system resources, like A_N, she/he can perform a brute-force attack to find out the pre-shared password between the appliance and the HAN controller. Brute-force attack is a time consuming attack, and the password is used only during the first few packet delivery between the two parties (A_N and H_C). Furthermore, we use a hash function to combine the password and random numbers to construct the key. Therefore, our model (similar to PAKE protocol) has the forward secrecy characteristic. Having said this, finding the password does not help the adversary to figure out or calculate any of the symmetric keys. The same situation is applicable to any of the original keys between the controllers. Indeed, obtaining any of the shared keys between the controllers does not help the adversary to calculate the constructed symmetric key after the fact. This is because the packets exchanged by our mechanism do not include all the items required for the key calculation. Besides, the aforementioned discussion shows that an adversary cannot perform a successful MITM attack.

To perform a DoS attack and overload the system (any controller), our adversary should initially run a spoof attack to masquerade one of the parties as well as performing a brute-force or dictionary attack to steal the shared key between the party and its neighbour. Then, she/he should manage sending the key agreement packets to the neighbour to perform the DoS attack. Even if the adversary is able to manage these attacks, the system can limit the number of requests from any ID for the key construction and prevents this scenario. Any misbehaviour can be monitored by other controllers, where an intrusion detection system like [86] would help in this regard.

4.5.2 Security Analysis

Consensus key establishment

Referring to the key construction (4.81), (4.82), (4.83) and (4.84), we need contribution of other downstream controllers for each controller. For instance in the case of K_{CA} as per (4.84), we utilize (x_c, y_c) that contains random numbers chosen by C_C & A_N as well as random numbers of H_C, B_C & N_C (4.44). Similarly, K_{NA} (4.83) uses random variables N_C & A_N as well as H_C & B_C (4.35). Furthermore, A_N verifies the received shared values all at the same time by checking S_{HA}. Therefore, if any of the controllers does not cooperate on the key constructions, none of the parties would have an appropriate key.

Mutual authentication

The utilization of password pw provides a mutual authentication between A_N and H_C. Furthermore, since pw is a part of the pw_b, pw_n & pw_c calculations that are used by other controllers for the verification and key formation, the mutual authentication is endorsed on the entire key-set.

Hierarchical/Conditional key formation

A_N needs to establish a symmetric key shared by H_C in order to establish a key with any of the higher layers controllers. Furthermore, all of the key construction are initiated by A_N and are forwarded to H_C as a gateway. To be more precise, only the downstream (and not the upstream) controllers random numbers are required by each controller.

Replay attack

Like D-H algorithm, since MCEPAK utilizes random numbers and hash functions to establish the keys, it delivers the replay attack resilience.

Key privacy and insider attack resilience

Depicted by (4.81)-(4.84), each key is only known by A_N and the corresponding controller. For instance, K_{NA} is only known by A_N and N_C, which supports the key privacy. Other controllers in-between only attend in the key construction; however, they do not gain access to any data to be used to decrypt the messages.

Off-line guessing attack resilience

An eavesdropper may perform an off-line dictionary attack over password pw by having access to the $\widetilde{H}$ (4.18) and obtains access to Q_A; nevertheless, based on ECC-CDH (Elliptic Curve Cryptography Cofactor Diffie-Hellman) assumption [6], s/he is not able to find d_A. As a result, the adversary does not have complete information and data to compute any of the keys.

Denning-Sacco attack resilience

If an eavesdropper gains access to $\widetilde{H}(ID_i|pw_x|ID_j)$, still she/he is not able to obtain pw_x value used in the key establishment. Furthermore, she/he is not able to guess $\widetilde{H}(ID_k|pw_y|ID_l)$ where $i \neq k$ & $j \neq l$ & $\forall x, y$.

Compromised impression resilience

Referring to our adversary models as well as (4.81)-(4.84), finding any of the keys does not enable an intruder to obtain any other controllers key.

Ephemeral key compromise impersonation

Even if an adversary finds any of the pw_u passwords, she/he is not able to calculate Q_{Az} since she/he does not have access to the random number d_A. Also, Q_{Az} is required by the her/him to obtain K_{zA} (z is a controller).

Unknown key-share attack

S_{HA} assures A_N that the controllers have the required parameters to calculate the keys. On the other hand, parameters T_{AH}, T_{AB}, T_{AN} & T_{AC} assure the controllers that A_N has the required values. So, if a controller is able to perform the key formation, the appliance would be able to do it too, and wise versa.

MITM attack

Per the aforementioned discussion on the adversary models, since all of the packets between A_N & H_C are encrypted by pw based temporary keys k^t_{aZ}, MCEPAK enjoys MITM attack resilience. Also, the communications between the controllers required for key formation are secured by the preliminary keys of the controllers.

4.5.3 Formal Validation Using Software Tool

We apply AVISPA to analyse appropriate shared keys of the five entities A_N, H_C, B_C, N_C and C_C. Simulation results presented in Figure 4.5a and Figure 4.5b show that the symmetric keys constructed by our mechanism are secure and safe to be used by the system entities.

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL

/ubc/ece/home/vl/grads/hasennic/De
sktop/avispa-
1.1/testsuite/results/MCEPAK.if

GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.03s
  visitedNodes: 5 nodes
  depth: 2 plies
```

(a) OFMC

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
/ubc/ece/home/vl/grads/hasennic/De
sktop/avispa-
1.1/testsuite/results/MCEPAK.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS
  Analysed   : 3 states
  Reachable  : 1 states
  Translation: 0.00 seconds
  Computation: 0.00 seconds
```

(b) Cl-AtSe

Figure 4.5: AVISPA Results

The evaluation program and AVISPA related HLPSL codes for the session, environment and goal sections, as well as each party HLPSL related codes are presented in Appendix A.

4.5.4 Performance Analysis

Low implementation cost

Let us consider the following two scenarios:

- *Sen*.1: A_N establishes an individual symmetric key by each controller, following the PAKE or EPAK protocols.
- *Sen*.2: A_N follows MCEPAK protocol.

An overall comparison between the two scenarios is presented by Table 4.3. Based on Figure 4.3, *Sen*.1 performs four iterations to construct four symmetric keys between appliance A_N and the upstream controllers.

Table 4.3: Overhead Improvement

Scenario	Hash Function	Password	Phase	Random number	Transferred packet between entities
Sen.1	5	4	16	8	30
Sen.2	1	1	4	5	12
Improvement	80%	75%	75%	37.5%	60%

Also, referring to Sections 4.2 and 4.3, one password between A_N and the controller is required for each key agreement. Furthermore, *Sen*.1 proceeds during $4 \times 4 = 16$ phases and requires five hash functions, while *Sen*.2 has only four phases and needs only one hash function and one password. In terms of random numbers, in *Sen*.1 two random numbers per key are required, and in total eight random numbers are needed. On the other hand, *Sen*.2 requires only five random numbers. Also, *Sen*.1 transfers three packets per iteration and in total needs $3+6+9+12 = 30$ packets to be delivered. In contrast, *Sen*.2 needs three packets per phase for a total of $3 \times 4 = 12$ packets.

Fast packet delivery

Packet delivery between A_N and H_C are the same for both scenarios, although MCEPAK provides a faster packet delivery between A_N and the upper-layer controllers for regular communications. Let us define t_0 as the required time for the encryption or decryption process at each party, which is assumed to be the same. Let us consider the following two scenarios:

- *Sen*.3: System has one symmetric key per layer. If a packet is encrypted by controller B_C to be sent to A_N, H_C should decrypt it by the key between B_C & H_C, and then encrypts it again by the key between H_C & A_N. Finally, the packet can be decrypted by A_N using the shared symmetric key between H_C & A_N.
- *Sen*.4: System provides a shared symmetric key between B_C & A_N. Therefore, the packet that needs to be sent by B_C to A_N, needs to be encrypted and decrypted once (by B_C & A_N).

Table 4.4 presents the required time in each case by each controller, and also presents the time saving (improvement).

Table 4.4: Improvement of Encryption/Decryption Time

Scenario	$A_N \leftrightarrow H_C$	$A_N \leftrightarrow B_C$	$A_N \leftrightarrow N_C$	$A_N \leftrightarrow C_C$
*Sen.*3	$2 \times\ t_0$	$4 \times\ t_0$	$6 \times\ t_0$	$8 \times\ t_0$
*Sen.*4	$2 \times\ t_0$	$2 \times\ t_0$	$2 \times\ t_0$	$2 \times\ t_0$
Improvement	0%	50%	66.67%	75%

Note: An analysis and evaluation on EPAK protocol (cost of key agreement) in comparison to the literature is presented at the end of Section 4.3 as well.

Cost

Although there are many benefits of using our proposal, as we discussed about them in above subsections, the controllers need to make sure that they receive the packets on time. In fact, the entire controllers should participate in the key calculations. Somehow, we need to trust that the entities will do their duty. Furthermore, fast responding to the mechanism is another requirement. In fact, the initial password shared between an appliance and HAN controller is not too strong, and only is set up to construct a key in a short time and quick, to prevent attacks such as MITM. If entities delay the mechanism, an adversary can attack the mechanism by e.g. brute-forcing the password, and perform the attack.

Chapter 5

Maintaining Privacy by Using Enhanced Network Coding

In this chapter, we consider the privacy aspect of users in SG system and provide a mechanism that utilizes the advances in network coding to maintain data privacy. We address privacy issues associated with gathering metering information of clients in a SG system. In SG systems, wireless multi-hop communications are mainly used to gather metering information through exchanging data and control messages between SMs and the utility. We argue that any communication paradigm used in a SG should support all aspects of privacy such as anonymity, unlinkability, unobservablity, and undetectablity. We propose innovative schemes for traffic routing and encryption that benefit from the enhanced network coding technology. Note that we use a selective network coding as well as a clustering the topology. These two features enhance our proposal comparing to a full network coding mechanism.

5.1 Introduction

Different communication technologies have been proposed for the AMI such as PLC and wireless communication [27]. As per our previous discussion in previous chapters, in North America, wireless multi-hop communication technologies (e.g., ad-hoc and mesh networks) are proposed to be used for exchanging data and control messages over the AMI between SMs or gateways of HANs and the utility [28, 31, 87–89]. In this case, data traffic is transmitted from a SM to the utility and vice versa over multi-hop wireless links with intermediate network nodes forwarding traffic (Figure 5.1).

Privacy in the SG is identified as one of the biggest concern by the research community, considering the uncertainty in the environment [90]. Although it may be tempting to try to patch existing protocols such as ran-

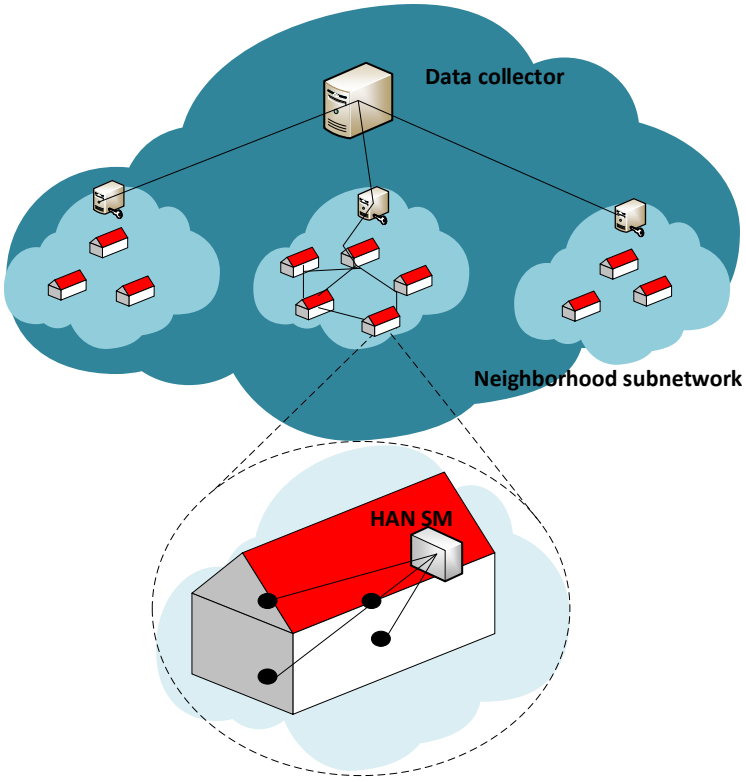


Figure 5.1: Smart Grid Network Architecture

dom paths and anonymous routing to provide some level of privacy [91], the privacy of the users in the SG system needs to consider more precise specifications such as anonymity, unobservability, unlinkability, and undetectability. This requires different designs of traffic routing in order to meet the required privacy properties. For example, when using anonymous routing protocols, an adversary may detect data traffic generated by an individual smart meter to infer information about appliances existed in a HAN (by monitoring trends of power consumed by different appliances), and information about behavior of the users (by monitoring amount of power usage in the HAN). Although a trivial scheme that generates dummy packets may solve the unobservability problem, it fails to address anonymity, unlinkabil-

ity and undetectability while introducing high amount of the overhead to the system. We refer to the Pfitzmann-Hansen definitions of the privacy [92], which we describe in Chapter 1.

Contribution: Our proposed schemes address the problem of preserving privacy of users in a SG system by maintaining all necessary features required for privacy in such a system including anonymity, unlinkability, undetectability and unobservability communications.

None of the existing schemes in the literature simultaneously address all these properties together. We identify five privacy measures for the CPS communication such as hiding source, destination, path, traffic volume and content. We address this problem using an enhanced network coding technique. Our proposed schemes basically benefit from the capability of the network coding in encoding transmitted linear combination of packets.

5.2 Background

Network coding has been widely used to improve the robustness and bandwidth efficiency of multicast routing in special network topologies. However, the inherit feature of packet encryption in the network coding can be exploited to provide privacy for users in a SG. Furthermore, the distributed nature of the network coding increases its robustness against possible attempts of attackers. The simplest coding scheme is linear coding [93, 94]. Linear network coding treats a block of data as a vector over a certain base field of coefficients. Each intermediate node performs a linear transformation and achieves a linear combination of the incoming edges before delivering them to the next node(s).

Network coding is used in communication to target maximizing throughput, minimizing energy per bit and Minimizing delay [95]. A linear combination of received packets at the encoding nodes is transmitted with a linear coding coefficient vector or Local Encoding Vector (LEV). The Global Encoding Vector (GEV) is used to form the transfer matrix for the entire system. Practical instances of the network coding constitute the following: (i) Random coding [96] which allows the encoding to be done in a distributed fashion, (ii) Packet tagging of each packet with LEV allows the decoding to be done in a distributed manner, and (iii) Buffering which is required for asynchronous packet arrivals and departures with arbitrarily varying rates, delay, and loss.

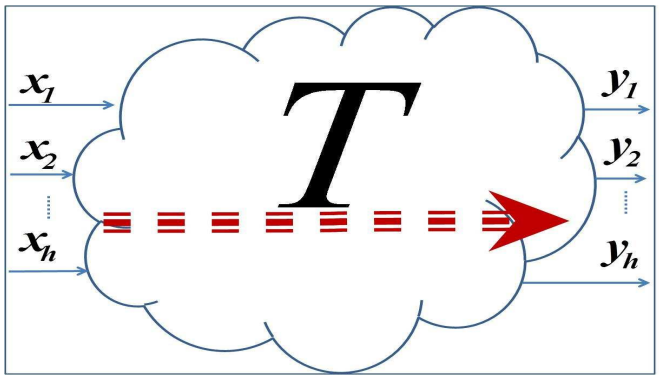


Figure 5.2: Matrix of Transfer

Let us assume an acyclic network (V, E, c) with unit capacity edges $c(e) = 1$ for all $e \in E$. Let $x_1, x_2, ..., x_h$ be the h packets that our graph, from an over all point of view, wishes to carry. Bringing the coefficients of all nodes $v \in V$ into account and in short, if we assume an "$h \times h$" model, (5.1) shows the relationship between received packets (y_is) and sent packets (x_is). Matrix T presented by (5.2) is called transfer matrix of the network, therefore, receiver(s) can use (5.3) to extract the original x_i out of y_i. T is based on each node coefficient and should be an invertible matrix, which having a random coefficient guarantees that.

$$\begin{bmatrix} y_1 \\ \vdots \\ y_h \end{bmatrix} = \begin{bmatrix} t_1(e_1) & \dots & t_h(e_1) \\ \vdots & \ddots & \vdots \\ t_1(e_h) & \dots & t_h(e_h) \end{bmatrix} \times \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix} \tag{5.1}$$

$$T = \begin{bmatrix} t_1(e_1) & \dots & t_h(e_1) \\ \vdots & \ddots & \vdots \\ t_1(e_h) & \dots & t_h(e_h) \end{bmatrix} \tag{5.2}$$

$$\begin{bmatrix} y_1 \\ \vdots \\ y_h \end{bmatrix} = T \times \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix} \Rightarrow \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix} = T^{-1} \times \begin{bmatrix} y_1 \\ \vdots \\ y_h \end{bmatrix} \tag{5.3}$$

Depicted by Figure 5.2, and since transfer matrix T is not fix due to dynamic and randomness of the coefficients, a receiver requires to calculate T^{-1} each time based on received tags. To improve the calculations of (5.3),

[97] proposes using sub-graph in order to handle different sources' traffics to different destination. More specifically, the main graph is divided to parallel sub-graphs, and packets from a source to a destination traverse in only one sub-graph. The aim in [98] is finding the minimum cost multicast sub-graph, where delay values associated with each link, limited buffer-size of the intermediate nodes and link capacity variations over time are taken into account.

5.3 Related Work

In [99], the CPS is studied as a combination of multiple fields of science such as computing, communication and control systems. The author compared the evolution of the CPS to the Internet, and provided some applications of the CPS in real world, e.g. smart grid for the power sector. He also mentioned that privacy should be preserved by the CPS: "*These CPSs will have embedded and distributed intelligence, operating dependably, securely, safely, and efficiently in real time, while satisfying privacy constraints*". The author also presented advances of the CPS, such as fully autonomous vehicles, smart power grids and extreme-yield agriculture, as well as the impact of the CPS on society and education.

The work in [100] considers the case of smart grid as an application of the CPS, which is related to the scope of our work in this chapter. The research work presented in [101] considers security of the smart grid. Author discussed the security aspects of the cyber-physical controls required to support the smart grid, which takes into account the power application. They analyzed the security from the risk point of view, and address the security concerns in control systems of the generation, transmission and distribution of the power in the smart grid. Furthermore, they studied the security of the infrastructure support and devices as well as security management and intrusion detection systems, followed by list of research challenges in this area. In this chapter, however, we focus on the privacy aspect of the SG system. To the best of our knowledge, we are the first to propose comprehensive schemes to address all features required to preserve privacy of clients in a smart grid system.

The scope of the work in [102] is the SG as well, in which the authors presented a security-oriented cyber-physical state estimation system. Their proposed system identifies the compromised set of hosts in the cyber network and the maliciously modified set of measurements obtained from power system sensors, at each time instant. They used the concept of the IDS, which

utilizes stochastic information fusion algorithms and merges sensor information from both the cyber and electrical infrastructures. The innovation of their proposed work is using the IDS system to monitor the cyber infrastructure for malicious or abnormal activity, in conjunction with knowledge about the communication network topology.

M. Stegelmann *et al.* proposed a scheme, wherein smart meter sends the metering data to a local aggregator, and then the aggregator applies the anonymity before sending the data to service providers. Although data for the billing is not anonymous, the same data is anonymous when it is sent to the service provider for the planning [103]. However, this scheme provides only source anonymity in portion of the data deliveries. The presented system in [104] aimed at anonymity of the SMs by combining the data collected by each SM with an *ortho code*, in a ring architecture, to the utility via an aggregator. The utility, without realizing the identification of each SM, can obtain the meters by summation information processed by aggregator. As the authors mentioned as well, they only provided anonymity of the sender (SM).

A Secured routing protocol for ad-hoc network is presented in [105], which enables anonymity of the source, destination and path. In this protocol, a source initiates and broadcasts a path request including a path sequence number and the encrypted destination address. The relay nodes only rebroadcast the path request after recording it. The destination responds back (unicast) to the path request, and nodes along the path reserve the path by matching information about the previous and next hops. However, this protocol is vulnerable to the flow tracing attack.

In [106], a network coding based scheme is used for privacy preserving, which extends the work in [105] by providing source anonymity. The scheme forwards a random-based linear vector encrypted GEV at each intermediate node in which only the destination is capable of decrypting the GEV. The receiver has to undergo the decryption of the tags, forming transfer matrix, and heavy process of the reverse matrix calculation. The scheme presented in [107] also utilizes network coding to support security and privacy.

In [108], the linear network coding is used to maintain privacy of the mobile nodes in a wireless mesh network environment. The proposed mechanism is aimed at flow untraceability and movement untraceability of the nodes. However, the proposal mainly pay attention to the flow of the information of the mobile nodes, and does not preserve anonymity of the nodes, especially when an attacker is listening to the first mesh router that receives the data/packet from the mobile node.

The proposal scheme in [109] aimed at flow anonymity of the data to

provide the anonymity of the communicating parties by tacking advantage of mixing characteristic of the coding. Although the scheme concentrates on anonymity of the source and destination by hiding the flow identifies causes by mixing the flows, it does not address other aspects of the privacy.

5.4 System Design

In this section, we first describe our assumptions. Then we present our proposed enhanced network coding mechanism and describe our privacy-preserving scheme.

5.4.1 Assumptions and System Setup

Our assumption are as follows:

- Public key encryption system that has a PKG responsible for the key management.
- Nodes have already performed an authentication scheme. They have also received their private key as well as the system parameters from the PKG.
- Topology is almost static: For instance in case of the SG, the maximum movement of nodes are within a HAN, although the SM of the HAN is static.

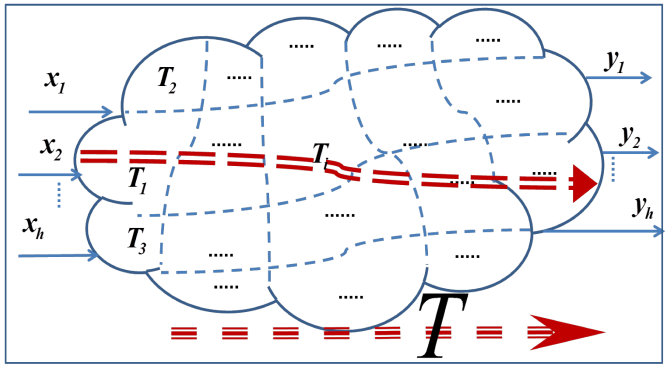


Figure 5.3: Matrix of Transfer, With Sub-graphs

- A SG server, which can be in charge of the PKG duties as well, is aware of the topology and graph of the network.

5.4.2 Enhanced Network Coding

As shown in Figure 5.3, the system administrator divides the main topology/graph G into m sub-graphs $SubG_i$ (he may consider the proposed solution in [98] for sub-graphing) and forms sub-graphs set $\widetilde{SubGS}$ such that:

$$\begin{cases} \widetilde{SubGS} = \{SubG_i |\ i = 1, 2, ..., m\} & \text{(5.4a)} \\ G = \bigcup_{i=1}^{m} SubG_i = \bigcup_{SubG_i \in \widetilde{SubGS}} SubG_i & \text{(5.4b)} \end{cases}$$

In each sub-graph $SubG_i$, system administrator selects n_s nodes to be the network coding nodes, which perform the network coding activities such as encoding. Furthermore, system administrator nominates one of the nodes to be head cluster of the sub-graph, which can be shown by HC_i.

We consider transfer matrices set $\widetilde{TS}$, which T_i represents transfer matrix of $SubG_i$ such that:

$$\widetilde{TS} = \{T_i | i = 1, 2, ..., m\} \tag{5.5}$$

Similarly, we consider inverse of transfer matrices set $\widetilde{TRS}$, which TR_i represents inverse of the transfer matrix of the sub-graph $SubG_i$, such that:

$$\widetilde{TRS} = \{TR_i | i = 1, 2, ..., m\} \tag{5.6}$$

Furthermore, we introduce a new parameter "α_i" as follows:

$$\alpha_i = \begin{cases} 1\ , & \textit{data crosses } SubG_i & \text{(5.7a)} \\ 0\ , & \textit{data does not cross } SubG_i & \text{(5.7b)} \end{cases}$$

Finally, we define "$h \times h$" transfer matrix $\widehat{T}$ which converts an input data matrix $\widehat{X} = \begin{bmatrix} x_1 & x_2 & ... & x_h \end{bmatrix}^T$ to the output data matrix $\widehat{Y} = \begin{bmatrix} y_1 & y_2 & ... & y_h \end{bmatrix}^T$, following (5.8a) and (5.8b).

$$\begin{cases} \widehat{T} = \prod_{T_i \in \widetilde{TS}\ \&\ \alpha_i = 1} T_i \quad ,\ i = 1, 2, ..., m & \text{(5.8a)} \\ \widehat{Y} = \widehat{T} \times \widehat{X} & \text{(5.8b)} \end{cases}$$

Similarly and at the receiver side, (5.9a) and (5.9b) are used to decode $\widehat{X}$ out of $\widehat{Y}$. Note that $\widehat{TR} = \widehat{T}^{-1}$.

$$\begin{cases} \widehat{TR} = \prod_{T_i \in \widehat{TS} \ \& \ \alpha_i = 1} T_i^{-1} \quad , \ i = 1, 2, ..., m \\ \quad = \prod_{TR_i \in \widehat{TRS} \ \& \ \alpha_i = 1} TR_i \quad , \ i = 1, 2, ..., m \qquad (5.9a) \\ \widehat{X} = \widehat{TR} \times \widehat{Y} \qquad (5.9b) \end{cases}$$

5.4.3 Privacy-Preserving Scheme

Referring to Section 5.2, a receiver requires the LEVs of a graph (over which the data has passed through) in order to compute the transfer matrix. In a linear network coding, there are two parameters that can be changed, such as network topology (path) and coefficient factors (LEVs). One solution is having one of these two values to be fixed and the other one changes dynamically (or, in some cases both of them can be dynamic). To be more precise, we can keep the topology (path) static, and randomly choose the coefficients, which in this case the coefficients information should be transferred (some how, and securely) to the receivers to make the receiver capable of decoding the data. On the other hand, we can fix the coefficients and randomly choose the path, which in this case information about the path, or the network coding nodes (that have performed network coding operation/encoding), should be transferred to the receiver.

Note that LEV is a function of the coefficient factors [95]. Without loss of generality:

$$T_i = Function(LEV_{SubG_i}) \quad , \quad i = 1, 2, ..., m \qquad (5.10)$$

Since we keep the sub-graph structure fix, only knowing coefficients is missing to compute the transfer matrix(ces) of the sub-graphs, which the server is capable of doing it. From an abstract point of view, in our system, we keep the topology, nodes coefficients and structure of the sub-graphs fix, although the sub-graphs that the data is crossing is being selected randomly. Our mechanism phases are as follows:

Phase I: setup

Firstly (Algorithm 2), PKG provides a *One-Way* hash function $F_{coef}(.)$ to the nodes. Each node applies $F_{coef}(.)$ to its own private key to obtain its

Algorithm 2 System Setup

Define:
$PrvK_{ID_j}$: Private key of node ID_j.
$Coef_{ID_j}$: Coefficient factor of node ID_j.
PKG : Private Key Generator.
$F_{coef}(.)$: Shared hash function.
$SubG_i$: "i^{th}" sub-graph in sub-graph set $\widetilde{SubGS}$.
T_i : Transfer matrix of the sub-graph $SubG_i$.
$\widetilde{TRS}$: Set of inverses of transfer matrices of the sub-graphs.

Algorithm:
$PKG \leftarrow ID_j$
$PKG: (PrvK_{ID_j}\ ,\ F_{coef}(.)\ ,\ i) \rightarrow ID_j$ {PKG calculates the private key}
$Coef_j \leftarrow F_{coef}(PrvK_{ID_j})$ {Perform by PKG and ID_j}
$\widetilde{SubGS} = \{SubG_i |\ i = 1, 2, ..., m\}$ {Defined by system administrator}
$PKG \leftarrow \widetilde{SubGS}$ {Receive from the system administrator}
$T_i^{-1} \leftarrow T_i \leftarrow (SubG_i\ ,\ Coef_j\ s.t.\ ID_j \in SubG_i)$ {Performed by PKG}
$\widetilde{TRS} = \{T_i^{-1} | i = 1, 2, ..., m\} = \{TR_i | i = 1, 2, ..., m\}$
$\widetilde{TRS} \rightarrow Destination$

coefficient (5.11):

$$Node\_Coefficient = F_{coef}(Node\_PrivateKey) \tag{5.11}$$

In a PKI-based system, only PKG and each node know the private key of the node. System administrator provides all information about the topology and graph consists of the participating nodes in each sub-graph to PKG. PKG calculates T_i and T_i^{-1} of each $SubG_i$ and provides the T_i^{-1}s to a destination.

Note that a private key can be considered as a random-based secret value managed by PKG. For instance, in an IBC approach, like [52], the private key of a node is multiplication of a secret random value generated by PKG and the public key of the node. Since the coefficient is a function of the private key (5.11), the randomness is implied for the coefficient as well, and referring to [95], T_i is invertible.

Since $F_{coef}(.)$ is a *One-Way* function, even if any of the receivers acts maliciously, an attacker would not be able to utilize matrix T_i^{-1} and performs a reverse operation to obtain the private keys of the nodes. We discuss more about this in Section 5.5. Furthermore, a private key is a dynamic value [110], therefore, transfer matrices T_i (and T_i^{-1}) are also dynamic. Note that the PKG is responsible to maintain and update the matrices and informing the receivers, for instance in case of the SG, the SG servers, which collect the data, should be notified by this server (PKG).

Algorithm 3 Generating and Sending the Packets

Define:
$PubK_{ID_a}$: Public key of node ID_a.
$PrvK_{ID_a}$: Private key of node ID_a.
NSG_{ID_s} : Set of next optional sub-graphs to the destination for sender ID_s.
$e_{ek}(.)$: Encrypting with key ek.
$sign_{ek}(.)$: Signature of data using key ek.
X : "$1 \times h$" size matrix of plain packets to be sent.
$\widehat{X}$: "$1 \times h$" size matrix of encrypted packets to be sent.
TAG : "m" bit size vector; each bit represent one sub-graph.
ID_{TAG} : A nonce value represents the identification of the TAG.
$F_{nc}(k)$: A nonce generator function in "k" bits size.

Algorithm:
{ID_s chooses one SG_i out of $\widetilde{NSG}_{ID_s}$ with an equal probability}
$MyNSG \leftarrow Random(\widetilde{NSG}_{ID_s})$ Random choosing a sub-graph out of $\widetilde{NSG}_k$ set
$TAG \leftarrow F_{nc}(m)$ {Encryption of the tag. "m" is total number of sub-graphs}
$ID_{TAG} \leftarrow F_{nc}(m)$ {Choosing a nonce vale for the tag identification}
$DataH \leftarrow (ID_s, ID_r, TAG, ID_{TAG})$ {Data header}
$SgnH \leftarrow sign_{PrvK_{ID_s}}(DataH)$ {Signing the data header}
{ID_s encrypts data (packet by packet) using public key of the receiver}
for $(l = 1 \rightarrow h)$ **do**
  $\widehat{X}.[1, l] \leftarrow e_{PubK_{ID_r}}(X.[1, l])$ {Encryption}
end for
$(\widehat{X}, e_{PubK_{ID_r}}(DataH), SgnH, TAG, ID_{TAG}) \rightarrow MyNSG$ {Sending encrypted data, data header, signature of the header, TAG and ID_{TAG} to the next sub-graph}

Phase II: generating and sending the packets

Presented by Algorithm 3, a sender chooses a nonce and assigns it to the TAG, and a nonce random identity for the TAG, which we show it as ID_{TAG}. Then, the sender chooses one of the adjacent sub-graphs with equal probability to send the data. Then, the sender forms the data header including the nonce values and address of the receiver. Furthermore, the sender signs the header with its own private key in order to preserve the source authentication as well as the data header integrity. Finally, the sender sends the encrypted data (packets) and data header, signature of data header and plain form of the tag and its ID to the next sub-graph toward the receiver.

Note: TAG is an array that traverses with the data. Each bit of the TAG represents α_i of a sub-graph ((5.7a) and (5.7b)). To be more precise, the i^{th} bit of the array is converted to one if the data passes through $SubG_i$. Therefore, initially TAG consists of only zeros ($TAG = 0$). Since TAG is sent in a plain format, we load it with a nonce value, and forward the nonce (encrypted) to the destination. Then, in each sub-graph, the head

cluster only reverses the value of the i^{th} bit. In other words, we XOR this bit with α_i. Consequently, at the destination only needs to XOR the result with the original nonce value to decrypt the tag and obtain list of the sub-graphs that the data has passed through. Comparing to the network coding operation, especially at the receiver, changing one bit per sub-graph is negligible overhead added cost by our mechanism.

Note: Referring to our discussion in Section 5.2 about the network coding, normally the coefficient that each network coding node use to handle the coding process, needs to be sent to the receiver for encoding process (by receiver). In our design, we eliminate sending this overhead data (coefficients) in cost of sending the tag and tag identity. In fact, tag ID is similar to the flow ID that is being used by the network coding, and our additional overhead cost is the tag itself. The overhead cost of sending the tag is much less than sending the coefficients, since in network coding there is one coefficient per network coding node, and we only have one tag from source to destination.

Algorithm 4 Relaying the Packets

Define:
NSG_i : A set of next optional sub-graphs to the destination for "i^{th}" sub-graph.
$\widehat{Y}_i$: Input "$1 \times h$" size data matrices at sub-graph $SubG_i$.
$\widehat{X}_i$: Output "$1 \times h$" size data matrices at sub-graph $SubG_i$.

Algorithm:
$SubG_i \leftarrow (\widehat{Y}_i, DataH, SngH, TAG, ID_{TAG})$ {Receiving data, data header, signature, tag and tag ID}
if ((Looks up ID_{TAG}) == NO) **then**
  $\widehat{X}_i \leftarrow SubG\_Function(\widehat{Y}_i)$ {The result of $SubG_i$ internal process}
  $SHFT\alpha_i \leftarrow 2^{i-1}$ {Shift "α_i" to the "i^{th}" bit position}
  $TAG \leftarrow (TAG \otimes SHFT\alpha_i)$ {Record "α_i" into TAG}
  Records ID_{TAG}
end if
$MyNSG \leftarrow Random(NSG_k)$ {Choosing $SubG_k$ out of $\widetilde{NSG}_k$ set}
$(\widehat{X}_i, DataH, SngH, TAG, ID_{TAG}) \rightarrow MyNSG$ {Sending data, tag, tag ID and data header to the next sub-graph}

Phase III: relaying the packets

As it is shown in Algorithm 4, we consider a situation that our data is entering to the $SubG_i$. The data passes through $SubG_i$ concerning the defined connections and coefficient values of the nodes (network coding nodes are already identified by the administrator). The head cluster of the sub-graph needs to record α_i into TAG by changing the i^{th} bit of TAG. Similar

to the previous step (sending data), the head cluster of the sub-graph $SubG_i$ randomly selects one of its neighbour sub-graphs to transfer the data to toward the receiver.

Note: Since the next sub-graph is chosen randomly, the data may get entered to the same sub-graph more than once. In order to prevent this looping situation, the identity of the tag (ID_{TAG}) is referred by the header of the sub-graph (HC_i). Indeed, HC_i keeps a record of the ID_{TAG} that is processed by the sub-graph, in addition to IDs the sub-graphs that it is received from and is sent to, for some time in order to prevent processing it twice. The reasonable expiry time of keeping the record can be same as SMs periodic collecting time, e.g. 15 minutes. In this case, the assumption is that the data will be received and decoded by the receivers during 15 minutes. Therefore, first of all, HC_i does not lead the processed (coded) information to be sent to the same sub-graph that is coming from. Secondly, if it receives the same data (ID_{TAG}) from another sub-graph, it will forward the data as-is and without coding it again, to the next randomly chosen sub-graph excluding the sub-graphs that are received from as well as the data has been sent previously to. It is obvious that in a worse case scenario, the data will reach the destination after being processed by the entire sub-graphs only once.

Phase IV: receiving and decoding the packets

Presented by Algorithm 5, when a receiver receives the data:

- Utilizes its own private key to decrypt the header to obtain addresses of the sender and receiver, and the nonce.
- Referring to the sender address, verifies the signature, and if it is valid, XORes the nonce with the received tags for decryption.
- Referring to the bit values of TAG, selects T_i^{-1} (TR_i) of sub-graphs that data has passed through, and multiplies them together to obtain the reverse value of the path transfer matrix $\widehat{TRS}$ via (5.9a).
- Obtains original packets sent by the sender via (5.9b).

Algorithm 5 Receiving and Decoding the Packets

Define:
$\widehat{T}$: Transfer matrix from source to destination.
$\widehat{TR}$: Inverse of the transfer matrix from source to destination.
y & x : Received packet and sent packet.
$\widehat{Y}$: Matrix of the received packets with size of "$1 \times n$".
$\widehat{X}$: Matrix of the sent packets with size of "$1 \times n$".
$e_{ek}(.)$: Encrypting with key ek.
$d_{dk}(.)$: Decrypting with key dk.

Algorithm:
$Receiver \leftarrow (\widehat{Y}, DataH, SgnH, TAG, ID_{TAG})$ {Receiving packets, data header, signature, tag and tag ID}
$OrgNonceEnc \leftarrow DataH$
$OrgNonce \leftarrow d_{PrvK_{ID_r}}(OrgNonceEnc)$
Verify Sgn {If verification result is positive, proceed}
$TAG \leftarrow (TAG \otimes OrgNonce)$ {XOR with the original nonce for decryption}
$\widehat{TR} \leftarrow I$ {I is identical matrix}
for $(i = 1 \rightarrow m)$ **do**
  if $(TAG.[i] == 1)$ **then**
    $\widehat{TR} \leftarrow (\widehat{TR} \times TR_i)$
  end if
end for
$\widehat{X} \leftarrow \widehat{TR} \times \widehat{Y}$
for $l = 1 \rightarrow h$ **do**
  $X.[1, l] \leftarrow d_{PrvK_{ID_r}}(\widehat{X}.[1, l])$ {Decryption}
end for

5.5 System Evaluation

In this section, we present our analysis from privacy and system performance point of views. First we propose two adversary models, then compare our delivered privacy factors comparing to the literature, and finally in the communication and network performance subsection, we discuss complexity and reliability of our design.

5.5.1 Adversary Models

We refer to Dolev-Yao model [25] to design our two adversary models including external and internal adversaries, in case of the SG system.

External adversary

In this case, the adversary is an external party and is not an entity of the system.

Objectives The adversary objective is obtaining information about the HAN occupancy and its resident behaviour.

Initial capabilities The adversary knows the detail information about the initial security system as well as our proposed privacy mechanism. For instance, the adversary knows public keys of the entire parties and has the detail knowledge about the network topology, graph and sub-graphs. Furthermore, the adversary knows the detail design of our mechanism including algorithms shown by Algorithm 2-5. Finally, the adversary has enough technical knowledge and is fully-equipped to be able to listen to the channels and analyze the traffic.

Capabilities during the attack The adversary receives all of the packets entering to a HAN (SM of the HAN) and departure from the HAN. Beside, the adversary can listen to the channel of any other entity of the system like PKG and any destination, to collect their receiving data.

Note: By using the term *data*, we mean and refer to the exact data that is in the channels (encrypted and/or encoded).

Discussion: Refer to our assumption, a HAN gateway (SM) acts as relay node in a mesh-based topology. We also implement and perform enhanced network coding that mixes the packets utilizing sub-graphs. Since source and destination addresses are encrypted inside the header, our scheme delivers the anonymity and undetectability, which yields to unobservability. If the adversary listens to entering and departing data from a HAN, he does not gain any useful information, since the entering packets plus HAN packet are encoded into one packet, which hides the HAN packet. If the origin of a packet is an appliance, listening to the channel does not help the adversary to obtain anything about the existence of the appliance (undetectability over appliances). In the proposed schemes in the literature (Section 5.1), he can understand HAN is generating a packet by listening to the first node, so, mostly those schemes only make a private path.

The packets entering a SM to be relayed, also do not have the source address, and are entering to the sub-graphs randomly. Therefore, the adversary cannot trace back the packets or monitor flow of the data, so unlinkability is delivered since he cannot observe direction of the data.

Last position for the adversary is at receiver side and listening to the receiving data. Considering above discussion about the hidden address of the receiver, he only obtain the flow of information to the destination. Indeed,

since the data travels through random chosen sub-graphs to reach the destination, he cannot trace back the data. Consequently, our scheme maintains anonymity and unlinkability here too.

Note that in any of the above situations, gaining access to TAG does not help the adversary. Indeed, encoding TAG with a random nonce makes sub-graphs capable of inserting α_i without decoding TAG. He does not obtain anything by having an encoded TAG, even at the first or last sub-graphs.

Internal adversary

Adversary is an internal party, e.g., he has access to one of the HANs and can particularly monitor gateway of the HAN or analyze the gateway information.

Objectives Gaining access to the neighbour HANs information by receiving their data for relay.

Initial capabilities The malicious node is already authenticated and receives the system parameters and its own private key, so our adversary has these information.

Capabilities during the attack The malicious node is under control of the adversary and performs the Algorithm 4.

Discussion: Having access to a malicious node only improves the adversary situation on modifying its HAN data. The relay nodes only mix the packets and do not perform any encryption and decryption. Furthermore, the data that he receives does not show any sign of the source or destination. Consequently, his capability and behave is almost same as the previous scenario.

Table 5.1: Delivery of the Privacy Measures

Scheme	[91]	[103]	[105]	[106]	[107]	[104]	[111]	[112]	Ours
Anonymity	✔	✖ & ●	✔	✔	✔	✔	●	●	✔
Unlinkability	●	✖ & ●	●	●	●	✔	✖	●	✔
Undetectability	✖	✖	●	●	●	✖	✖	✖	✔
Unobservability	✖	✖	✖	✖	✖	✖	✖	✖	✔

5.5.2 Privacy Performance Analysis

Referring to Sections 5.2 and 5.3 as well as our proposal in Section 5.4, Table 5.1 presents performance of our scheme comparing to the discussed schemes in Section 5.1. We consider two types of the attackers such as a neighbour and a relay node. Some of the schemes may deliver the anonymity in case of relay nodes; however, the data is not anonymous for a neighbour. We also use the following symbols to describe each deliverable:

- "✖": Does not deliver the measure.
- "●": Delivers the measure only against relay nodes.
- "✔": Delivers the measure against all nodes.

5.5.3 Communication and Network Performance Analysis

In this subsection, we provide an analysis and evaluation on the aspects of probability of success and complexity as well as intrusion success likelihood, and reliability for the proposed approach. Throughout the discussion we consider a square grid network topology. The communication performance evaluation of our proposed coordinated method is evaluated against the random network coding approach of [113] where authors claim a throughput performance gain over no coding. However, while there are advantages to network coding approaches, the success of these methods highly depends on the characteristics of topology. In this method, nodes continuously replicate and forward messages to newly discovered nodes.

Complexity

One of the overheads with the network coding is that nodes must have the processing capability to perform arithmetic operations over finite fields in real time. This processing will determine whether a decoded content chunk is innovative and makes a decision to either encode, forward, or decode. The processing complexity involved in operations over fields depends on the size of each generation h, and size of the field n. It takes $O(h^2)$ operations in F_{2^n} for linear operations with generations of size h. Multiplications and inversions over field F_{2^n} is of complexity $O(n^2)$. Furthermore, matrix inversions and Gaussian elimination to solve the system takes $O(h^3)$.

As shown in Figure 5.4, the cost of computing in our method is lower since the transfer metric at the receiver is implied and need not to be recalculated every interval. The computational cost in our algorithm is reduced

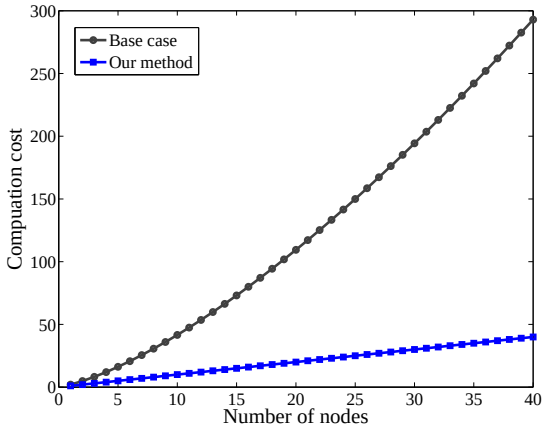


Figure 5.4: Cost of Computing

because enhanced network coding is performed on a selected set of nodes within each cluster.

Reliability

Our method aims at minimizing the number of nodes that shall perform the network coding operations. Therefore, we can take advantage of opportunities for fixed the network coding where possible. It is intuitive that as the system size increases, random network coding on large number of node compromise the overall computational complexity and degrades the overall probability of success.

The probability that a random network coding problem is solvable depends on whether the global coding vector has a full rank. If the coefficients are randomly chosen from a field F_q, then probability for a generation to be invalid is at most $\frac{|T|}{|q|}$. The extension of the *Schwartz-Zippel* theorem yields the probability of success at each random coded node as follows:

$$Pr(success) = (1 - \frac{|T|}{q})$$

where $Pr(success)$ is the probability of success within the cluster of random network coding. The following theorem from [96] states the probability of success by a valid network code.

Theorem 5.5.1 *The probability of a random network code with coefficients*

from field F_q being valid and being successfully decoded in a multicast connection problem with $|T|$ number of receivers and $|S|$ number of sources is $(1 - \frac{|T|}{q})^{\eta}$ where $q > |S|$ and η is the number of intermediate links with associated random coefficients.

As depicted by Figure 5.5, in contrast to the base case scenario, where random network coding is used, our proposed method utilizes a fixed network coding approach where the coefficients are dependent on the private key. Therefore, the uncertainty about the existence of a solution for the system is being resolved.

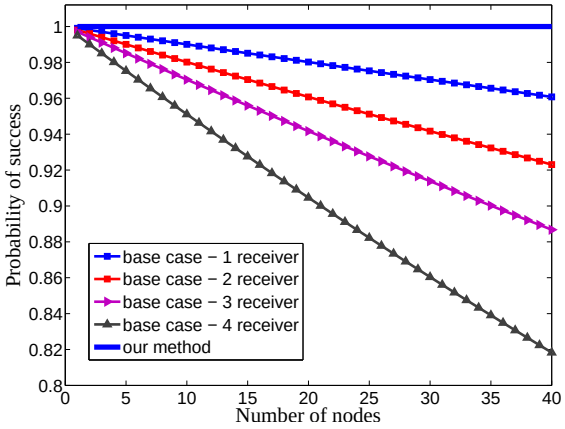


Figure 5.5: Probability of Success

Cost

In our mechanism, security server responsible for the private key generation, is in charge of calculating the transfer matrices as well as their inverses. Also, the inverse matrices should be transfered to the receivers. Finally and in each iteration and receiving the coded data, each receiver needs to multiply inverse matrices of the subnets that the data has been transfered. However, comparing to calculating the entire matrix of transfer for the entire network, and solving the network coding problem, our proposal costs less.

Chapter 6

Privacy Preservative Context-Aware Security Solution for Mobile Devices

The technology, security and privacy requirements of the electric vehicle (EV) in the SG context, especially when the EV acts as mobile power storage, have gained much attention from the research community and market recently. This role of the EV is motivated by the increase in capacity of the power storages in the EV. In this chapter, first we present different situations that an EV can be in the SG system and their privacy issues. In fact we consider different contexts of an EV in the SG, and our mechanism preserves privacy in all of the contexts. We provide two authentication schemes, first one between the EV and a trusted SG server directly, and second one via a non-trusted third party entity with a robust privacy-preserving agenda.

6.1 Introduction

The growth of interest in EVs and implementation of the SG introduces a new collaborative domain in transportation and resource management. In the SG system, EVs, PEVs or hybrid PEVs (HPEV), receive charging power from the SG network. However, EVs have recently gained attention about using the battery power stored in the EV as a SG mobile power storage that can saves and carries the power energy. Since the EV is mobile, it can store the power in one location and return the surplus back to the grid in another location (we will describe different situations that an EV can be in the SG system in Section 6.2).

Although bringing *ICT* to grid infrastructure [1] and new concepts like *"Electric Vehicle as Power Energy Storage"* are aimed at improving the power grid consumption and provision, they can be successfully support the new SG system implementation as long as the security and privacy concerns of the stockholders, especially consumers, are appropriately and

fully addressed [114, 115]. The survey presented in [116] describes charging infrastructure and PHEV/PEV batteries, intelligent energy management, vehicle-to-grid (V2G) and its communication requirements. In [117], some of the vulnerabilities and security concerns of the infrastructure for PEV are reviewed. Han et al. [118] proposed a method to estimate the achievable power capacity for practical V2G services. In [119], a management system was presented. It defined a temporary energy buffer that facilitates increased utilization of alternative energy sources. In this system EVs can be considered as distributed auxiliary batteries to support energy grids either in household domain or in a local energy network.

Contribution: Our first contribution is a mutual authentication scheme between the EV and the SG server that is followed by registration and binding of the EV to a person/owner who is held accountable for the power costs. The second one is a privacy-preserving communication mechanism which utilizes the concept of pseudonymous communication to hide the identity of the EV (and its owner) from any third party in the SG system. For instance, a power station operator in case of receiving (buying) and returning back (selling) the power energy to the grid.

The mechanism that we propose as V23PPA (vehicle to third party privacy-preserved authentication) assumes that originally the EV and utility server are authenticated using the V2GA (vehicle to grid authentication) scheme, which will be elaborated more in Section 6.4.

6.2 Problem Definition

Based on the seven domains structure model presented in [1] and Chapter 1 (in 1.4.1), electric power is delivered to the customer area (where the EV is normally residing) via the distribution network. However, an EV can be in any of the six locations shown in Figure 6.1, which are referred as Charging Points (CPs):

I. *HAN:* The EV is charging while connected inside the residence of the EV owner, as part of the HAN.

II. *BAN:* The EV is charging inside the building complex in which the owner resides, which is equipped with a BAN.

III. *Host:* The EV is connected to the power plug of a home other than the EV owner's residence (e.g. the owner is a guest and visiting a friend).

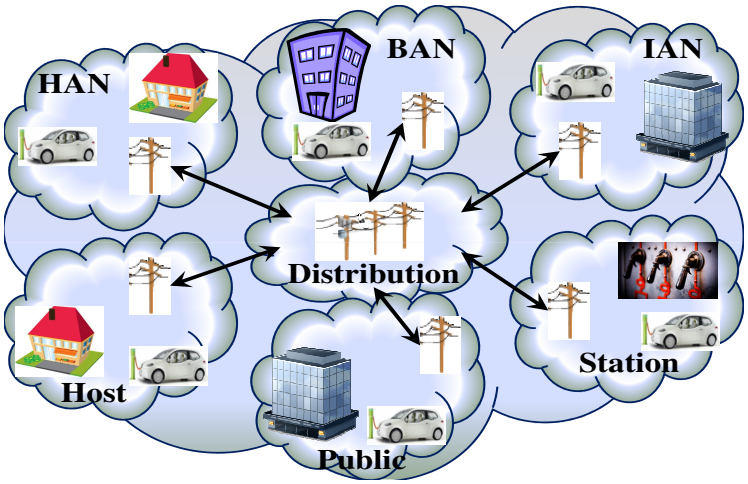


Figure 6.1: Charging Points in Smart Grid

IV. *Industry Area Network(IAN):* The EV is being charged inside the commercial (industrial) building where the owner is working and the company is providing the power charging services, so the EV is connected to IAN.

V. *Public:* Any public infrastructure that provides the power charging service to the visitors of the entity. For instance, a host commercial building, a public parking spot or a shopping mall.

VI. *Station:* Any third party power station that the EV can plug and receive the power energy.

Table 6.1: Power and Service Charge

Situation	Service Credit	Power Charge Debit
I	NIL	Smart Grid
II	Building	Smart Grid
III	Host	Host
IV	NIL/Company	Company
V	Entity/Service provider	Entity/Service provider
VI	Station	Smart Grid

In any of the use cases that there is a need to give or take power to/from EVs with authentication, there exists a transaction of charging or debiting the EV owner's account. For instance, when the EV receives the power from a host home, the host's account is initially charged; however, the charge should be forwarded to the EV (owner's account) and credited back to the host's account. Table 6.1 presents the appropriate account that in each situation should be accessed.

When a person buys an EV, or somehow becomes in-charge of the EV, she/he needs to register the EV in the SG system. In order to perform this registration, the EV needs to (mutually) authenticate itself to the sever. In the first part of our proposal, we state the vehicle to grid authentication scheme that covers this requirement. Referring to our discussion about the power charging of the EV as well as the concept of using the EV as SG power storage, the second part of our proposal is a mechanism for vehicle to third party privacy-preserving communication/authentication scheme. The third party is an entity that provides the power charging services, e.g. a station or a host home. The V2GA mechanism refers to the vehicle identification number (VIN) of the EV for identifying the EV; however, the V23PPA mechanism utilizes the pseudonym of the EV along with identity management service provided by SG server in order to provide anonymity in communication. In fact, the SG back end, i.e. utility server, (trusted entity) is the only entity that can manage and map the user's identity and pseudonyms to manage user account. The presented pseudonym identity of the vehicle is changing for any entity that the EV is being connected to which includes any scenario such as charging or returning to the grid. As soon as the EV leaves the entity (CP), the pseudonym is expired and a new one (which only the EV and the utility server are aware of it) will be used.

6.3 Literature Review

In [120], authors concentrated on security in Vehicular Ad-Hoc Network (VANET) by using pseudonyms and determining their expiration times. In [121], the anonymity and unlinkability of vehicles is studied, in which the role of battery information and how it can influence vehicle mixing is studied. The suggested model reduces the amount of transmitted data, aims at anonymity for tracking protection. The methods presented in [122] and [58] are aimed at hiding the identity of the vehicle by using a pseudonymous identity; however, the pseudonym is fixed (for 24 hours). Therefore, if an intruder can physically obtain the real identity of the EV once and prepare

a mapping between the real identities and pseudonyms, he can still attack the privacy of the vehicles by tracking the pseudonyms.

Using IBC is proposed in [123] for designing an authentication scheme for the EV although their scope is the VANET platform. In this work, the authors have used the pseudonym of the EV for the authentication, where the pseudonym has an expiry date. The issue of having an expiry date is, for instance in our scope, that the EV may require to contact two third parties during the period that the pseudonym is not expired yet. Therefore, during that period, the EV can be traced via tracing the pseudonym. A similar work is presented in [77] that uses IBC as well as pseudonym of the EV in VANET. Similar to the previous design in [123], this methods has an expiry time for the private key, which has the same issue as [123]. In [124] the authors proposed to use the standard Authorization, Availability, Accountability (AAA) infrastructure and Extensible Authentication Protocol (EAP) protocol for authentication and key management. Their method is proposed for VANETs with intermittent connectivity. The contribution in [124] is to use pre-arrival authentication to Road Side Units (RSUs). The mobile node uses EAP-Transport Layer Security (EAP-TLS) to authenticate itself the most probable contacted RSUs in future. However the mechanism to find the next RSU in the future has not been discussed. Moreover, the EAP protocol needs several message exchanges and therefore is not very compatible method for dynamic networks like VANET.

In [125], a method for batch authentication and key exchange (ABAKA) between the vehicle and service providers is proposed. The focus in this work is to reduce the delay imposed by authentication and key exchange mechanism in order to minimize the security cost over the short-lived wireless connectivity between moving vehicles and RSU. To provide anonymity, ABAKA also uses pseudonym IDs which are generated by the tamper proof on-board device. However, to verify a message, CP needs to know the secret key of the on-board device. This key is supposed to have been placed in the on-board unit during initial phase and is known to CP. Assuming that this value is uniquely dedicated to a vehicle, by using a pseudonym (which provides anonymity about the vehicle's identity), it is not clear how the CP is able to verify the message with the matching secret value for that car.

The authors in [126] proposed PAAVE, an anonymous authentication mechanism in VANETs which considers Vehicle to Roadside (V2R) communication authentication using smart cards in vehicles. They have used PbKE system in which the public key for every vehicle is signed by trusted authority before being saved inside the vehicle. Their approach covers V2R communication only. PAAVE is designed to provide anonymity for user but

traceability is not addressed. Since the public key for vehicle is constant, a third party can trace the previous locations of a vehicle.

Implementing the privacy preserving mechanism that deals with hiding the footprint of the vehicle, especially when the vehicle is allowed to have multiple identity, may lead to performing a Sybil attack by a malicious EV, which is the scope of the work presented in [127]. However, in our mechanism the EV can have a maximum of two identity (a real identity and a pseudonym) at each time, which prevents the attacker from performing a Sybil attack.

6.4 Proposal

Let us consider the topology shown in Figure 6.2, and four shown states for the EV:

I. This the first time that an EV connect to the SG system, when the EV and SG server (SGS) perform a mutual authentication process.

II. The EV connects to an entity, like a third part power station, (CP) to receive the power from.

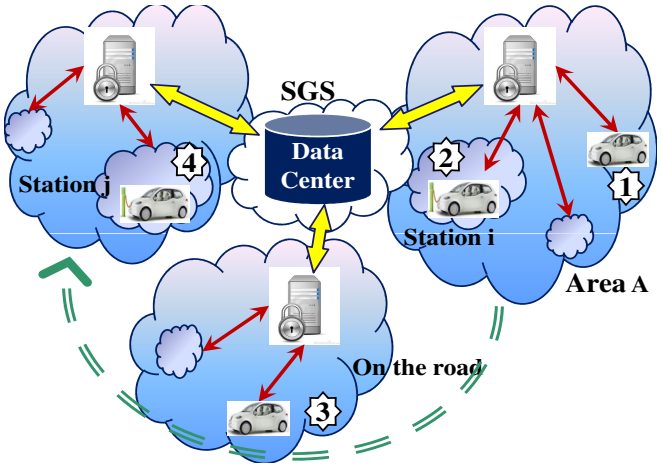


Figure 6.2: Electric Vehicle Communication with CP

III. The EV is disconnected from the CP entity and is on the road before connecting to another CP to receive (or deliver) the electric power.

IV. The EV connects to the next CP for service.

In this part, first we introduce our V2GA scheme that is designed for the first state, followed by V23PPA mechanism that covers the second, third and fourth states.

6.4.1 V2GA Scheme

Our authentication scheme (V2GA) between the EV and SGS is presented in Figure 6.3. The V2GA scheme, which is an SRP based protocol, is designed having into account the following assumptions:

A.1. When the owner of the EV receive the vehicle, she/he should register the EV to the SG system. In this stage, the owner should give her/his required information along with the VIN of the EV to the SGS.

A.2. The owner also keys in a simple (secret) password to the SGS as well as the EV.

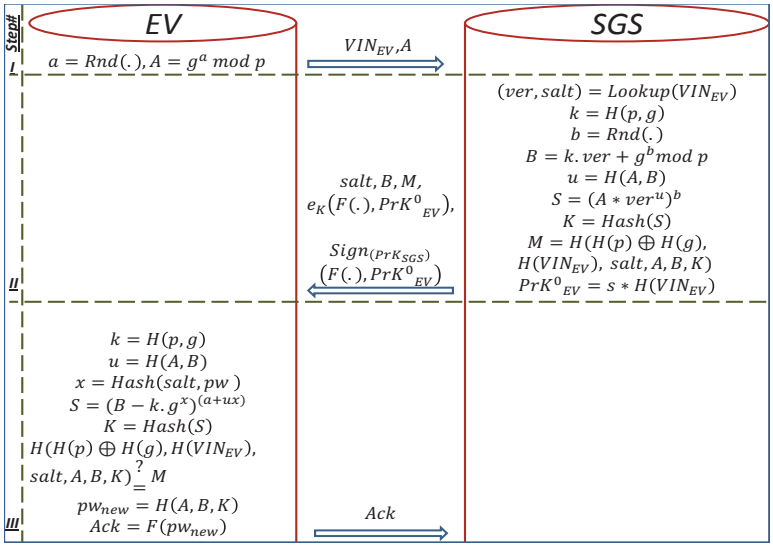


Figure 6.3: Authentication Between EV and SG Server

A.3. The EV is equipped with a temper-proof device that can support the security mechanism, e.g, by keeping the private key of the EV safe.

A.4. The EV is equipped initially with the system parameters "g & p" as well as the hash function $H(.)$ for the SRP-based calculation.

A.5. There are communication technology supports for communications between the EV and SG server.

A.6. ID of the SG servers (SGS) is known by the EV.

A.7. The duties of the PKG is assigned to the SGS.

A.8. As presented in Figure 6.2, the entities that play the role of the SGS, shared their data by saving them in the cloud, e.g. a data center.

A.9. The system follows IBC and the public key and private key of each entity can be calculated by (1.1) and (1.2).

Our mutual authentications steps including the initial step are as follows:

Preparation

Followed by entering the password pw by the owner of the EV, the connected SGS picks a random value $salt$ and calculates the verifier ver as per (6.1) and saves it in the database of the SGSs, located in the data center, as record of the EV consists of the VIN and the $salt$.

$$ver = g^{H(salt,pw)} \bmod p \tag{6.1}$$

Initialization step (I)

The EV picks a random number A, calculates "$a = g^A \bmod p$", and sends a along with its VIN to the SGS.

SGS response step (II)

SGS looks up the database and obtains the ver and $salt$ aligned with the received VIN. Then, the SGS follows the tasks shown in Figure 6.3, and also calculates the original private key of the EV following the IBC technique, in which s is the secret value of the system, kept by the SGSs. The SGS sends the private key along with the $salt$, B and M to the EV while encrypts the private key with the constructed key K and signs them with its own private

key. Note that, as per Figure 6.3, the SGS also sends system parameter set $Parm(VIN_{EV})$ prepared for the EV, e.g. including the hash function $Psd(.)$, which we will use them for the authentication of the EV and third parties as part of the V23PPA mechanism (will be described shortly). The entire secret values, as per our above mention assumption, will be saved in the temper-proof device of the EV.

Final step (III)

First of all, the EV applies the hash function $H(.)$ to ID of the SGS to obtain public key of the SGS and verifies the signature. Then, the EV calculates the shared key K utilizing the received values $salt$ and B, and verifies the key K based on received M from the SGS. Then, the EV decrypts the received encrypted private key from the SGS using K. Beside, the EV calculates the new password pw_{new} as "$pw_{new} = H(A, B, K)$", applies the received function $Psd(.)$ to obtain the Ack as "$Ack = Psd(pw_{new})$" and sends it to SGS to verify finishing the algorithm. Note that, SGS is capable of calculating the values pw_{new} and Ack.

Since the EV is a mobile node and for any reason may lose the connection, or its private key gets expired, therefore, the EV utilizes the new password pw_{new} and performs the V2GA protocol for the re-authentication. Also, the SGS reselects a $salt_{new}$ and calculates the new verification value ver_{new} via (6.1), and then saves them in the database for the next time authentication of the EV.

Table 6.2: Definitions

Item	Description
SGS	Smart Grid Server.
$H(.)$	System one-way hash function for public key creation.
$Psd(.)$	One-way hash function for Pseudonymous creation ID.
CP_k	Identification Number of the k^{th} CP.
VIN_j	Vehicle Identification Number of the j^{th} EV.
$PubK_j^0$	Master Public Key of the EV VIN_j.
$PrvK_j^0$	Master Private Key of the EV VIN_j.
i	The security state of the EV VIN_j, $i = 1, 2, ...$
a_j & b_j	PRNG Parameters of the EV VIN_j.
$Parm$	System parameter set.
ID_j^i	i^{th} Pseudonym (ID) of the EV VIN_j.
$PubK_j^i$	i^{th} Pseudonymous Public Key of the EV VIN_j.
$PrvK_j^i$	i^{th} Pseudonymous Private Key of the EV VIN_j.
$Crdt_j$	The Credit Value of the EV VIN_j.

6.4.2 V23PPA Scheme

As the result of performing the V2GA mechanism, the EV has received its private key and is able to communicate with any node. Indeed, all an EV needs is the ID of the party, which calculates the public key of the party/node via (6.3) to establish a secure communication. In order to preserve the privacy in this mentioned communication, we propose that the EV uses its pseudonym instead of its real identity (VIN). The parameters used in our design are presented by Table 6.2. Our assumptions are as follows, based on the involved parties in the system presented in Figure 6.2, which can be considered as part of the initialization phase of the system:

B.1. There are communication technology supports for communications between the SG server and CP, e.g. station, wired or wireless, between EV and SG server (wireless) as well as between EV and the charging point (most likely wired).

B.2. The system follows IBC in which the public key and private key of each entity can be calculated by (1.1) and (1.2).

B.3. The duties of the PKG is assigned to the SG server. ID of the SG server (SGS) is known by the parties (The SGS that covers the area).

B.4. Each CP has an ID (CP_k) that is known by the SGS as well as by the EV that wants to be connected to the CP entity.

B.5. The initial authentication has been already done, and each CP has received the required system parameters and its own private key.

To describe the mechanism, let us consider four stages presented in the Figure 6.2. Without loss of generality and refer to our discussion in Section 6.1, we consider the sixth situation of the EV (EV to Power Station Communication) as per Table 6.1, which will cover other situations as well.

First stage: registration of the EV

In this stage, the initial authentication is done between the EV and SG server in its area, via V2GA scheme. Following to the initial authentication, the electric vehicle VIN_j receives the required system parameters ($Parm$), as per (6.2).

$$Parm = \{Psd(.), PrvK_j^0, a_j, b_j, PrvK_j^1, Crdt_j\} \tag{6.2}$$

Table 6.3: Smart Grid Server Database

VIN	$salt$	ver	EV State	a_j	b_j	Pseudonym	Credit Value
VIN_1	$salt_1$	ver_1	i_1	a_1	b_1	$ID_1^{i_1}$	$Crdt_1$
VIN_2	$salt_2$	ver_2	i_2	a_2	b_2	$ID_2^{i_2}$	$Crdt_2$
...	...	...	...	...	...	...	...
VIN_n	$salt_n$	ver_n	i_n	a_n	b_n	$ID_n^{i_n}$	$Crdt_n$
...	...	...	...	...	...	...	...

An EV follows (1.1) to calculate public key of any party, including the SG server and each station (CP), e.g. CP_k, that the EV wants to have communication with. In this system, the public key of the EV follows (6.3) aligned with the private key (6.4) that SGS generates for the EV.

$$PubK_j^0 = H(VIN_j) \tag{6.3}$$
$$PrvK_j^0 = s \times PubK_j^0 = s \times H(VIN_j) \tag{6.4}$$

The SGS entity also calculates the first pseudonym of the EV via (6.5) and then calculates the first pseudonymous private key of the EV as per (6.7) which is aligned with the EV first pseudonymous public key (6.6). SGS keeps these information in the database, as it is shown in Table 6.3, to be used by other SGSs as well.

$$ID_j^1 = Psd(a_j \times VIN_j + b_j) \tag{6.5}$$
$$PubK_j^1 = H(ID_j^1) \tag{6.6}$$
$$PrvK_j^1 = s \times PubK_j^1 = s \times H(ID_j^1) \tag{6.7}$$

Second stage: EV and CP communication

The first time that the EV communicates to a CP/station, it follows (6.5) to calculate its first pseudonym in order to introduce itself to the CP. Then, the EV pseudonymous private key is used by the EV to sign the request while the CP follows (6.6) to obtain the pseudonymous public key of the EV to verify signature of the EV. At this state, the CP does not know the state of EV (i). Indeed the CP/station only receives ID of the EV that is pseudonym of the EV.

As soon as the massage consists of the EV pseudonym is received by the SGS, the SGS searches its database and obtains the real VIN. Then, SGS sends the credit of the EV to the CP (station) to make the CP and EV able to manage their required steps to charge the EV or receive the power from the EV. The entire communications between the EV and CP are based on the pseudonym of the EV. Note that in these communication the IBC system is used. Finally, the CP will let the SGS know the EV credit changes, in which the SGS will update its database accordingly.

Third stage: on the road, between two CPs

When the EV leaves a CP, updates its pseudonym and its public key for the next connection point (CP). Precisely, if the EV state is i, the EV as well as the SGS calculate the next pseudonym of the EV via (6.8).

$$ID_j^{i+1} = Psd(a_j \times ID_j^i + b_j) \tag{6.8}$$

Furthermore, SGS calculates the new pseudonymous private key of the EV via (6.10) to be sent to the EV via the secure channel supported by the EV public/private key pair.

$$PubK_j^i = H(ID_j^i) \tag{6.9}$$

$$PrvK_j^i = s \times PubK_j^i = s \times H(ID_j^i) \tag{6.10}$$

Fourth stage: EV and next CP communication

This stage is similar to the second stage, indeed, only the state of the system is changed from i to "$i+1$". Similarly, the EV uses its current pseudonym as ID to contact to the new CP, in which has its current pseudonymous private key (state "$i+1$") for the secure communication.

6.5 Analysis and Evaluation

In this section, we analyze our design by reviewing the security and privacy attacks against our mechanisms. Furthermore, we evaluate the secrecy of the mechanism utilizing AVISPA, followed by studying the overall cost of the mechanism to analyze its performance.

6.5.1 Privacy Characteristics

In comparison to references models presented in Section 6.3, although the pseudonym and private key are proposed in [122] (and [58]), they only refreshed the pseudonym and private key once a day (every 24 hours). Therefore, the adversary is able to trace the victim EV path (footprint) during one day. Also, using the proposed model in [128] has the pre-assumption of trusted station, which is not valid. The trust of the station needs to be analyzed and managed e.g. based on proposed model in [129]. Stations are only service providers that deliver the power receiving and charging the EVs although are not as secure/trusted as the SG back end infrastructure. Even if by establishing the rules and regulations to enforce them to keep the customer information private, still the leakage of the customer information is the customers' concern. In fact, letting the third party to have private information of the customer decreases the cost of the social engineering attack, which can be easily performed by an adversary.

6.5.2 Analyzing the Attacks

Our V2GA mechanism has inherited the security features of the SRP protocol, which is secure versus most of the well-known attacks. For instance, since we use the verifier in the V2GA mechanism, it helps preventing the unknown key share attack. Also, using the verifier and salt make our mechanism secure versus the man-in-the-middle (MITM) attack as well as compromised impression attack. Also the packets in step II, which delivers the private key to the EV, is encrypted that makes the mechanism secure versus Key privacy & insider attack. Since all the time, the entity use a random number, the mechanism is secure versus reply attack, and by assuming the random key has an enough bit size, the mechanism is secure versus brute-force, on-line dictionary and off-line dictionary attacks. Finally, since we hash the password to obtain the verifier, and then hash the random values and other exchanged items as per V2GA mechanism, steps, the mechanism is secure versus denning-sacco as well as ephemeral key compromise impersonation attacks.

Theorem 6.5.1 *The main privacy attack as per our scope is tracing the EV, which the V23PPA mechanism is secure versus this attack.*

Proof Tracing the EV can be done only via the real identity of the vehicle. Referring to a pseudonym of the EV does not give any information since

it is the product of a hash (one-way) function. Therefore, the attacker cannot obtain the past pseudonym. Furthermore, the next pseudonym is the product of the current pseudonym in combination with two other parameter "a_j & b_j" that are only known by the EV and SGS. Therefore, the attacker is not able to calculate the next pseudonym neither. Consequently, knowing only one pseudonym does give any information to the attacker about the past and next pseudonyms, and therefore the attacker is not able to trace where the EV was and/or will be. As a result, the mechanism is secure versus this attack.

6.5.3 Formal Validation Using Software Tool

In this analysis, first we evaluate the V2GA mechanism, and then use the the password generated as the result of the first round, and re-authenticate the parties. The result of the evaluation are presented in Figure 6.4a and 6.4b, which evaluate the secrecy of the mechanism. Furthermore, the required HLPSL codes are shown in Appendix A.

6.5.4 Cost Analysis

Privacy is one of the major concerns of the SG implementation, in all levels. In fact, it does not matter how much the new smart power grid, SG, can be efficient and save money in customer and/or providers side, as long as

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
/ubc/ece/home/vl/grads/hasennic/
Desktop/avispa-1.1/testsuite/
results/PEV-Final.if

GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.79s
  visitedNodes: 51 nodes
  depth: 6 plies
```

(a) OFMC

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL

/ubc/ece/home/vl/grads/hasennic/
Desktop/avispa-1.1/testsuite/
results/PEV-Final.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 28 states
  Reachable  : 18 states
  Translation: 94.52 seconds
  Computation: 0.02 seconds
```

(b) ATSE

Figure 6.4: AVISPA Results

the privacy concerns of the stockholders are not fully addressed, the society does not accept the SG implementation. In our proposed mechanism, we only keep two private keys in the EV each time, which makes it efficient from the capacity and resource point of view. Also, in each state changing, the EV needs to run a hash function once ($Psd(.)$), and the SGS needs to run two hash functions ($Psd(.)$ & $H(.)$), once each, that are the cost of our mechanism. Although this is an efficient mechanism and system, and efficiency should be addressed all the time, the privacy cannot and should not be sacrificed as cost of the efficiency and improvement.

6.5.5 Summary of Security Analysis

Our proposed V2GA mechanism efficiently authenticates the EV to SGS in a mutual fashion, to register the EV based on the real identity of the EV. From then, the EV communicates with any CP for receiving or returning the power station utilizing its dynamic pseudonym. The pseudonym is refreshed by any connection of the EV to a plug, e.g. a station, which prevents an eavesdropper/attacker to trace the EV. The used pseudonym by the EV in each CP can be mapped to the real entity of the EV only by the EV and SGS. Precisely, since we use a hash function to obtain the new pseudonym from the current one, and the hash function is a one-way function, the adversary cannot find the past pseudonym. On the other hand, since the new pseudonym is application of the hash function on a pseudo random number generator with seed value of the real identity of the EV, the adversary is not able to calculate the future pseudonym neither. The pseudonym of the EV is only valid and active between each two plugs (CPs), which shows our mechanism fully delivers forward and backward secrecies.

6.5.6 Other Benefits

One of the problems that motivated the researchers to propose a privacy-preserving mechanisms for EV in the context of SG is monitoring status of the battery of the EVs in an area to manage participation of the EVs in power distribution in the area. Indeed, it schedules the returning the power back to the grid and/or schedules charging the EV, normally as part of an optimization problem. For instance our references in this thesis such as [118], [121], [130], [122] and [58], dealt with this problem, which we have review partially in this chapter. Our mechanism indeed responds to this problem as well. To be more precise, the monitoring agent/application collects the information, which consists of the location, ID and status of the battery

of the EVs. An EV in our proposed system, and between every two stations/plugs, uses pseudonym instead of the VIN. Therefore, the information collected by the monitoring system is based on the pseudonym. In case of directing an EV to a CP to returns the power back to the grid, the EV will refresh its pseudonym after leaving the station, and will starts using the new pseudonym from that point of view in communication with the monitoring entity as well. Hence the monitoring agent is not able to trace the vehicle since there is no connection between the new and old pseudonyms of the EV from the monitoring entity point of view. Consequently, our proposed mechanism maintains the privacy of the EVs in this application.

Chapter 7

Conclusion and Future Works

Referring to Chapter 1, this thesis is aimed at improving the efficiency of authentication and key management, and brought into account the privacy of the users.

7.1 Conclusion

Understanding the smart grid concept is related to the accuracy of information about the power consumption, actual and planned, by the consumers. This data needs to be collected for billing purposes and efficient power provisioning. In Chapter 2 we addressed this requirement by providing efficient and secure authentication and key management mechanisms tailored for communication between smart appliances and smart meters, as well as between smart meters and smart grid server in utility network.

In order to efficiently plan and consume (generate) the power by customers (suppliers), they need to collaborate with each other as a group. To make this communication secure, having a group key management is essential. In Chapter 3 we addressed this requirement by providing an efficient cluster-based group key mechanism that handles forward and backward secrecies.

The smart grid system needs to send some controlling commands by smart grid controllers to the smart home appliances. These controllers can be located in different layers, such as the home level, building level, neighbourhood level, and central smart grid controller unit level. Although they all can send the controlling commands, in case of emergency to turn the appliances off, the hierarchy should be preserved between them as well. In Chapter 4 we presented our multi-layer key mechanism that maintains this requirement efficiently.

The second part of this thesis concentrated on privacy, compared to the first that was mainly about the security. Although the power consumption

information is essential in the smart grid system, and mostly are encrypted to secure them, the user information can be traced from even encrypted data that jeopardizes user privacy. In Chapter 5 we utilized the enhanced network coding for this purpose and provided our privacy preserved mechanism. Our proposed design preserves anonymity, unlinkability, unobservablity, and undetectablity in collecting power consumption information in the Advanced Metering Infrastructure.

One of the smart ideas in the smart grid system, motivated by increasing capacity of the power storages in electric vehicles, is using the electric vehicles as mobile power storages. There are different options for an electric vehicle to plug in and receive/return the power from/to the grid, which are mostly owned by third parties, who could be untrusted, or at least curious, and want to trace the electric vehicle. Maintaining the user privacy in this situation was the aim of our proposal presented in Chapter 6, our last chapter of the thesis.

7.2 Suggested Future Works

In addition to the contributions that have been conducted during this dissertation, which was mainly founded by NSERC Smart Grid research project at UBC, the following directions and steps are suggested:

I. Cryptography: Initially we enhanced the identity based cryptography (IBC) and provided a conceptual solution called enhanced IBC (EIBC). The proposed mechanism is used in our other work, for instance our proposal in Chapter 2. However, more improvement and works can be done in this area, and a detail design of the mechanism in mathematics needs to be conducted. EIBC can improve the key management very well, as it has shown partially in Chapter 2 and related published papers as well. We even proposed a solution based on EIBC in Information-Centric Networking also [2]. After a few tests and re-design, EIBC showed its benefits very well, and it can be developed more to be used in other applications and areas.

II. Our design presented in Chapter 2, efficient authentication and key management mechanisms, mainly developed for a mesh topology network used in the smart grid system. After detail design of the EIBC

[2] H. Nicanfar, P. TalebiFard, C. Zhu and V.C.M. Leung, "Efficient Security Solution for Information-Centric Networking", in Proc. IEEE SymCPS, Beijing, China, Aug. 2013

(our above mentioned suggestion), the design of our mechanisms presented in Chapter 2 can be revisited and improved. Moreover, as it can be seen from our publications supporting Chapter 2, we proposed two designs, one for inside a home and another one for outside the home (the one for outside of the HAN is discussed in this thesis). A trust model around smart meter (or a home gateway) can be added to these two designs, and make a more enhanced and completed design that covers from a home appliances up to the utility server.

III. The group key protocol proposed in Chapter 3 can be extended to be elliptic curve cryptography (ECC) based design and to be more efficient and to have other benefits of the ECC based approaches as well. For instance, especially in a light and efficient communication and to use a smaller key size, the ECC based design can be more fit to address the efficiency.

IV. The multi-layer ECC based design presented in Chapter 4 is a completed and detail proposal. All can be added to improve it, is to run a test for instance in a test bed facility, and as per outcome, modify and improve the design accordingly.

V. Using other techniques from other areas, such as network coding (NC), to improve or even maintain a security (and privacy) related solution is an interesting approach. The reason is that e.g. in case of using NC; the NC area by itself is improving and has its own benefits in communication. So, applying it to a security and/or privacy mechanism, as we used in Chapter 5, will deliver an enhanced solution, which will maintain better network performance and enhancement (because of NC itself) as well as preserve security/privacy requirements. We provided two mechanisms that use NC. One addresses privacy, as it is described and presented in Chapter 5, and the second one provides security[3]. To be more precise, our second work focused on confidentiality of data transmission. In fact, we only provided the initial work, and more improvement needs to be added to make the second one a completed work as well. Both of the works can be much improved by applying new enhancements coming from the NC development area.

VI. In Chapter 6, we presented our privacy preservative security solution

[3]H. Nicanfar, A. Alasaad, P. TalebiFard and V.C.M. Leung, "Network Coding Based Encryption System for Advanced Metering Infrastructure", in Proc. IEEE ICCCN, Nassau, Bahamas, July 2013

for mobile devices, which in the chapter, we presented our solution for power electric vehicles in the SG context. We also presented our applied solution in mobile devices in the heterogeneous network (Hetnet) as per supporting papers of the chapter[4]. Now-a-day, the privacy is a serious concern of the users and society as well as governments (Chapter 1). The proposed solution can be more enhanced by design a privacy-aware secure mechanism to manage credits between the entities, e.g. by leveraging the electronic money concept. Also the communications between an electric vehicle and smart grid server (trusted entity) after each connection with a contact point can be improved.

7.2.1 Future Technology and our Mechanisms

Most parts of our contributions in this thesis are about the efficiency. Although the new technology can solve e.g. the smart meters or appliances issues in terms of limited computing capacity, our mechanism still can be used in other areas such as sensor networks or Internet of things. In fact, in any environments that entities have low computing capacities, our efficient mechanisms will be good answer to the computing constraints.

[4]H. Nicanfar, J. Hajipour, F. Agharebparast, P. TalebiFard and V.C.M. Leung, "Privacy-Preserving Handover Mechanism in 4G", in Proc. IEEE CNS, Washington, DC, Oct. 2013

Bibliography

[1] NIST Smart Grid, Cyber Security Working Group, "Introduction to NISTIR 7628 Guidelines for Smart Grid Cyber Security," Guideline, Sep. 2010. [Online]. Available: www.nist.gov/smartgrid

[2] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[3] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[4] A. Shamir, "Identity-based Cryptosystems and Signature Schemes," in *Advances in CryptologyCRYPTO 1984*. Springer, 1984, pp. 47–53.

[5] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in CryptologyCRYPTO 2001*. Springer, 2001, pp. 213–229.

[6] A. Koblitz, N. Koblitz, and A. Menezes, "Elliptic curve cryptography: The serpentine course of a paradigm shift," *Journal of Number Theory*, vol. 131, no. 5, pp. 781–814, 2011.

[7] M. Strangio, "Efficient diffie-hellmann two-party key agreement protocols based on elliptic curves," in *ACM symposium on Applied computing*. ACM, 2005, pp. 324–331.

[8] A. Muniyandi, R. Ramasamy *et al.*, "Password based remote authentication scheme using ecc for smart card," in *International Conference on Communication, Computing & Security*. ACM, 2011, pp. 549–554.

[9] H. Boumerzoug, B. Amar Bensaber, and I. Biskri, "A key management method based on an avl tree and ecc cryptography for wireless sensor networks," in *7th ACM symposium on QoS and security for wireless and mobile networks*. ACM, 2011, pp. 57–62.

[10] S. Wang, Z. Cao, M. Strangio, and L. Wang, "Cryptanalysis and improvement of an elliptic curve diffie-hellman key agreement protocol," *IEEE Commu. Letters*, vol. 12, no. 2, pp. 149–151, 2008.

[11] E. Yooni and K. Yoo, "A new elliptic curve diffie-hellman two-party key agreement protocol," in *7th International Conference on Service Systems and Service Management (ICSSSM)*. IEEE, 2010, pp. 1–4.

[12] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Advances in Cryptology*. Springer, 1985, pp. 10–18.

[13] "IEEE Standard Specifications for Password-Based Public-Key Cryptographic Techniques," IEEE Std 1363.2-2008, 2009.

[14] "Password-authenticated key exchange (PAK) protocol," Telecommunication Standardization Sector of International Telecommunication Union (ITU-T), Recommendation X.1035,.

[15] T. Wu *et al.*, "The secure remote password protocol," in *Internet Society Symposium on Network and Distributed System Security*, 1998.

[16] ——, "SRP-6: Improvements and Refinements to the Secure Remote Password Protocol," P1363.2 working group,.

[17] M. Burmester and Y. Desmedt, "A secure and efficient conference key distribution system," in *Advances in CryptologyEUROCRYPT'94*. Springer, 1995, pp. 275–286.

[18] NIST Smart Grid, Cyber Security Working Group, "Guidelines for Smart Grid Cyber Security: Vol. 2, Privacy and the Smart Grid," Guideline, Aug. 2010. [Online]. Available: www.nist.gov/smartgrid

[19] Y. Huang, S. Tang, and J. Yuan, "Steganography in inactive frames of voip streams encoded by source codec," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 2, pp. 296–306, 2011.

[20] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing source-location privacy in sensor network routing," in *25th IEEE International Conference on Distributed Computing Systems (ICDCS 2005)*. IEEE, 2005, pp. 599–608.

[21] H. Wang, B. Sheng, and Q. Li, "Privacy-aware routing in sensor networks," *Computer Networks*, vol. 53, no. 9, pp. 1512–1529, 2009.

[22] B. Blakley, "What is Privacy, Realy?" Digital ID World, Presentation, Oct. 23, 2006. [Online]. Available: http://podcast.burtongroup.com/ip/2006/10/what_is_privacy.html.

[23] A. Pfitzmann and M. Hansen, "A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management," http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf, Aug. 2010, v0.34. [Online]. Available: http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf

[24] "AVISPA-Automated Validation of Internet Security Protocols," 2006. [Online]. Available: http://www.avispa-project.org

[25] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[26] P. McDaniel and S. McLaughlin, "Security and privacy challenges in the smart grid," *IEEE Security & Privacy*, vol. 7, no. 3, pp. 75–77, 2009.

[27] Z. Fan, P. Kulkarni, S. Gormus, C. Efthymiou, G. Kalogridis, M. Sooriyabandara, Z. Zhu, S. Lambotharan, and W. H. Chin, "Smart Grid Communications: Overview of Research Challenges, Solutions, and Standardization Activities," *IEEE Commun. Surveys & Tutorials*, vol. 15, no. 1, pp. 21–38, 2013.

[28] J. Wang and V. Leung, "A survey of technical requirements and consumer application standards for ip-based smart grid ami network," in *International Conference on Information Networking (ICOIN)*. IEEE, 2011, pp. 114–119.

[29] T. M. Chen, "Survey of cyber security issues in smart grids," *Cyber Security, Situation Management, and Impact Assessment II; and Visual Analytics for Homeland Defense and Security II (part of SPIE DSS 2010)*, pp. 77 090D–1, 2010.

[30] P. D. Ray, R. Harnoor, and M. Hentea, "Smart power grid security: A unified risk management approach," in *Security Technology (ICCST), 2010 IEEE International Carnahan Conference on*, 2010, pp. 276–285.

[31] H. Gharavi and B. Hu, "Multigate communication network for smart grid," *Proceedings of the IEEE*, vol. 99, no. 6, pp. 1028–1045, 2011.

[32] M. Fouda, Z. Fadlullah, and N. Kato, "Assessing attack threat against zigbee-based home area network for smart grid communications," in *International Conference onComputer Engineering and Systems (ICCES)*. IEEE, 2010, pp. 245–250.

[33] H. Kim, Y. Kim, K. Yang, and M. Thottan, "Cloud-based demand response for smart grid: Architecture and distributed algorithms," in *IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2011, pp. 398–403.

[34] A. Patel, J. Aparicio, N. Tas, M. Loiacono, and J. Rosca, "Assessing communications technology options for smart grid applications," in *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*. IEEE, 2011, pp. 126–131.

[35] S. Karnouskos, "Crowdsourcing information via mobile devices as a migration enabler towards the smartgrid," in *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*. IEEE, 2011, pp. 67–72.

[36] M. Alizadeh, A. Scaglione, R. J. Thomas, and D. Callaway, "Information infrastructure for cellular load management in green power delivery systems," in *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*. IEEE, 2011, pp. 13–18.

[37] V. C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, and G. P. Hancke, "Smart grid technologies: communication technologies and standards," *Industrial informatics, IEEE transactions on*, vol. 7, no. 4, pp. 529–539, 2011.

[38] K. De Craemer and G. Deconinck, "Analysis of state-of-the-art smart metering communication standards," in *Proceedings of the 5th Young Researchers Symposium*, 2010.

[39] P. P. Parikh, M. G. Kanabar, and T. S. Sidhu, "Opportunities and challenges of wireless communication technologies for smart grid applications," in *Power and Energy Society General Meeting, 2010 IEEE*. IEEE, 2010, pp. 1–7.

[40] C. Gomez and J. Paradells, "Wireless home automation networks: A survey of architectures and technologies," *IEEE Communications Magazine*, vol. 48, no. 6, pp. 92–101, 2010.

[41] T. Zhang, W. Lin, Y. Wang, S. Deng, C. Shi, and L. Chen, "The design of information security protection framework to support smart grid," in *Power System Technology (POWERCON), 2010 International Conference on*, Oct 2010, pp. 1–5.

[42] W. Yanliang, D. Song, L. Wei-Min, Z. Tao, and Y. Yong, "Research of electric power information security protection on cloud security," in *International Conference on Power System Technology (POWERCON)*. IEEE, 2010, pp. 1–6.

[43] R. Zhang, Z. Zhao, and X. Chen, "An overall reliability and security assessment architecture for electric power communication network in smart grid," in *Power System Technology (POWERCON), 2010 International Conference on*, Oct 2010, pp. 1–6.

[44] D. Wei, Y. Lu, M. Jafari, P. Skare, and K. Rohde, "An integrated security system of protecting smart grid against cyber attacks," in *Innovative Smart Grid Technologies (ISGT)*. IEEE, 2010, pp. 1–7.

[45] J. Zerbst, M. Schaefer, and I. Rinta-Jouppi, "Zone principles as cyber security architecture element for smart grids," in *IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT Europe)*. IEEE, 2010, pp. 1–8.

[46] Z. Lu, X. Lu, W. Wang, and C. Wang, "Review and evaluation of security threats on the communication networks in the smart grid," in *MILITARY COMMUNICATIONS CONFERENC (MILCOM)*. IEEE, 2010, pp. 1830–1835.

[47] Y. Wang, W. Lin, and T. Zhang, "Study on security of wireless sensor networks in smart grid," in *2010 International Conference on Power System Technology*, 2010, pp. 1–7.

[48] C. Efthymiou and G. Kalogridis, "Smart grid privacy via anonymization of smart metering data," in *First IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2010, pp. 238–243.

[49] E. Ayday and S. Rajagopal, "Secure, intuitive and low-cost device authentication for smart grid networks," in *IEEE Consumer Communications and Networking Conference (CCNC)*. IEEE, 2011, pp. 1161–1165.

[50] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile," 2008.

[51] Z. Zhang and Q. Zhang, "Verifier-based password authenticated key exchange protocol via elliptic curve," in *IEEE International Conference on Information Theory and Information Security (ICITIS)*. IEEE, 2010, pp. 407–410.

[52] H. Nicanfar and V. C. Leung, "EIBC: Enhanced Identity-Based Cryptography, a Conceptual Design," in *IEEE InternationalSystems Conference (SysCon)*. IEEE, 2012, pp. 1–7.

[53] S. Kim, E. Kwon, M. Kim, J. Cheon, S. Ju, Y. Lim, and M. Choi, "A Secure Smart-Metering Protocol Over Power-Line Communication," *IEEE Transactions on Power Delivery*, vol. 26, no. 4, pp. 2370–2379, 2011.

[54] J. Kamto, L. Qian, J. Fuller, and J. Attia, "Light-weight key distribution and management for advanced metering infrastructure," in *IEEE GLOBECOM Workshops (GC Wkshps)*. IEEE, 2011, pp. 1216–1220.

[55] F. Zhao, Y. Hanatani, Y. Komano, B. Smyth, S. Ito, and T. Kambayashi, "Secure authenticated key exchange with revocation for smart grid," in *IEEE PES Innovative Smart Grid Technologies (ISGT)*. IEEE, 2012, pp. 1–8.

[56] X. He, M. Pun, and C. Kuo, "Secure and efficient cryptosystem for smart grid using homomorphic encryption," in *IEEE PEIS nnovative Smart Grid Technologies (ISGT)*. IEEE, 2012, pp. 1–8.

[57] J. Xia and Y. Wang, "Secure Key Distribution for the Smart Grid," *IEEE Transactions on Smart Grid*, vol. 3, no. 3, pp. 1437–1443, 2012.

[58] H. Tseng, "A secure and privacy-preserving communication protocol for v2g networks," in *IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2012, pp. 2706–2711.

[59] Q. Gao, "Biometric authentication in Smart Grid," in *International Energy and Sustainability Conference (IESC)*. IEEE, 2012, pp. 1–5.

[60] S. Das, Y. Ohba, M. Kanda, D. Famolari, and S. Das, "A key management framework for ami networks in smart grid," *IEEE Communications Magazine*, vol. 50, no. 8, pp. 30–37, 2012.

[61] J. Wang, M. A. Biviji, and W. M. Wang, "Lessons learned from smart grid enabled pricing programs," in *Power and Energy Conference at Illinois (PECI), 2011 IEEE*. IEEE, 2011, pp. 1–7.

[62] Z. Vale, H. Morais, and H. Khodr, "Intelligent multi-player smart grid management considering distributed energy resources and demand response," in *Power and Energy Society General Meeting, 2010 IEEE*. IEEE, 2010, pp. 1–7.

[63] E. Heiskanen and K. Matschoss, "Exploring emerging customer needs for smart grid applications," in *Innovative Smart Grid Technologies (ISGT Europe), 2011 2nd IEEE PES International Conference and Exhibition on*. IEEE, 2011, pp. 1–7.

[64] S. Rusitschka, K. Eger, and C. Gerdes, "Smart grid data cloud: A model for utilizing cloud computing in the smart grid domain," in *First IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2010, pp. 483–488.

[65] Y. L. Sun and K. R. Liu, "Analysis and protection of dynamic membership information for group key distribution schemes." *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 2, pp. 213–226, 2007.

[66] D. Choi, H. Kim, D. Won, and S. Kim, "Advanced key-management architecture for secure scada communications," *Power Delivery, IEEE Transactions on*, vol. 24, no. 3, pp. 1154–1163, 2009.

[67] G. Yao, H. Wang, and D. Feng, "A group pake protocol using different passwords," in *International Conference on Networks Security, Wireless Communications and Trusted Computing (NSWCTC)*, vol. 1. IEEE, 2009, pp. 270–273.

[68] S. Jarecki, J. Kim, and G. Tsudik, "Flexible robust group key agreement," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 22, no. 5, pp. 879–886, 2011.

[69] Y.-H. Lin, A. Studer, H.-C. Hsiao, J. M. McCune, K.-H. Wang, M. Krohn, P.-L. Lin, A. Perrig, H.-M. Sun, and B.-Y. Yang, "Spate: small-group pki-less authenticated trust establishment," in *Proceedings of the 7th international conference on Mobile systems, applications, and services*. ACM, 2009, pp. 1–14.

[70] M. Wang, J. Pan, and J. Wang, "Password-based group authenticated key exchange protocol: From 3-party to group," in *International Conference on Network Computing and Information Security (NCIS)*, vol. 1. IEEE, 2011, pp. 239–241.

[71] Y. Chen and G. Yang, "Efficient and secure group key management based on EBS and attribute encryption," in *IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, vol. 2. IEEE, 2011, pp. 661–665.

[72] N. Mailloux, A. Miri, and M. Nevins, "Forward secure identity-based key agreement for dynamic groups," in *Ninth Annual International Conference on Privacy, Security and Trust (PST)*. IEEE, 2011, pp. 102–111.

[73] J. Teng and C. Wu, "A provable authenticated certificateless group key agreement with constant rounds," *Journal of Communications and Networks*, vol. 14, no. 1, pp. 104–110, 2012.

[74] E. Konstantinou, E. Klaoudatou, and P. Kamparmpakis, "Performance evaluation of id-based group key agreement protocols," in *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*. IEEE, 2011, pp. 377–384.

[75] P. Sakarindr and N. Ansari, "Survey of security services on group communications," *Information Security, IET*, vol. 4, no. 4, pp. 258–272, 2010.

[76] E. Klaoudatou, E. Konstantinou, G. Kambourakis, and S. Gritzalis, "A survey on Cluster-Based Group Key Agreement protocols for WSNs," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, pp. 429–442, 2011.

[77] Z. Sun and J. Ma, "Efficient key management for advanced distribution automation system," in *2nd IEEE International Conference on Network Infrastructure and Digital Content*. IEEE, 2010, pp. 794–798.

[78] J. Kim and H. Choi, "An efficient and versatile key management protocol for secure smart grid communications," in *IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2012, pp. 1823–1828.

[79] D. Dolev and A. Yao, "On the security of public-key protocols," *IEEE Transactions Information Theory*, vol. 29.

[80] M. Fouda, Z. Fadlullah, N. Kato, R. Lu, and X. Shen, "Towards a light-weight message authentication mechanism tailored for smart grid communications," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2011, pp. 1018–1023.

[81] Q. Li and G. Cao, "Multicast authentication in the smart grid with one-time signature," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 686–696, 2011.

[82] W. Juang, S. Chen, and H. Liaw, "Robust and efficient password-authenticated key agreement using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 6, pp. 2551–2556, 2008.

[83] D. Xiao-fei, M. Chuan-gui, and C. Qing-feng, "Password authenticated key exchange protocol with stronger security," in *First International Workshop on Education Technology and Computer Science (ETCS)*, vol. 2. IEEE, 2009, pp. 678–681.

[84] L. Liu and Z. Cao, "Improvement of one password-based authenticated key exchange protocol," in *Second International Symposium on Information Science and Engineering (ISISE)*. IEEE, 2009, pp. 372–375.

[85] C. Hauser, "Trust research to address uncertainty in security for the smart grid," in *IEEE PES Innovative Smart Grid Technologies (ISGT)*. IEEE, 2012, pp. 1–2.

[86] P. Jokar, H. Nicanfar, and V. C. M. Leung, "Specification-based Intrusion Detection for Home Area Networks in Smart Grids," in *IEEE SmartGridComm*, Brussels, Belgium, Oct. 2011.

[87] P. Kulkarni, S. Gormus, Z. Fan, and F. Ramos, "Ami mesh networksa practical solution and its performance evaluation," *IEEE Transactions on Smart Grid*, vol. 3, no. 3, pp. 1469–1481, 2012.

[88] Z. Fan, P. Kulkarni, S. Gormus, C. Efthymiou, G. Kalogridis, M. Sooriyabandara, Z. Zhu, S. Lambotharan, and W. H. Chin, "Smart grid communications: Overview of research challenges, solutions, and standardization activities," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 1, pp. 21–38, 2013.

[89] A. Rossello-Busquet, "G. hnem for AMI and DR," in *International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2012, pp. 111–115.

[90] J. Shi, J. Wan, H. Yan, and H. Suo, "A survey of cyber-physical systems," in *International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2011, pp. 1–6.

[91] V. Mohanty, D. Moliya, C. Hota, and M. Rajarajan, "Secure Anonymous Routing for MANETs Using Distributed Dynamic Random Path Selection," *Intell. and Security Informatics*, pp. 65–72, 2010.

[92] A. Pfitzmann and M. Hansen, "A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management," *URL: http://dud. inf. tu-dresden. de/literatur/Anon_Terminology_v0*, vol. 34, 2010.

[93] S. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[94] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, 2003.

[95] P. A. Chou and Y. Wu, "Network coding for the internet and wireless networks," *IEEE Signal Processing Magazine*, vol. 24, no. 5, pp. 77–85, 2007.

[96] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory,*, vol. 52, no. 10, pp. 4413–4430, 2006.

[97] P. Bao-xing, Y. Lu-ming, W. Wei-ping, and X. Xiao, "Linear Network Coding Construction for Multi-Source Multicast Network," in *First International Workshop on Education Technology and Computer Science (ETCS)*, vol. 3. IEEE, 2009, pp. 114–118.

[98] H. Ghasvari, M. Raayatpanah, B. Khalaj, and H. Bakhshi, "Optimal sub-graph selection over coded networks with delay and limited-size buffering," *IET Commun.*, vol. 5, no. 11, pp. 1497–1505, 2011.

[99] R. Rajkumar, "A Cyber–Physical Future," *Proceedings of the IEEE*, vol. 100, no. 13, pp. 1309–1312, 2012.

[100] H. Li, L. Lai, and H. Poor, "Multicast routing for decentralized control of cyber physical systems with an application in smart grid," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 6, pp. 1097–1107, 2012.

[101] S. Sridhar, A. Hahn, and M. Govindarasu, "Cyber–physical system security for the electric power grid," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 210–224, 2012.

[102] S. Zonouz, K. M. Rogers, R. Berthier, R. B. Bobba, W. H. Sanders, and T. J. Overbye, "SCPSE: Security-Oriented Cyber-Physical State Estimation for Power Grid Critical Infrastructures," *IEEE Transactions on Smart Grid*, vol. 3, no. 4, pp. 1790–1799, 2012.

[103] M. Stegelmann and D. Kesdogan, "GridPriv: A Smart Metering Architecture Offering k-Anonymity," in *IEEE 11th International Conference onTrust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2012, pp. 419–426.

[104] S. Li, K. Choi, and K. Chae, "An enhanced measurement transmission scheme for privacy protection in smart grid," in *Information Networking (ICOIN), 2013 International Conference on*. IEEE, 2013, pp. 18–23.

[105] X. Lin, R. Lu, H. Zhu, P.-H. Ho, X. Shen, and Z. Cao, "ASRPAKE: An anonymous secure routing protocol with authenticated key exchange for wireless ad hoc networks," in *IEEE International Conference on-Communications (ICC)*. IEEE, 2007, pp. 1247–1253.

[106] Y. Fan, Y. Jiang, H. Zhu, J. Chen, and X. S. Shen, "Network coding based privacy preservation against traffic analysis in multi-hop wireless networks," *IEEE Transactions on Wireless Communications*, vol. 10, no. 3, pp. 834–843, 2011.

[107] J. Wang, J. Wang, C. Wu, K. Lu, and N. Gu, "Anonymous communication with network coding against traffic analysis attack," in *IEEE INFOCOM*. IEEE, 2011, pp. 1008–1016.

[108] J. Wang, K. Lu, J. Wang, and C. Qiao, "Untraceability of mobile devices in wireless mesh networks using linear network coding," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 270–274.

[109] F. Atya, A. Osama, T. ElBatt, and M. Youssef, "On the flow anonymity problem in network coding," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*. IEEE, 2013, pp. 225–230.

[110] H. Nicanfar, P. Jokar, K. Beznosov, and V. C. M. Leung, "Efficient Authentication and Key Management Mechanisms for Smart Grid Communications," *accepted for publication in IEEE Systems Journal*, Jan. 2013.

[111] J. Wang, K. Lu, J. Wang, and C. Qiao, "Untraceability of mobile devices in wireless mesh networks using linear network coding," in *INFOCOM, 2013 Proceedings IEEE*, 2013, pp. 270–274.

[112] A. O. Fathy Atya, T. ElBatt, and M. Youssef, "On the flow anonymity problem in network coding," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*, 2013, pp. 225–230.

[113] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 4. IEEE, 2005, pp. 2235–2245.

[114] H. Khurana, M. Hadley, N. Lu, and D. Frincke, "Smart-grid security issues," *IEEE Security & Privacy*, vol. 8, no. 1, pp. 81–85, 2010.

[115] NIST Smart Grid, Cyber Security Working Group, "Guidelines for Smart Grid Cyber Security: Vol. 1, Smart Grid Cyber Security Strategy, Architecture, and High-Level Requirements," Guideline, Aug. 2010. [Online]. Available: www.nist.gov/smartgrid

[116] W. Su, H. Eichi, W. Zeng, and M. Chow, "A survey on the electrification of transportation in a smart grid environment," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 1–10, 2012.

[117] H. Chaudhry and T. Bohn, "Security concerns of a plug-in vehicle," in *IEEE PES Innovative Smart Grid Technologies (ISGT)*. IEEE, 2012, pp. 1–6.

[118] S. Han, S. Han, and K. Sezaki, "Estimation of Achievable Power Capacity From Plug-in Electric Vehicles for V2G Frequency Regulation: Case Studies for Market Participation," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 632–641, 2011.

[119] J. Keiser, J. Glass, N. Masuch, M. Lutzenberger, and S. Albayrak, "A distributed multi-operator W2V2G management approach," in *IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2011, pp. 273–278.

[120] A. Adigun, B. Amar Bensaber, and I. Biskri, "Proof of concept of a security based on lifetime of communication's pseudonyms for the VANETs," in *Proceedings of the second ACM international symposium on Design and analysis of intelligent vehicular networks and applications*. ACM, 2012, pp. 111–114.

[121] M. Stegelmann and D. Kesdogan, "Location privacy for vehicle-to-grid interaction through battery management," in *Ninth International Conference on Information Technology: New Generations (ITNG)*. IEEE, 2012, pp. 373–378.

[122] Z. Yang, S. Yu, W. Lou, and C. Liu, "P2: Privacy-preserving communication and precise reward architecture for V2G networks in smart grid," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 697–706, 2011.

[123] J. Sun, C. Zhang, Y. Zhang, and Y. Fang, "An identity-based security system for user privacy in vehicular ad hoc networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 21, no. 9, pp. 1227–1239, 2010.

[124] J. Martinez, P. Ruiz, and R. Marin, "Impact of the Pre-Authentication Performance in Vehicular Networks," in *Vehicular Technology Conference Fall (VTC 2010-Fall), 2010 IEEE 72nd*. IEEE, 2010, pp. 1–5.

[125] J. Huang, L. Yeh, and H. Chien, "ABAKA: An Anonymous Batch Authenticated and Key Agreement Scheme for Value-Added Services in Vehicular Ad Hoc Networks," *Vehicular Technology, IEEE Transactions on*, vol. 60, no. 1, pp. 248–262, 2011.

[126] V. Paruchuri and A. Durresi, "PAAVE: Protocol for Anonymous Authentication in Vehicular Networks using Smart Cards," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*. IEEE, 2010, pp. 1–5.

[127] S. Chang, Y. Qi, H. Zhu, J. Zhao, and X. Shen, "Footprint: Detecting Sybil Attacks in Urban Vehicular Networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, no. 6, pp. 1103–1114, 2012.

[128] H. Guo, Y. Wu, F. Bao, H. Chen, and M. Ma, "UBAPV2G: A Unique Batch Authentication Protocol for Vehicle-to-Grid Communications," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 707–714, 2011.

[129] A. Boukerche and X. Li, "An agent-based trust and reputation management scheme for wireless sensor networks," in *Global Telecommunications Conference, 2005. GLOBECOM'05. IEEE*, vol. 3. IEEE, 2005, pp. 5–pp.

[130] Y. Ota, H. Taniguchi, T. Nakajima, K. Liyanage, J. Baba, and A. Yokoyama, "Autonomous Distributed V2G (Vehicle-to-Grid) Satisfying Scheduled Charging," *IEEE Transactions on Smart Grid*, vol. 3, no. 1, pp. 559–564, 2012.

Appendix A

AVISPA codes

In this appendix, we present the detail of AVISPA codes for each of our simulations, in High Level Protocol Specification Language (HLPSL).

A.1 Related HLPSL Codes of SGMA and SGKM: Chapter 2

The HLPSL codes for AVISPA to define the SM and SAS roles are presented in Figure. A.1 and A.2, respectively. Also, the required session and environment HLPSL codes are shown in Figure A.3. Note that since AVISPA does not support arithmetic operations, we have used instead the "*xor* & *exp*" (raise to power) operators besides other security functions. The *xor* operator is used for the mod 2 "+ & −" (addition and subtraction) operations required for the authentication algorithms.

```
role sgas_Init (SM,SAS :        agent,
                PW :            symmetric_key,
                Hsh :           hash_func,
                G,N :           text,
                Snd,Rcv :       channel(dy))
played_by SM
def=
 local          State :                 nat,
        Rsm :                   text,
        Salt :                  protocol_id,
        PubKsm, PubKsas:        public_key,
        FFi, Ffi :              hash_func,
        STi, SNsm, Gsm, Gsas, Ver, K0, K, M1, M2, M, Ru, X, S0, S1, S2, S :         message

 const          sec_init_Si, sec_init_K :       protocol_id

 init  State := 0

 transition

 1. State = 0   /\ Rcv(start) =|>                       % start
   State':= 2   /\ Rsm' := new()                        % R_sm = Rnd(.)
                /\ SNsm' := new()                       % SN_sm = Rnd(.)
                /\ Gsm' := exp(G,Rsm')                          % Gsm = g^R_sm
                /\ Snd(SM.Gsm'.SNsm')                   % Sending ID_sm, g^R_sm, SN_sm

 2. State = 2   /\ Rcv(Salt'.Gsas'.FFi'.STi') =|>       % Receiving Salt, B, Encrypted F_i(.) & State i with K
   State':= 4   /\ K0':= Hsh(N.G)                       % k= hash(N,g)
                /\ Ru':= Hsh(Gsm.Gsas')                 % u= hash(A,B)
                /\ X':= Hsh(Salt'.PW)                           % x= hash(salt, pw)
                /\ Ver' := exp(G,X')                    % ver = g^x
                /\ S0' := xor(Gsas',Hsh(K0'.Ver'))      % g^b + k.g^x - k.g^x = g^b
                /\ S1' := exp(S0',Rsm)                          % (g^b)^a = g^ab
                /\ S2' := exp(exp(S0',Ru'), X')                 % ((g^b)^u)^x = g^bux
                /\ S' := xor(S1',S2')                   % S = g^ab xor g^bux
                /\ K' := Hsh(S')                                % K = hash(S)
                /\ witness(SM,SAS,k1,K')                % Checking K
                /\ secret(K',sec_init_K,{SM,SAS})       % Checking K
                /\ M1' := xor(Hsh(N),Hsh(G))                    % M1 = hash(N) xor hash(g)
                /\ M2' := Hsh(xor(SM,SNsm))                     % M2 = hash(ID xor SN)
                /\ M' := Hsh(M1'.M2'.Salt.Gsm.Gsas'.K')  % M = hash(M1,M2,salt,A,B,K)
                /\ STi' := {STi'}_inv(K')                       % Decrypting i with K
                /\ FFi' := {FFi'}_inv(K')                       % Decrypting F_i with K
                /\ PubKsas' := FFi(SAS)                 % PubK_sas = F_i(ID_SAS)
                /\ Snd({STi'}_inv(PubKsas'))                    % sending i uncrypted by PubK_sas
                /\ witness(SM,SAS,si1,STi')             % Checking state i
                /\ secret(STi',sec_init_Si,{SM,SAS})    % Checking state i

 2. State = 4   /\ Rcv(M) =|>                                   % receiving hash(M)
   State' := 6
                /\ request(SM,SAS,k2,K)                 % Checking K
                /\ request(SM,SAS,si2,STi)              % Checking state i

end role
```

Figure A.1: Chapter 2: Smart Meter (SM) HLPSL Codes

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role sgas_Resp (SAS, SM:      agent,
                Ver :         message,
                Salt :        protocol_id,
                Hsh :         hash_func,
                G,N :         text,
                Snd, Rcv :    channel(dy))


played_by SAS

def=
 local State :  nat,
                Rsas :                  text,
                FFi, Ffi :              hash_func,
                PubKsm, PubKsas :       public_key,
                Resi, Resj, STi, SNsm, Ru, M1, M2, M, K0, K, Gsm, Gsas, X, S0, S1, S :      message

 const          sec_resp_Si, sec_resp_K :      protocol_id

 init  State := 1

 transition

 1. State = 1   /\ Rcv(SM.Gsm',SNsm') =|>                       % A and g^a
   State':= 3   /\ K0' := Hsh(N.G)                              % k = hash(N,g)
                /\ Rsas' := new()                               % b = Rnd()
                /\ Gsas' := xor(exp(G,Rsas'),Hsh(K0'.Ver))      % B = g^b + k.g^x
                /\ Ru' := Hsh(Gsm'.Gsas')                       % u = hash(A,B)
                /\ S0' := exp(Gsm',Rsas')                       % (g^a)^b = g^ab
                /\ S1' := exp(exp(Ver,Ru'),Rsas')               % ((g^x)^u)^b = g^bux
                /\ S' := xor(S0',S1')                           % S = g^ab xor g^bux
                /\ K' := Hsh(S')                                % K = hash(S)
                /\ M1' := xor(Hsh(N),Hsh(G))                    % M1 = hash(N) xor hash(g)
                /\ M2' := Hsh(xor(SM,SNsm'))                    % M2 = hash(ID xor SN)
                /\ M' := Hsh(M1'.M2'.Salt.Gsm'.Gsas'.K')        % M = hash(M1,m2,salt,A,B,K)
                /\ STi' := new()                                % State i
                /\ PubKsas' := FFi(SAS)                         % PubK_sas = F_i(ID_sas)
                /\ Snd(Salt.Gsas'.{FFi}_K'.{STi'}_K')           % sending salt,B
                /\ witness(SAS,SM,k2,K')                        % Checking K
                /\ secret(K',sec_resp_K,{SM,SAS})               % Checking K

 2. State = 3   /\ Rcv({Resi'}_inv(PubKsas)) =|>                % receiving state i
   State':= 5    /\ witness(SAS,SM,si2,Resi')                  % Checking state i
                /\ secret(Resi',sec_resp_Si,{SM,SAS})           % Checking state i
                 /\ Snd(M)                                      % sending M
                /\ request(SAS,SM,si1,Resi')                    % Checking state i
                /\ request(SAS,SM,k1,K)                         % Checking K


end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure A.2: Chapter 2: Server (SAS) HLPSL Codes

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role session(SM,SAS :        agent,
             PW :            symmetric_key,
             Salt :          protocol_id,
             Hsh :           hash_func,
             G,N :           text)
def=

  local  SndSM, RcvSM, SndSAS, RcvSAS :    channel (dy)

  composition
      sgas_Init(SM,SAS,PW,Hsh,G,N,SndSM,RcvSM)  /\
      sgas_Resp(SAS,SM,exp(G,Hsh(Salt.PW)),Salt,Hsh,G,N,SndSAS,RcvSAS)     % x = hash(Salt, pw) & Ver = g^x

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role environment()

 def=
  const si1, si2, k1, k2 :       protocol_id,
        sm, sas, intruder :      agent,
        kab, kai, kbi :          symmetric_key,
        s_ab,s_ai,s_bi :         protocol_id,
        hsh :                    hash_func,
        g, n :                   text

 intruder_knowledge = {i, kai, kbi, s_ai, s_bi}

  composition
          session(sm,sas,kab,s_ab,hsh,g,n)
       /\ session(sm,intruder,kai,s_ai,hsh,g,n)
       /\ session(sas,intruder,kbi,s_bi,hsh,g,n)

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

goal

   secrecy_of   sec_init_Si, sec_init_K, sec_resp_Si, sec_resp_K

   authentication_on k1
   authentication_on k2

   authentication_on si1
   authentication_on si2

end goal

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

environment()
```

Figure A.3: Chapter 2: Session and Environment HLPSL Codes

A.2 Related HLPSL Codes of Group Key: Chapter 3

The following are the HLPSL codes of entities of Chapter 3, Group Key mechanism. Figure A.15 presents the evaluation program and AVISPA related HLPSL codes for the session, environment and goal sections. Also, HLPSL codes of the four entities A, B, C, and D roles are shown in Figure A.10, A.12, A.14 and A.8, respectively.

```
%% PROTOCOL*: SGGM
%%HLPSL:
role session (A,B,C,D: agent, G: text, Hsh: hash_func,
              Kab,Kbc,Kcd,Kda: symmetric_key, Pw: symmetric_key)

def=
   local   SA,RA,SB,RB,SC,RC,SD,RD: channel(dy)

   composition
          sgsk_1(D,A,B,G,Hsh,Kda,Kab,SA,RA,Pw)  /\
          sgsk_2(A,B,C,G,Hsh,Kab,Kbc,SB,RB,Pw)  /\
          sgsk_3(B,C,D,G,Hsh,Kbc,Kcd,SC,RC,Pw)  /\
          sgsk_4(C,D,A,G,Hsh,Kcd,Kda,SD,RD,Pw)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role environment() def=

  const   gk_ab, gk_bc, gk_cd, gk_da         : protocol_id,
          gk_ba, gk_cb, gk_dc, gk_ad         : protocol_id,
          a,b,c,d                            : agent,
          kab,kbc,kcd,kda                    : symmetric_key,
          kai,kia,kbi,kib,kci,kic,kdi,kid    : symmetric_key,
          pw                                 : symmetric_key,
          g                                  : text,
          hsh                                : hash_func

  intruder_knowledge  = {a,b,c,d,kai,kia,kbi,kib,kci,kic,kdi,kid}

  composition

     session(a,b,c,d,g,hsh,kab,kbc,kcd,kda,pw)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
goal
  % secrecy_of GK
     secrecy_of     sec_GK_AB, sec_GK_BC, sec_GK_CD, sec_GK_DA,
                    sec_GK_AD, sec_GK_BA, sec_GK_CB, sec_GK_DC

% authentication
  authentication_on gk_ab
  authentication_on gk_ba
  authentication_on gk_bc
  authentication_on gk_cb
  authentication_on gk_cd
  authentication_on gk_dc
  authentication_on gk_da
  authentication_on gk_ad
end goal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
environment()
```

Figure A.4: Chapter 3: Main HLPSL Codes

```
%% PROTOCOL*: SGGM
%%HLPSL:
role sgsk_1 (D,A,B   : agent,
          G          : text,
          Hsh        : hash_func,
          Kda,Kab    : symmetric_key,
               Snd,Rcv    : channel(dy),
               Pw         : symmetric_key)

played_by A

def=
  local   State          : nat,
     Ra                  : text,
     Gd,Gcd,Gbcd         : message,                          % in
     Gbcda,GK            : message,                          % mine
     Ga,Gda,Gcda         : message,                          % out
     ST_DA,ST_AB         : message                           % verifiers

  const sec_GK_AB, sec_GK_AD : protocol_id

  init  State := 0

  transition

 1. State = 0  /\ Rcv(start) =|>                  % start
    State':= 1 /\ Ra' := new()                    % a
               /\ Ga' := exp(G,Ra')               % g^a
               /\ Snd(A.{{Ga'}_Kab}_Pw.B)         % send:g^a                  s1

 2. State = 1  /\ Rcv(D.{{Gd'}_Kda.{Gcd'}_Kda.{Gbcd'}_Kda}_Pw.A) =|>
                                                  % receive: g^d, g^cd, g^bcd
    State':= 2 /\ Gda' := exp(Gd',Ra)             % g^da
               /\ Gcda' := exp(Gcd',Ra)           % g^cda
               /\ Gbcda' := exp(Gbcd',Ra)         % g^bcda
               /\ Snd(A.{{Gda'}_Kab.{Gcda'}_Kab}_Pw.B)% send: g^da, g^cda
               /\ GK' := Hsh(Pw.Gbcda')           % group key: ash(pw,g^abcd)
               /\ ST_AB' := Hsh(Pw.Gcda'.Gbcda')  % A<-->B:hash(pw,g^cda,g^abcd)
               /\ ST_DA' := Hsh(Pw.Gbcd'.Gbcda')  % A<-->D:hash(pw,g^bcd,g^abcd)
               /\ witness(A,B,gk_ab,GK')          % Checking group key with B
               /\ secret(GK',sec_GK_AB,{A,B})     % Checking group key with B
               /\ witness(A,D,gk_ad,GK')          % Checking group key with D
               /\ secret(GK',sec_GK_AD,{A,D})     % Checking group key with D s5

 3. State = 2  /\ Rcv(D.ST_DA.A) =|>              % receive verifier from D
    State':= 3 /\ Snd(A.ST_AB.B)                  % send verifier to B
               /\ request(A,B,gk_ba,GK)           % Checking group key with B
               /\ request(A,D,gk_da,GK)           % Checking group key with D s9
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure A.5: Chapter 3: First Entity HLPSL Codes

```
%% PROTOCOL*: SGGM
%%HLPSL:
role sgsk_2 (A,B,C       : agent,
             G           : text,
             Hsh         : hash_func,
             Kab,Kbc     : symmetric_key,
             Snd,Rcv     : channel(dy),
             Pw          : symmetric_key)

played_by B

def=
  local State            : nat,
     Rb                  : text,
     Ga,Gda,Gcda         : message,                    % in
     Gcdab,GK            : message,                    % mine
     Gb,Gab,Gdab         : message,                    % out
     ST_AB,ST_BC         : message                     % verifiers

  const sec_GK_BC, sec_GK_BA  : protocol_id

  init  State := 0

transition

 1. State = 0  /\ Rcv(A.{{Ga'}_Kab}_Pw.B) =|>     % receive: g^a
    State':= 1 /\ Rb' := new()                    % b
               /\ Gb' := exp(G,Rb')               % g^b
               /\ Gab' := exp(Ga',Rb')            % g^ab
               /\ Snd(B.{{Gb'}_Kbc.{Gab'}_Kbc}_Pw.C) % send: g^b, g^ab        s2

 2. State = 1  /\ Rcv(A.{{Gda'}_Kab.{Gcda'}_Kab}_Pw.B) =|>% receive: g^da, g^cda
    State':= 2 /\ Gdab' := exp(Gda',Rb)           % g^dab
               /\ Gcdab' := exp(Gcda',Rb)         % g^cdab
               /\ Snd(B.{{Gdab'}_Kbc}_Pw.C)       % send: g^dab
               /\ GK' := Hsh(Pw.Gcdab')           % group key: hash(pw,g^abcd)
               /\ ST_BC' := Hsh(Pw.Gdab'.Gcdab')  % B<->C: hash(pw,g^dab,g^abcd)
               /\ ST_AB' := Hsh(Pw.Gcda'.Gcdab')  % B<->A: hash(pw,g^cda,g^abcd)
               /\ witness(B,C,gk_bc,GK')          % Checking group key with C
               /\ secret(GK',sec_GK_BC,{B,C})     % Checking group key with C
               /\ witness(B,A,gk_ba,GK')          % Checking group key with A
               /\ secret(GK',sec_GK_BA,{B,A})     % Checking group key with A s6

 3. State = 2  /\ Rcv(A.ST_AB.B) =|>              % receive verifier from A
    State':= 3 /\ Snd(B.ST_BC.C)                  % send verifier to C
               /\ request(B,C,gk_cb,GK)           % Checking group key with C
               /\ request(B,A,gk_ab,GK)           % Checking group key with As10
%
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure A.6: Chapter 3: Second Entity HLPSL Codes

```
%% PROTOCOL*: SGGM
%%HLPSL:
role sgsk_3 (B,C,D       : agent,
             G           : text,
             Hsh         : hash_func,
             Kbc,Kcd     : symmetric_key,
             Snd,Rcv     : channel(dy),
             Pw          : symmetric_key)

played_by C

def=
  local State            : nat,
         Rc              : text,
         Gb,Gab,Gdab     : message,                    % in
         Gdabc,GK        : message,                    % mine
         Gc,Gbc,Gabc     : message,                    % out
         ST_BC,ST_CD     : message                     % verifiers

  const sec_GK_CD, sec_GK_CB  : protocol_id

  init  State := 0

  transition

 1. State = 0  /\ Rcv(B.{{Gb'}_Kbc.{Gab'}_Kbc}_Pw.C) =|>    % receive: g^b, g^ab
    State':= 1 /\ Rc' := new()                          % c
               /\ Gc' := exp(G,Rc')                     % g^c
               /\ Gbc' := exp(Gb',Rc')                  % g^bc
               /\ Gabc' := exp(Gab',Rc')                % g^abc
               /\ Snd(C.{{Gc'}_Kcd.{Gbc'}_Kcd.{Gabc'}_Kcd}_Pw.D)
                                              % send: g^c, g^bc, g^abc         s3

 2. State = 1  /\ Rcv(B.{{Gdab'}_Kbc}_Pw.C) =|>   % receive: g^dab
    State':= 2 /\ Gdabc' := exp(Gdab',Rc)          % g^abcd
               /\ GK' := Hsh(Pw.Gdabc')            % group key:hash(pw,g^abcd)
               /\ ST_CD' := Hsh(Pw.Gabc.Gdabc')    % C<->D: hash(pw,g^abc,g^abcd)
               /\ ST_BC' := Hsh(Pw.Gdab'.Gdabc')   % C<->B: hash(pw,g^dab,g^abcd)
               /\ Snd(C.ST_CD'.D)                  % send verifier to D
               /\ witness(C,D,gk_cd,GK')           % Checking group key with D
               /\ secret(GK',sec_GK_CD,{C,D})      % Checking group key with D
               /\ witness(C,B,gk_cb,GK')           % Checking group key with B
               /\ secret(GK',sec_GK_CB,{C,B})      % Checking group key with B s7

 3. State = 2  /\ Rcv(B.ST_BC.C) =|>               % receive: hash(hash(g^abc),1)
    State':= 3 /\ request(C,D,gk_dc,GK)            % Checking group key with D
               /\ request(C,B,gk_bc,GK)            % Checking group key with Bs11
%
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure A.7: Chapter 3: Third Entity HLPSL Codes

```
%% PROTOCOL*: SGGM
%%HLPSL:
role sgsk_4 (C,D,A  : agent,
          G         : text,
          Hsh       : hash_func,
          Kcd,Kda   : symmetric_key,
          Snd,Rcv   : channel(dy),
          Pw    : symmetric_key)

played_by D

def=
  local State            : nat,
          Rd             : text,
          Gc,Gbc,Gabc    : message,              % in
          Gabcd,GK       : message,              % mine
          Gd,Gcd,Gbcd    : message,              % out
          ST_CD,ST_DA    : message               % verifiers

  const sec_GK_DA, sec_GK_DC  : protocol_id

  init  State := 0

  transition

 1. State = 0  /\ Rcv(C.{{Gc'}_Kcd.{Gbc'}_Kcd.{Gabc'}_Kcd}_Pw.D) =|>
                                              % receive: g^c, g^bc, g^abc
    State':= 1 /\ Rd' := new()                % d
               /\ Gd' := exp(G,Rd')           % g^d
               /\ Gcd' := exp(Gc',Rd')        % g^cd
               /\ Gbcd' := exp(Gbc',Rd')      % g^bcd
               /\ Gabcd' := exp(Gabc',Rd')    % g^abcd
               /\ Snd(D.{{Gd'}_Kda.{Gcd'}_Kda.{Gbcd'}_Kda}_Pw.A)
                                              % send: g^d, g^cd, g^bcd
               /\ GK' := Hsh(Pw.Gabcd')       % group key: hash(pw,g^abcd)
               /\ ST_DA' := Hsh(Pw.Gbcd'.Gabcd')  % D<->A: hash(pw,g^bcd,g^abcd)
               /\ ST_CD' := Hsh(Pw.Gabc'.Gabcd')  % D<->C: hash(pw,g^abc,g^abcd)
               /\ witness(D,A,gk_da,GK')      % Checking group key with A
               /\ secret(GK',sec_GK_DA,{D,A})     % Checking group key with A
               /\ witness(D,C,gk_dc,GK')      % Checking group key with C
               /\ secret(GK',sec_GK_DC,{D,C})     % Checking group key with C s4


 2. State = 1  /\ Rcv(C.ST_CD.D) =|>          % receive verifier from C
    State':= 2 /\ Snd(D.ST_DA.A)              % send verifier to A
               /\ request(D,A,gk_ad,GK)       % Checking group key with A
               /\ request(D,C,gk_cd,GK)       % Checking group key with C s8

end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure A.8: Chapter 3: Forth Entity HLPSL Codes

A.3 Related HLPSL Codes of MCEPAK: Chapter 4

The following are the HLPSL codes of entities of Chapter 4, ECC based multi-layer key construction mechanism. Figure A.15 presents our evaluation program and AVISPA related HLPSL codes for the session, environment and goal sections. Also, HLPSL codes of the appliance is shown in Figure A.10 while HLPSL codes of the controllers roles are shown in Figure A.11, A.12, A.13 and A.14.

```
role session (A,H,B,N,C: agent, G: text, Hsh: hash_func,
              Khb,Kbn,Knc: symmetric_key, Pwd: symmetric_key)
def=
   local   SA,RA,SH,RH,SB,RB,SN,RN,SC,RC: channel(dy)

   composition

           sk_1(A,H,B,N,C,G,Hsh,SA,RA,Pwd)  /\
           sk_2(A,H,B,N,C,G,Hsh,Khb,SH,RH,Pwd)  /\
           sk_3(A,H,B,N,C,G,Hsh,Khb,Kbn,SB,RB)  /\
           sk_4(A,H,B,N,C,G,Hsh,Kbn,Knc,SN,RN)  /\
           sk_5(A,H,B,N,C,G,Hsh,Knc,SC,RC)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role environment() def=

  const k_ah, k_ab, k_an, k_ac                    : protocol_id,
        k_ha, k_ba, k_na, k_ca                    : protocol_id,
        a,h,b,n,c                                 : agent,
        khb,kbn,knc                               : symmetric_key,
        kai,kia,khi,kih,kbi,kib,kni,kin,kci,kic   : symmetric_key,
        pwd                                       : symmetric_key,
        g                                         : text,
        hsh                                       : hash_func

  intruder_knowledge  = {a,h,b,n,c,kai,kia,khi,kih,kbi,kib,kni,kin,kci,kic}

  composition

     session(a,h,b,n,c,g,hsh,khb,kbn,knc,pwd)

end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
goal

  % secrecy_of GK
  secrecy_of   sec_K_AH, sec_K_HA, sec_K_AB, sec_K_BA,
               sec_K_AN, sec_K_NA, sec_K_AC, sec_K_CA

  % authentication
  authentication_on k_ah
  authentication_on k_ha
  authentication_on k_ab
  authentication_on k_ba
  authentication_on k_an
  authentication_on k_na
  authentication_on k_ac
  authentication_on k_ca

end goal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
environment()
```

Figure A.9: Chapter 4: Main HLPSL Codes

```
role sk_1 (A,H,B,N,C      : agent,
           G              : text,
           Hsh            : hash_func,
           Snd,Rcv        : channel(dy),
           Pwd            : symmetric_key)
played_by A
def=
  local State             : nat,
      Ra                  : text,                  % Random variable
      TKah                : symmetric_key,         % temp key A  H
      TKab                : symmetric_key,         % temp key A  B
      TKan                : symmetric_key,         % temp key A  N
      TKac                : symmetric_key,         % temp key A  C
      Xah                 : message,               % out
      Yh,Yhb,Yhbn,Yhbnc   : message,               % in
      Xhb,Xbn,Xnc,Xcc     : message,               % mine
      Kh,Kb,Kn,Kc         : message                % keys
  const sec_K_AH, sec_K_AB, sec_K_AN, sec_K_AC : protocol_id
  init  State := 0
  transition
 1. State = 0   /\ Rcv(start) =|>                  % start
    State':= 1  /\ Ra' := new()                    % a
                /\ Xah' := exp(G,Ra')              % g^a
                /\ TKah' := Hsh(A.Pwd.H)           % Temp key
                /\ TKab' := Hsh(TKah'.B)           % Temp key
                /\ TKan' := Hsh(TKab'.N)           % Temp key
                /\ TKac' := Hsh(TKan'.C)           % Temp key
                /\ Snd(A.{Xah'}_TKah'.H)           % send to h: g^a  s1
 2. State = 1   /\ Rcv(H.{Yh'}_TKah.{Yhb'}_TKah.{Yhbn'}_TKah.{Yhbnc'}_TKah.A) =|>
                                                   % receive: g^h, g^hb, g^hbn , g^hbnc
    State':= 2  /\ Xhb' := exp(Yh',Ra)             % g^ah
                /\ Xbn' := exp(Yhb',Ra)            % g^ahb
                /\ Xnc' := exp(Yhbn',Ra)           % g^ahbn
                /\ Xcc' := exp(Yhbnc',Ra)          % g^ahbnc
                /\ Kh' := Hsh(Xhb'.TKah.TKab.Xah)  % Shared key A  H
                /\ Kb' := Hsh(Xbn'.TKab.TKan.Xhb') % Shared key A  B
                /\ Kn' := Hsh(Xnc'.TKan.TKac.Xbn') % Shared key A  N
                /\ Kc' := Hsh(Xcc'.TKac.TKac.Xnc') % Shared key A  C
                /\ witness(A,H,k_ah,Kh')           % Check shared key with H
                /\ secret(Kh',sec_K_AH,{A,H})      % Check shared key with H
                /\ witness(A,B,k_ab,Kb')           % Checking shared key with B
                /\ secret(Kb',sec_K_AB,{A,B})      % Checking shared key with B
                /\ witness(A,N,k_an,Kn')           % Checking shared key with N
                /\ secret(Kn',sec_K_AN,{A,N})      % Checking shared key with N
                /\ witness(A,C,k_ac,Kc')           % Checking shared key with C
                /\ secret(Kc',sec_K_AC,{A,C})      % Checking shared key with C
                /\ request(A,H,k_ha,Kh')           % Checking shared key with H
                /\ request(A,B,k_ba,Kb')           % Checking shared key with B
                /\ request(A,N,k_na,Kn')           % Checking shared key with N
                /\ request(A,C,k_ca,Kc')           % Checking shared key with C
end role
```

Figure A.10: Chapter 4: HLPSL Codes of New Appliance (A_N)

```
role sk_2 (A,H,B,N,C     : agent,
           G             : text,
           Hsh           : hash_func,
           Khb           : symmetric_key,
           Snd,Rcv       : channel(dy),
           Pwd           : symmetric_key)

played_by H

def=
  local State            : nat,
     Rh                  : text,
     TKah                : symmetric_key,         % Temp key A  H
     TKab                : symmetric_key,         % Temp key A  B
     Temp1               : message,               % temp
     Xah                 : message,               % in
     Xhb                 : message,               % out
     Yb,Ybn,Ybnc         : message,               % in
     Yh,Yhb,Yhbn,Yhbnc   : message,               % out
     Kh                  : message                % key

  const sec_K_HA         : protocol_id

  init  State := 0

  transition

 1. State = 0  /\ Rcv(A.Temp1'.H) =|>
    State':= 1 /\ TKah' := Hsh(A.Pwd.H)           % receive from A: g^a%
               /\ Xah' := {Temp1'}_TKah'          % Temp key A  H
               /\ Rh' := new()                    % h
               /\ Xhb' := exp(Xah',Rh')           % g^ah
               /\ TKab' := Hsh(TKah'.B)           % Temp key A  B
               /\ Snd(H.{TKab'}_Khb.{Xhb'}_Khb.B) % send to B: g^ah

 2. State = 1  /\ Rcv(B.{Yb'}_Khb.{Ybn'}_Khb.{Ybnc'}_Khb.H) =|>
                                % receive fron B: g^b , g^bn , g^bnc
    State':= 2 /\ Yh' := exp(G,Rh)                % g^h
               /\ Yhb' := exp(Yb',Rh)             % g^hb
               /\ Yhbn' := exp(Ybn',Rh)           % g^hbn
               /\ Yhbnc' := exp(Ybnc',Rh)         % g^hbnc
               /\ Snd(H.{Yh'}_TKah.{Yhb'}_TKah.{Yhbn'}_TKah.{Yhbnc'}_TKah.A)
                                     % send to A: g^h , g^hb , g^hbn , g^hbnc
               /\ Kh' := Hsh(Xhb.TKah.TKab.Xah)   % Shared key A  H
               /\ witness(H,A,k_ha,Kh')           % Check shared key with A
               /\ secret(Kh',sec_K_HA,{H,A})      % Check shared key with A
               /\ request(H,A,k_ah,Kh')           % Check shared key with A


end role
```

Figure A.11: Chapter 4: HLPSL Codes of Home Controller (H_C)

```
role sk_3 (A,H,B,N,C    : agent,
           G            : text,
           Hsh          : hash_func,
           Khb,Kbn      : symmetric_key,
           Snd,Rcv      : channel(dy))

played_by B

def=
  local State       : nat,
    Rb              : text,
    TKab            : symmetric_key,            % Temp key A  B
    TKan            : symmetric_key,            % Temp key A  N
    Xhb             : message,                  % in
    Xbn             : message,                  % out
    Yn,Ync          : message,                  % in
    Yb,Ybn,Ybnc     : message,                  % out
    Kb              : message                   % key

  const sec_K_BA    : protocol_id

  init  State := 0

  transition

 1. State = 0  /\ Rcv(H.{TKab'}_Khb.{Xhb'}_Khb.B) =|>
                                                % receive from H: g^ah
   State':= 1  /\ Rb' := new()                  % b
               /\ Xbn' := exp(Xhb',Rb')         % g^ahb
              /\ TKan' := Hsh(TKab'.N)          % temp key A  N
              /\ Snd(B.{TKan'}_Kbn.{Xbn'}_Kbn.N)% snd to N: g^ahb

 2. State = 1  /\ Rcv(N.{Yn'}_Kbn.{Ync'}_Kbn.B) =|>
                                               % rcve from N: g^n , g^cn
   State':= 2 /\ Yb' := exp(G,Rb)               % g^b
              /\ Ybn' := exp(Yn',Rb)            % g^bn
              /\ Ybnc' := exp(Ync',Rb)          % g^bnc
              /\ Snd(B.{Yb'}_Khb.{Ybn'}_Khb.{Ybnc'}_Khb.H)
                                % send to H: g^b , g^bn , g^bnc
              /\ Kb' := Hsh(Xbn.TKab.TKan.Xhb)     % Shred key A  B
              /\ witness(B,A,k_ba,Kb')     % Check shred key with A
              /\ secret(Kb',sec_K_BA,{B,A})% Check shred key with A
              /\ request(B,A,k_ab,Kb')     % Check shred key with A
end role
```

Figure A.12: Chapter 4: HLPSL Codes of Building Controllers (B_C)

```
role sk_4 (A,H,B,N,C    : agent,
           G            : text,
           Hsh          : hash_func,
           Kbn,Knc      : symmetric_key,
           Snd,Rcv      : channel(dy))

played_by N

def=
  local State       : nat,
    Rn              : text,
    TKan            : symmetric_key,    % Temp key A  N
    TKac            : symmetric_key,    % Temp key A  C
    Xbn             : message,          % in
    Xnc             : message,          % out
    Yc              : message,          % in
    Yn,Ync          : message,          % out
    Kn              : message           % key

  const sec_K_NA    : protocol_id

  init  State := 0

  transition

 1. State = 0   /\ Rcv(B.{TKan'}_Kbn.{Xbn'}_Kbn.N) =|>
                                        % receive from B: g^ahb
    State':= 1  /\ Rn' := new() % n
                /\ Xnc' := exp(Xbn',Rn') % g^ahbn
                /\ TKac' := Hsh(TKan'.C) % temp key A  C
                /\ Snd(N.{TKac'}_Knc.{Xnc'}_Knc.C)
                                    % send to C: g^ahbn

 1. State = 1 /\ Rcv(C.{Yc'}_Knc.N) =|>  % rcve frm C: g^c
   State':= 2 /\ Yn' := exp(G,Rn)        % g^n
              /\ Ync' := exp(Yc',Rn)     % g^cn
              /\ Snd(N.{Yn'}_Kbn.{Ync'}_Kbn.B)
                                        % send to B: g^n , g^cn
             /\ Kn' := Hsh(Xnc.TKan.TKac.Xbn)      % Shred key A N
             /\ witness(N,A,k_na,Kn')      %Check shred key with A
             /\ secret(Kn',sec_K_NA,{N,A}) %Check shred key with A
             /\ request(N,A,k_an,Kn')      %Check shred key with A
end role
```

Figure A.13: Chapter 4: HLPSL Codes of Neighbourhood Controller (N_C)

```
role sk_5 (A,H,B,N,C  : agent,
           G          : text,
           Hsh        : hash_func,
           Knc        : symmetric_key,
           Snd,Rcv    : channel(dy))

played_by C

def=
  local State     : nat,
    Rc            : text,
    TKac          : symmetric_key,  % Temp key A  C
    Xnc           : message,        % in
    Yc            : message,        % out
    Xcc           : message,        % mine
    Kc            : message         % key

  const sec_K_CA  : protocol_id

  init  State := 0

  transition

 1. State = 0  /\ Rcv(N.{TKac'}_Knc.{Xnc'}_Knc.C) =|>
                                % receive from N: g^ahbn
   State':= 1  /\ Rc' := new()            % c
               /\ Yc' := exp(G,Rc')       % g^c
               /\ Xcc' := exp(Xnc',Rc')   % g^ahbnc
               /\ Snd(C.{Yc'}_Knc.N)      % send to N: g^c
               /\ Kc' := Hsh(Xcc'.TKac'.TKac'.Xnc')
                                          % Shared key A  C
               /\ witness(C,A,k_ca,Kc')
                                % Checking shared key with A
               /\ secret(Kc',sec_K_CA,{C,A})
                                % Checking shared key with A
               /\ request(C,A,k_ac,Kc')
                                % Checking shared key with A
end role
```

Figure A.14: Chapter 4: HLPSL Codes of Central Controller (C_C)

A.4 Related HLPSL Codes of Privacy-Preserved Security Solution: Chapter 6

Figure A.15 shows the evaluation program and HLPSL codes for the session, environment and goal sections. In addition, HLPSL codes of the server and EV are shown in Figure A.16, Figure A.17 respectively.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role session(EV,SGS     : agent,
             PW         : symmetric_key,
             Salt       : protocol_id,
             H          : hash_func,
             G,P        : text)
def=

   local SndEV,RcvEV,SndSGS,RcvSGS: channel (dy)

   composition
           pev_Init(EV,SGS,PW,H,G,P,SndEV,RcvEV)  /\
           pev_Resp(SGS,EV,PW,Salt,H,G,P,SndSGS,RcvSGS)

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role environment()
def=

  const k11,k12,k21,k22 : protocol_id,
      ev,sgs,I          : agent,
      kab,kai,kib       : symmetric_key,
      s_ab,s_ai,s_ib    : protocol_id,
      hsh               : hash_func,
      g,p               : text

  intruder_knowledge = {i, kai, kib, s_ai, s_ib}
  composition
         session(ev,sgs,kab,s_ab,hsh,g,p)
      /\ session(ev,i,kai,s_ai,hsh,g,p)
      /\ session(i,sgs,kib,s_ib,hsh,g,p)

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

goal

    secrecy_of sec_init_K1, sec_resp_K1, sec_init_K2, sec_resp_K2

    authentication_on k12
    authentication_on k11

    authentication_on k22
    authentication_on k21

end goal

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


environment()
```

Figure A.15: Chapter 6: Main HLPSL Codes

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role pev_Resp (SGS, EV    : agent,
               PW1        : symmetric_key,
               Salt1      : protocol_id,
               H          : hash_func,
               G,P        : text,
               Snd, Rcv   : channel(dy))
played_by SGS                                                    % smart grid server

def=
  local      State        : nat,
             SGSr1, SGSr2 : text,
             Salt2        : protocol_id,
             EVg1, SGSg1, U1, Ver1, S1, K0, K1, M1  : message,
             EVg2, SGSg2, U2, Ver2, S2, K2, M2      : message,
             PW2, PW3     : symmetric_key

  const sec_resp_K1, sec_resp_K2       : protocol_id

  init  State := 0

  transition

  1. State = 0  /\ Rcv(EV.EVg1') =|>                             % ID_EV and A (g^a)
     State':= 1 /\ K0' := H(P.G)                                 % k = hash(p,g)
                /\ SGSr1' := new()                               % b = Rnd()
                /\ Ver1' := exp(G,H(Salt1.PW1))                  % ver = g^x
                /\ SGSg1' := xor(exp(G,SGSr1'),H(K0'.Ver1'))     % B = g^b + k.g^x
                /\ U1' := H(EVg1'.SGSg1')                        % u = hash(A,B)
                /\ S1' := exp(exp(Ver1',U1'),SGSr1')             % ((g^x)^u)^b = g^bux
                /\ K1' := H((exp(EVg1',SGSr1')).S1')             % K = hash(g^ab , g^bux)
                /\ M1' := H((xor(H(P),H(G))).(H(xor(EV,PW1))).Salt1.EVg1'.SGSg1'.K1')
                                                                 % M = hash(M1,M2,salt,A,B,K)
                /\ Snd(Salt1.{SGSg1'}_(exp(G,H(Salt1.H(EV.PW1)))))      % sending salt,B
                /\ witness(SGS,EV,k12,K1')
                /\ secret(K1',sec_resp_K1,{EV,SGS})

  2. State = 1  /\ Rcv(M1) =|>                                  % A and g^a
     State':= 3 /\ PW2' := H(EVg1.SGSg1.K1)
                /\ Snd(H(PW2'))
                /\ request(SGS,EV,k11,K1)

  3. State = 3  /\ Rcv(EV.EVg2') =|>                            % A and g^a
     State':= 4 /\ Salt2' := new()                               % New salt = Rnd()
                /\ SGSr2' := new()                               % b = Rnd()
                /\ Ver2' := exp(G,H(Salt2'.PW2))                 % ver = g^x
                /\ SGSg2' := xor(exp(G,SGSr2'),H(K0.Ver2'))      % B = g^b + k.g^x
                /\ U2' := H(EVg2'.SGSg2')                        % u = hash(A,B)
                /\ S2' := exp(exp(Ver2',U2'),SGSr2')             % ((g^x)^u)^b = g^bux
                /\ K2' := H((exp(EVg2',SGSr2')).S2')             % K = hash(g^ab , g^bux)
                /\ M2' := H((xor(H(P),H(G))).(H(xor(EV,PW2))).Salt2'.EVg2'.SGSg2'.K2')
                                                                 % M = hash(M1,M2,salt,A,B,K)
                /\ Snd(Salt2'.{SGSg2'}_(exp(G,H(Salt2'.H(EV.PW2)))))    % sending salt,B
                /\ witness(SGS,EV,k22,K2')
                /\ secret(K2',sec_resp_K2,{EV,SGS})

  4. State = 4  /\ Rcv(M2) =|>
     State':= 5 /\ PW3' := H(EVg2.SGSg2.K2)
                /\ Snd(H(PW3'))
                /\ request(SGS,EV,k21,K2)

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure A.16: Chapter 6: HLPSL Codes of Smart Grid Server

```
%%  HLPSL:
role pev_Init (EV,SGS     : agent,
               PW1        : symmetric_key,
               H          : hash_func,
               G,P        : text,
               Snd,Rcv    : channel(dy))

played_by EV                                          % electric vehicle

def=
  local      State        : nat,
             EVr1, EVr2   : text,
             Salt1, Salt2 : protocol_id,
             EVg1, SGSg1, X1, Ver1, U1, S1, K0, K1, M1     : message,
             EVg2, SGSg2, X2, Ver2, U2, S2, K2, M2         : message,
             PW2, PW3     : symmetric_key

  const sec_init_K1, sec_init_K2       : protocol_id

  init  State := 0

  transition

  1. State = 0  /\ Rcv(start) =|>
     State':= 1 /\ EVr1' := new()                       % a = rand()
                /\ EVg1' := exp(G,EVr1')                % A = g^a
                /\ Snd(EV.EVg1')                        % sebding A and ID_EV to SGS

  2. State = 1  /\ Rcv(Salt1'.{SGSg1'}_(exp(G,H(Salt1'.H(EV.PW1))))) =|>
     State':= 2 /\ K0' := H(P.G)                        % k= hash(p,g)
                /\ U1' := H(EVg1.SGSg1')                % u= hash(A,B)
                /\ X1' := H(Salt1'.PW1)                 % x= hash(salt, pw)
                /\ Ver1' := exp(G,X1')                  % ver = g^x
                /\ S1' := xor(SGSg1',H(K0'.Ver1'))      % g^b + k.g^x - k.g^x = g^b
                /\ K1' := H((exp(S1',EVr1)).(exp(exp(S1',U1'),X1')))    % K = hash(g^ab , g^bux)
                /\ M1' := H((xor(H(P),H(G))).(H(xor(EV,PW1))).Salt1'.EVg1.SGSg1'.K1')
                                                        % M = hash(M1,M2,salt,A,B,K)
                /\ PW2' := H(EVg1.SGSg1'.K1')
                /\ Snd(M1')
                /\ witness(EV,SGS,k11,K1')
                /\ secret(K1',sec_init_K1,{EV,SGS})

  3. State = 2  /\ Rcv(H(PW2)) =|>
    State':= 3
                /\ request(EV,SGS,k12,K1)
                /\ EVr2' := new()
                /\ EVg2' := exp(G,EVr2')
                /\ Snd(EV.EVg2')

  4. State = 3  /\ Rcv(Salt2'.{SGSg2'}_(exp(G,H(Salt2'.H(EV.PW2))))) =|>
     State':= 4 /\ U2' := H(EVg2.SGSg2')                % u= hash(A,B)
                /\ X2' := H(Salt2'.PW2)                 % x= hash(salt, pw)
                /\ Ver2' := exp(G,X2')                  % ver = g^x
                /\ S2' := xor(SGSg2',H(K0.Ver2'))       % g^b + k.g^x - k.g^x = g^b
                /\ K2' := H((exp(S2',EVr2)).(exp(exp(S2',U2'),X2')))    % K = hash(g^ab , g^bux)
                /\ M2' := H((xor(H(P),H(G))).(H(xor(EV,PW2))).Salt2'.EVg2.SGSg2'.K2')
                                                        % M = hash(M1,M2,salt,A,B,K)
                /\ PW3' := H(EVg2.SGSg2'.K2')
                /\ Snd(M2')
                /\ witness(EV,SGS,k21,K2')
                /\ secret(K2',sec_init_K2,{EV,SGS})

  5. State = 4  /\ Rcv(H(PW3)) =|>
    State':= 5
              /\ request(EV,SGS,k22,K2)
end role
```

Figure A.17: Chapter 6: HLPSL Codes of Electric Vehicle