



Forward Secrecy Enhancement for Named Data Networking Access Control over
Mobile Ad-Hoc Networks

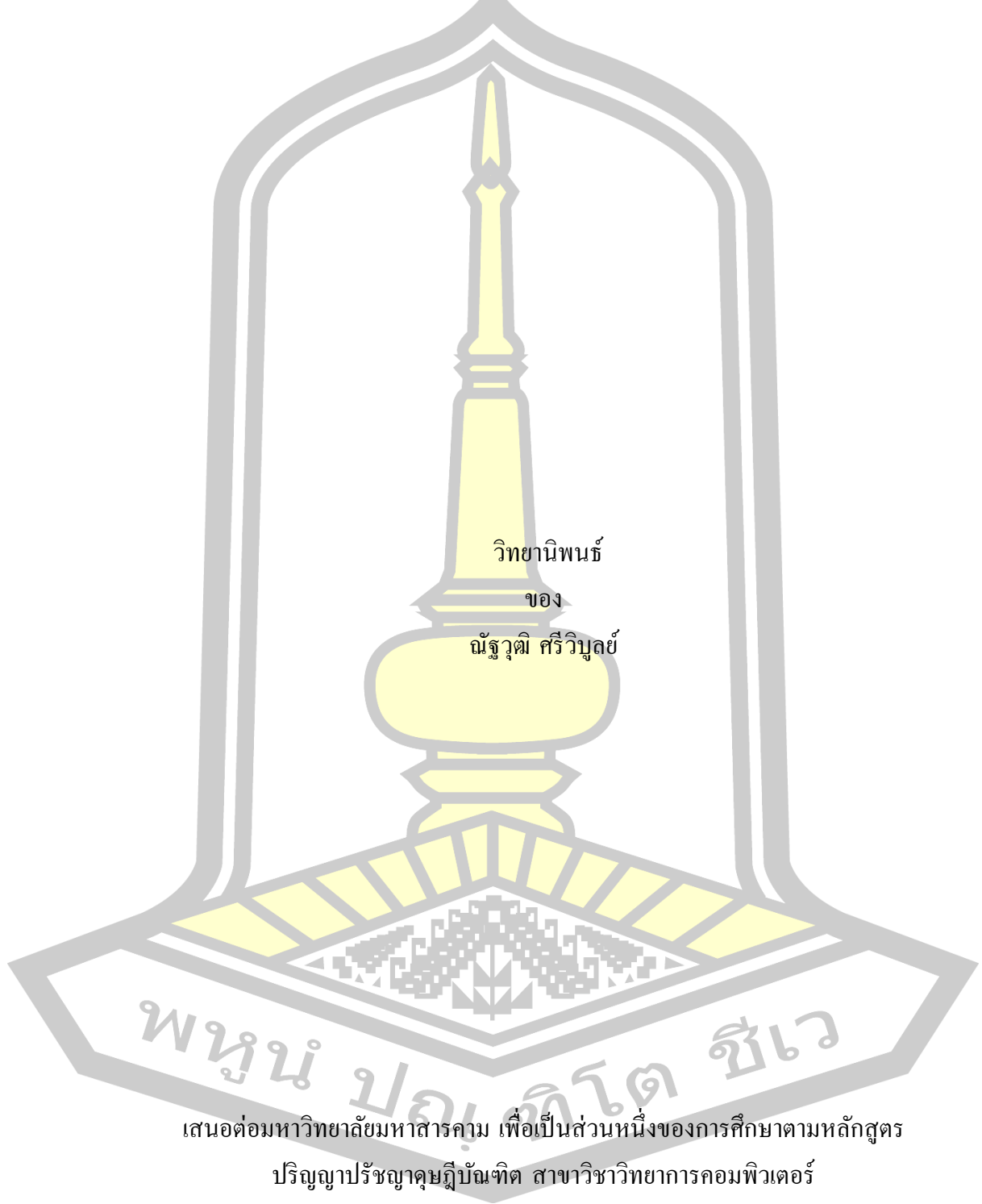
Nattavut Sriwiboon

A Thesis Submitted in Partial Fulfillment of Requirements for
degree of Doctor of Philosophy in Computer Science

June 2020

Copyright of Mahasarakham University

การเสริมสร้างความลับส่งต่อสำหรับการควบคุมการเข้าถึงของเครือข่ายเนตดาต้าบนโครงข่าย
เคลื่อนที่เฉพาะกิจ



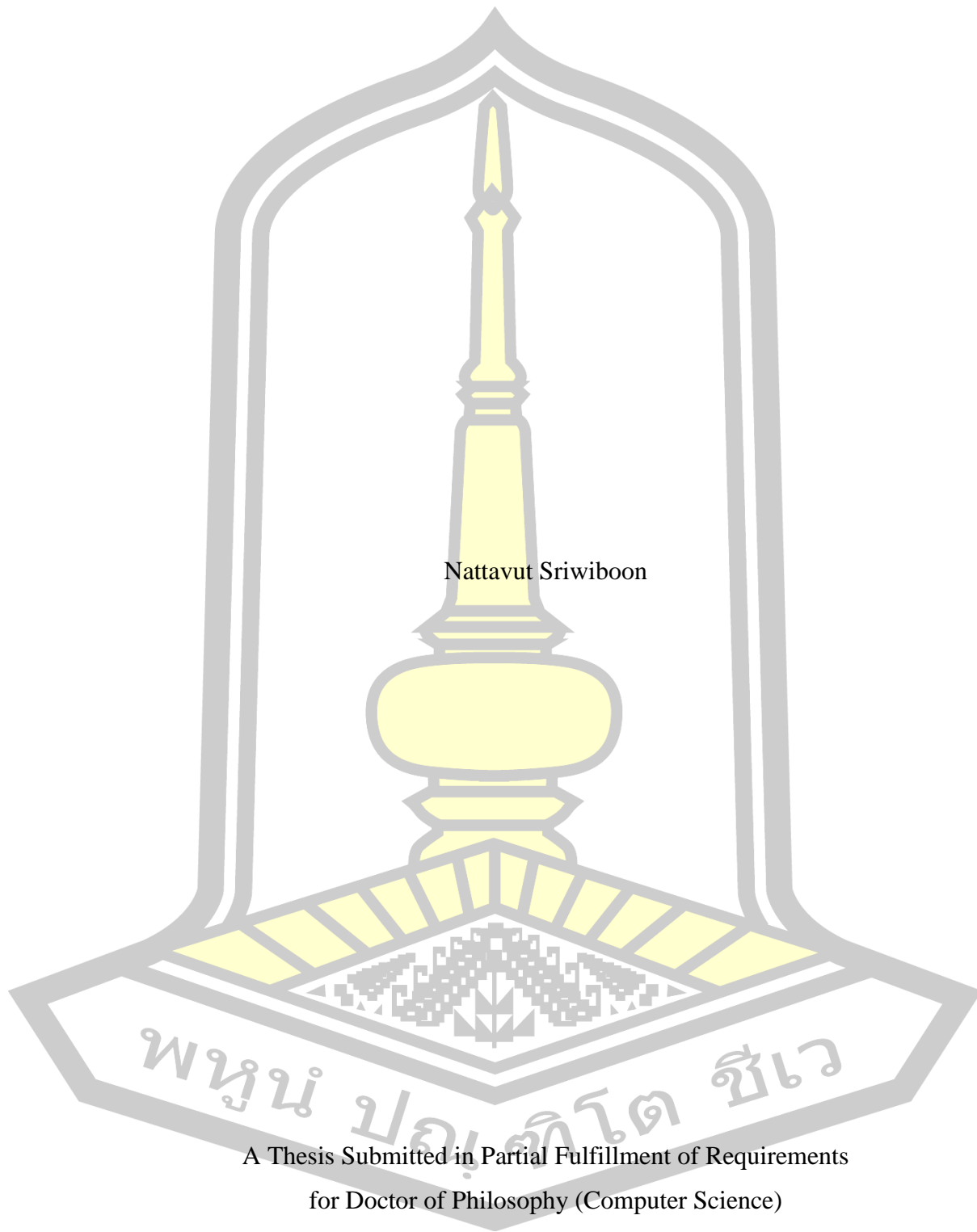
พหุบัน ปญญะโกโต ชีเว

เสนอต่อมหาวิทยาลัยมหาสารคาม เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาปรัชญาดุษฎีบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

มิถุนายน 2563

ลิขสิทธิ์เป็นของมหาวิทยาลัยมหาสารคาม

Forward Secrecy Enhancement for Named Data Networking Access Control over
Mobile Ad-Hoc Networks



Nattavut Sriwiboon

A Thesis Submitted in Partial Fulfillment of Requirements
for Doctor of Philosophy (Computer Science)

June 2020

Copyright of Maharakham University



The examining committee has unanimously approved this Thesis, submitted by Mr. Nattavut Sriwiboon , as a partial fulfillment of the requirements for the Doctor of Philosophy Computer Science at Maharakham University

Examining Committee

Chairman

(Asst. Prof. Kornchawal Chaipah ,
Ph.D.)

Advisor

(Asst. Prof. Somnuk Puangpronpitag
, Ph.D.)

Committee

(Asst. Prof. Suchart Khummanee ,
Ph.D.)

Committee

(Asst. Prof. Phatthanaphong
Chompoowises , Ph.D.)

Maharakham University has granted approval to accept this Thesis as a partial fulfillment of the requirements for the Doctor of Philosophy Computer Science

(Asst. Prof. Sasitorn Kaewman)
Dean of The Faculty of Informatics

(Assoc. Prof. Krit Chaimoon , Ph.D.)
Dean of Graduate School

พหุบัณฑิต ชีวะ

TITLE Forward Secrecy Enhancement for Named Data Networking
Access Control over Mobile Ad-Hoc Networks

AUTHOR Nattavut Sriwiboon

ADVISORS Assistant Professor Somnuk Puangpronpitag , Ph.D.

DEGREE Doctor of Philosophy **MAJOR** Computer Science

UNIVERSITY Mahasarakham **YEAR** 2020
University

ABSTRACT

Named Data Networking (NDN) is a new paradigm for the future Internet, aiming for efficient content delivery using in-network cache and information-centric communication. Security is built into NDN by embedding a public key signature in each data packet to enable verification of authenticity and integrity of contents. For NDN access control, the encryption-based model is the main proposed scheme where sensitive data are encrypted by legitimate producers and then decrypted by only authorized consumers. There have been several proposals using this encryption-based model. However, most of them still suffer from Perfect Forward Secrecy (PFS) issues. Without PFS, the previously published contents for an authorized consumer could be compromised if the private key of the consumer is leaked. This could be a serious security threat in several scenarios, particularly the battlefield scenario. Hence, this dissertation proposes EKAC: Ephemeral Key-based Access Control scheme for NDN to address the problem. EKAC considers Mobile Ad-Hoc Network scenarios, like battlefield network, where the connection can be intermittently lost, and provides a new mechanism to provide PFS, immediate revocation. We have evaluated the performance of EKAC by comparing to previous encryption-based access control schemes. From the evaluation results, EKAC is more suitable for security to prevent attackers to access the previously published contents, in case that the attackers compromise consumer devices to gain the consumer's private key. So, EKAC can support ephemeral key shares. Furthermore, our policy key distribution is more suitable for intermittent connection. For the efficiency of EKAC, we have also found that EKAC computation and communication cost is less than the previous NDN access control scheme.

Keyword : Named Data Networking, Ephemeral Key-based Access Control, Ephemeral Key, Mobile Ad-Hoc Network, Intermittent Connection

ACKNOWLEDGEMENTS

This dissertation would not have been accomplished if without the help from several people. First of all, I would like to thank Dr. Somnuk Puangpronpitag for providing invaluable support throughout my Ph.D. program. I also thank to my colleagues in the Information Security and Advanced Network (ISAN) and members of Distributed System & Services (DSS) research group (University of Leeds, UK) for a sharp discussion.

I was very fortunate to have many friends both within and outside the Faculty of Informatics during my doctoral life. I thank them all for their being very supportive. In addition, this dissertation is partly supported by the Newton Mobility Grant (No: NI160138) from the UK's Official Development Assistance together with Office of Higher Education Commission (OHEC) Thailand, University of Leeds (UK), and Mahasarakham University (Thailand). I am also grateful to Prof. Karim Djemame for all supports, during a few months of collaboration in Leeds (UK).

Nattavut Sriwiboon

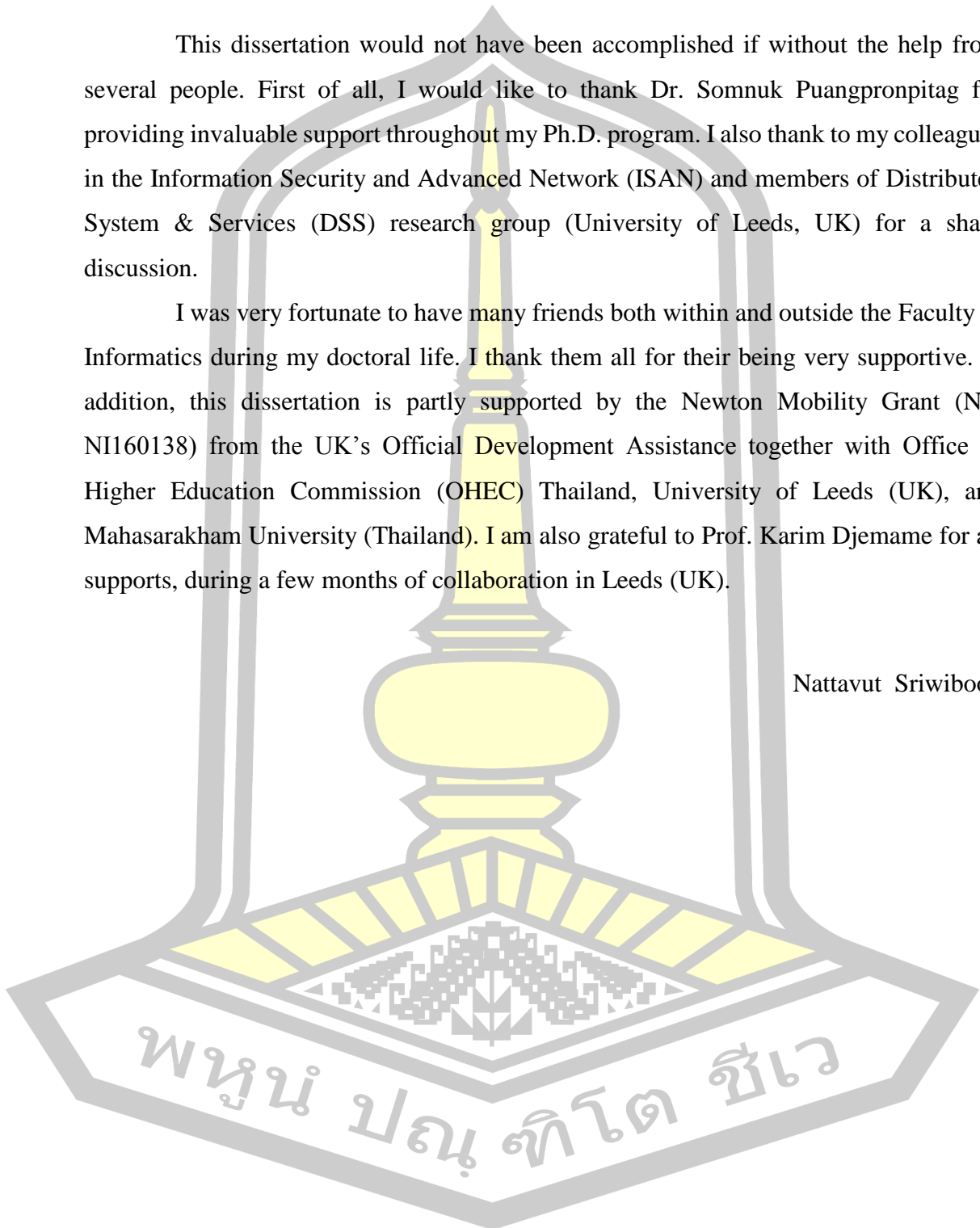


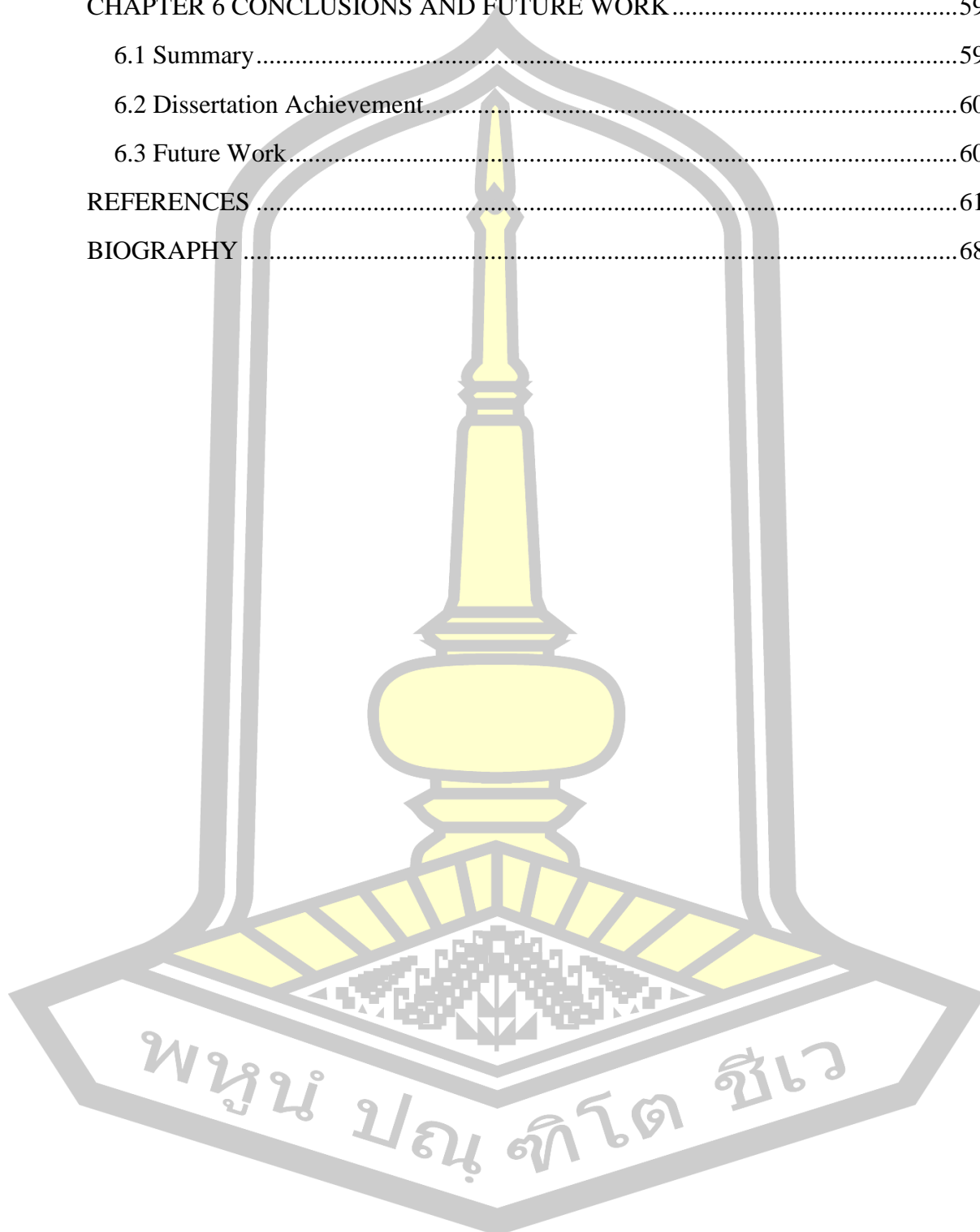
TABLE OF CONTENTS

	Page
ABSTRACT.....	D
ACKNOWLEDGEMENTS.....	E
TABLE OF CONTENTS.....	F
LIST OF TABLES.....	J
LIST OF FIGURES.....	K
CHAPTER 1 INTRODUCTION.....	1
1.1 Research Motivation.....	1
1.2 Objectives.....	1
1.3 Scope.....	2
1.4 Dissertation Outline.....	2
1.5 Abbreviations.....	2
CHAPTER 2 BACKGROUND AND RELATED WORK.....	4
2.1 Named Data Networking (NDN).....	4
2.1.1 Brief Introduction to NDN.....	4
2.1.2 Types of Packets.....	5
2.1.3 Router Architecture.....	6
2.1.4 Names.....	7
2.2 Security Design in NDN.....	8
2.3 The Trust Model in NDN.....	8
2.3.1 Automated Certificate Issuance.....	8
2.3.2 Hierarchical Naming System.....	9
2.3.3 Obtaining Trust Anchors.....	9
2.3.4 Using Trust Model to Define Trust Policies.....	9
2.3.5 NDN Certificate Management Process.....	11
2.4 Signed Interest.....	13

2.5 The Benefits of NDN.....	13
2.6 NDN Research Challenges	15
2.7 Access Control.....	15
2.7.1 General Concepts	15
2.7.2 Granularity in Access Control.....	16
2.8 Why NDN needs Access Control	16
2.9 NDN Access Control Model.....	17
2.9.1 PKI-based Access Control.....	17
2.9.2 Attribute-based Access Control.....	18
2.9.3 Interest-based Access Control.....	18
2.9.4 Proxy Re-encryption-based Access Control.....	18
2.10 Access Control Mechanism over the Classical Internet and NDN.....	18
2.11 Device Compromise	19
2.12 Perfect Forward Secrecy.....	20
2.13 Related Works	20
2.14 Summary of Previous Studies on Access Control Schemes for NDN.....	24
2.15 Access Control Model of NAC and NAC-ABE.....	24
2.15.1 NAC and NAC-ABE Assumptions	24
2.15.2 NAC and NAC-ABE Overview	25
2.15.3 NAC and NAC-ABE Schemes.....	26
2.15.4 The Advantages of NAC and NAC-ABE.....	29
2.15.5 The Disadvantages of NAC and NAC-ABE	31
CHAPTER 3 RESEARCH METHODOLOGY	34
3.1 Introduction.....	34
3.2 Analyzing the Problems of NDN Access Control and Previous Work	34
3.3 Proposing a New NDN Access Control Scheme.....	36
3.4 Performance Evaluation Techniques	36
3.4.1 Implementation Testbed.....	36
3.4.2 Network Emulation	37

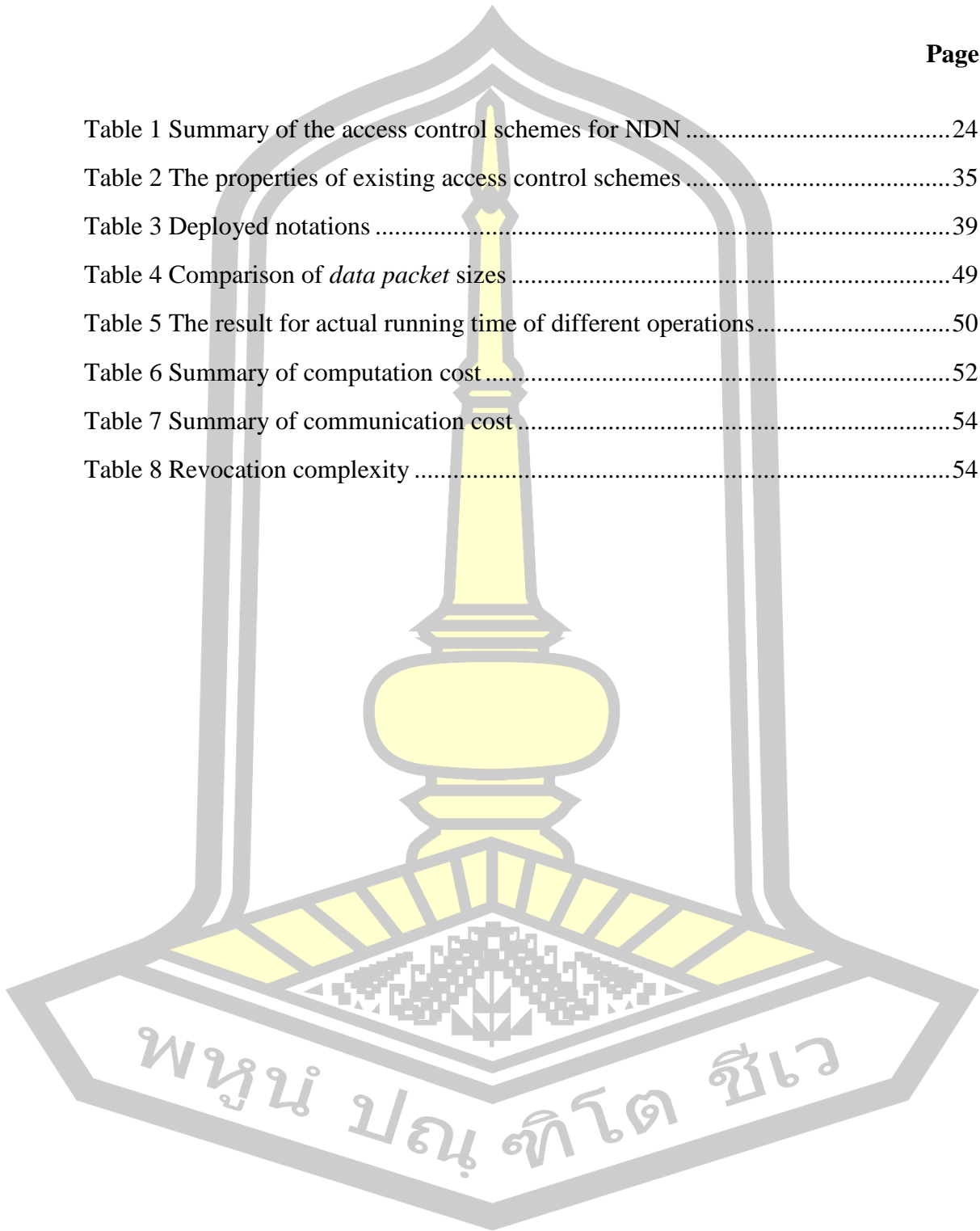
3.4.3 PFS and Efficiency Analysis	37
3.4.4 Result Analysis and Confidence Interval	37
CHAPTER 4 OUR ACCESS CONTROL SCHEME.....	39
4.1 The Design of EKAC.....	39
4.2 EKAC Assumptions.....	39
4.3 EKAC Design Overview	40
4.4 EKAC Naming Conventions	41
4.5 EKAC Scheme.....	42
4.6 EKAC Access Revocation.....	45
4.7 EKAC Properties	46
CHAPTER 5 PERFORMANCE EVALUATION.....	48
5.1 Introduction.....	48
5.2 Experimental Setup.....	48
5.2.1 Experimental Parameters.....	48
5.2.2 Size of Data Packet Transmission.....	49
5.2.3 Access Revocation Cost	49
5.2.4 Cryptographic Operations	50
5.3 Evaluation	51
5.3.1 Computation Cost.....	51
5.3.2 Communication Cost.....	52
5.3.3 Revocation Complexity.....	54
5.4 Security Assessment.....	54
5.5 Discussion.....	55
5.5.1 Efficiency for Threat Mitigation from Consumer Compromise	55
5.5.2 Performance and Resource Consumption	55
5.5.3 Suitability for Intermittent Connectivity	56
5.5.4 Effective Consumer Revocation.....	56
5.5.5 EKAC with Better Scalability	57
5.5.6 Effective Access Control and Robust Privacy	57

5.5.7 Efficient of NDN Architecture Usage	57
CHAPTER 6 CONCLUSIONS AND FUTURE WORK	59
6.1 Summary	59
6.2 Dissertation Achievement.....	60
6.3 Future Work.....	60
REFERENCES	61
BIOGRAPHY	68



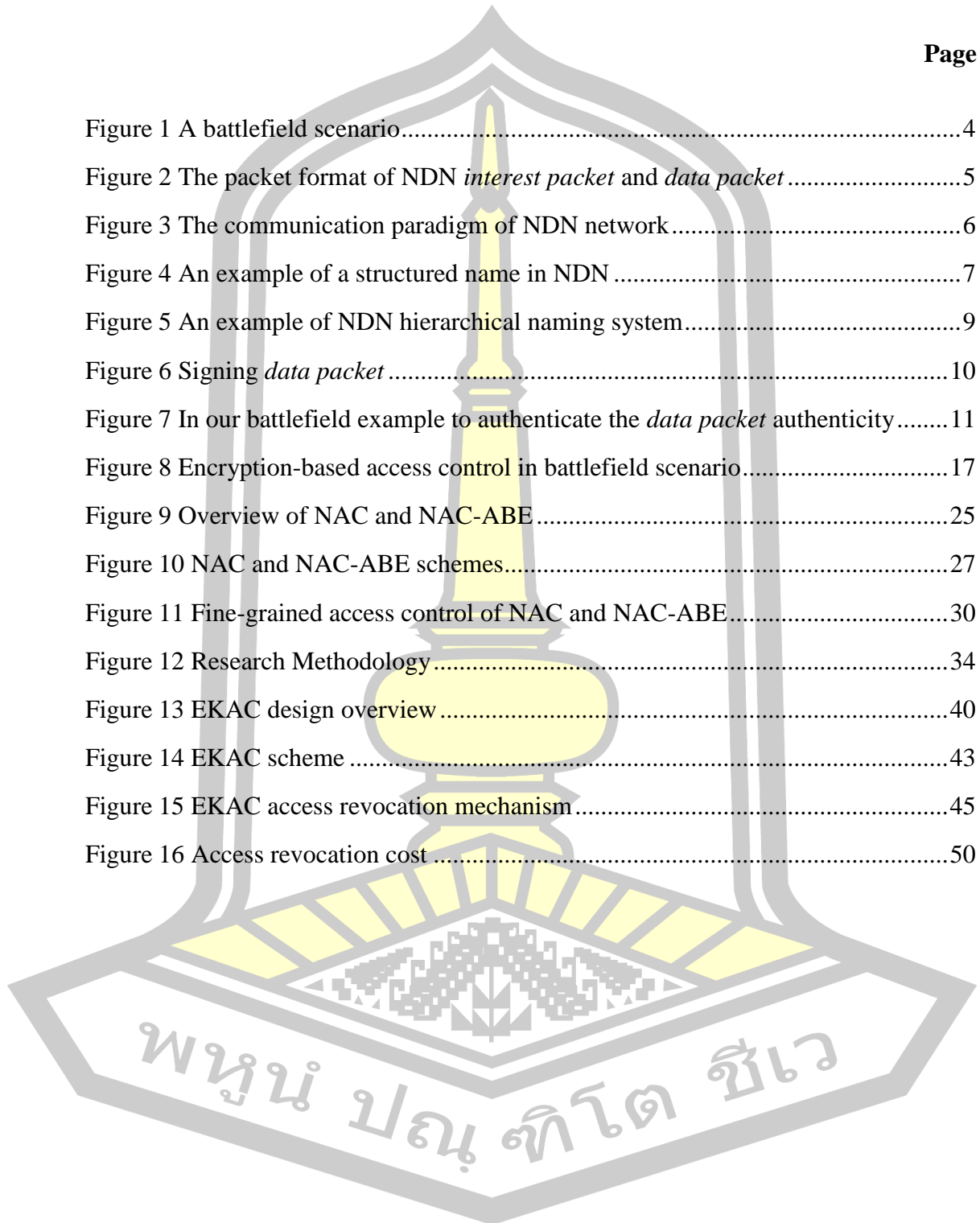
LIST OF TABLES

	Page
Table 1 Summary of the access control schemes for NDN	24
Table 2 The properties of existing access control schemes	35
Table 3 Deployed notations	39
Table 4 Comparison of <i>data packet</i> sizes	49
Table 5 The result for actual running time of different operations	50
Table 6 Summary of computation cost	52
Table 7 Summary of communication cost	54
Table 8 Revocation complexity	54



LIST OF FIGURES

	Page
Figure 1 A battlefield scenario.....	4
Figure 2 The packet format of NDN <i>interest packet</i> and <i>data packet</i>	5
Figure 3 The communication paradigm of NDN network.....	6
Figure 4 An example of a structured name in NDN.....	7
Figure 5 An example of NDN hierarchical naming system.....	9
Figure 6 Signing <i>data packet</i>	10
Figure 7 In our battlefield example to authenticate the <i>data packet</i> authenticity.....	11
Figure 8 Encryption-based access control in battlefield scenario.....	17
Figure 9 Overview of NAC and NAC-ABE.....	25
Figure 10 NAC and NAC-ABE schemes.....	27
Figure 11 Fine-grained access control of NAC and NAC-ABE.....	30
Figure 12 Research Methodology.....	34
Figure 13 EKAC design overview.....	40
Figure 14 EKAC scheme.....	43
Figure 15 EKAC access revocation mechanism.....	45
Figure 16 Access revocation cost.....	50



CHAPTER 1

INTRODUCTION

1.1 Research Motivation

The Named Data Networking (NDN) [1, 2] has considered shortcomings of the mobility, content distribution and security in the classical Internet architecture (Transmission Control Protocol/Internet Protocol: TCP/IP), and proposed a new Internet architecture, shifting away from the host-centric model to the data-centric model. The data-centric model desires a security model that secures data chunks directly rather than securing hosts and channels. The producer operates named and secured *data packets* by a cryptographic signature to bind the content and producer's name at the time of production. In order to access a specific content, the consumers request the content by sending the *interest packets* carrying the content name. The corresponding *data packets* are then sent back following the reverse path of the *interest packets*. The received *data packet* is authenticated by names and digital signatures to ensure the content integrity and producer authentication. The consumer can validate the original data producer's signature rather than authenticating the host. By this way, producers can publish and store their contents in NDN content store (in-network caches). This paradigm enables contents being efficiently delivered to the consumers and increases mobility, delay-tolerant networking.

Although the design of NDN is efficient in retrieving contents and securing contents directly, access control is one of its major challenges. Since NDN contents could be available in-network caches and be accessible by all the entities, the content access control is critical. Several previous studies ([3-14]) have been proposed the access control scheme for NDN, but there are still several drawbacks, particularly forward secrecy and access revocation issues.

Hence, we propose the Ephemeral Key-based Access Control (EKAC) scheme for NDN. EKAC provides a scheme to prevent the attackers to access the previously published content, when the consumer devices are compromised. The EKAC scheme enforces the access control by using ephemeral key shares. So, the access keys will be thrown away after the communication session is done. EKAC enables the efficient revocation mechanisms by immediate privilege revocation. In addition, the EKAC provides levels of scalability in the access control policies for intermittent network scenarios, such as a battlefield scenario. Furthermore, a prototype of our scheme have been implemented in an NDN platform. The performance evaluation has been done, and we have compared our scheme to the existing schemes.

1.2 Objectives

1. Analyzing the problems of NDN access control and previous work, particularly in intermittent network
2. Proposing a new NDN access control scheme
3. Evaluating the proposed NDN access control scheme and comparing it with the previous

1.3 Scope

NDN names data and secures data chunks directly as contrary to host-centric model in the classical Internet architecture. This powerful paradigm shift has been facilitate mobile ad hoc communications, and intermittently connected communication channels, which is difficult to achieve by the classical Internet. Several of the research in the NDN community have focused on the benefit of data-centric model to apply in the application. This includes communication where information tends to be confidential, e.g. mobile health [15], and others communication are ad-hoc and intermittent, e.g. battlefield networks [12, 16-19]. In this dissertation, we focus on some of the challenges of battlefield networks, where information is sensitive and needs strong access control.

A review of literatures on access control [3-14, 20-27] suggests that the encryption-based access control model is the most successful access control scheme for NDN so far. Nonetheless, all previous studies still have a drawback in term of forward secrecy when a consumer and his/her private key is compromised.

We propose the access control scheme to address the threat from the consumer compromised over NDN by using the Diffie-Hellman (DH) [28] key exchange algorithm to establish ephemeral key shares between a producer and a consumer. Our access control scheme can be guaranteed the robust security by Perfect Forward Secrecy (PFS) properties. An enhanced revocation scheme has also been enhanced by this dissertation. The focus environment for the proposed access control scheme is intermittent connection of the battlefield Mobile Ad-hoc Network (MANET).

1.4 Dissertation Outline

CHAPTER 2 provides the background of NDN access control. We illustrate the NDN challenges for the security models, and an appropriate summary of related work to this dissertation.

CHAPTER 3 illustrates research methodology used for this dissertation. We describe why NDN needs access control, and describe possibility mechanism to enable access control over NDN. Our contributions are included in this chapter.

CHAPTER 4 proposes our access control scheme, entitled EKAC: Ephemeral Key-based Access Control scheme for NDN. The design goals of the EKAC scheme are to achieve ephemeral key shares. Our access control policy mechanism is suitable for intermittent connection, and enables immediate revocation of the consumer. In addition, an overview of implementing the EKAC prototype is illustrated.

CHAPTER 5 illustrates the experiments and evaluation of our proposed scheme. Comparative discussion between the EKAC and related work is described also in this chapter.

CHAPTER 6 summaries the contribution of this dissertation, and also recommend future work.

1.5 Abbreviations

ABAC	Attribute Based Access Control
AES	Advanced Encryption Standard
CA	Certificate Authority
CCN	Content Centric Networking

CGAC	Coarse Grained Access Control
CK	Content Key
CORE	Common Open Research Emulator
CP-ABE	Ciphertext-Policy Attribute-based Encryption
DNS	Domain Name System
DH	Diffie-Hellman (DH) key exchange
ECDSA	Elliptic Curve Digital Signature Algorithm
EKAC	Ephemeral Key-based Access Control Scheme for NDN
EK	Ephemeral Key
EAK	Ephemeral Access Key
ESK	Ephemeral Share Key
FGAC	Fine-Grained Access Control
SHA	Secure Hash Algorithm
ICN	Information Centric Networks
IP	Internet Protocol
KDK	Key-Decryption Key
KEK	Key-Encryption Key
KP	Key Policy
MANET	Mobile Ad-Hoc Network
MITM	Man-In-The-Middle Attack
NAC	Name-based Access Control
NAC-ABE	Name-based Access Control with CP-ABE
NIC	Network Interface Card
NDN	Named Data Networking
NDN-CXX	Named Data Networking C++ library with eXperimental eXtension
NDNCERT	NDN Certificate Management System
NFD	NDN Forwarding Daemon
PFS	Perfect Forward Secrecy Property
PKI	Public Key Infrastructure
RBAC	Role Based Access Control System
UAV	Unmanned Aerial Vehicle

พหุ มั บณุ ทิโต ชีเว

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Named Data Networking (NDN)

2.1.1 Brief Introduction to NDN

The classical Internet (Transmission Control Protocol/Internet Protocol: TCP/IP) was created in the 1970's [29], which provided a point-to-point communication. The main concept is an end host sending packets to the other host, using IP addresses. Currently, the application communication model however has shifted from delivering content to the end-host to fetching the content from wherever available. Content accessing is more important than host addressing. For example, Content Distribution Network (CDN) [30] has been created on top of the TCP/IP to ease such the content access.

With all the reasons above, Information-Centric Networking (ICN) was introduced in 2010. The initial similar concept was called Content Centric Networking (CCN) [31], previously presented by Van Jacobsen at a Google Talk in 2006. The ICN treats naming content as the main idea in the Internet architecture, instead of naming host addresses. A branch of the ICN is Named Data Networking (NDN) [32] research project, started in 2010, as a promising model for the future Internet. In November 2010, NDN directed by University of California, Los Angeles (UCLA), is one of five research projects funded by a grant from the National Science Foundation (NSF) under its Future Internet Architecture (FIA), and then the Internet Research Task Force (IRTF) has established an ICN research working group in 2012.

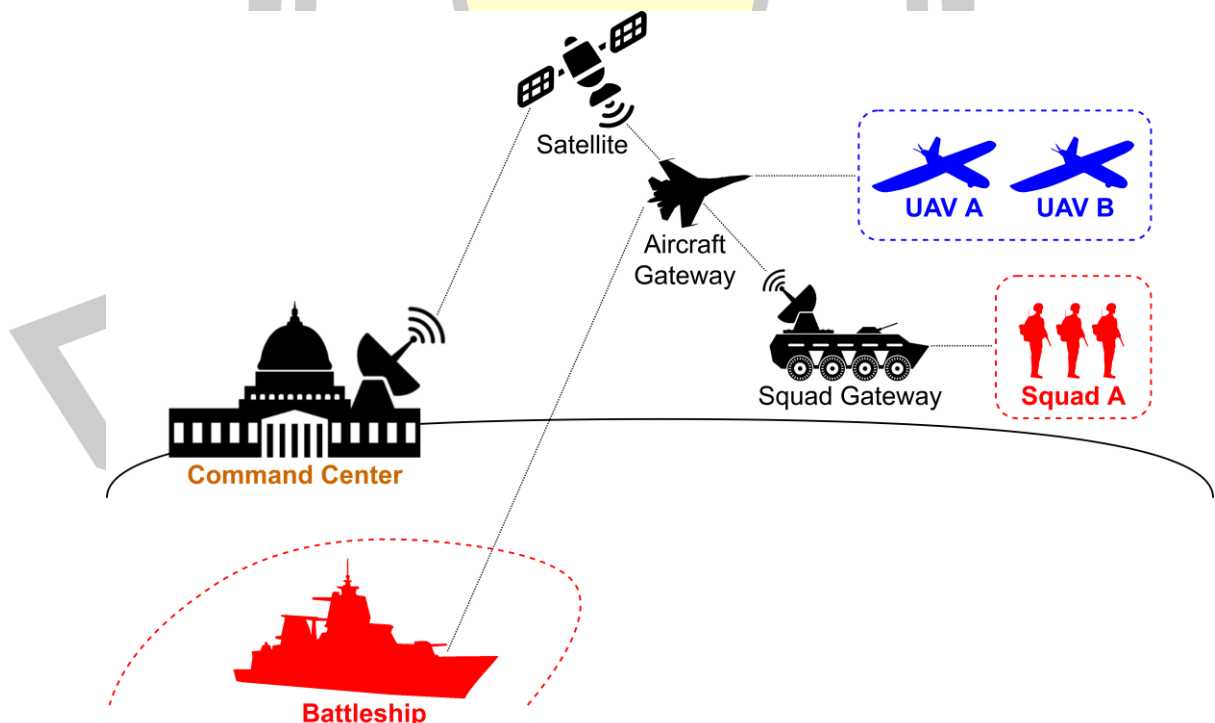


Figure 1 A battlefield scenario

The NDN is a prominent data-centric communication [2, 16, 33-35], which enables the location independence that delivers packets by content names. The consumer can then fetch the content from the third party storages over capricious channels. Therefore, the content-based security model appears to be a more suitable for NDN architectures. To discuss in the rest of this dissertation, we use a battlefield scenario as shown in Figure 1 to facilitate explanation and describe how NDN enables application design patterns that have been successful in a variety of Internet communication. All entities are the participants in the NDN networks, and each entity has a semantically meaningful name, as follows: (1) A command center, acting as an access manager to directly control the access rights and production rights, (2) Unmanned Aerial Vehicles (UAV) A and B acting as producers, and (3) squad A and a battleship acting as consumers. For example, the command center can control over all contents that share between all entities. The UAV A can share the content with only the squad A, while the UAV B can share the content only with a battleship. NDN network nodes, including a satellite, an aircraft gateway, and a squad gateway are NDN forwarders that distribute contents among entities.

2.1.2 Types of Packets

NDN is a new Internet architecture that changes the network communication model from “request a content from a source address” to “fetch content by a given name”. Peers participating in communication either produce (as a server in a classical network) or consume (as a client in a classical network) named data. Communication in NDN architecture relies on two types of packets: *interest packets* and *data packets* as shown in Figure 2.

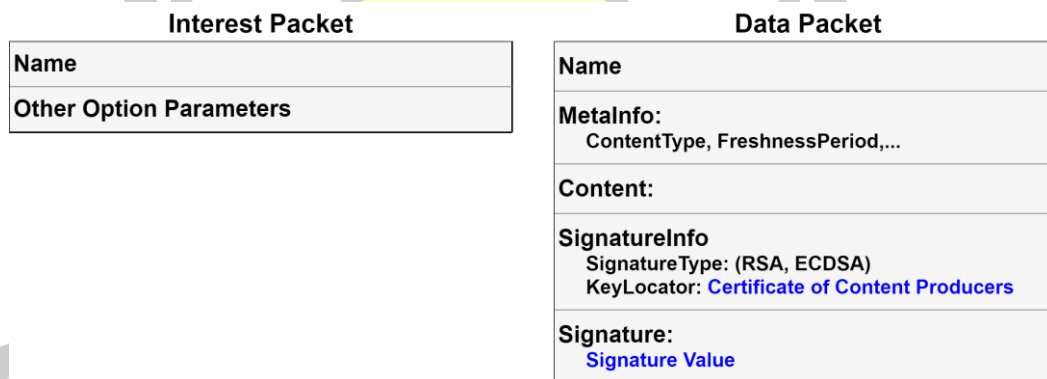


Figure 2 The packet format of NDN *interest packet* and *data packet*

As shown in Figure 2, both types of packets carry a name that identifies a content that can be communicated in one *data packet*. *Interest packets* include the “Name” field and other option parameters (e.g., “Nonce” field) as described in [36]. The “Name” field is a hierarchical name for data content. When a consumer request a content by sending an *interest packet*, which specifies name (or name prefix) of the requested content into the network. A *data packet* are then fetched, when the *interest packet* reaches a producer of the requested content. The *data packet* contains the “Name”, “MetaInfo”, “Content”, “SignatureInfo” and “SignatureValue” fields. The “MetaInfo” can include a “Content Type” field, defining a type of the content. A value of Content Type is “BLOB”, that means this *data packet* is a normal *data packet*. In

addition, the “MetaInfo” can include the “FreshnessPeriod” field, defines a value number of milliseconds for expiration of the *data packet*. That means, how long an NDN network node should cache of this *data packets*. If the value of FreshnessPeriod is zero, a network node should consider no cache. Every *data packet* has a digital signature, which is generated by the producer to assure the producer and the integrity of a content. A “KeyLocator” is a part of the “SignatureInfo” field, specifying a name that points to an NDN certificate [37] to verify the content.

2.1.3 Router Architecture

The plan of NDN is to reshape the Internet into a content distribution architecture. Entities, participating in NDN networks, consist of producers, consumers and forwarders. The producers are the applications that produce contents. These entities are equivalent to servers in the classical network. The consumers are the applications that request contents. These entities are equivalent to clients in the classical network. The forwarders act as NDN routers and the third party storage (for caching the contents). The communication paradigm of NDN network considers the named contents rather than IP addresses, as shown in Figure 3.

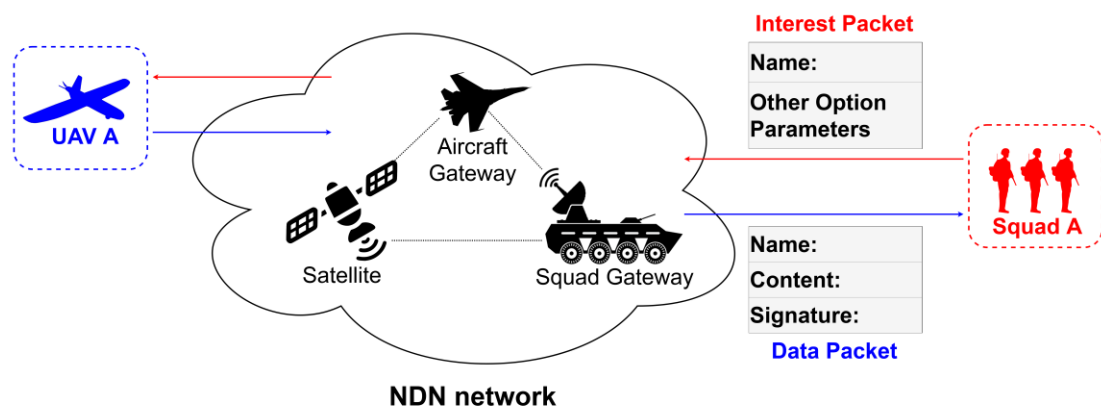


Figure 3 The communication paradigm of NDN network

Named data are the essential elements of NDN network, due to the consumers access each content, called a chunk by identifying a unique name. All consumer applications fetch a content by using the *interest packets*. The NDN routers forward each *interest packet* toward the potential location of the corresponding content. In each NDN router upon receipt of an *interest packet*, it has four components for data transmission.

- 1) If the name of content exists in Content Store (CS), the corresponding *data packet* will be responded from the CS, following the reverse path of *interests*.
- 2) If the name of content not exists in CS, the NDN router forwards the *interest packet* using Pending Interest Table (PIT). If a matching name exists in PIT, it records the *interest packet* and the incoming interface to the PIT. The NDN router is not required to repeat a forwarding process, it certainly records the name of this *interest* in the PIT, and waits for incoming desired contents.

- 3) If no matching PIT, the NDN router will make a new PIT entry and records the outgoing interfaces, and then perform a name of this *interest packet* match lookup in the Forwarding Information Base (FIB) to select routing paths to get the requisite content either in a next NDN router or from the data source.
- 4) When a *data packet* arrives, the router finds the matching PIT. If match is existing PIT, the *data packet* is cached in the CS, and then returned to the consumer by following the incoming interfaces of the *interest packet* as recorded in the PIT, and then the corresponding name in the PIT entry must be removed.

2.1.4 Names

In the classical Internet, each entity has an IP address, that must be managed before them communication in IP network. The end-host sends packets to another host identified by IP address, and need of mapping services between name and IP addresses to alleviate the pain of recognizing an IP address by using some system such as the Domain Name System (DNS).

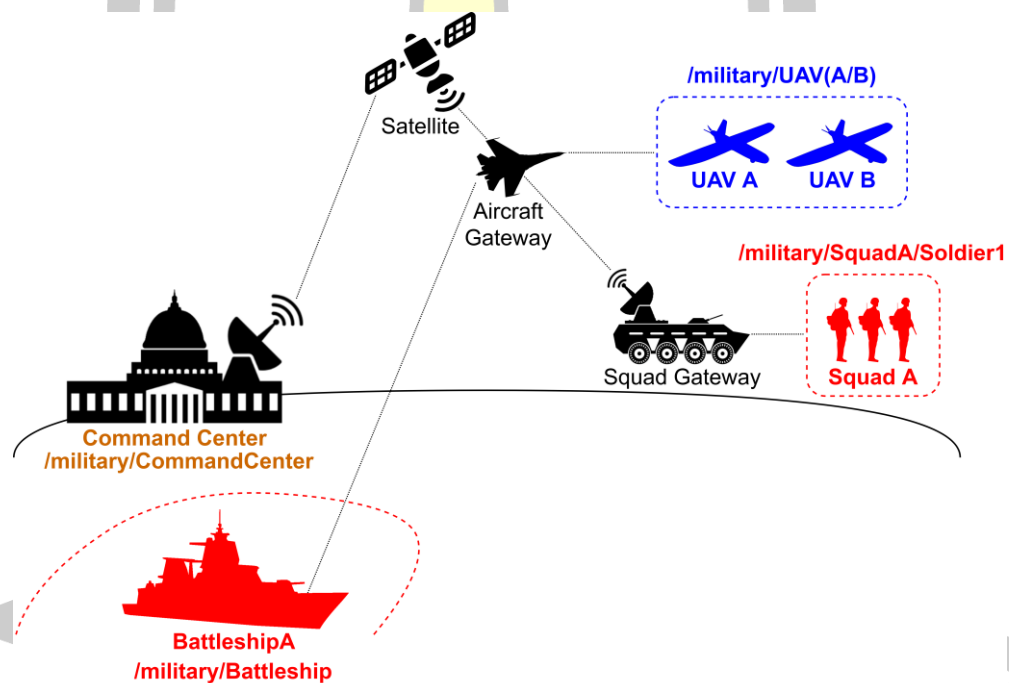


Figure 4 An example of a structured name in NDN

In contrast to the classical Internet, as shown in Figure 4 is the military communications over NDN networks, all entities are the participants in the systems, and each entity has a semantically meaningful name, that in our example. A command center (`/military/CommandCenter`) that directly controls the access rights and production rights. The UAV A and UAV B (`/military/UAV(A/B)`) are the producers, and the soldiers in squad A (`/military/SquadA/Soldier(1/2/3)`) and the battleship (`/military/Battleship`) are the consumers.

All names of the entities are under the name of the system (`/military`) as the anchor of trust for this system. For example, when a consumer (e.g., a soldier) needs

some contents from the producer “UAV A”, it sends an *interest packet* with a name (e.g., “/military/UAVA/info”) to the NDN network. When the NDN network nodes (e.g., Satellite, Aircraft Gateway, and Squad Gateway) receive the *interest packet*, each node uses the name of the content in an *interest packet* to ask the potential location in the NDN network of the corresponding content. A *data packet* of the UAV A with a name corresponding to the name in the *interest packet* may exist in the NDN network or in the UAV A’s devices, it is returned to the soldier by following the reverse route of the *interest packet*. That means, the names in NDN replace the IP address in the classical Internet, and no need for mapping systems (e.g., DNS) anymore.

2.2 Security Design in NDN

The NDN shifts in paradigm for networking from a host-centric model to a data-centric model by providing in-network caching. The NDN enables a producer to create a digital signature on each *data packet* at the time of production. So, the integrity and data source authentication of the *data packet* is independent from where the data are retrieved. To facilitate the integrity and data source authentication, the NDN uses the trust model [38] to develop a public key management system. This is similar to traditional Public Key Infrastructure (PKI) [39]. Two endpoints (i.e., a consumer and a producer) can verify signatures by retracing and verifying the certificates along with the certificate chain to the trust anchor whose public key is pre-configured in the endpoints. Each *data packet* has a “SignatureInfo” field that indicating the name of the producer’s certificate. A certificate is a normal NDN *data packet*, but it carries a public key. As a result, the data receiver can fetch the certificate inside in-network caches, and uses it to validate a received content. In this way, the *data packet* is considered as valid if all certificate chains, along with the trust anchor, have valid signatures.

2.3 The Trust Model in NDN

2.3.1 Automated Certificate Issuance

The certificate issuance is the first step of enabling the trust model in NDN. Certificates are serious to NDN security, where certificate issuers must issue certificates to suitable certificate owners (also producers). A certificate owner must make its certificates highly stable as to certificate users (also consumers) can automates retrieve the certificate to operate data authentication by using the NDN certificate management system (NDNCERT) [40]. For example, as shown in Figure 4, every entity has a semantically meaningful name and all entities have their own public/private key pair. The system (“/military”) is a Certificate Authority (CA), it issues the certificates for all entities by using its private key to sign the certificates to bind between a name and a public key of each entity. The name of the certificate has to be under the issuer’s name prefix. In our example, a name “/military/CommandCenter/KEY/<key-id>” is the certificate name of command center. A name “/military/UAV(A/B)/KEY/<key-id>” is the certificate name of UAV A and UAV B. A name “/military/SquadA/Soldier(1/2/3)/KEY/<key-id>” is the certificate name of the soldiers in squad A, and “/military/Battleship/KEY/<key-id>” is the certificate name of the battleship, where the “prefix” is the name of certificate owner, and the components after “KEY” are the key-id, using an identifier of the certificate. Like normal NDN *data packets*, a certificate carrying the public key information can be cached and fetched like any other contents.

2.3.2 Hierarchical Naming System

Hierarchical naming system in the NDN network, forces naming of *data packets* or certificate in the same hierarchical naming system to obviously express a trust model. For example as shown in Figure 5, the trust anchor of participants in a system is “/military”. The trust anchor signs their members (e.g., command center and UAV A) using its private keys. The name of command center’s certificate is a “/military/CommandCenter/KEY/<key-id>”, and the name of UAV A’s certificate is “/military/UAVA/KEY/<key-id>”. A name prefix of their certificates is under a name “/military” of trust anchor in this system. The UAV A can produce any contents by defining the name of *data packet* by using the hierarchy name, where the name of *data packet* has to be under the name prefix of UAV A’s certificate, e.g. “/military/UAVA/info”, and then sign a *data packet* by its own private key.

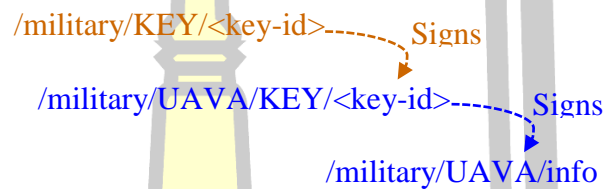


Figure 5 An example of NDN hierarchical naming system

2.3.3 Obtaining Trust Anchors

All entities in the system need the certificate of trust anchors to verify the identity of all the received contents [38]. The trust anchors are either pre-installed or usually received through usually installed via out-of-band mechanisms to all entities in the system. For example, the military system creates a certificate of trust anchor by defining its name as “/military/KEY/<key-id>” during the installation of the system. The certificate of trust anchors can then be obtained in a chain in accordance with the name hierarchy. In our battlefield example, we take a simple approach of pre-installing the command center’s certificate into the device of UAV A and the device of all soldiers.

2.3.4 Using Trust Model to Define Trust Policies

The trust model leverages naming conventions to establish systematic representations of trust policies, consisting of (1) how to names in the *data packet* (2) how to sign and define the components in the *data packet* (3) how to authenticate the *data packet* authenticity.

1. How to names in the *data packet*

To produce the contents, all producers must first obtain their own certificate. The producer must name *data packets* under a prefix to specify its own certificate, and to secure of *data packets* by a cryptographic signature, binding the content and producer’s name at the time of production. For our example, as shown in Figure 5, the certificate name “/military/UAVA/KEY/<key-id>” is only allowed to define a name of *data packet* under the prefix “/military/UAVA”. For example, an information produced by UAV A would have the name “/military/UAVA/info”.

2. How to sign and define the components in the *data packet*

As shown in Figure 6, the producer should put the name of the own certificate used to sign this content in “KeyLocator” field [41] during *data packet* generation.

Data Packet

Name: <code>/military/UAVA/info</code>
MetaInfo: <code>ContentType, FreshnessPeriod,...</code>
Content: <code>Content</code>
SignatureInfo KeyLocator: <code>/military/UAVA/KEY/<key-id></code>
SignatureValue: <code>SignatureSha256WithEcdsa(Name, MetaInfo, Content, SignatureInfo)</code>

Figure 6 Signing *data packet*

Additionally, *data packets* should have at least an SHA-256 hash that must be digitally signed. Also, the signature type value must be defined to indicate the signature algorithm. For example, “SignatureSha256WithEcdsa” indicates ECDSA public key signature that is computed over SHA-256 hash of the “Name”, “MetaInfo”, “Content”, and “SignatureInfo” fields.

3. How to authenticate the *data packet* authenticity

The advantages of the NDN architecture come from naming data hierarchically and guaranty named content with a digital signature. Every *data packet* is particularly named and this name is bound to the content using a digital signature. To authenticate the *data packet* authenticity, the consumer can fetch the certificate inside in-network caches, and uses it to validate a received content. In this way, the *data packet* is respected to be valid if all certificate chains, along with the trust anchor, have valid signatures. In our battlefield example, as shown in Figure 7, the verification steps are as follows:

- 1) Check the “KeyLocator” field of the *data packet*. If the field does not contain a name of certificate, a signing key a *data packet* cannot be verified. However, the only exceptions from this rule is when the content illustration corresponds to a certificate of trust anchor. That means a certificate of trust anchor is pre-authenticated.
- 2) If the “KeyLocator” field of the *data packet* contains a name of the certificate, the consumer can then retrieve the certificate according to the “KeyLocator” field and recursively validate the retrieved certificate according to the chain until reaching a trust anchor.



Squad A
Data Packet

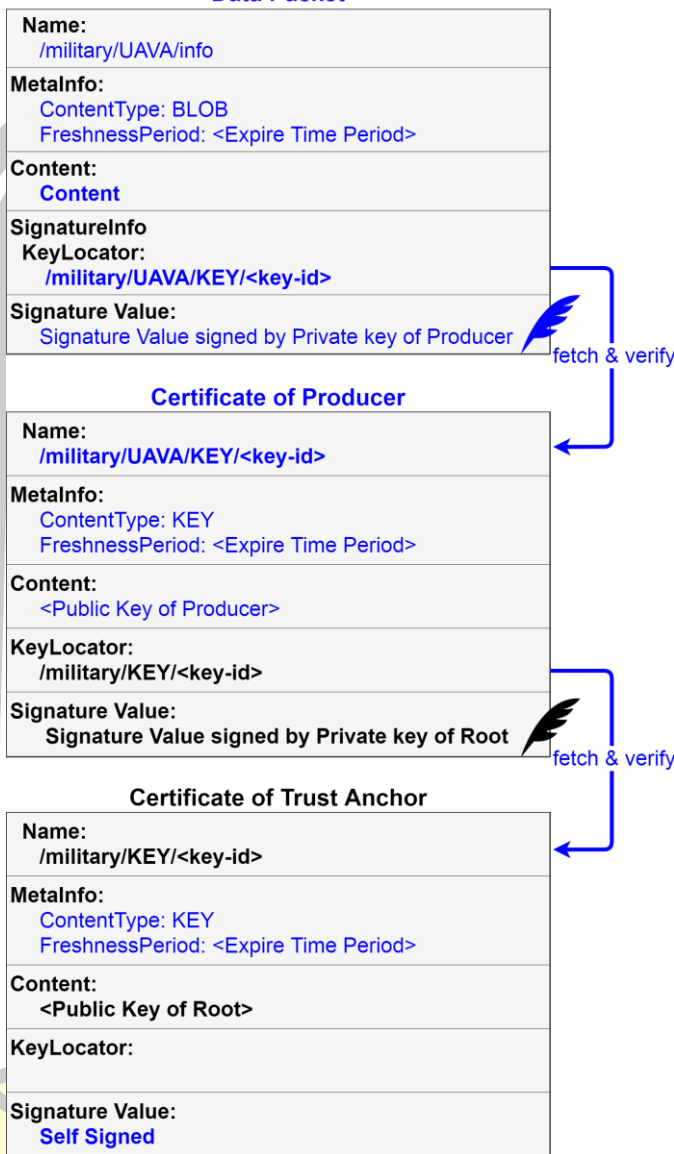


Figure 7 In our battlefield example to authenticate the *data packet* authenticity

2.3.5 NDN Certificate Management Process

To enable the communication over NDN network, the end nodes (e.g., soldier devices) need to install the NDN Forwarding Daemon (NFD), as explained in [42]. NDN architecture enables the naming data instead of data locations [34], and securing content directly by letting applications cryptographically sign each *data packet* to bind its name with the content. Therefore, all entities in the system need to install the NDN CERT, as explained in [43], to manage the certificate usage in the securing networking system. To manage the certificate usage, the end nodes can use NDN testbed, as explained in [44] to achieve the data-centric authenticity, or enable the trust relationship between local nodes by using the local trust management [45]. In this

dissertation, we focused on the local trust management to enable the securing system over NDN network.

For our example, all entities can install the local trust anchors (e.g., /military). The trust relationship between local data and key names can be enforced within the scope of the local network without the intervention of cloud services. The UAV A and the soldier can communicate securely by establishing a trust relationship between the UAV A and the soldier, through the identity certificates in the local network under the identity name prefix “/military”. We use the NDN CERT to issue the certificate to each node. We can illustrate a certificate issuance in local network, as following steps:

1. We define a machine n1 as a trust anchor, that can generate own certificate as following.

```
n1# ndnsec key-gen -i /military -t e
Bv0BKAcqCAhtaWxpdGFyeQgDS0VZCAi3crHoE7qtGwgEc2VsZggJ/QAA
AXBmDn83FAkYAQIZBAA27oAVWzBZMBMGBYqGSM49AgEGCCqGSM49AwEH
A0IABI fGwNg1EfZ2GHahUgafnXfG8B0xI2r+crZfJsji2SdJOBDDi/6D
p1ZnAp4HTJ53+BkrYSSumGmK8ptm5/1HkQYWSHsBAxwbBxkICG1pbG10
YXJ5CANLRVkiCLdysegTuq0b/QD9Jv0A/g8xOTcwMDEwMVQwMDAwMDD9
AP8PMja0MDAymTZUMDQ0MzExF0YwRAIgbOUseb9a5XcQ2QHcDiDVTeZy
FR1GGR8zzJoaGBY6irgCIA5KLNDHMRmOpgaA/8oM/jo43Qnkd9g6xYrt
88nXpvsI
```

2. A machine n2 is an UAV A. It needs to issue a certificate to its own machine. An UAV A generates a certificate request as file name “UAVA.ndncertreq”, where determines its own name of the certificate under the name prefix of a trust anchor.

```
n2# ndnsec key-gen -tr /military/UAVA > UAVA.ndncertreq
```

3. An UAV A sends own certificate request to n1 machine. After obtaining a certificate request as file name “UAVA.ndncertreq”, a trust anchor issues the certificate to UAV A by signing a certificate request of UAV A with its own private key.

```
n1$ ndnsec cert-gen -s /military - < UAVA.ndncertreq >
UAVA.ndncert
```

4. After obtaining a certificate as file name “UAVA.ndncert”, where an UAV A can install and preview the new certificate in its own machine.

```
n2$ ndnsec cert-install UAVA.ndncert
OK: certificate with name
[/military/UAVA/KEY/%B5%DA%BD%84%0A%87%C6%BE/NA/%FD%00%0
0%01pf%13%A9%BA] has been successfully installed
```

```
n2$ ndnsec list -vvv
/military/UAVA
+--> /military/UAVA/KEY/%B5%DA%BD%84%0A%87%C6%BE
+-->
/military/UAVA/KEY/%B5%DA%BD%84%0A%87%C6%BE/NA
/%FD%00%00%01pf%13%A9%BA
```



```

Certificate name:
/military/UAVA/KEY/%B5%DA%BD%84%0A%87%C6%BE/NA/%FD%00%00
%01pf%13%A9%BA
Validity:
  NotBefore: 20200221T044850
  NotAfter: 20210220T044849
Public key bits:
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA5083OWE
TMod5JkO/ieLmvy0HSJpPyQNu5O34VKuFBzP0HEbS6sv8nW6cSp
+7d02OYHNgeu33CavjKuEaNqbebamZfQm/9XX1CaUkEn4iXtMGe
gdPIK5w2YHKL6TqIWE/zTLn/oak9FyBuglSQ1i2iiUi2Ja2kB2j
Nh3/6Xg1lU0ENrhaAXvwTu3rH6OQDDV1z3XRBJSz604AWXuw5U1
Fgzkzh/chXIObj6Mj8iGcU+fwcewnOa7jFrk8zLrBPUb1FN8Dha
W9uXyrqRWAM9lCy9MkkQsYcVtyQHvo5v6dRVuA2uM1eplgmqE6/
cPUmksnpwqU/oChH8wVCWZ7vBdCXwIDAQAB
Signature Information:
  Signature Type: SignatureSha256WithEcdsa
  Key Locator:
Name=/military/KEY/%B7r%B1%E8%13%BA%AD%1B

```

5. To enable the secure trust model, a trust anchor can export its own certificate as file name “root.ndncert” and install it to UAV A’s machine.

```

n1$ ndnsec cert-dump
/military/KEY/%0F%AF%D8%EBz%AA_%B6/self
/%FD%00%00%01pf%17%1C%FF > root.ndncert

n2$ ndnsec cert-install root.ndncert
OK: certificate with name
[/military/KEY/%0F%AF%D8%EBz%AA_%B6/self/%FD%00%00%01pf%
17%1C%FF] has been successfully installed

```

2.4 Signed Interest

The *interest packets* can also establish a signature, called the *signed interest* [46]. *Signed interest* is an optional mechanism to establish an authenticated the sender, supporting the receivers can authenticate the command. *Signed interest* also contains a timestamp and a nonce to prevent replay attacks (with novel timestamp and nonce). For example, when a UAV A receives a *signed interest*, can be able to validate the *signed interest* by using the trust model to similar to the *data packet* validation. If the verification succeeds, the UAV A’s will operate the commands that define in *signed interests*.

2.5 The Benefits of NDN

The battlefield scenario as shown in Figure 1, is used to describe the benefits of NDN.

By naming content and securing content directly, the NDN shifts in paradigm for networking from a host-centric model to a data-centric model. The remainders of this section describes several design patterns that are applicable in tactical scenarios, difficult to achieve over classical Internet, and inherently enabled by the NDN architecture.

1. Securing Content Directly

The classical Internet requires both sides of the channel online to enable the communication channel, to provide security for point-to-point channels to realize the secure communications. The classical Internet requires the channel always online between both nodes to enable the communication channel. The security of classical Internet is established by using point-to-point channels to realize the secure communications, where the communication approaches the challenge in the dynamic network environment, e.g., battlefield that all nodes are dynamic. Due to the nature of all entities must be moved, that means any of them can fail at any time, where one cannot establish a secure channel between nodes of different environments. By securing content directly [38], the *data packets* are named and signed by the producer at the time of content creation using a cryptographic signature to bind together the name of data owner and content. In this way, the NDN enables the content to store in network storage, such as the non-trusted data repository, which the communication is suitable for intermittent network. The receiver can be used to satisfy future requests for the same data. The *data packet* carries a cryptographic signature generated by its producer, together with the name of the signing key. This allows the data consumer to verify the provenance of received data regardless of its source. In this way, NDN provides better communication availability compared to the current Internet that the communication over a predefined secure channel.

2. Local Trust Management

In the battlefield reality, the network environment is likely to rely on intermittent connections and the change in topologies is so dynamic. By naming data instead of data locations, and securing content directly, this means the network nodes can store a copy of the reliable *data packets* in the local caches to satisfy future requests for the same content. By using trust schema to achieve local trust management [45], all entities can list the local trust anchors and specify the trust relationship between local data and key names. This can be enforced within the scope of the local network without the intervention of cloud services. For a soldier and UAV to communicate securely, the process of establishing a trust relationship between the UAV and the squad A through the identity certificates are all published as *data packets* in the local NDN network under the identity namespace. All entities can check the authentication chain against a pre-defined trust model, which the security bootstrapping process in the trust anchor is typically a root key created by the trust anchor. That means a soldier can be able to validate the authenticity of all *data packets* received from UAV A by the certificates, which are securely stored on an NDN network such as an aircraft gateway or a soldier's device without reliance on cloud connectivity to the trust anchor.

3. Establishing Host-Independent

An IP host usually obtains an IP address from the network, and delivering IP packets from any host to any other hosts over point-to-point communication. However, the classical Internet approaches are either unavailable or work poorly in tactical networks, which the topology is dynamic changing in the nature of mobility node and infrastructure-less environments. To compare the NDN hosts with IP hosts, when an IP host moves to a new network, it must be re-provisioned with new IP addresses to suit itself into the new network location, and to establish the security mechanisms identified by the source and destination address, where both end nodes must be online at the same time. In contrast, a host on an NDN network shares the same content's name. By

naming data and securing content directly on an NDN architecture automatically enables independent from location, storage, and also provides content reliability. The hosts are not identified by the source or destination addresses, since every entity has a specific name, which is associated with a cryptographic public key, and private key to guarantee the data source, by using trust model and a certificate is available in the local network environment. The *interest packets* and *data packets* may come and go through any of the multiple wireless interfaces. All entities can simply request contents by names, which are built into applications, from other encountered devices, with no additional configuration or no more need for IP address allocation or DNS services to translate names used by applications to addresses used by IP for delivery. That means the capability in applying NDN to facilitate mobile ad hoc communications, and intermittently connected communication channels, which is difficult to achieve by the classical Internet.

4. Allowing Storage Everywhere

In the classical Internet, due to the source and destination hosts must be stably connected to deliver data, where all data are destroyed after successful or unsuccessful forwarding them. In contrast, the NDN architecture enables the naming data instead of data locations [34], and securing contents directly by letting applications cryptographically sign each *data packet* to bind its name with the content. Every content can be cached in the NDN node, all entities can retransmit an *interest* to find the credible cached content. Therefore, caching and allowing all entities to share their contents are critical as battlefield applications, where the entities are limited capabilities and intermittent connectivity to forward the contents. For example, allowing storage everywhere enables all entities to store and share their data that means more than one aircrafts, battleships, and UAV A to share the same data without relying on a central server. In this way, the *interest packet* does not send to all the way to the data source. The consumer can retransmit the *interest packet* to find the previously cached data, as long as an entity on the NDN network has cached the content. As a result, the entities are delay tolerant, and also allowing efficient recovery the contents from losses.

2.6 NDN Research Challenges

The NDN enables the data centric security model. All contents directly protected by a cryptographic signature, where inconsiderate of the secure communication channels and any intermediaries. However, open challenges in NDN are the security, privacy and access control. With network-wide content caching and location-independent, that the enforcement of content access control policies become issues. The producers can satisfy corresponding requirements in *interest packets* irrespective of whether the requests are from authorized or unauthorized consumers. That means the content which is available in the network is accessible by all the users and there is no access restriction applied to the contents available.

2.7 Access Control

2.7.1 General Concepts

The access control is one of the general fundamental of computer network security, related to other security mechanisms such as authentication, validation, and authorizations [47]. Once an entity is authenticated, that entity is allowed to access a

content or not depending on whether that granularity role of an access control policy of authorized entities. The granularity is general to any mechanism of constraining a set of access right (e.g., users, roles, entities) to access the resources (e.g., content, databases, and files) [48]. The type of requested access may vary depending on the level or scale of details, and once the obvious future has ended or a role change is required or once an access right of an entity has expired. An entity will no longer receives the access right. Therefore, granularity in access control is an evaluating a decision to grant or reject the access to put in authorization rules for all entities in the system to access a resource.

2.7.2 Granularity in Access Control

Granularity in access control is the level or scale of details to specify the access control. There are two types of access control according to the granularity: Coarse Grained Access Control (CGAC) and Fine-Grained Access Control (FGAC) [20, 49]. They can be explained as follows:

1. Coarse-grained Access Control (CGAC)

The access control is coarse-grained if authorization rule for a resource access is just based on particular check (like associated roles) only. Role Based Access Control (RBAC) is a CGAC. For example, as shown in Figure 1 the following permission sets which defines the access based on role assigned to the entities.

Rule 1: The squad A can access the data, generated by UAV A.

Rule 2: The battleship can access the data, generated by UAV B.

In several situations, CGAC is not enough for access control system. FGAC may be needed.

2. Fine-grained Access Control (FGAC)

The access control is fine-grained if authorization rule for a resource access requires more details, regarding end users, current environment (such as time, date). A typical FGAC decision is an Attribute Based Access Control (ABAC) [50] decision using identity of entities or attributes to make an authorization to perform some action on a resource. That means ABAC is very flexible. For a common example, as shown in Figure 1 the following permission sets, if we want to restrict the access based on other additional conditions as well for the same scenario.

Rule 1: The squad A having the role can access the UAV A all the time

Rule 2: The battleship having with role can access the UAV B for 9am-5pm every day or belonging to region Pacific Ocean

From the above rule sets, the access control is FGAC.

2.8 Why NDN needs Access Control

One of the open challenges in NDN is the access control. With network-wide content caching and location-independent, the enforcement of content access control policies become issues [10, 20, 51-53]. The producers can satisfy corresponding requirements in *interest packets* irrespective of whether the requests are from authorized or unauthorized consumers. That means this mechanism can damage the privacy of contents. The content which is available in the network is accessible by all the users and there is no access restriction applied to the contents. So, there have been several studies, proposing access control schemes for NDN.

2.9 NDN Access Control Model

Access control for NDN is a tough challenge to solve due to network-wide content caching and location-independent. So, that means unauthorized consumers could be able to access any contents from in-network caches. A promising approach is encrypting contents by a producer [1], and the corresponding decrypting key must be securely delivered to the authorized consumers. This access control scheme is called an encryption-based access control [54, 55]. At the moment, it is the most popular access control model over NDN. There are several sub-categories of encryption-based access control as follows.

2.9.1 PKI-based Access Control

The PKI-based access control is one of the most popular access control models over NDN. It is actually a kind of encryption-based access control scheme, using cryptographic techniques to provide the confidential contents to the authorized consumers. This process effectively protects the access of sensitive contents in NDN networks [1, 4]. The PKI-based access control scheme leverages the NDN trust model to impose the access privileges of all entities in the system. All entities can communicate securely after establishing a trust relationship via the PKI-based NDN trust model. The symmetric and asymmetric cryptographic algorithms are generally deployed. The authorized consumers can access the contents by using the corresponding decryption key for decrypting the encrypted content that could be accessed only under the access policy.

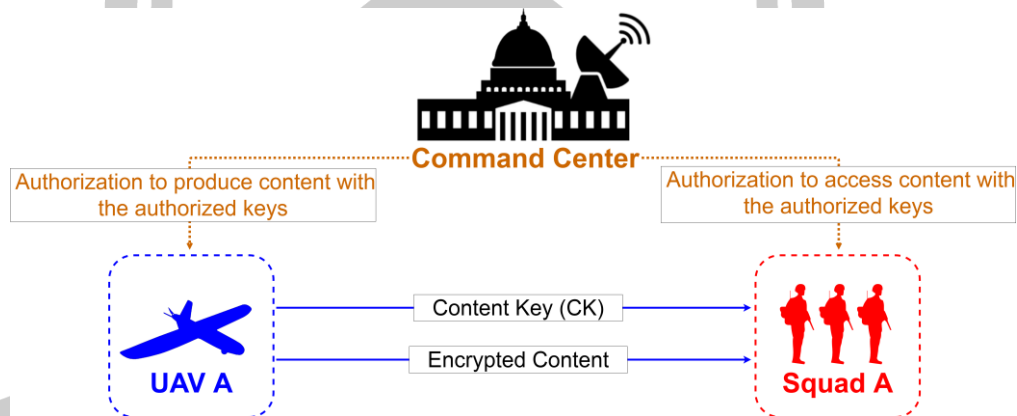


Figure 8 Encryption-based access control in battlefield scenario

An example of PKI-based access control is shown in Figure 8. From the figure, the command center defines authorization details as rules of access control policy; for example, UAV A can only produce the contents for a soldier in squad A. This access control scheme explicitly specifies the privilege via authorized keys (a pair of public/private asymmetric keys generated by the command center). When the UAV A would like to produce a content for the squad A. After generating the content, UAV A will encrypt it using a symmetric cryptography and a Content Key (CK). To control the access rights, an asymmetric encryption and a pair of public/private keys are deployed. The UAV A uses the authorized key (a public key) received from the command center to encrypt the CK. After that, an authorized soldier uses a corresponding authorized key (a corresponding private key), received from the

command center to decrypt the encrypted CK, and uses the CK to access the content. To distribute the authorized key to the soldier, the command center publishes the authorized key by encrypting it using the soldier's authentic certificate. The soldier then uses his own private key to decrypt the authorized key. So, that means only the soldier can access the encrypted content through the access rights via the asymmetric keys.

2.9.2 Attribute-based Access Control

In this model, contents or symmetric keys are encrypted with the attributes of the authorized consumers. This attribute-based access control is actually based on the identity-based encryption [56]. An access manager (which is the command center in the example in Figure 8) grants different access keys to authorized consumers, according to their attributes and access control policies. Only the authorized consumers that have the corresponding access keys and attributes can decrypt the encrypted contents. For our example, this model scheme is in contrast to the PKI-based access control. The UAV A would like to produce an encrypted content for soldiers in squad A. To control the access rights, the UAV A uses the attribute of the soldiers in squad A to encrypt the CK, which the UAV A can control the access right through the asymmetric cryptography. All soldiers in squad A can then use the same access key received from the command center to decrypt the encrypted CK, and uses the CK to access the content.

2.9.3 Interest-based Access Control

This model needs the consumer to be authenticated before producing any contents [22]. For our example, the soldier in squad A (the authorized consumer) must send the *signed interest* to the UAV A (the producer). After obtaining a *signed interest*, the UAV A verifies the soldier's signature by using the soldier's public key. If the verification succeeds, the UAV A will send back encrypted contents to a soldier. Likewise, to control access rights, the soldier's will send a *signed interest* to the command center to request an access key for accessing encrypted contents.

2.9.4 Proxy Re-encryption-based Access Control

For this access control model, a piece of encrypted contents is re-encrypted [57] by the intermediate nodes (proxy servers or NDN routers) for each authorized consumer. For our example, the UAV A has a responsibility to produce an encrypted content for the soldiers in squad A. The UAV A uses its own key to encrypt the content. A soldier in squad A request for the encrypted content. Then, the authentic proxy servers use its own key to re-encrypts the encrypted contents. The soldier can decrypt the encrypted contents using corresponding proxy keys. This model enables the immediate revocation. The producer can indicate the revoked consumer's identification (e.g., ID) to the proxy servers. The proxy server must use the ID of the authorized consumer to generate the new proxy keys. Only unrevoked consumers can decrypt the encrypted content by using the proxy keys.

2.10 Access Control Mechanism over the Classical Internet and NDN

The access control is a system that allows an authority to control access to resources so that a content is only accessible by the authorized consumers. The authorization of a consumer to access a resource is the responsibility of an access

manager (e.g., command center), which uses authentication to verify the identity of consumers, and provides a mechanism to create the authorization rules.

Traditional access control models over the classical Internet are generally centralized where a central entity is responsible for managing the authorization mechanisms, allowing or denying requests from external entities. However, the contents must be through a secure channel (such as using the Transport Layer Security (TLS), requiring both the access manager and the consumer to be always online. The requirement is not suitable for several communication situations, for example, the battlefield network where the communication is intermittent.

In contrast, an NDN node can freely communicate with any nodes it encounters because authenticity and access control are built into the data and independent from the data containers or communication channels. So, the NDN access control requires the implementation of per-packet verification, schematized trust (leveraging name relationships to manage trust), and name-based access control [8]. Particularly, the concept of access control shifts from securing communication channel to securing the contents/data. The encryption-based access control is an one effective way of access control [4, 7] in NDN. A producer can encrypt a content at the time of production, and also it is able to control the shared content by management of the corresponding decryption key distributions. That can achieve real end-to-end confidentiality and access control. Therefore, the NDN fits more to establish the access control system, when the underlying network situation is unstable in the battlefield.

2.11 Device Compromise

In encryption-based access control scheme, an end device in NDN must store its private key at its own device. Device compromise is so a serious threat. So, this section discusses three cases of device compromise by using the sample of battlefield scenario as follows.

Case 1. The device of a command center is compromised. In this case, the attacker may impersonate the command center by using a command center's private key to define the access policy for all entities. Threats to the command center can be a central point for attacking private key of the anchor such as case of attack DigiNotar, Comodo and Verisign in the classical IP network [58-60].

Case 2. The device of an UAV A is compromised. The attacker may impersonate an UAV A by using the private key of UAV A to assert the authenticity of fake content as mentioned in literature review [61-64].

Case 3. The device of a soldier is compromised. Since the UAV A encrypts a CK by using the authorized key (also known public key) to encrypt the CK. After obtaining the encrypted CK and encrypted content, a soldier uses its own private key to decrypt the corresponding authorized keys (also known private key), and then decrypt the encrypted CK, and use CK to decrypt the encrypted content. Therefore, the compromise of a soldier's device may allow the attacker to access the private key of a soldier. The attacker can use a private key of a soldier to access all the contents that a soldier has accessed before. Moreover, the aforementioned attacks require the

attacker to be neither real-time presence nor proximity [51, 65, 66]. The attacker can easily access a content inside network caches as long as the NDN network has cached the content. While the aforementioned attacks in classical IP, the attacker must perform to eavesdrop real-time presence by an attacker who must also be physically near the victim.

2.12 Perfect Forward Secrecy

Perfect Forward Secrecy (PFS), also known as Forward Secrecy, has been defined by Gunther [67] since 1990. It is a cryptographic feature of key agreement schemes to ensure that a compromise of the private key should not lead to the compromise of the session keys. Hence, by enabling the forward secrecy, the past encrypted communications/sessions should not be decrypted and still confidential, even if the private key is compromised in the future. For example, attackers can compromise a soldier's mobile device, and then attackers may obtain the long-term private key of the entities. The PFS can guarantee the security of the previously published contents between an UAV and a soldier, where the contents cannot be decrypted if one of the private key of the entities are compromised. The Diffie-Hellman (DH) key exchange protocol is one of the important practice for the exchange the shared ephemeral key to achieve PFS property [68-70].

2.13 Related Works

The work in this dissertation builds upon main concepts of the challenges of access control for NDN. There have been several existing access control schemes for enforcing access control over the NDN.

In 2013, Hamdane et al. [3] proposed an access control scheme by using encryption-based access control to allow an authorized consumer can access the encrypted content in-network caching. Their scheme defines an encryption/decryption key pairs and the symmetric key. The contents must be encrypted by a symmetric key. To control access rights, the symmetric keys must be encrypted by decryption key. The access manager can securely transmit the decryption key using the authorized consumer's public key to encrypt a decryption key. The consumer revocation is considered in their scheme by using key update and re-encryption, where need content re-encryption and distribute the new encryption/decryption key pairs for each revocation. However, the revoked consumer can access all contents previously published, because their scheme does not mechanism to remove access or the content residing within in-network caching. In addition, their scheme is not considered to support the intermittent communication, and the PFS properties.

In 2014, Chen et al. [4] proposed access control scheme by using an encryption-based access control scheme. The contents must be encrypted by an access key based on symmetric cryptography. The NDN trust model is used to securely transmit the access key. The producer can use the authorized consumer's public key to encrypt an access key. After obtaining the encrypted content, the consumer authenticates itself to the producer to securely transmit the access key. The privilege revocation is considered in their scheme by using key update and re-encryption, where need content re-encryption for each revocation. However, the revoked consumer can access all contents previously published due to their scheme not has the mechanism to remove any content

in-network caching. In addition, their scheme not suitable for the intermittent connectivity, because the producer must be always online to send the access key. Moreover, this scheme could not provide the PFS property.

In 2015, Mangili et al. [5] proposed an access control scheme using an encryption-based access control. A producer encrypts the contents using a symmetric key, and the authorized consumer can retrieve a decryption key from the directly producer. The access policy is enforced by the producer by using key-regression [71]. In this way, the producer can establish a new version of the encrypted content by using a new key. There is a mechanism to mitigate a threat of the consumer device compromise by periodically updating access policy and re-encryption. For consumer revocation, the producer generates a new key and publishes the re-encrypted content. However, the revoked consumer can access all contents previously published due to their scheme not has the mechanism to remove any content in-network caching. In addition, their scheme is not suitable for the intermittent connectivity because this access control scheme requires the consumer to authenticate itself with a centralized access manager every time accessing the contents.

In 2015, Da Silva et al. [6] proposed an access control scheme using the interest-based and attribute-based access control. Their scheme suggested the Ciphertext-Policy Attribute-based Encryption (CP-ABE) [29] to operate the policy key. The content is encrypted using the access policy, and the encrypted content can be cached on the network. After receiving the encrypted content, the consumer sends the specific *interest packet* by defining its own identity to the proxy server. To control access rights, the proxy sends access policy to the consumer if there is no revocation. However, their scheme does not have the mechanism to remove any content in-network caching. Hence, the revoked consumer can access all contents previously published even if the access privileges have been revoked. In addition, their scheme not suitable for the intermittent connectivity, because the proxy server must be always online to authenticate consumers. Moreover, this scheme used asymmetric keys to enable access control and cannot maintain the PFS property, when the consumer's device is compromised.

In 2015, Kurihara et al. [7] proposed an access control scheme, based on the encryption-based access control scheme. Their scheme uses in-network caching to distribute all keys. An access manager generates a symmetric key, and use asymmetric cryptography to encrypt this symmetric key. The producers encrypt any contents using symmetric keys. To control access rights, the producer can use the access key to encrypt the symmetric key, and use the public key of authorized consumers to encrypt access keys. After receiving the encrypted content, consumers can fetch all keys from in-network caching. The consumers use its own private key to decrypt access keys. Then, its use access keys to decrypt symmetric keys, and use this symmetric key to access content. For consumer revocation, the access manager generates a new key, and the producer can publish the re-encrypted content with new keys. However, the revoked consumer can access all contents previously published due to their scheme not has the mechanism to remove any content in-network caching. Moreover, the scheme used the asymmetric key to enable access control, but cannot provide the PFS property in case of consumer device compromise.

In 2015, Yu et al. [8] proposed an access control scheme to provide data confidentiality and access control over NDN architecture. Their scheme is named

Name-based Access Control (NAC), and based on the encryption-based access control approach. The access control policy is determined by leveraging the NDN naming convention, and all the keys can be cached in the network. For consumer revocation, the access manager generates a new key, and the producer can publish the re-encrypted content with new keys. The PFS properties are provided in their scheme by using the ephemeral key usage to protect policy keys requires the access manager and the consumer to be exchanged policy keys, if the access policies are changed. However, this access control model not suitable for the battlefield scenario, because the scheme requires the access manager to be online to manage key distribution.

In 2015, Shang et al. [9] proposed the access control framework, it is a lightweight access control protocol for constrained environments over NDN. The access manager receives the *signed interest*, then verifies the signature of *signed interest* using the consumer's public key. If the verification succeeds, it encrypts the access key and distributes over the network, using the scheme proposed in [8]. Only the authorized consumer can decrypt the access key with a corresponding key. Then, the authorized consumers can use this access key to access the services offered by the actuator using *signed interests*.

In 2017, Misra et al. [10] proposed an access control scheme by using an encryption-based access control scheme. The consumer needs to operate a one-time enrollment process at the producer for assigning the access privileges. The producers encrypt any contents by using a symmetric key. To control access rights, the producer can use the access key to encrypt this symmetric key, and publishes the encrypted content and access keys into the network caches. Only authorized consumers can use the access keys to decrypt the symmetric key, and decrypt encrypted contents after that. Consumer revocation is achieved by replacing the access keys. However, the revoked consumer can access all contents previously published due to their scheme not has the mechanism to remove any content in-network caching. In addition, their scheme not suitable for the intermittent connectivity because the consumer is needed to perform of access rights authentication at the producers, and PFS properties are not considered in their scheme.

In 2018, Feng and Guo [11] proposed an access control model based on the attribute-based access control approach. Their scheme can be effective to enable access control over intermittent connection by using the key policy as normal NDN *data packets*. The encrypted contents and all the keys can be cached in the network caches. The PFS properties are provided in their scheme by defining the short-lived of the policy keys. Consumer revocation is achieved by updating a new key, and the producer can publish the re-encrypted content with new corresponding attributes. However, the revoked consumer can access all contents previously published due to their scheme has no mechanism to remove any contents from in-network caches. In addition, this access control model is not suitable for the battlefield scenario, because it requests the producer be periodically online to send the policy keys. So, if the entities are outside the access manager network, they will not be able to acquire the policy keys when the communication is disconnected. Moreover, this scheme cannot provide the PFS property.

In 2018, Zhang et al. [12] proposed two general approaches to provide data confidentiality and access control in NDN architecture by improving the NAC [8] to enable the automatic key retriever. The first scheme, they proposed the NAC achieves

end-to-end confidentiality as well as fine-grained access control. NAC provides the data access control by using an encryption-based access control scheme, which leverages symmetric/asymmetric cryptographic operations. An access control is achieved at the same time by using encryption that allows an access manager to enforce and provides automatic access control policy distribution. The data owners can simply publish access control policies into the network caches, so that the producers and authorized consumers can proceed to work without connecting with the access managers. The NAC and NAC-ABE achieve the fine-grained access control and there are several existing applications [35, 45, 72] over NDN network use these schemes to provide data confidentiality and access control. However, NAC requires the access managers perform an access control policy, which generates the key pairs and encrypts a decryption key for each kind of data for all authorized consumers. The main problem of this scheme is that it cannot provide the PFS property when using in a very sensitive situation, like the battlefield scenario.

In 2019, Wu et al. [13] proposed an access control scheme using the interest-based and attribute-based access control. Their scheme achieves the access control by splitting the content into two parts, where consisting of the small part, and the large part. The small part must be encrypted by the producer using CP-ABE, the large part is not encrypted. To control access rights, the authorized consumer gets to the small part from only by the producer, and can get the large part from the neighbor caches by sending the specific *interest packet* by defining its own identity. This method achieves the access control because the authorized consumer can access the small part by using the access key corresponding the attribute to decrypt the small part. The neighbor caches can service the request by checking the authorized consumer with the access control structure information. To revoke the consumer, the producer can update its own access control structure information, and sends identity of the unauthorized consumer to the in-network caches for updating the access control structure information. However, the revoked consumer can access all contents previously published by using the existing access key, due to their scheme not has the mechanism to remove any content in-network caching. Furthermore, the PFS properties are not considered in their scheme.

In 2020, Wu et al. [14] proposed an access control scheme using the interest-based and attribute-based access control. To enable access control, the consumer needs to obtain the access privilege by sending the *signed interest* into the producer when the consumer first visits. Their scheme can be effective to enable access control over intermittent connection. The producer can encrypt the contents by using the CP-ABE, and publish encrypted contents into the proxy server, which may be an in-network caches. After receiving the *signed interest*, the proxy server can re-encrypt the cipher-texts by using the authorized consumer's proxy key. The authorized consumers can access the encrypted content by using the access key corresponding the attribute that received the access manager. The consumer revocation is considered. The producer can send the revoked consumer's identity to the proxy server, and the proxy key of the revoked consumer's must be destroyed. In this way, the revoked consumer unable decrypt new cipher-texts with a new proxy key. However, the revoked consumer can access all contents previously published by using the existing access key, due to their scheme not has the mechanism to remove any content in-network caching. Moreover, the PFS properties are not considered in their scheme.

2.14 Summary of Previous Studies on Access Control Schemes for NDN

This dissertation considers the access control scheme can be supported the consumer revocation, intermittent connectivity and PFS properties. We can summarize of previous studies on access control schemes for NDN as shown in Table 1.

Table 1 Summary of the access control schemes for NDN

Scheme	Consumer Revocation	Intermittent Connectivity	PFS Properties
Hamdane et al. [3]	✓	✓	✗
Chen et al. [4]	✓	✗	✗
Da Silva et al. [6]	✓	✗*	✗
Kurihara et al. [7]	✓	✓	✗
Misra et al. [10]	✓	✓	✗
Wu et al. [13]	✓	✗*	✗
Wu et al. [14]	✓	✗*	✗
Mangili et al. [5]	✓	✗	✓
Yu et al. [8]	✓	✓	✓
Feng and Guo [11]	✓	✓	✓
Zhang et al. [12]	✓	✓	✓

✓ Considered ✗ Not Consider *Proxy is Needed

Consumer revocation is considered in all existing schemes. The [3-5, 7, 8, 10-12] schemes use lazy revocation [73] by using key update and re-encryption the contents for each revocation. The [6, 13, 14] schemes use proxy re-encryption to enable the immediate revocation [18], which this mechanism requires consumer authentication from an always online proxy server. The several existing access control schemes [4-6, 13, 14] request the access managers always online to manage the policy keys, which is impossible and leads to inefficiency on the intermittent network, e.g., the battlefield scenario. The PFS properties are not considered in [3, 4, 6, 7, 10, 13, 14], while these schemes [5, 8, 11, 12] consider the PFS properties.

2.15 Access Control Model of NAC and NAC-ABE

Among the literature review, NAC and NAC-ABE should be considered as states of the arts. They can also overcome intermittent network connectivity [11, 12]. The new access control scheme, proposed by this dissertation, will also be compared to these two work later on. In this section, two of previous NDN access control proposals (NAC and NAC-ABE) will be discussed in details.

2.15.1 NAC and NAC-ABE Assumptions

The NAC and NAC-ABE uses the NDN trust model to enable the trust relationships among entities. For our battlefield scenario, all entities must already pre-installed one or more the certificate of trust anchors, e.g., a certificate of trust anchor is named “/military/KEY/<key-id>”. Each entity has a semantically meaningful name and all entities have its own public/private key pair. The trust anchor issues the certificates for all entities by using own private key to sign the certificates to binding between a name and a public key of each entity. The name of the certificate have to be under the

issuer's name prefix. In our example, a name “/military/CommandCenter/KEY/<key-id>” is the certificate name of command center. A name “/military/UAVA/KEY/<key-id>” is the certificate name of UAV A. A name “/military/SquadA/Soldier1/KEY/<key-id>” is the certificate name of a soldier in squad A.

In addition, the NAC-ABE needs the additional trust model to enable the trust relationship between all entities and the command center as described in [74]. In practice, the NAC-ABE has suggested the access manager and attribute authority may be in the identical node. That is why, in our example, the command center performs the additional mission of attribute authority. In contrast, the NAC does not need such additional.

2.15.2 NAC and NAC-ABE Overview

For discussion of the NAC and NAC-ABE schemes, we use an example in a battlefield scenario to explain of the access control scheme, advantages and disadvantages of NAC and NAC-ABE. In the scenario, an access manager (i.e., the command center) in the system will generate and publish a policy key pair for each access control policy. The keys are a KEK (Key-Encryption Key) and a KDK (Key-Decryption Key). The KEK and KDK are represented in a normal form of NDN *data packet*, where KEK carries a public key and KDK carries the corresponding private key. For the cryptographic implementation, NAC uses RSA to generate the public/private key pair, while NAC-ABE uses the CP-ABE [29]. With the reasons above, the KEK of NAC-ABE has no need to define public keys into the KEK *data packet*, but the producer can use the attribute policy in a name of the KEK *data packet* to encrypt the CK, and the consumer can fetch the corresponding KDK *data packet* from the command center, that it carries a private key, generated by CP-ABE.

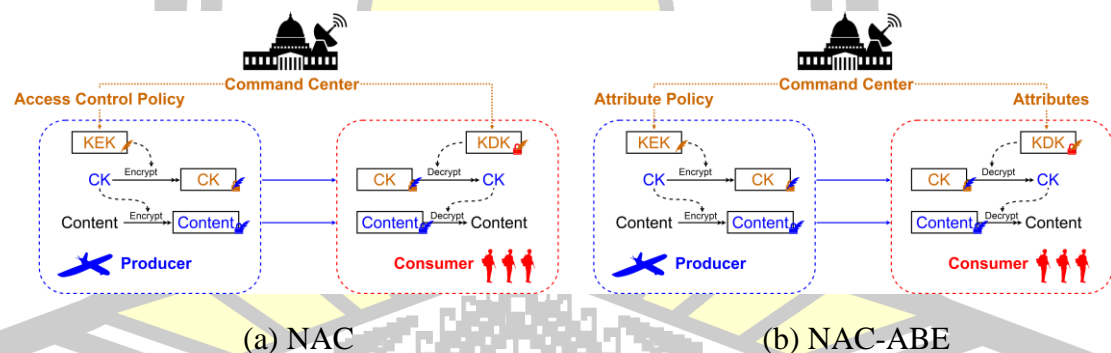


Figure 9 Overview of NAC and NAC-ABE

NAC and NAC-ABE achieve the encryption-based access control by leveraging symmetric and asymmetric cryptography, and data access control by leveraging a specially established NDN naming convention to name the key policies. As shown in Figure 9, the producers enforce access control policy by using the symmetric encryption to generate encrypted contents, and use asymmetric encryption to distribute the access keys to the authorized consumers. After receiving the request from the consumer. The producer calculates a symmetric key, called Content Key (CK). In NAC scheme, the producers use a public key in KEK to encrypt the CK, while the NAC-ABE uses policy attribute in a name of the KEK *data packet* to encrypt the CK. Therefore, the access keys can be distributed from the producer to the authorized

consumer under the control of the command center. To enable fine-grained access control, both NAC and NAC-ABE schemes allow the producer to produce encrypted contents under the granularity by defining the name prefix of encrypted contents under the granularity in KEK. To control access rights, the consumer uses the name of CK in the encrypted contents to fetch the *CK data packet*, and uses the name of KDK in the *CK data packet* to fetch the KDK. To access the encrypted contents, the consumer then uses its own private key to decrypt the KDK, so that only authorized consumers can access the encrypted contents. The consumer then uses the KDK to decrypt the encrypted CK, and use CK to decrypt the encrypted contents.

2.15.3 NAC and NAC-ABE Schemes

The workflow of NAC and NAC-ABE are shown in Figure 10. This section explains NAC and NAC-ABE schemes steps by steps in details.

- Step 1. The consumer can generate an *interest packet* (“military/UAVA/info”) automatically to fetch a content.
- Step 2. After obtaining the request, the producer fetches a KEK.
- Step 3. The command center fetches a certificate of the consumer from the network.
- Step 4. The NDN router responds by sending back a certificate of the consumer.
- Step 5. After obtaining a certificate of the consumer, the command center uses the NDN trust model to verify the authenticity of the certificate. If the certificate is authentic, the consumer can be reliably authorized by the command center.
- Step 6. The command center generates the KEK and KDK, where defines the KEK and KDK Data Name as a specific NAC naming conventions as follows.

KEK Data Name_{NAC} = “/<command center prefix>/NAC/<granularity>/KEK/
<key-id>”

KDK Data Name_{NAC} = “/<command center prefix>/NAC/<granularity>/KDK/
<key-id>/ENCRYPTED-BY/<consumer prefix>
/KEY/<consumer key-id>”

where the granularity is the name prefix of the *data packet* that is being produced by the producer, and the key identified by consumer key-id is the consumer’s certificate that is used to encrypt the KDK, and the key-id is the identity acts as an identifier of the KEK and KDK.

KEK Data Name_{NAC-ABE} = “/<command center prefix>/NAC/<granularity>/KEK
/<attribute name>”

KDK Data Name_{NAC-ABE} = “/<command center prefix>/ATTRIBUTE/<attribute
name>
/ENCRYPTED-BY/<decryptor prefix>/KEY/<consumer key id>”

where the attributes are prospective to be established before the system starts, and the key identified by consumer key-id is the consumer's certificate that is used to encrypt the KDK.

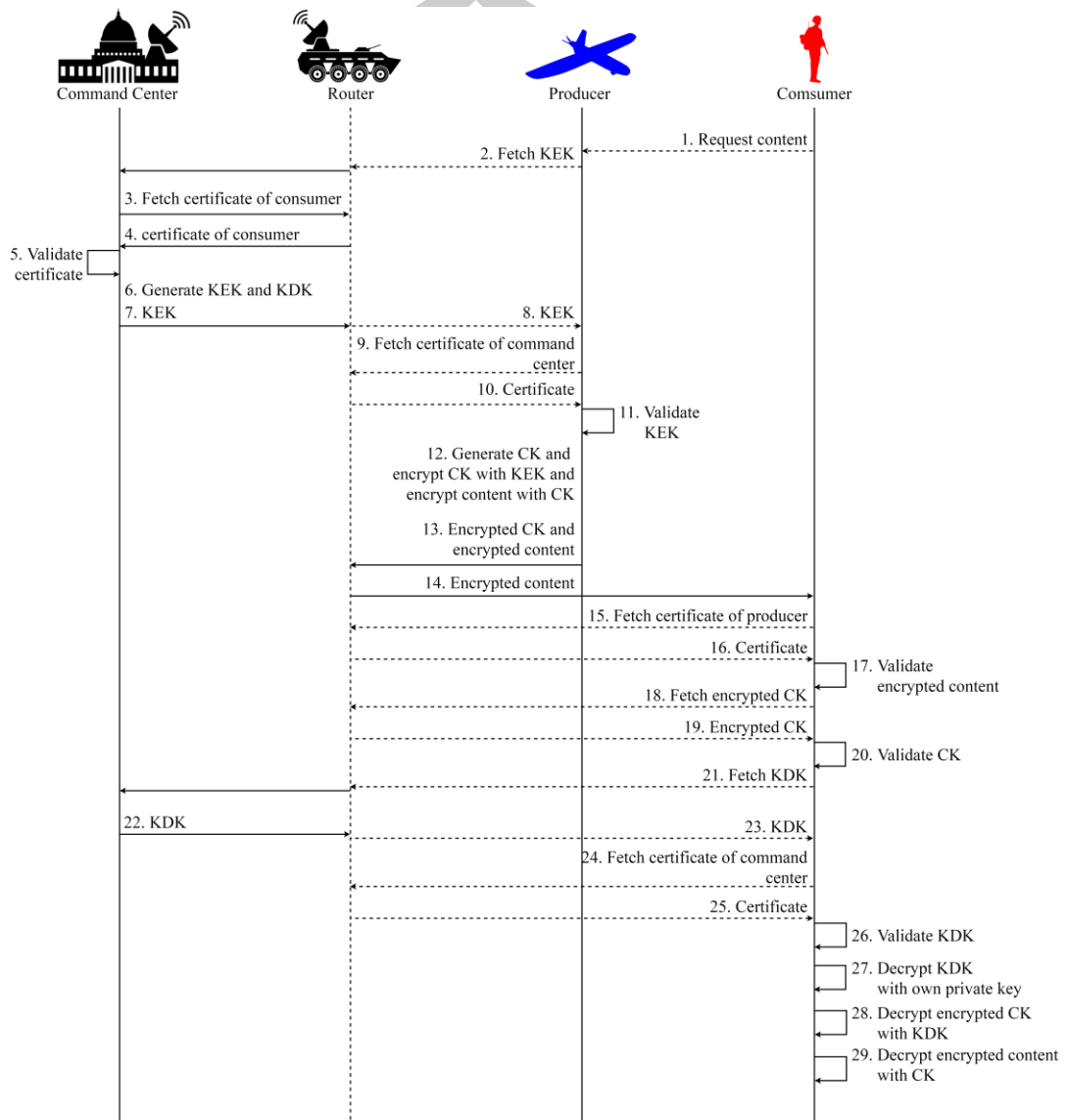


Figure 10 NAC and NAC-ABE schemes

Step 7. The command center publishes its access control policies as the KEK in the network.

Step 8. In the NAC and NAC-ABE, KEK carries the key policies in a form of normal NDN *data packet*. Therefore, the command center can distribute these contents within the in-network caches, e.g., NDN routers. The NDN router then responds by sending back a KEK of the command center, following the reverse path of the KEK *interest packet*.

Step 9. After obtaining the KEK, the producer generates an interest to fetch the certificate of command center from the network.

Step 10. The NDN router responds by sending back a certificate of the command center.

Step 11. After obtaining the certificate of the command center, the producer uses the NDN trust model to verify the authenticity of the certificate. The producer can then reliably authenticate legitimate KEK with the certificate of command center through the sequence of *data packet* signature verification.

Step 12. The producer generates CK, and encrypts the CK with KEK. The producer generates a content and encrypt it with the CK, and will wrap the encrypted content with the CK name prefix, e.g., “/<producer prefix>/CK/<key-id>/” into each *data packet*. The producer generates a CK *data packet* that carries the encrypted CK, and then signs the CK *data packet*, and signs all encrypted contents to fully authenticate the content with the own private key. For naming, the producer defines the name of encrypted content by following the granularity field in KEK, and defining the name of CK *data packet* by the NAC naming convention as follows.

$$\text{CK Data Name}_{\text{NAC}} = \text{“/}<\text{producer prefix}>/\text{CK}/<\text{key-id}>/\text{ENCRYPTED-BY} \\ /<\text{command center prefix}>/\text{NAC}/<\text{granularity}>/\text{KEK}/<\text{key-id}>\text{”}$$

$$\text{CK Data Name}_{\text{NAC-ABE}} = \text{“/}<\text{producer prefix}>/\text{CK}/<\text{key-id}>/\text{ENCRYPTED-BY} \\ /<\text{command center prefix}>/\text{NAC}/<\text{granularity}>/\text{KEK}/<\text{attribute name}>\text{”}$$

Step 13. The producer publishes an encrypted content and encrypted CK in the network.

Step 14. NDN router responds by sending back a *data packet* in an encrypted format containing the requested content.

Step 15. After obtaining the encrypted content, the consumer can learn the name of producer’s certificate in the “KeyLocator” from the received *data packet*. The name of producer’s certificate can then be put into the *interest packet* as to fetch the producer’s certificate from the network.

Step 16. The NDN router responds by sending back the certificate of the producer.

Step 17. After obtaining the certificate of producer, the consumer uses the NDN trust model to verify the authenticity of the certificate and encrypted content.

Step 18. After validating an encrypted content, the consumer can learn the CK name from the received *data packet*. The CK name can then be put into the *interest packet* as to fetch the encrypted CK from the network.

Step 19. The NDN router responds by sending back the encrypted CK.

Step 20. After obtaining the encrypted CK, the consumer uses the NDN trust model to verify the authenticity of the encrypted CK.

Step 21. The consumer can learn the KDK name from the received encrypted CK. The KDK name can then be put into the *interest packet* as to fetch the KDK from the network.

- Step 22. The command center responds by sending back a KDK.
- Step 23. In the NAC and NAC-ABE, KDK is in a form of normal NDN *data packet*. Therefore, the command center can distribute these contents within the in-network caches, e.g., NDN routers. The NDN router then responds by sending back the KDK of the command center, following the reverse path of the KDK *interest packets*.
- Step 24. After obtaining a KDK, the consumer can learn the name of command center's certificate in the "KeyLocator" from received KDK. The name of command center's certificate can then be put into the *interest packet* as to fetch the command center's certificate from the network.
- Step 25. The NDN router responds by sending back the certificate of the command center.
- Step 26. After obtaining the certificate of the command center, the consumer uses the NDN trust model to verify the authenticity of the certificate. The consumer can then reliably authenticate the legitimate KDK with the certificate of command center through the sequence of *data packet* signature verification.
- Step 27. The consumer decrypts the KDK with its own private key.
- Step 28. The consumer decrypts the encrypted CK with the KDK.
- Step 29. The consumer decrypts the encrypted content with the CK.

2.15.4 The Advantages of NAC and NAC-ABE

NAC and NAC-ABE enable the encryption-based access control over NDN, and there are several existing applications [35, 45, 72] over NDN network use the NAC and NAC-ABE schemes to provide data confidentiality and access control. In this section, we have been summarized the advantages of NAC and NAC-ABE as follows.

- **Automatic Policy Key Retrieval:** NAC and NAC-ABE leverage an NDN naming convention to define names of all keys in the system. In our example, the UAV A produces *data packets* under the name prefix "/military/UAVA" that define in the KEK. It can automatically establish the *interest packet* by defining a name "/military/CommandCenter/NAC/military/UAVA". That means the *interest packet* will have the name of the command center's name prefix appending the name prefix of the expected *data packet*. Therefore, with the leveraging NDN naming convention, NAC and NAC-ABE no longer require the naming service like Domain Name System (DNS) [75] to manually configure of all keys in the system. In addition, the NDN naming convention also supports a soldier in squad A to choose the right CK and KDK to decrypt the encrypted content, received from the UAV A. In Step 12, the UAV A generates a content and encrypt it with a CK, and will wrap the encrypted content with the CK name prefix into each *data packet*. Thus, a soldier can use the CK name from the *data packet* to fetch the CK *data packets*, as show in Figure 11. Likewise, the name in CK *data packet* can directly extract the KEK name by simply changing the "KEK" to "KDK", and the soldier uses its own certificate key-id appending to establish an *interest packet* for the KDK.

Data packet: `/military/UAVA/info` Extract CK name from Data Packet

CK Interest: `/military/UAVA/CK/<key-id>`

CK Data: `/military/UAVA/CK/<key-id>`

/ENCRYPTED-BY

`/military/CommandCenter/NAC/military/UAVA/KEK/<key-id>` Extract KDK name from CK Data Name

KDK Interest: `/military/CommandCenter/NAC/military/UAVA/KDK/<key-id>`

KDK Data: `/military/CommandCenter/NAC/military/UAVA/KDK/<key-id>`

/ENCRYPTED-BY

`/military/SquadA/Soldier1/KEY/<key-id1>`

Figure 11 Fine-grained access control of NAC and NAC-ABE

- Fine-grained Access Control:** NAC and NAC-ABE are based on NDN. All *data packets* must be named with a structured name. For our example in Figure 1, the soldier in squad A can access the content from the UAV A, and the battleship can access the content from the UAV B. To grant the soldier to access the encrypted content produced by an UAV A. The command center can establish the KDK with name “`/military/CommandCenter/NAC/military/UAVA/KDK/<key-id>/ENCRYPTED-BY/military/SquadA/Soldier1/KEY/<key-id1>`” and KEK with name “`/military/CommandCenter/NAC/military/UAVA/KEK/<key-id>`”. In Step 12 of NAC and NAC-ABE schemes, the UAV A must be used granularity in KEK to define the name of encrypted content under “`/military/UAVA`”. Consequently, all *data packets* produced by the UAV A, cannot be accessed by the battleship, because it cannot access the KDK named “`/military/CommandCenter/NAC/military/UAVA/KDK/<key-id>/ENCRPYPTED-BY/military/SquadA/Soldier1/KEY/<key-id1>`” that encrypted using the soldier’s public keys.
- Support for Intermittent Connectivity:** In Step 8 and Step 23 of NAC and NAC-ABE schemes, the policy key pairs (KDK and KEK) are represented in a form of normal NDN *data packet*. Therefore, the command center can distribute these contents within the in-network caches, e.g., NDN routers. The NDN router then responds by sending back these key policies following the reverse path of the *interest packet*. In the scenario shown in Figure 1, the communications can face with the intermittent loss; for example, the link between the satellite and the command center can go down periodically. Yet, due to the NDN concept, the copies of KEK still remain in the network, such as in the cache of the aircraft gateway. The UAV A can be fetched KEK by using an *interest packet* to generate the encrypted content. Similarly, the soldier requests the encrypted content from the UAV A, and the KDK from the command center by sending two *interest packets* into the NDN network after obtaining an encrypted content from the UAV A, as long as there is a copy for the KDK in the network (for example, in the cache

of the squad gateway). The soldier can always decrypt the encrypted content from the UAV A, even with the intermittent connectivity. In contrast, it is impossible for the current IP network that communication based on the channel-based communication model [33].

- **The NAC-ABE is Better Scalability:** In Step 12, CK must be encrypted with KEK. For NAC-ABE, CK must be encrypted by using attribute-based KEK. For example, the UAV A may produce an encrypted content for the soldiers in squad A by using the KEK named “/military/CommandCenter/NAC/UAVA/KEK/(“SquadA” AND “Soldier”) or “Battleship”. The UAV A uses the authorized attribute string (“SquadA” AND “Soldier”) or “Battleship” to encrypt the CK for the squad A. Similarly, the UAV B can use the authorized attribute string (“SquadA” AND “Soldier”) or “Battleship” to encrypt the CK for the battleship. The squad A and battleship then use the corresponding private keys from their attributes in KDK to decrypt the CK. Therefore, to grant the access rights for three soldiers in squad A and a battleship to access the encrypted contents, produce by UAV A and UAV B, the command center can generate only one policy key, and distributes four KDK *data packets* for three soldiers in squad A and the battleship. That means, the policy key must be generated $O(a)$ times and distributes $O(a \times n)$ the KDK *data packets*, where a is the different set of attributes, and n is the number of consumers in NAC-ABE. In contrast, the scalability issues in NAC come from the number of consumers are increasing. The command center needs to generate two policy keys and eight KDK *data packets*. That means, the policy key must be generated $O(m)$ times and the KDK *data packets* must be distributed $O(m \times n)$ in NAC, where m is the number of granularities.
- **Consumer Revocation:** NAC and NAC-ABE generate the policy key pairs as a normal NDN *data packet*. Therefore, the command center can distribute these contents within the NDN network by defining the expiration of key policies to periodically update KEK and KDK. The producer must re-encrypt all its own contents with a new CK and KEK. The policy key renewals are transparent to the consumers, which can learn a name of new CK and KDK from receiving *data packets*.
- **Threat Mitigation of Man-In-The-Middle Attack:** NAC and NAC-ABE perform data authentication, based on the NDN trust model. Therefore, the attackers may perform Man-In-The-Middle (MITM) attacks and modify some *data packets* (e.g., KEK and KDK). The entities can check the illicit changes of all *data packets* by verifying the signature.

2.15.5 The Disadvantages of NAC and NAC-ABE

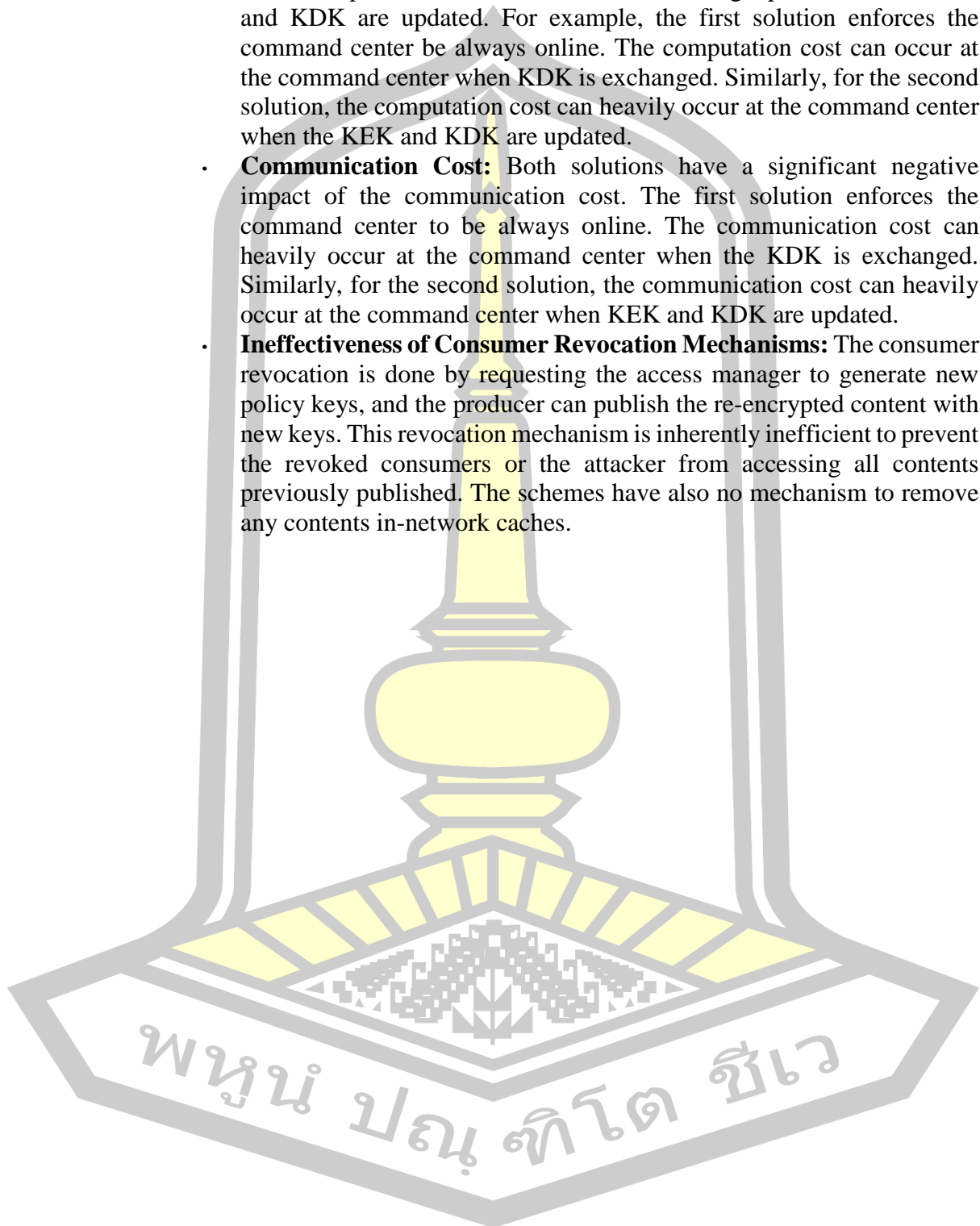
- **Consumer Compromise:** NAC and NAC-ABE need a trust model that imposes which the NDN certificates are authorized to validate a *data packet*. For example, as shown in Figure 4, the first step of enabling secure communication in NDN is the certificate issuance. To enable the access right, the command center distributes the KDK to the authorized soldier by encrypting KDK using the soldier’s public key. Therefore, the

compromise of the end device of the soldier may allow an attacker to access the private key of the soldier. This compromise could cause the attacker to use the soldier private key to access all the encrypted contents that the soldier has accessed before. This leak of past accessed contents could cause a serious problem in many situations, particularly in the battlefield situation. So, the aforementioned attacks have shown a significant negative design of NAC and NAC-ABE. When the producer produces and distributes an encrypted content to the NDN network, the NAC and NAC-ABE are inefficient to prevent the attacker to access the previously published content. Technically, both scheme has no PFS property, and they don't have a mechanism to remove the previously published content inside network caches. The producer cannot control the lifetime of the encrypted contents, where the attacker to be neither real-time presence nor proximity. NAC and NAC-ABE have been proposed two solutions to mitigate a threat of the consumer device compromise. However, it is still in the initial phase.

- **Unsuitability for Intermittent Connectivity:** For the first PFS solution, Yu et al. [8] have proposed to use the ephemeral key to encrypt KDK, where the ephemeral key can establish by using the key agreement protocol such as Diffie-Hellman (DH) key exchange [28]. However, this solution requires the access manager to be always online to set the access control policies. The communication between both entities must be always online. This is not suitable for the intermittent connectivity, e.g., in the battlefield scenario. In our example, if the communications are faced with intermittent loss, the soldier would not be able to decrypt the encrypted contents, because the soldier could not receive KDK from the command center.

- **Ineffectiveness of Solutions from Consumer Compromise:** For the second solution, Zhang et al. [12] has proposed to use the short-lived of the policy key pairs. NAC and NAC-ABE may define the expiration of key policies in a short-lived period to revoke the policies and periodically update KEK and KDK, to reduce the vulnerability of content leakage, when the private key of the consumer is compromised. However, NAC and NAC-ABE are inefficient to prevent the attacker to access the previously published content, because they don't have a mechanism to remove the previously published content inside network caches from any access. The producer cannot control the lifetime of the encrypted contents. In addition, the urgency is the main concern to report a notification to producers to limit a consumer compromise, where the revocation report may not endorsed to be received by every producer [62]. This problem has occurred in the classic Internet. For example, DigiNotar [55], a Dutch CA, has been attacked by an attacker to issue a fraudulent certificate for Google. On September 2011, the major browsers (i.e., Mozilla Firefox, Google Chrome, Internet Explorer and Apple Safari) have revoked the trust in the DigiNotar root certificate.

- **Computation Cost:** Both solutions have a significant negative impact of the computation cost that can occur in the single point when the KEK and KDK are updated. For example, the first solution enforces the command center be always online. The computation cost can occur at the command center when KDK is exchanged. Similarly, for the second solution, the computation cost can heavily occur at the command center when the KEK and KDK are updated.
- **Communication Cost:** Both solutions have a significant negative impact of the communication cost. The first solution enforces the command center to be always online. The communication cost can heavily occur at the command center when the KDK is exchanged. Similarly, for the second solution, the communication cost can heavily occur at the command center when KEK and KDK are updated.
- **Ineffectiveness of Consumer Revocation Mechanisms:** The consumer revocation is done by requesting the access manager to generate new policy keys, and the producer can publish the re-encrypted content with new keys. This revocation mechanism is inherently inefficient to prevent the revoked consumers or the attacker from accessing all contents previously published. The schemes have also no mechanism to remove any contents in-network caches.



CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

This dissertation aims to describe the access control mechanism over NDN, and evaluates the previous access control schemes, and also propose the novel access control scheme for NDN to better solve the problems of the previous access control schemes. We illustrate the research methodology of this dissertation that shown in Figure 12.

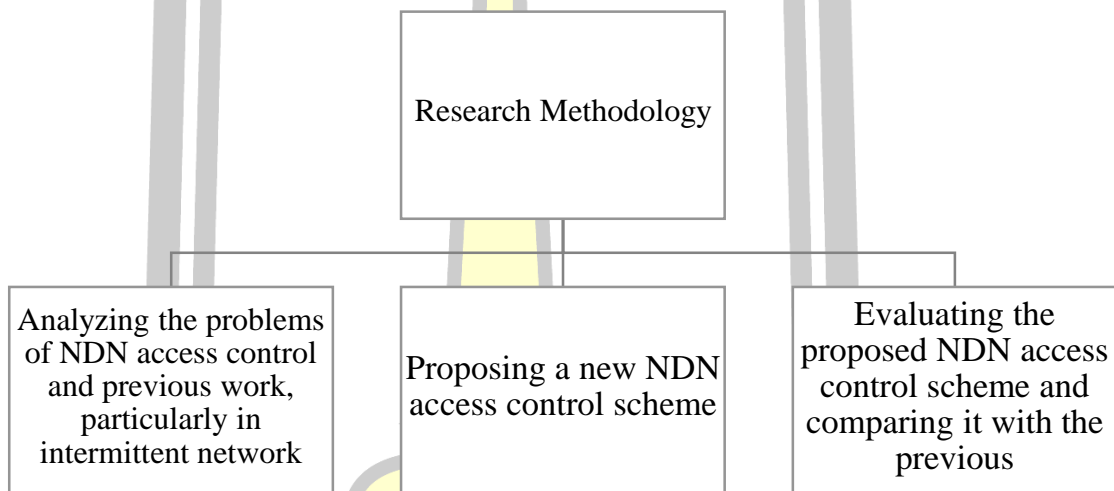


Figure 12 Research Methodology

We first analyze the problems of NDN access control and previous work, particularly in intermittent network. We describe analyzing the problems of NDN access control and previous work, particularly in intermittent network in a Section 3.2. In a Section 3.3, we propose a new NDN access control scheme. We describe the performance evaluation techniques to evaluate the proposed NDN access control scheme and comparing it with the previous in a Section 3.4.

3.2 Analyzing the Problems of NDN Access Control and Previous Work

The previous work has been analyzed in details in Chapter 2 (Section 2.14 and Section 2.15). This section will summarize the analysis. We describe about the properties of existing access control schemes, as shown in Table 2. Although the existing access control schemes can enable the access control for NDN, the following challenges must be discussed.

First, several existing access control schemes [4, 5] require the **access manager always online**, which is impossible and leads to inefficiency on the intermittently connected communication. Using the NDN naming convention to name policy keys and defining the policy keys as normal NDN *data packets* that proposed in [3, 7, 8, 10-12, 14] can overcome this problem.

Second, **consumer revocation** is considered in all existing schemes. However, the [3-5, 7, 8, 10-12] schemes use lazy revocation [73]. That is an inefficient approach of the consumer revocation mechanisms. Their scheme doesn't have the mechanism to remove any content in-network caching, and also the revoked consumer can access all contents previously published until the next update is performed [64, 76]. The [6, 13, 14] schemes use proxy re-encryption to enable the immediate revocation [18], which the drawback of this mechanism is the need for an always online proxy server [64].

Third, the **Perfect Forward Secrecy (PFS)** has not been considered by many previous proposals [3, 4, 6, 7, 10, 13, 14]. Only some previous work has proposed some solutions for PFS. Some work [8] proposed to use an ephemeral key to protect policy keys. Some work [5, 11, 12] proposed to use the short-lived policy keys. However, both solutions have the following problems:

1. The solution in [8] requires the access manager and the consumer to exchange policy keys, every time when the access policies are changed. This solution is not suitable for the intermittent connectivity, e.g., in the battlefield scenario, if the communication between the access manager and the consumer went down.
2. In the second solution, the short-lived policy key requires the access manager to frequently communicate to spread the access control policies. This requirement is hard to fulfill in battlefield network scenario, where the access manager could be in the intermittent connectivity and lost its communication quite often.
3. Both solutions are inefficient to prevent the attacker to access the previously published content, because they don't have the mechanism to remove the previously published content inside network caches from any access, due to the producer cannot control the lifetime of the encrypted contents.
4. The computation cost and the communication cost is quite high for all entities for the both solutions. Because the policy key must be regenerated by the access manager. Hence, the encrypted content must be also regenerated by producer under the new policy key, and also the consumer need for requesting for the new access policy and new encrypted content. While, the access control policy and the content has not changed.

Table 2 The properties of existing access control schemes

Scheme	Access Manager Always Online	Revocation Mechanism	PFS Properties
Hamdane et al. [3]	No Need	Lazy Revocation	Not considered
Chen et al. [4]	Need	Lazy Revocation	Not considered
Da Silva et al. [6]	A proxy is needed	Immediate Revocation	Not considered
Kurihara et al. [7]	No Need	Lazy Revocation	Not considered
Misra et al. [10]	No Need	Lazy Revocation	Not considered
Wu et al. [13]	A proxy is needed	Immediate Revocation	Not considered
Wu et al. [14]	A proxy is needed	Immediate Revocation	Not considered
Mangili et al. [5]	Need	Lazy Revocation	Considered
Yu et al. [8]	No Need	Lazy Revocation	Considered
Feng and Guo [11]	No Need	Lazy Revocation	Considered
Zhang et al. [12]	No Need	Lazy Revocation	Considered

3.3 Proposing a New NDN Access Control Scheme

To better solve the NDN access control problems, we propose the EKAC: Ephemeral Key-based Access Control scheme for NDN, as described later (in Chapter 4). The EKAC provides an access control scheme to prevent the attackers to access the previously published content, in case that the attacker compromises consumer devices to gain the consumer's private key. The EKAC design goals are established on an ephemeral key to provide the access key for authorization consumers, and utilizes symmetric and asymmetric cryptography algorithms, to provide content confidentiality and automate data access control. By using ephemeral key shares, this access control scheme is suitable for intermittent connection. We use a battlefield scenario as an example throughout the dissertation to illustrate our proposed approach. The design goals of the EKAC scheme are the following properties:

- **Fine-grained access control** must be able to achieve access control on a specific user. For Example, an access manager can control the read access to content from two dimensions: specifying the privilege of individual consumption credential and restricting the set of credentials that a consumer can obtain. As shown in Figure 1, between the two rules: (1) the squads having the role can access the UAV A all the time. (2) In some cases, the battleship having the role can access the UAV B for 9am-5pm every day or belonging to region Pacific Ocean.
- **Scalability**. For NDN encryption-based access control, policy key pairs are very significant to specify the access control policy. Scalability here means that the number of policy keys (used for access control) should be manageable. The smaller number of policy keys is preferred.
- **Perfect Forward Secrecy (PFS)** is required, particularly for data sensitive situation, such as battlefield scenarios. It's so important that the past session could not be compromised even the private key of a consumer is hacked after end device compromise.
- **Suitable for intermittent connection**. An access manager and a producer do not require to be online for setting the access control policies, and authenticating consumers. This is because the connection can be lost periodically for the intermittent connection. In the battlefield scenario, the access manager can even under the on and off satellite link.
- **Revocation** occurs when a consumer leaves, or an access key expires, or an access key in the system is abolished. The access control scheme enables immediate revocation of the consumer. The revoked entities cannot access all contents previously or forwards published.

3.4 Performance Evaluation Techniques

In this dissertation, we use the performance evaluation techniques in two aspects: experimental using a network emulation and a testbed (further explained in Section 5.2) and efficiency analysis (further explained in Section 5.3). We can illustrate that the performance evaluation techniques as follows.

3.4.1 Implementation Testbed

To perform experiments by using a testbed, we evaluate the performance in terms of the size of *data packet* transmission (Section 5.2.2), access revocation cost

(Section 5.2.3) and cryptographic operations (Section 5.2.4). We have implemented the EKAC scheme as an application prototype in C++ version 7.2.0, and using the NDN-CXX version 0.7.0. This library has been used by existing applications [77] over NDN network, including the NAC [32] and NAC-ABE [78]. In addition, we use Crypto++ library version 8.3 for DH key exchange protocol.

3.4.2 Network Emulation

The Common Open Research Emulator (CORE) [79] and network emulation techniques are deployed to evaluate the performance of EKAC's access revocation mechanisms. CORE emulator has been developed by the U.S. Naval Research Laboratory, running on Linux platform (such as the Ubuntu operating system) with a simple Graphical User Interface (GUI). CORE emulator can be used to manage the Mobile Ad-Hoc Network (MANET) by using a virtual network environment. Our implementation of CORE extensions has also been vigorously validated [73].

3.4.3 PFS and Efficiency Analysis

In addition, we evaluate the performance in terms of computation cost (Section 5.3.1), communication cost (Section 5.3.2) and revocation complexity (Section 5.3.3). We compare in the solution to achieve the PFS properties of the EKAC scheme against two solutions in the NAC and NAC-ABE schemes. The first solution has been proposed by Yu et al. [8] that is the ephemeral key usage to protect the access keys. The second solution has been proposed by Zhang et al. [12] is the short-lived policy keys. All schemes operate under the same scenario, according to perform the comparative evaluation in [8]. For the computation and communication costs, the policy keys of Zhang et al. [12] solution (the short-lived policy keys) are assumed to be renewed for every day and every hour to provide near PFS. To analyze of computation cost, we count all cryptographic operations of EKAC, NAC and NAC-ABE comparatively. We run the EKAC, NAC [32] and NAC-ABE [78] application testbed in the same node, and record experimental results of cryptographic operations to evaluate the computation time of operations.

For the revocation complexity, we analyze EKAC revocation complexity against the previous work, both lazy revocation [8, 12] and immediate revocation [14]. We assume that all mechanisms have already obtained the revocation command from the access manager. For the analysis, we specify the following variables. The system has an n authorized consumers, x data packets. R is the number of revocation commands. N is the number of the immediate nodes (e.g., proxy servers or NDN routers).

3.4.4 Result Analysis and Confidence Interval

The experimental results from the testbed and emulation of the EKAC, NAC and NAC-ABE would be different for each experiment. So, we consider a confident result, where an average value from the results is established with confident interval of 95%. They can be calculated as follows:

The average results \bar{x} can be calculated as:

$$\bar{x} = \frac{\sum_{i=1}^n x_i}{n}$$

where n is a number of runs, x_i and is the result of a run.

To establish \bar{x} with a confidence interval, which significant level (α) of 95% confidence level is 0.05. Thus, the average results with confidence intervals are:

$$\bar{x} \pm \left(t_{\frac{\alpha}{2}} \times \frac{s}{\sqrt{n}} \right)$$

where s is the standard deviation and can be calculated as:

$$s = \sqrt{\frac{\sum_{i=1}^n x_i^2 - \left(\sum_{i=1}^n x_i \right)^2}{n(n-1)}}$$

พหุบัน ปณ กิโต ชีเว

CHAPTER 4

OUR ACCESS CONTROL SCHEME

4.1 The Design of EKAC

The design of the EKAC achieved as design goals by using the Diffie-Hellman (DH) [28] key exchange protocol to establish ephemeral key shares between a producer and consumer to enable the Perfect Forward Secrecy (PFS) properties. EKAC achieves fine-grained access control by leveraging an NDN naming convention to define naming conventions to express the access rights of the keys policy and the access keys, which supports access control with scalability of access control policies. The notations used to explain EKAC scheme is described in Table 3.

Table 3 Deployed notations

Notations	Definition
p	A large prime integer of public parameter
g	An integer of public parameter, $g < p$ and g is a primitive root modulo for p [80]
a	A random secret key integer: $1 \leq a \leq p-1$
b	A random secret key integer: $1 \leq b \leq p-1$
DH_P	DH public key of producer
DH_C	DH public key of consumer

4.2 EKAC Assumptions

EKAC is based on NDN's security mechanisms. The first step is that a suitable trust relationship among all entities must be enabled via security bootstrapping process. Before performing in the battlefield, all entities need trusted keys to bootstrap the trust (trust anchors). For example, as depicted in Figure 4, the trust anchor in EKAC system is "/military". Therefore, the certificate of the command center is either pre-installed or usually received through a usual installation via out-of-band mechanisms to all entities in the system. Every entity has a semantically meaningful name, that in our example: (1) command center ("/military/CommandCenter") that directly controls the access rights and production rights; (2) The UAV A and UAV B ("/military/UAV(A/B)") are the producers; and (3) the squad A ("/military/SquadA/Soldier1") and the battleship ("/military/Battleship") are the consumers. All entities have their own public/private key pairs, and can communicate with the command center and other units in the system. The trust anchor in this military system may then be named "/military". It issues the certificates for all entities by using its own private key to sign the public key of the entity. The name of the key has to be under the issuer's name prefix. In our example, a name "/military/CommandCenter/KEY/<key-id>" is the certificate name of command center. A name "/military/UAV(A/B)/KEY/<key-id>" is the certificate name of UAV A and UAV B. A name "/military/SquadA/Soldier1/KEY/<key-id>" is the certificate name of a soldier in squad A, and "/military/Battleship/KEY/<key-id>" is the certificate name of the battleship, where the "prefix" is the name of certificate owner, and the

component after “KEY” is the key-id that uses as an identifier of the certificate. Like normal NDN *data packets*, a certificate carrying the public key information can be cached and fetched like any other content. So, the command center and the UAV A can produce any *data packets*, where the soldier can authenticate a content received from the command center or the UAV A whenever needed via the NDN trust model as described in [38].

In addition, since EKAC uses the DH key exchange protocol to establish the ephemeral key shares, EKAC needs an addition mechanism to agree on the public parameters p and g . We take the simple approach of pre-installing a p and g into the device of all entities.

4.3 EKAC Design Overview

EKAC is established on an ephemeral key to provide the access key for authorization consumers, and utilizes symmetric and asymmetric cryptography algorithms, to provide content confidentiality and automate data access control. To be suitable deployed for intermittent connection, we use a battlefield scenario to illustrate a possible access control scheme and our proposed approach as follows.

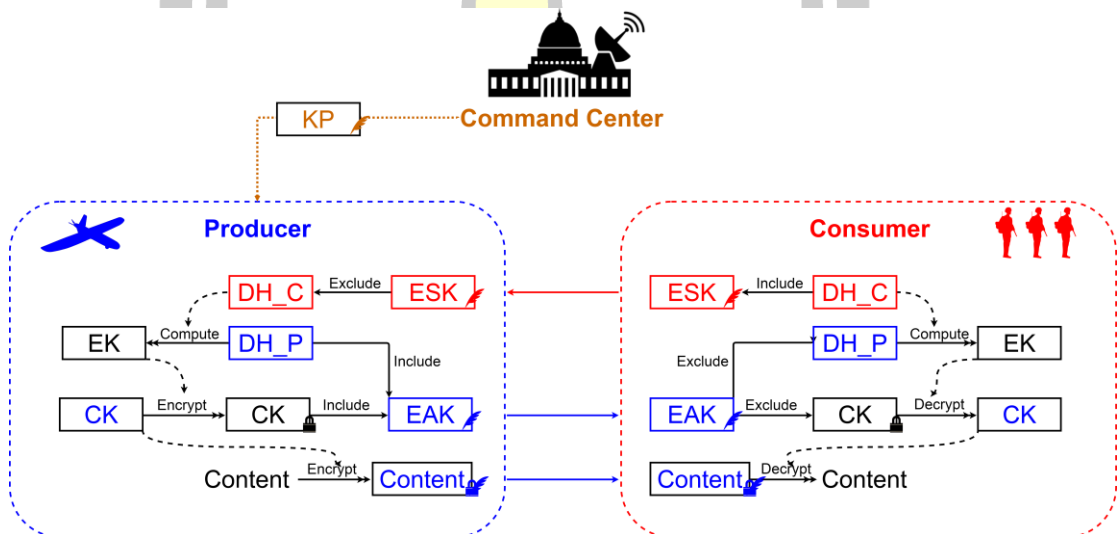


Figure 13 EKAC design overview

In EKAC design as shown in Figure 13, EKAC leverages naming conventions to establish fine granularity access control. To establish effective access control, the command center needs to generate the policy keys, called Key Policy (KP). It directly controls both content production and the access rights.

EKAC enables the PFS properties by requesting the consumer to generate and publish the Ephemeral Share Keys (ESK), carrying the DH_C (a public parameter of Diffie-Hellman algorithm, further mentioned in Section 4.5). After that, the producer fetches the content of KP and learns for a granularity in the KP. KP name should be used for defining its own content name. Producers can then learn the ESK name from the KP name to get an ESK. After obtaining an ESK, the producer generates its own DH_P , and then computes the Ephemeral Key (EK) from two DH public keys (the other one is contained in the corresponding ESK). To enforce access control, the producer generates Content Key (CK) and encrypts a content with the CK. To establish

ephemeral key shares between a producer and an authorized consumer, the producer encrypts CK with EK. The producer then publishes the Ephemeral Access Keys (EAK) *data packet*, carrying the encrypted CK and its own DH_P. The producer also publishes encrypted contents by wrapping it with the EAK name in a *data packet*. After obtaining an encrypted content, the consumer learns which corresponding EAK should be used to compute the EK by checking the EAK name from the received encrypted content. If its corresponding EAK name has never used to compute EK, the consumer uses its own DH_C to compute the EK with DH_P of the producer in EAK. If the consumer has already got a corresponding EK, the consumer will use its own EK for decrypting an encrypted CK in EAK, and then uses the CK to decrypt an encrypted content.

EKAC establishes all the keys as normal NDN *data packet*. Therefore, the NDN network allows all entities to distribute the keys within the network. In this way, the producer and consumer can always perform a communication, as long as there is a copy the keys on the network even with intermittent connectivity.

Besides, the keys can define the expiration to revoke the policies and periodically update. EKAC scheme can also remove the encrypted contents in-network caches, and revoke the KP. The command center can use a *signed interest* to carry the revoked command in a name. Likewise, the producer can revoke access to published content in the network by sending a *signed interest* that carries the revoked command to in-network caches. To prevent the sensitive information, the contents must be re-encrypted to replace all encrypted contents, which are created previously by using a new KP, CK and EK. By this way, the revoked entity cannot access new encrypted content.

4.4 EKAC Naming Conventions

The EKAC leverages naming convention to establish fine-grained access control. To establish effective access control, the command center needs to generate the policy key, called Key Policy (KP). The KP in EKAC is just like any other NDN *data packet*, can directly be fetched through *interest packets* carrying the corresponding key names. EKAC is named all the keys under a specific naming convention as discussed as follows:

Key Policy (KP): The command center needs to generate a KP, and it can simply publish KP to in-network data repositories, so that the producers can continue to work without communicating with the command center. The EKAC defines the naming convention for KP as follows:

KP Interest = “/<command center prefix>/EKAC/<granularity>/KP/<key-id>”

KP Data name = “/<command center prefix>/EKAC/<granularity>/KP/<key-id>/AUTHORIZED/<consumer prefix>/KEY/<consumer key-id>”

where key identified by consumer key-id is the consumer’s certificate, and the key-id is the unique identifier of the KP key.

Ephemeral Share Keys (ESK): The consumer generates and publishes the ESK to achieve the content access. The EKAC defines the naming convention for ESK as follows:

ESK Interest = “<consumer prefix>/<consumer key-id>/**ESK**”

ESK Data Name = “<consumer prefix>/<consumer key-id>/**ESK**/**<key-id>**”

where signed by the private key of consumer, and the key-id is the unique identifier of the key.

Ephemeral Access Keys (EAK): After obtaining the request, the producer fetches the corresponding ESK by using the ESK Interest to generate an EAK. The EKAC defines the naming convention for EAK as follows:

EAK Interest = “/<granularity>/**EAK**”

EAK Data Name = “/<granularity>/**EAK**/**<key-id>**”

where the key-id is the same as its corresponding ESK.

4.5 EKAC Scheme

The workflow of EKAC is shown in Figure 14. This section explains each of steps in EKAC schemes.

- Step 1. The consumer can generate the *interest packet* (“military/UAVA/info”) automatically by following the naming convention to fetch the content.
- Step 2. After obtaining the *interest packet* in Step 1, the producer then requests a KP by sending KP Interest as a specific EKAC naming convention.
- Step 3. The command center generates KP, where defines the KP Data Name as a specific EKAC naming convention.
- Step 4. The command center publishes its access control policies as the KP in the network.
- Step 5. If the KP still stores inside in-network caches, the producer can fetch the KP content directly from the NDN network nodes, e.g., the squad gateway.
- Step 6. After obtaining the KP, the producer generates an *interest packet* to fetch the certificate of command center from the network, where it can learn the name of command center’s certificate from the received KP.
- Step 7. The NDN router responds by sending back a certificate of the command center.
- Step 8. After obtaining a certificate of the command center, the producer uses the NDN trust model to verify the authenticity of the certificate. The producer can then reliably authenticate the legitimate of KP with the certificate of command center through the sequence of *data packet* signature verification.

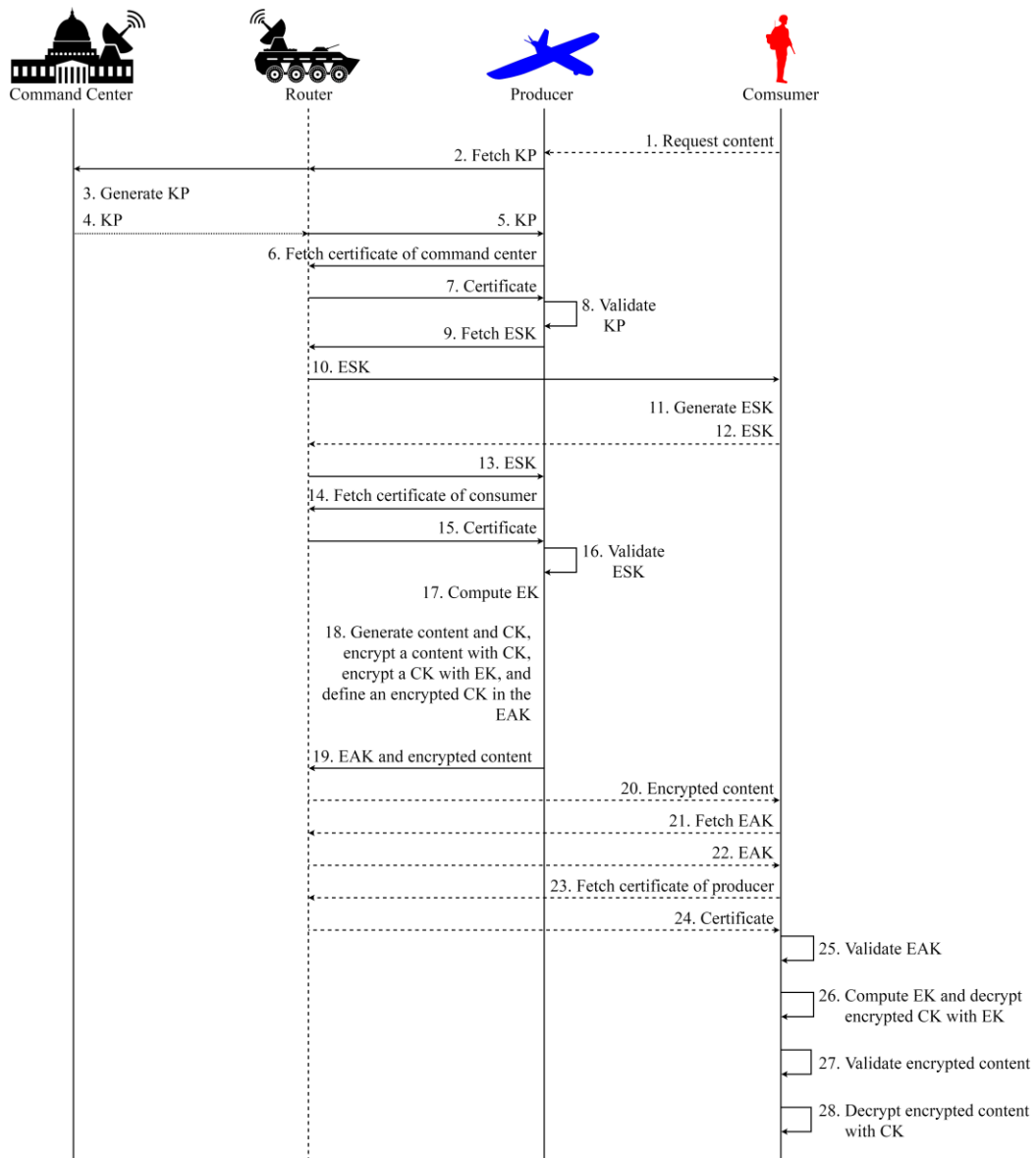


Figure 14 EKAC scheme

Step 9. The producer requests an ESK by using an ESK Interest as a specific EKAC naming convention. So, the producer can learn which ESK Interest should be used for requesting an ESK by using the KP Data Name.

Step 10. The NDN router forwards the ESK Interest to the consumer.

Step 11. The consumer generates its own DH_C , where $DH_C = g^a \pmod{p}$, and define a DH_C in the ESK *data packet*, where defined the ESK Data Name as a specific EKAC naming convention. The consumer then signs ESK *data packet* with its own private key.

Step 12. The consumer responds by sending back an ESK to the NDN router.

Step 13. The NDN router responds by sending back an ESK to the producer.

- Step 14. After obtaining ESK of the consumer, the producer uses the name of consumer's certificate in KP to requests the certificate of consumer to validate the ESK.
- Step 15. The NDN router responds by sending back a certificate of the consumer.
- Step 16. After obtaining the certificate of the consumer, the producer uses the NDN trust model to verify the authenticity of the certificate and ESK.
- Step 17. The producer generates own DH_P, where $DH_P = g^b \pmod p$, and then computes the EK with the DH_C in the corresponding ESK, where $EK = DH_C^b \pmod p$.
- Step 18. The producer generates CK, and encrypts CK with an EK. The producer then generates the EAK by defining the EAK Data Name as a specific EKAC naming convention, where the EAK *data packet* carrying the encrypted CK and own DH_P, and signing EAK *data packet* with its own private key. The producer generates and encrypts the content with the CK, and will wrap the encrypted content with the EAK name into a *data packet*. The producer then signs all encrypted contents to fully authenticate the content with its own private key.
- Step 19. The producer publishes an encrypted content and EAK in the network.
- Step 20. The NDN router responds by sending back a *data packet* in an encrypted format, containing the requested content.
- Step 21. After obtaining an encrypted content, the consumer can learn the EAK name from the received *data packet* or automatically generate *interest packet* by following the EKAC naming convention to fetch the EAK *data packets* from the NDN network.
- Step 22. The NDN router responds by sending back an EAK *data packet*.
- Step 23. After obtaining the EAK, the consumer can learn the name of producer's certificate from the received EAK, and generate an *interest packet* to fetch the certificate of the producer from the network.
- Step 24. The NDN router responds by sending back a certificate of the producer.
- Step 25. After obtaining the certificate of the producer, the consumer uses the NDN trust model to verify the authenticity of the certificate and EAK.
- Step 26. The consumer can learn which EAK should be used to compute the EK by checking the identifier of its own corresponding ESK. If its corresponding EAK has never used to compute EK, the consumer uses its own a , that has already got from Step 17, to compute the EK with DH_P of the producer in EAK, where $EK = DH_P^a \pmod p$. If the consumer has already a corresponding EK, the consumer will use its own EK for decrypting the encrypted CK.
- Step 27. The consumer uses the NDN trust model, where it can use the certificate from Step 25, to verify the authenticity of the encrypted content.
- Step 28. If the encrypted content is authentic, the consumer uses its own CK to decrypt the encrypted contents.

4.6 EKAC Access Revocation

The EKAC leverages the *signed interest* to revoke the KP and encrypted content in the network caches. To establish effective access revocation, the command center or the producer needs to generate the signed interest by defining the naming as follows:

Signed Interest = “/<Immediate>/EKAC/REVOCABLE/<command>”

where Signed Interest signed by the private key of the sender.

The workflow of EKAC access revocation is shown in Figure 15. This section explains the steps of access revocation as follows.

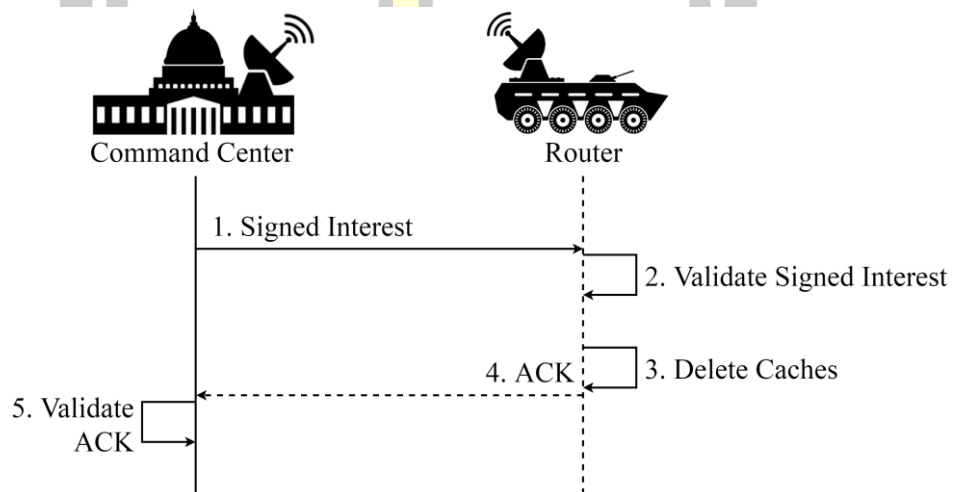


Figure 15 EKAC access revocation mechanism

Step 1. The sender sends *signed interest* to the network.

Step 2. After obtaining a *signed interest*, the router uses the NDN trust model to verify the authenticity of the signed interest. The router can identify the received command by using the NDN trust model. For example, if the *signed interest* is authenticated by the certificate of the command center. If success, it means that the *signed interest* comes from the command center. Likewise, if *signed interest* is successfully authenticated by the certificate of the producer. The *signed interest* would come from the producer.

Step 3. If the verification succeeds, the router will operate the commands, defined in the *signed interests* by deleting the named data as specified in the “command” field. For example, the command specified in “command” field is “/military/CommandCenter/EKAC/military/UAVA/KP”. The router can delete the data in its own caches by following a specified name in the “command” field that come from the command center. Likewise, the router can delete the data in its own caches by following the name “UAVA”, defining in the “command” field of the producer’s *signed interest*.

Step 4. The router sends back a *data packet* to the client with an acknowledgement (ACK) that signed by its own private key.

Step 5. After obtaining an ACK *data packet*, the command center or the producer uses the NDN trust model to verify the authenticity of an ACK *data packet*.

4.7 EKAC Properties

The EKAC aims to establish the encryption-based access control of *data packets* over the NDN. EKAC is established on a combination of the NDN naming convention and the ephemeral key shares between a producer and a consumer to enable PFS. We use a battlefield scenario as an example to illustrate the possibility of EKAC to achieve the design goal as follows.

- **Fine-grained Access Control:** The EKAC leverages the NDN naming convention to establish fine-grained access control, where the KP must be named with a structured name. For our example in Figure 1, we define that a soldier in squad A can access the content from the UAV A, and the battleship can access the content from the UAV B. To grant the soldier to access the encrypted content produced by an UAV A, the command center can establish the KP with the name “/military/CommandCenter/EKAC/military/UAVA/KP/<key-id>/AUTHORIZED/military/SquadA/Soldier1/KEY/<key-id1>”. From our example of the EKAC schemes and the work flow step 18, the UAV A must use the granularity in KP to define the name of the encrypted content under “/military/UAVA”. All *data packets*, produced by the UAV A, cannot be accessed by the battleship, because in Step 9, the UAV A uses the ESK name of the soldier, defined in KP, to request the ESK of the authorized soldier. Therefore, the requests of the battleship were rejected, because the battleship is impossible to produce the ESK under the name prefix of the soldier.
- **Scalability:** To establish access control policies, EKAC leverages the NDN naming convention, and the basic format of NDN *data packet* to achieve its scalability for the policy key. For example, the soldier in squad A can access the content from the UAV A. To enable access control, the command center only determines a name, e.g., “/military/CommandCenter/EKAC/military/UAVA/KP/<key-id>/AUTHORIZED/military/SquadA/Soldier1/KEY/<key-id1>”, and signs the KP with its own private key, without defining the content policy keys in KP. Therefore, in case of the number of granularities or the number of consumer increases, there is no scalability issues of the policy key generator.
- **Suitable for Intermittent Connection:** The management of access control policies in EKAC supports the suitability to intermittent connection, where the KP is a normal NDN *data packet*. Therefore, in Step 4 of the EKAC schemes, the command center can distribute this access control policy within the NDN network. With this reason, as long as there is a copy for the KP in the network, all entities in the system can always perform an operation even with intermittent connectivity. In addition, EKAC is strong against the intermittent communication, where

the encrypted contents can be cached into the network and accessed by the authorized consumer in the future.

- Preface Forward Secrecy (PFS):** EKAC scheme is efficient to prevent the attacker to access the previously published content, where the EK shares between the producer and the consumer has PFS properties. For our example, the consumer (in the step 11 of EKAC schemes) generates its own DH_C and defines the ESK for each communication session. In Step 17, the producer generates its own DH_P , and then computes the EK from the two DH public keys (the other one is contained in the corresponding ESK). In Step 18, the EK will then be used to encrypt the CK. In Step 26, the consumer obtains the encrypted content. The consumer then uses its own DH_C to compute the EK with DH_P of the producer in EAK, and use EK for decrypting the encrypted CK. Whenever a new request arrives at the producer, it fetches the ESK from the network, where it can learn the name of the ESK from the KP name. Therefore, all CKs are encrypted with the fresh EK. This allows the encrypted contents to be protected using keys, derived from the shared ephemeral keys. The encrypted contents can protect the damage, even if the consumer's device is compromised in the future.
- Revocation:** In EKAC, KP is the policy keys that represented in a normal NDN *data packet*. Therefore, the command center can distribute these contents within the NDN network by defining the expiration of policy keys to periodically update the KP. If an entity is compromised, the command center should not generate a renewed access privilege to the entity. In addition, the ESK and the EAK must restart the process by generating a fresh ephemeral DH public key and transmitting that key between the consumer and the producer in the system. The keys can define the expiration to revoke the policies and periodically update. Moreover, EKAC scheme can also remove the encrypted contents in-network caches, and revoke the KP. The command center can use a *signed interest* to carry the revoked command in a name. Likewise, the producer can revoke access to publish the encrypted content in the network by sending a *signed interest* that carries the revoked command to in-network caches. The contents must be re-encrypted to replace all encrypted contents (created previously) by using a new KP, CK and EK to prevent the sensitive information. So, EKAC can enable immediate revocation of the consumer, and also the revoked entity cannot access the new encrypted content.

CHAPTER 5

PERFORMANCE EVALUATION

5.1 Introduction

In the previous chapter, EKAC: Ephemeral Key-based Access Control scheme has been proposed. In this chapter, we present parameters of performance evaluation techniques in a testbed to compare the EKAC with NAC and NAC-ABE. We evaluate the performance in terms of cryptographic operations and packet size to consider a period of time to receive the encrypted content. We can illustrate the time in the consumer revocation phase of EKAC.

5.2 Experimental Setup

5.2.1 Experimental Parameters

The following hardware is deployed for our experiment: Intel (R) Core i5-4260U CPU @ 1.4 GHz, 4GB of RAM, and the operating system is Ubuntu 18.04. We use 100Mbps on the Network Interface Card (NIC). This node has installed the EKAC, NAC [32] and NAC-ABE [78] application testbed. Our experimental parameters are illustrated as follows:

- **Cryptography Algorithms:** For the Content Key (CK) exchange, EKAC uses the Diffie–Hellman (DH) key agreement to compute 256 bits of ephemeral keys that would be used to encrypt/decrypt CK with AES (Advanced Encryption Standard) algorithm. This is different from NAC that uses RSA, and NAC-ABE that uses CP-ABE to encrypt/decrypt CK. The implementation of EKAC uses the Crypto++ library version 8.3 to compute the ephemeral keys, based on the large prime number (2048 bits) for DH key agreement. NAC uses the NDN-CXX version 0.7.0 library for RSA (at 2048 bit key size) to encrypt/decrypt the CK, and NAC-ABE uses the PBC library [81] to compute elliptic curve group (at 160 bits) to setup the parameters in CP-ABE. For other security mechanisms, all schemes use the same NDN-CXX version 0.7.0 library to operate the security mechanism, including the ECDSA of the key size 256 bits to perform the digital signature. The SHA-256 hash functions are used to compute a 256-bit hash value, and the symmetric of the 256-bit key size is deployed for the AES.
- **Component Name:** To evaluate the performance of packet size. We define the experimental parameters for the name of the access manager as “/military/CommandCenter”, the name of granularity as “/military/UAVA”, the name of the *data packet* as “/military/UAVA/info”, and the name of consumer as “/military/SquadA/Soldier1”. For NAC-ABE we use the same component name, but differ in the set of attributes. We define three attributes, (“SquadA” AND “Soldier”) or “Battleship”, to encrypt the CK.
- **Plaintext Size:** We define the content as a plaintext of 1024 bits.

- **NDN Data Packets:** The size of all the imperative *data packets* depends on each application testbed. We only define the lifetime of all *data packets* to be 10,000 millisecond (ms) to cache these *data packets* with 10 seconds in-network cache.

5.2.2 Size of Data Packet Transmission

We assume the consumer requests the encrypted content x one time from the producer. Both NAC and NAC-ABE use the policy key pairs (KEK and KDK), and the content key (CK), as described in Section 2.15. The size of the essential *data packet* depends on the process of each application testbed, as illustrated the experimental results in Table 4.

Table 4 Comparison of *data packet* sizes

Scheme	Role	Data Type	Data Size (bit)	Total (bit)
NAC	Access Manager	KEK	491	$2747 + 1227x$
		KDK	1741	
	Producer	CK	515	
		Encrypted Content	1227	
NAC-ABE	Access Manager	KEK	216	$3141 + 1227x$
		KDK	1818	
	Producer	CK	1107	
		Encrypted Content	1227	
EKAC	Access Manager	KP	247	$1302 + 1227x$
		EAK	604	
	Producer	Encrypted Content	1227	
		Consumer	ESK	

5.2.3 Access Revocation Cost

In this section, we illustrate experimental parameters, related with the access revocation as follows:

- **Network Scenario:** We use the Common Open Research Emulator (CORE) [79] to evaluate the performance of EKAC's access revocation mechanism. All entities have installed the NFD version 0.6.6 for a network forwarding, and use the physical layer as IEEE 802.11b. We use the network architecture in the Mobile Ad-Hoc Network (MANET) that composes the infrastructureless wireless networks with mobile devices. We have to configure the local Network Interface Card (NIC) of all nodes to forward the *data packet* based on forwarding strategy in NDN network, and defining the transmission range is 250 meters.
- **NDN Data Packets:** The size of *data packets* is 1227 bits. We only define the lifetime of all *data packets* to be 10,000 milliseconds (ms) to cache these *data packets* with 10 seconds in-network cache.
- **Number of Nodes:** The number of router nodes, consisting of 5, 10, 15, 20 nodes.

Our evaluation uses a sender and different number of router nodes, consisting of 5, 10, 15, 20 nodes to delete 100 *data packets*. We run the experiment on the testbed for 30 times. The results of EKAC access revocation cost are represented with 95% confident interval, as shown in Figure 16.

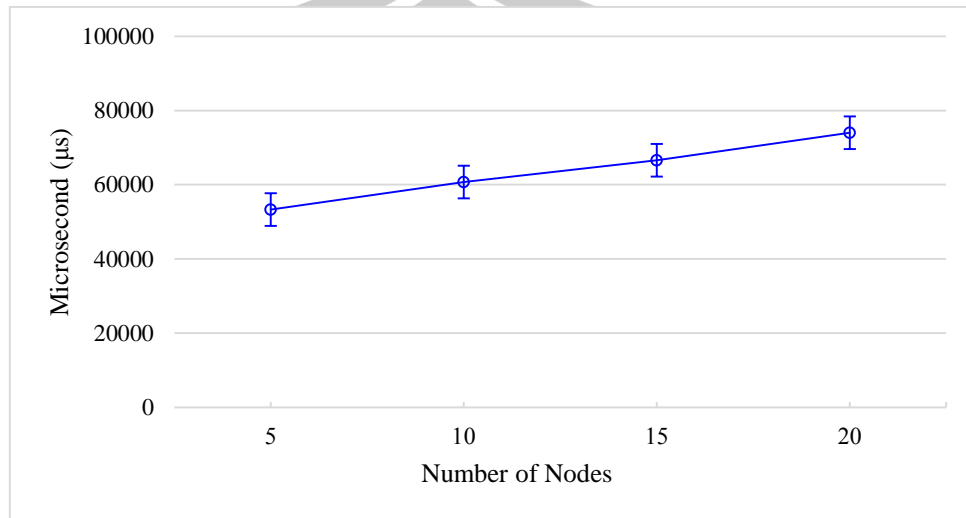


Figure 16 Access revocation cost

5.2.4 Cryptographic Operations

We use the experimental parameters to evaluate the cryptographic operations. We define a Content Key (CK) in encryption for m granularity in our experiments, and the Ephemeral Key (EK) is an ephemeral key to encrypt the CK in the EKAC scheme. The function process returns a time in microseconds (μ s). The notations used for different cryptographic operations, and the result for actual running time of different operations are shown in Table 5. The experiment results under our operations are equivalent to the experiment of the literature [82-84].

Table 5 The result for actual running time of different operations

Notations	Description	Execution Time (μ s)
T_{RSAGen}	Time of RSA Generation	85041.50 \pm 989.59
T_{AttGen}	Time of Attribute Generation	25515.60 \pm 67.57
T_{DHGen}	Time of Generation DH Public Key	2140.13 \pm 20.42
T_{EK}	Time of Computation an EK	2139.00 \pm 20.16
T_{RSAEnc}	Time of Encryption a CK with RSA	70.07 \pm 2.49
T_{RSADec}	Time of Decryption a CK with RSA	2067.73 \pm 18.87
T_{AttEnc}	Time of Encryption a CK with CP-ABE	18156.90 \pm 79.12
T_{AttDec}	Time of Decryption a CK with CP-ABE	7173.37 \pm 48.40
T_{EKEnc}	Time of Encryption a CK with EK	16.93 \pm 0.59
T_{EKDec}	Time of Decryption a CK with EK	9.93 \pm 0.26
T_{se}	Time of Encryption a Plaintext with AES	11.67 \pm 0.26
T_{sd}	Time of Decryption a Plaintext with AES	11.17 \pm 0.59
T_{gs}	Time of Generation Signature with ECDSA	82.83 \pm 2.76
T_{vs}	Time of Verification Signature with ECDSA	174.27 \pm 3.28

5.3 Evaluation

In this section, our evaluation aims to propose an analysis of the computation and communication cost. We assume the system has n consumers, m granularities, a attributes, and x data packets. We compare the solution to achieve the Perfect Forward Secrecy (PFS) properties of the EKAC scheme against two solutions in the NAC and NAC-ABE schemes. The first solution has been proposed by Yu et al. [8] that uses the ephemeral key to protect the access keys. The second solution has been proposed by Zhang et al. [12] is using the short-lived policy keys. NK is the number of new policy keys. Both NAC and NAC-ABE use the policy key pairs (KEK and KDK), as described in Section 2.15.

5.3.1 Computation Cost

We define N as a number of operations that the access manager will use to establish the policy keys. The operations of the access manager, proposed in [8], can be calculated as:

$$N_{\text{NAC}} = (T_{\text{DHGen}} + T_{\text{EK}} + T_{\text{EKEnc}} + T_{\text{se}} + T_{\text{RSAGen}} + 2T_{\text{gs}})(mn) \quad (1)$$

$$N_{\text{NAC-ABE}} = (T_{\text{DHGen}} + T_{\text{EK}} + T_{\text{EKEnc}} + T_{\text{se}} + T_{\text{AttGen}} + 2T_{\text{gs}})(an) \quad (2)$$

A number of operations that the access manager will use to establish the policy keys, proposed in [12], can be calculated as:

$$N_{\text{NAC}} = NK \cdot [(T_{\text{RSAGen}} + T_{\text{RSAEnc}} + T_{\text{se}} + 2T_{\text{gs}})(mn)] \quad (3)$$

$$N_{\text{NAC-ABE}} = NK \cdot [(T_{\text{AttGen}} + T_{\text{RSAEnc}} + T_{\text{se}} + 2T_{\text{gs}})(an)] \quad (4)$$

For EKAC, the policy keys can be established by leveraging the NDN naming convention and represented as a normal NDN data packet. The operations of the access manager can be calculated as:

$$N_{\text{EKAC}} = T_{\text{gs}}(mn) \quad (5)$$

We define NP as a number of operations of the producer to produce the encrypted content. Therefore, the operations of the producer, proposed in [8], can be calculated as:

$$NP_{\text{NAC}} = T_{\text{vs}} + T_{\text{RSAEnc}} + T_{\text{gs}} + (T_{\text{gs}} + T_{\text{se}})x \quad (6)$$

$$NP_{\text{NAC-ABE}} = T_{\text{vs}} + T_{\text{AttEnc}} + T_{\text{gs}} + (T_{\text{gs}} + T_{\text{se}})x \quad (7)$$

The computation cost of the producer, proposed in [12], can be calculated as:

$$NP_{\text{NAC}} = NK \cdot [T_{\text{vs}} + T_{\text{RSAEnc}} + T_{\text{gs}} + (T_{\text{gs}} + T_{\text{se}})x] \quad (8)$$

$$NP_{\text{NAC-ABE}} = NK \cdot [T_{\text{vs}} + T_{\text{AttEnc}} + T_{\text{gs}} + (T_{\text{gs}} + T_{\text{se}})x] \quad (9)$$

For EKAC, the encrypted contents can be produced once in a day. The operation of a producer can be calculated as:

$$NP_{EKAC} = 2T_{vs} + T_{DHGen} + T_{EK} + T_{EKEnc} + T_{gs} + (T_{gs} + T_{se})x \quad (10)$$

We define NC as a number of operations that the consumer to decrypt the encrypted content. The operations of the consumer, proposed in [8], can be calculated as:

$$NC_{NAC} = 2T_{vs} + T_{DHGen} + T_{EK} + T_{EKDec} + T_{RSADec} + T_{sd} + (T_{vs} + T_{sd})x \quad (11)$$

$$NC_{NAC-ABE} = 2T_{vs} + T_{DHGen} + T_{EK} + T_{EKDec} + T_{AttDec} + T_{sd} + (T_{vs} + T_{sd})x \quad (12)$$

The computation cost of the consumer, proposed in [12], can be calculated as:

$$NC_{NAC} = NK \cdot [2T_{vs} + 2T_{RSADec} + T_{sd} + (T_{vs} + T_{sd})x] \quad (13)$$

$$NC_{NAC-ABE} = NK \cdot [2T_{vs} + T_{RSADec} + T_{AttDec} + T_{sd} + (T_{vs} + T_{sd})x] \quad (14)$$

For EKAC, the operations of consumers to decrypt the encrypted contents can be calculated as:

$$NC_{EKAC} = T_{DHGen} + T_{vs} + T_{EK} + T_{EKDec} + (T_{vs} + T_{sd})x \quad (15)$$

To summarize the computation cost, we use the experimental results of system overhead in Table 5. The summary of computation cost of each solution is illustrated in Table 6.

Table 6 Summary of computation cost

Scheme		Execution Time (μ s)
Yu et al. [8]	NAC	$96557+279x$
	NAC-ABE	$60224+279x$
Zhang et al. [12]	NAC	$90110+279x$
	NAC-ABE	$53777+279x$
Our		$9272+279x$

5.3.2 Communication Cost

In this section, our evaluation aims to propose an analysis of the communication cost. We define N as a number of communication cost that the access manager will use to establish the policy keys. The communication cost of the access manager, proposed in [8], can be calculated as:

$$N_{NAC} = 2n + m \quad (16)$$

$$N_{NAC-ABE} = 2n + m \quad (17)$$

The communication cost of the access manager, proposed in [12], can be calculated as:

$$N_{\text{NAC}} = NK(n + m) \quad (18)$$

$$N_{\text{NAC-ABE}} = NK(n + m) \quad (19)$$

For EKAC, the communication cost of the access manager can be calculated as:

$$N_{\text{EKAC}} = m \quad (20)$$

We define NP as a number of operations that the producer to receive policy keys. Therefore, the communication cost of the producer, proposed in [8], can be calculated as:

$$NP_{\text{NAC}} = m \quad (21)$$

$$NP_{\text{NAC-ABE}} = m \quad (22)$$

The communication cost of the producer, proposed in [12], can be calculated as:

$$NP_{\text{NAC}} = NK \cdot m \quad (23)$$

$$NP_{\text{NAC-ABE}} = NK \cdot m \quad (24)$$

For EKAC, the communication cost of the producer can be calculated as:

$$NP_{\text{EKAC}} = m + n \quad (25)$$

We define NC as a number of operations for the consumer to access the encrypted content. Therefore, the communication cost of the consumer, proposed in [8], can be calculated as:

$$NC_{\text{NAC}} = m + 2n + x \quad (26)$$

$$NC_{\text{NAC-ABE}} = m + 2n + x \quad (27)$$

The communication cost of the consumer, proposed in [12], can be calculated as:

$$NC_{\text{NAC}} = NK(m + n + x) \quad (28)$$

$$NC_{\text{NAC-ABE}} = NK(m + n + x) \quad (29)$$

For EKAC, the communication cost of the consumer can be calculated as:

$$NC_{\text{EKAC}} = m + x \quad (30)$$

7. The summary of communication cost of each solution is illustrated in Table

Table 7 Summary of communication cost

Scheme		Total
Yu et al. [8]	NAC	$4n + 3m + x$
	NAC-ABE	$4n + 3m + x$
Zhang et al. [12]	NAC	$(2n + 3m + x) \cdot NK$
	NAC-ABE	$(2n + 3m + x) \cdot NK$
Our		$n + 3m + x$

5.3.3 Revocation Complexity

To analyze the revocation complexity of EKAC against lazy revocation [8, 12] and immediate revocation [14]. We assume all mechanisms have already obtained the revocation command from the access manager. We define that the system has n authorized consumers, x data packets. R is the number of revocation commands, and N is the number of intermediate nodes (e.g., proxy servers or NDN routers). The revocation complexity of the EKAC, lazy revocation and immediate revocation are shown in Table 8.

Table 8 Revocation complexity

Revocation Mechanism	Computation Burden			Communication Overhead
	Producer	Intermediate Node	Consumer	
Lazy revocation [8, 12]	$3x \cdot n$	x	x	0
Immediate revocation [14]	$3x + R$	$[(3x^2) \cdot nx] + R$	$2x$	NR
EKAC	$(3x \cdot n) + R$	$x + R$	$x + 1$	NR

5.4 Security Assessment

- **Impersonation Attack:** In the EKAC, an attacker cannot request the encrypted content as the authorized consumer. For our example, the authorized soldier establishes the ephemeral share keys (as known as ESK). To enable the secure access control, the ESK data packet will be signed by the soldier's private key. Therefore, the attacker cannot request to the resource of UAV A using the access privileges of the soldier. The UAV A will validate the signatures of ESK to ensure the authenticity of the authorized soldier.
- **Man-In-The-Middle (MITM) Attack:** EKAC performs the secure communication, based on NDN model. Therefore, by data authentication based on the NDN trust model, the attackers may try to perform MITM attack and modify some data packets (e.g., KP). However, all entities can check the illicit changes of all data packets by verifying the signature.

- **Name Confidentiality:** The design of EKAC leverages naming convention to establish fine-grained access control. For our example, a KP name “/military/CommandCenter/EKAC/military/UAVA/KP/<key-id>/AUTHORIZED/military/SquadA/Soldier1/KEY/<key-id1>”, which conveys that the encrypted content might be relative to the soldier and produced by the UAV A. However, the naming convention may disclose a secret information to a certain extent. To solve this problem, we suggest that the system can conceal the name of *data packet* by using the technique, as apparent by some papers [85, 86]. Yet, it is not in the scope of this dissertation.

5.5 Discussion

EKAC scheme is established on an ephemeral key to enforce the content access control to achieve the threat mitigation from device compromise with PFS properties. We combine the NDN naming convention to be applied in practice to create an access control scheme able to provide fine-grained access control. With considering to ephemeral key shares, the EKAC scheme is suitable for intermittent connection.

In this section, we discuss the properties of EKAC, comparing against existing access control schemes. We illustrate that EKAC can achieve the goals as follows.

5.5.1 Efficiency for Threat Mitigation from Consumer Compromise

The properties of EKAC has illustrated that efficient to prevent the attacker to access the previously published content. In contrast, the previous solutions are not fully adhering in the PFS properties when the private key is compromised. Yu et al. [8] have suggested the ephemeral key usage to protect the access keys (as known as KDK). This solution requires the access manager to be always online to set the access control policie. Furthermore, it cannot prevent the attacker from accessing of all the contents that the consumer had accessed before when the consumer device is compromised.

The access control scheme proposed in [5, 11, 12] only can reduce the content leakage if the consumer device is compromised by using short-lived policy key pairs. Yet, their schemes are inefficient to prevent the attacker to access the previously published content. For our example, the policy key would be new established in every day. With short-lived of the policy key pairs, the vulnerability is still there until one day passes.

5.5.2 Performance and Resource Consumption

The size of *data packet* transmission is defined as the size of *data packet* transmission of each application testbed to distribute the encrypted content x , as shown in Table 4. EKAC use only one policy key (KP), which has no content, and contains only a small value of ESK and EAK to enable the access control. With this reason, the size of *data packet* transmission of EKAC is smaller than NAC of 52.6%, and NAC-ABE of 58.55%.

From Table 6, the evaluation indicates that NAC and NAC-ABE can cause higher overhead, if they want to enable PFS properties by using the solution of Yu et al. [8]. From the evaluation, overall cryptographic operation of EKAC is less than NAC-BE by 84.6% and less than the NAC by 90.4% if using the proposed solution of Zhang et al. [12]. To have the short-lived policy keys, the computational cost of all

entities is increased. The operations need NK multiplication to create new policy keys. For our example, the system of Zhang et al. [12] must use 24 policy key per-hour in one day. An overall comparison cryptographic operation of EKAC is less than the NAC-ABE in this case by 99.26% and less than the NAC in this case by 99.56%. In some case, the policy key may be renewed six hours in one day to reduce the computational cost. However, an overall comparison cryptographic operation of EKAC is still less than the NAC-ABE by 95.58% and less than the NAC by 97.36%.

The communication cost is defined as the system uses to distribute the encrypted content x to n authorized consumers. From Table 7, EKAC has lower communication cost, compared to the proposed solution of Yu et al. [8], which an overall communication cost of EKAC is less than the NAC and NAC-ABE of 37.5%. Compared to the proposed solution of Zhang et al. [12], the operation needs NK multiplication. For our example, the solution of Zhang et al. [12] needs to explicitly use the policy key per-hour in one day. An overall communication cost of EKAC is less than the NAC and NAC-ABE for 96.53%. In some cases, the policy key may be renewed six hours in one day to reduce the communication cost. However, an overall communication cost of EKAC is still less than the NAC and NAC-ABE for 79.17%.

From the above discussion, it becomes clear that EKAC has a low computation and communication cost, and also provides stronger security than having to wait for a common expiry of short-lived of the policy keys, which could be until an hour or six hours.

5.5.3 Suitability for Intermittent Connectivity

From the experiments, EKAC has demonstrated the suitability for intermittent connectivity. The producer can use the policy keys as long as they are not expired or the granularities are not changed. The contents can protect the damage of private key compromise because the Content Key (CK) is protected with the fresh Ephemeral Key (EK).

In contrast, Yu et al. [8] have proposed to utilize ephemeral keys to encrypt the access keys (KDK). However, this solution requires the access manager to be always online to set the access control policies. So, it is not suitable for the intermittent connectivity. Zhang et al. [12] (also the schemes in [5, 11]) have proposed to use the short-lived policy keys to mitigate the vulnerability of content leakage, when the private key of the consumer is compromised. However, the short-lived policy keys cannot completely provide the real PFS. We have also evaluated the overhead of the short-lived policy keys and have found a significant negative impact of the computation and communication cost, and not suitable for intermittent connection. In fact, if the communications are faced with intermittent connectivity, e.g., the battlefield scenario. The command center can notify and re-generate the key policies when the private key of consumer is compromised. Yet, the UAV A and the soldier can be outside the command center network, and they may not be able to acquire the key policies if a communication is disconnected.

5.5.4 Effective Consumer Revocation

By EKAC scheme, the consumer revocation mechanism is effective to revoke the policy keys and the encrypted contents. Policy keys can define the expiration to revoke the consumers and periodically update. Moreover, EKAC scheme can also remove the encrypted contents, and remove the policy keys in-network caches. The

command center can use a *signed interest* to carry the revoked command in a name. Likewise, the producer can revoke the access to the published encrypted content in the network by sending a *signed interest* that carries the revoked command to in-network caches. The contents will then be re-encrypted to replace all encrypted contents, created previously by using new KP, CK and EK to prevent the sensitive information. So, the revoked consumer cannot access the new encrypted content. The experimental results of the cost of revocation of access are shown in Figure 16. It shows that the EKAC access revocation is possible in the communication, based on MANET, e.g., the battlefield scenario.

Upon comparing with the existing access control schemes, EKAC is effective to enable the immediate revocation of the consumer. The revoked consumer cannot access all contents previously or forwardly published by using the existing access key. In addition, EKAC has the overall computation burden less than the existing immediate revocation mechanisms as shown in Table 7.

5.5.5 EKAC with Better Scalability

The EKAC scheme leverages the NDN naming convention, and basic format of the NDN *data packet* to establish one policy key, while [5, 8, 11, 12] use asymmetric cryptography algorithms to establish the policy key pairs. With this reason, their schemes have a significant negative impact of the computation cost. Our evaluation has illustrated a significant negative impact of the computation cost if the short-lived key solution is used. The computation cost can occur in the single point when the new policy key must be established.

5.5.6 Effective Access Control and Robust Privacy

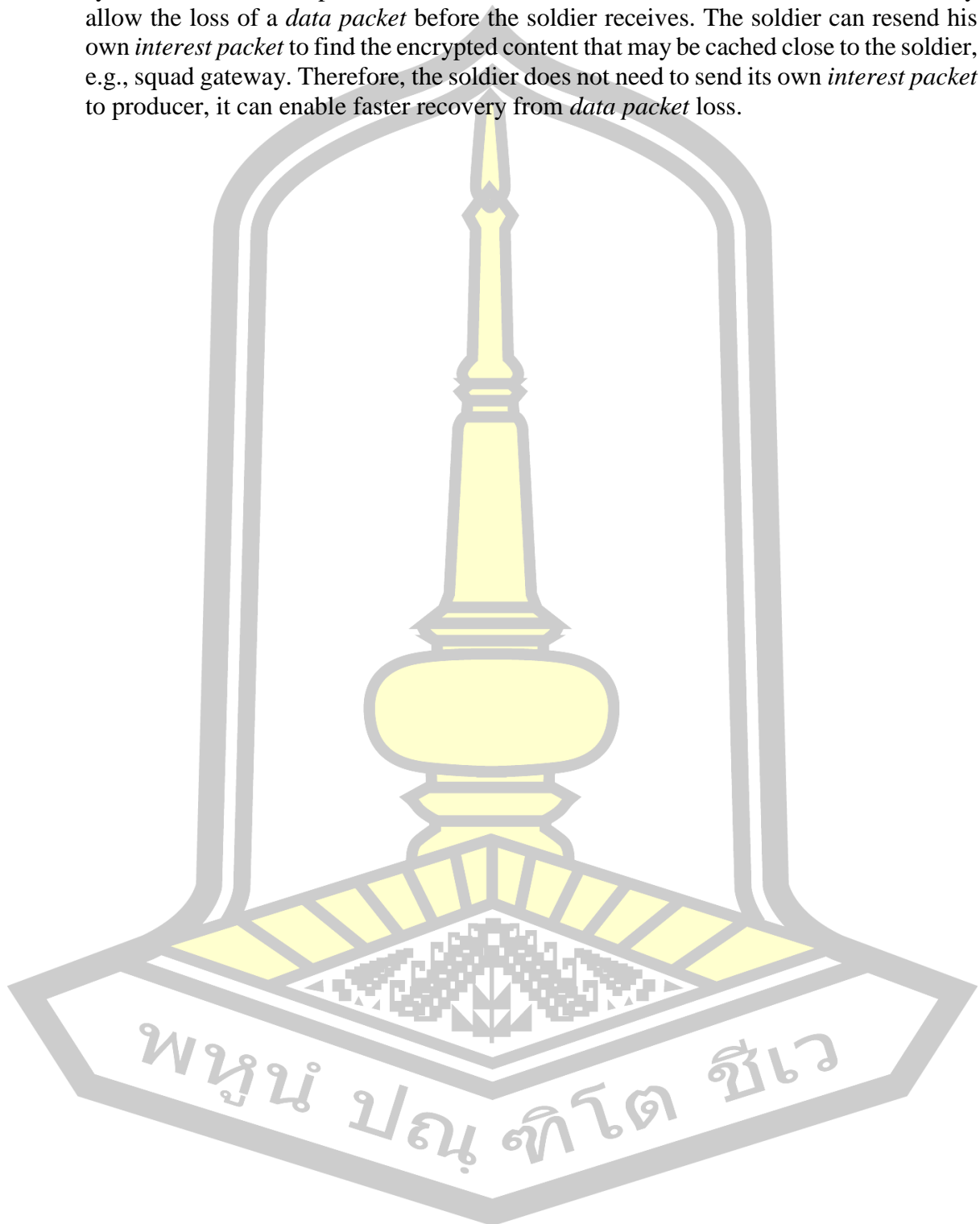
In our experiments, we define the cryptography algorithms according to the instructions of the National Institute of Standards and Technology (NIST) [87] that specifies the suitable key sizes. The EKAC uses the 2048 bits for DH key agreement, which is robust as same as 2048 bits for RSA, used in NAC. For NAC-ABE, the key size of default application support is lower than the guidance of NIST. This purpose can illustrate that EKAC is robust privacy, and provides effective access control. EKAC provides robust security with PFS properties to prevent a device compromise. In addition, as shown in Table 4. EKAC has the overall size of *data packet* transmission less than the NAC and NAC-ABE.

5.5.7 Efficient of NDN Architecture Usage

The EKAC scheme requires the producer to encrypt and sign individually for each consumer. Therefore, the communication may not be completely benefited from the NDN architecture, e.g., caching. However, compared to previous work [3-14], EKAC provides a secured content, which the sensitive data is protected by shared ephemeral key, and the access control policy mechanism that exerts suitable for intermittent connection, e.g., battlefield.

We can argue that such the efficiency of NDN architecture usage of the EKAC in two scenarios. First, the policy key management is suitable for intermittent connectivity, the KP can cache in the NDN network as long as it is not expired and the granularities are not changed. Second, although the encrypted content must be encrypted by the producer and signed individually for each consumer, EKAC can be benefited from the NDN architecture in the efficient recover an encrypted content from

losses [88, 89], which suitable than the classical Internet to establish the access control system. For our example, the wireless communications of battlefield scenarios may allow the loss of a *data packet* before the soldier receives. The soldier can resend his own *interest packet* to find the encrypted content that may be cached close to the soldier, e.g., squad gateway. Therefore, the soldier does not need to send its own *interest packet* to producer, it can enable faster recovery from *data packet* loss.



CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Summary

Access control is very significant for security issues. However, it is a challenging issue for NDN, due to the contents can be cached in several nodes. So, the access control scheme cannot rely on securing channels or hosts, but the scheme must move to securing the contents instead. So, several previous studies have proposed several access control schemes [3-14] by encrypting the contents for authorized consumers. The schemes are called encryption-based access control. NAC and NAC-ABE are two of states of the arts in the field. They have proposed to handling cryptographic keys by using naming system of NDN. Furthermore, several studies [61-64] have proposed several solutions to the challenges in NDN access control. However, all of previous works have not solve several problems, such as perfect forward secrecy issues, effective revocation, efficiency of operations in terms of communication and computation. For some scenarios (such as a battlefield scenario), perfect forward secrecy is very crucial to ensure that the past sensitive data will not be compromised even when an end device is hacked. This PFS problem has not been solved by the previous work. Only some initial suggestions [5, 8, 11, 12], have been made. In addition, access control for NDN over the intermittent network (especially, battlefield scenario) would be more difficult to manage. The connection of an access manager, producers and consumers could be interrupted periodically for such a connection. The access manager may be connected via a satellite link, that could be most interrupted. So, the previous proposed schemes that require the access manager or producers which are always-online could be failed.

So, this PhD dissertation has evaluated the NDN access control problems, particularly for the battlefield scenario. The previous proposals have been analyzed and point out several drawbacks. After that, this dissertation has proposed a new access control scheme, named EKAC: Ephemeral Key-based Access Control. EKAC aims at providing PFS, immediate revocation and suitable to intermittent network like the battlefield scenario. We have prototyped EKAC for experiments over a network testbed and CORE network emulator. The performance evaluation has been done using network testbed and emulation techniques as well as computation and communication analysis, by comparing with NAC [32], NAC-ABE [78] and other previous works [8, 12, 14]. The evaluation results have shown that EKAC can provide PFS, immediate revocation with lower communication and computation overheads. The size of *data packet* transmission of EKAC is smaller than NAC for 52.6%, and smaller than NAC-ABE for 58.55%. An overall cryptographic operation of EKAC is less than NAC-ABE around 84.6% and less than NAC around 90.4% for the proposed solution of Yu et al. [8]. For the communication cost, EKAC can perform better than the proposed solution of Yu et al. [8] with less overhead around 37.5%.

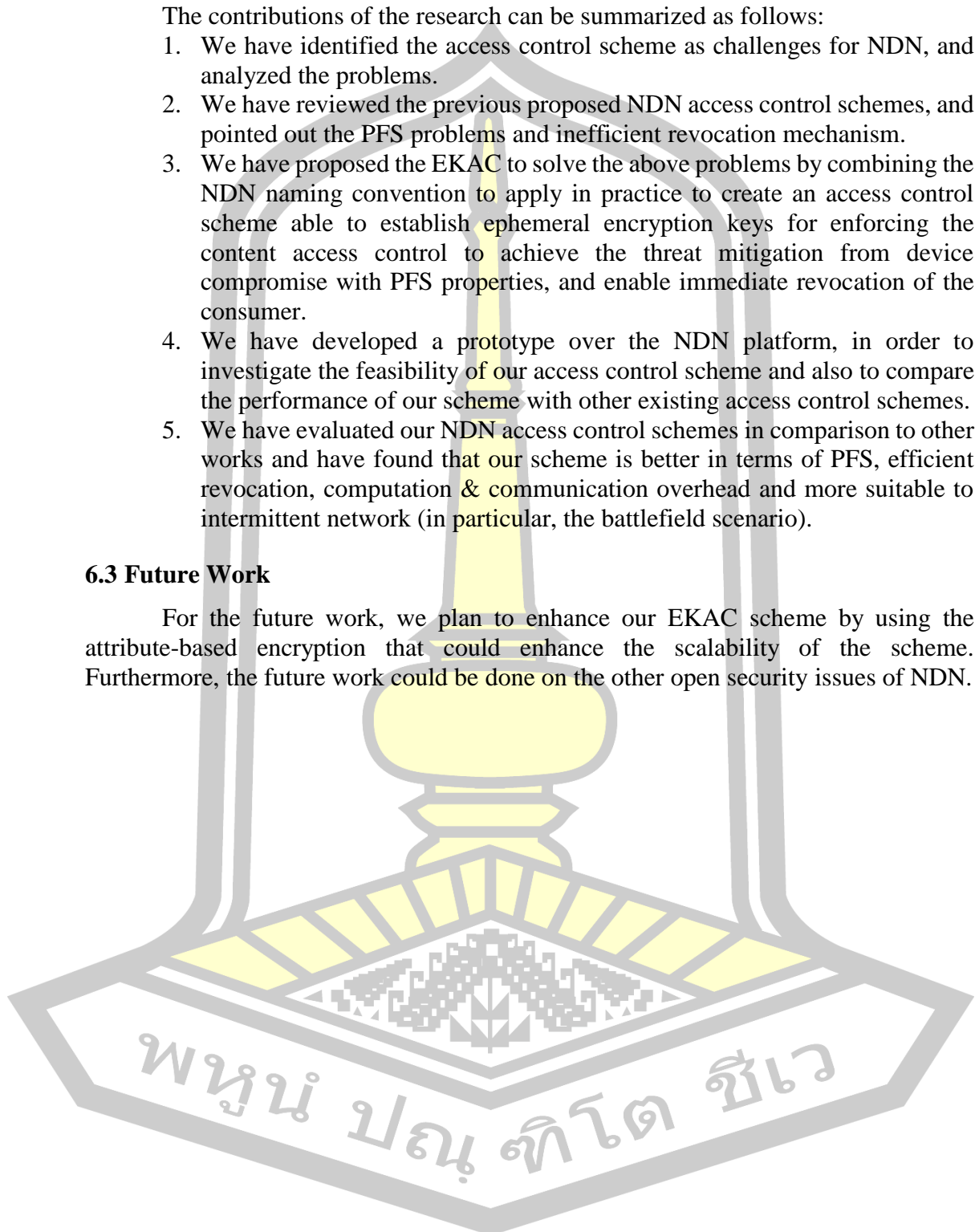
6.2 Dissertation Achievement

The contributions of the research can be summarized as follows:

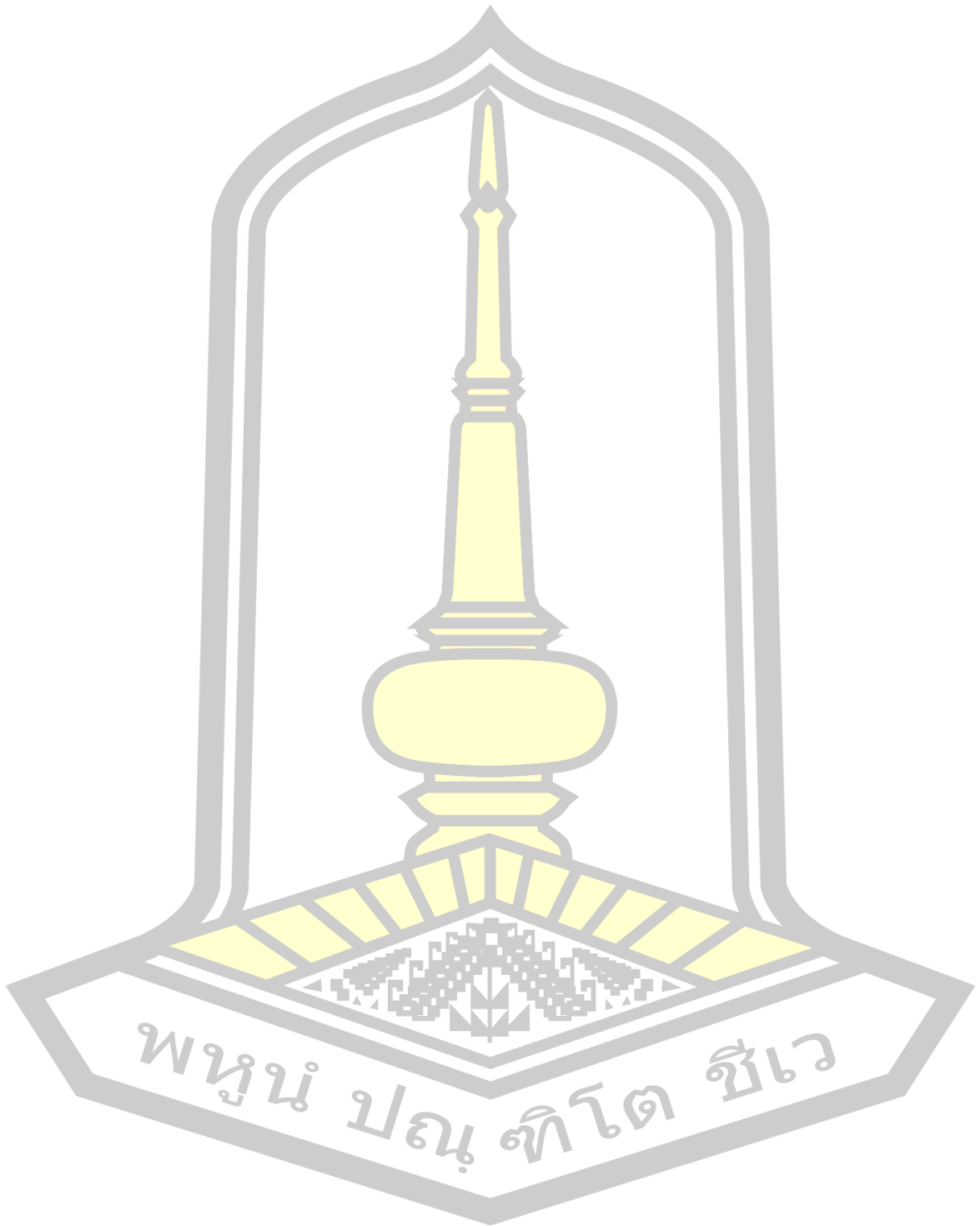
1. We have identified the access control scheme as challenges for NDN, and analyzed the problems.
2. We have reviewed the previous proposed NDN access control schemes, and pointed out the PFS problems and inefficient revocation mechanism.
3. We have proposed the EKAC to solve the above problems by combining the NDN naming convention to apply in practice to create an access control scheme able to establish ephemeral encryption keys for enforcing the content access control to achieve the threat mitigation from device compromise with PFS properties, and enable immediate revocation of the consumer.
4. We have developed a prototype over the NDN platform, in order to investigate the feasibility of our access control scheme and also to compare the performance of our scheme with other existing access control schemes.
5. We have evaluated our NDN access control schemes in comparison to other works and have found that our scheme is better in terms of PFS, efficient revocation, computation & communication overhead and more suitable to intermittent network (in particular, the battlefield scenario).

6.3 Future Work

For the future work, we plan to enhance our EKAC scheme by using the attribute-based encryption that could enhance the scalability of the scheme. Furthermore, the future work could be done on the other open security issues of NDN.



REFERENCES



REFERENCES

- [1] D. K. Smetters and V. Jacobson, "Securing Network Content," PARC Technical Report, 2009.
- [2] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, *et al.*, "Named Data Networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, pp. 66–73, July 2014.
- [3] B. Hamdane, A. Serhrouchni, and S. Fatmi, "Access control enforcement in named data networking," in *Proceedings of International Conference for Internet Technology and Secured Transactions (ICITST)*, 2013, pp. 576–581.
- [4] T. Chen, K. Lei, and K. Xu, "An Encryption and Probability based Access Control Model for Named Data Networking," in *Proceedings of the 2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC)*, Austin, 2014, pp. 1-8.
- [5] M. Mangili, F. Martignon, and S. Paraboschi, "A cache-aware mechanism to enforce confidentiality, trackability and access policy evolution in content-centric networks," *Computer Networks*, vol. 76, pp. 126-145, 2015.
- [6] R. Silva and S. Zorzo, "An access control mechanism to ensure privacy in named data networking using attribute-based encryption with immediate revocation of privileges," in *Proceedings of the 2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, 2015, pp. 128-133.
- [7] J. Kurihara, E. Uzun, and C. Wood, "An Encryption-Based Access Control Framework for Content-Centric Networking," in *Proceedings of the IFIP Networking* Toulouse, France, 2015.
- [8] Y. Yu, A. Afanasyev, and L. Zhang, "Name-Based Access Control," Technical Report NDN-0034 Revision 2, January 2016.
- [9] W. Shang, Y. Yu, T. Liang, B. Zhang, and L. Zhang, "NDN-ACE: Access Control for Constrained Environments over Named Data Networking," Technical Report NDN-0036, December 2015.
- [10] S. Misra, R. Tourani, F. Natividad, T. Mick, N. E. Majd, and H. Huang, "AccConF: An Access Control Framework for Leveraging In-Network Cached Data in the ICN-Enabled Wireless Edge," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, pp. 5-17, 2017.
- [11] T. Feng and J. Guo, "A New Access Control System Based on CP-ABE in Named Data Networking," *Journal of International Journal of Network Security*, vol. Vol. 20, pp. 710-720, July 2018.
- [12] Z. Zhang, Y. Yu, S. Ramani, A. Afanasyev, and L. Zhang, "NAC: Automating Access Control via Named Data," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*, Los Angeles, CA, 2018, pp. 626-633.
- [13] Z. Wu, E. Xu, L. Liu, and M. Yue, "CHTDS: A CP-ABE Access Control Scheme Based on Hash Table and Data Segmentation in NDN," in *Proceedings of 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, Rotorua, New Zealand, 2019, pp. 843-848.

- [14] Z. Wu, Y. Zhang, and E. Xu, "Multi-Authority Revocable Access Control Method Based on CP-ABE in NDN," *Future Internet*, vol. 12, pp. 1-13, 2020.
- [15] H. Zhang, Z. Wang, C. Scherb, C. Marxer, J. Burke, and L. Zhang, "Sharing mHealth Data via Named Data Networking," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, 2016, pp. pp. 142-147.
- [16] J. Burke, A. Afanasyev, T. Refaei, and L. Zhang, "NDN Impact on Tactical Application Development," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*, Los Angeles, CA, 2018, pp. 640-646.
- [17] T. Refaei and A. Afanasyev, "Enabling a Data-Centric Battlefield Through Information Access Gateways," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*, Los Angeles, CA, 2018, pp. 634-639.
- [18] R. Tourani, R. Stubbs, and S. Misra, "TACTIC: Tag-Based Access Control Framework for the Information-Centric Wireless Edge Networks," in *Proceedings of the IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, Vienna, 2018, pp. 456-466.
- [19] N. Leshov, M. A. Yaqub, M. T. R. Khan, S. Lee, and D. Kim, "Content Name Privacy in Tactical Named Data Networking," in *Proceedings of the Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, Zagreb, Croatia, 2019, pp. 570-572.
- [20] V. Hemanathan and N. Anusha, "Role Based Content Access Control in NDN," *Journal of Innovative Technology and Education*, vol. 2, pp. 143 - 152, 2015.
- [21] S. Singh, "A trust based approach for secure access control in information centric network," *International Journal of Information and Network Security (IJINS)*, vol. 2, pp. 97-104, 2012.
- [22] C. Ghali, M. Schlosberg, G. Tsudik, and C. Wood, "Interest-Based Access Control for Content Centric Networks," in *Proceedings of the 2nd ACM Conference on Information-Centric Networking*, 2015 pp. 147-156.
- [23] C. A. Wood and E. Uzun, "Flexible end-to-end content security in CCN," in *Proceedings of IEEE 11th Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, 2014, pp. 858-865.
- [24] Q. Zheng, G. Wang, R. Ravindran, and A. Azgin, "Achieving secure and scalable data access control in information-centric networking," in *Proceedings of IEEE International Conference on Communications (ICC)*, London, 2015, pp. 5367-5373.
- [25] B. Li, D. Huang, Z. Wang, and Y. Zhu, "Attribute-based Access Control for ICN Naming Scheme," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, pp. 194-206, 2018.
- [26] M. Raykova, H. Kazmi, H. Lakhani, and A. Gehani, "Decentralized Authorization and Privacy-Enhanced Routing for Information-Centric Networks," in *Proceedings of the 31st Annual Computer Security Applications Conference*, ACM, 2015, pp. 31-40.
- [27] M. Aiash and J. Loo, "A formally verified access control mechanism for information centric networks," in *Proceedings of the International Conference on Security and Cryptography*, 2015, pp. 377-383.
- [28] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, pp. 644-654, November 1976.

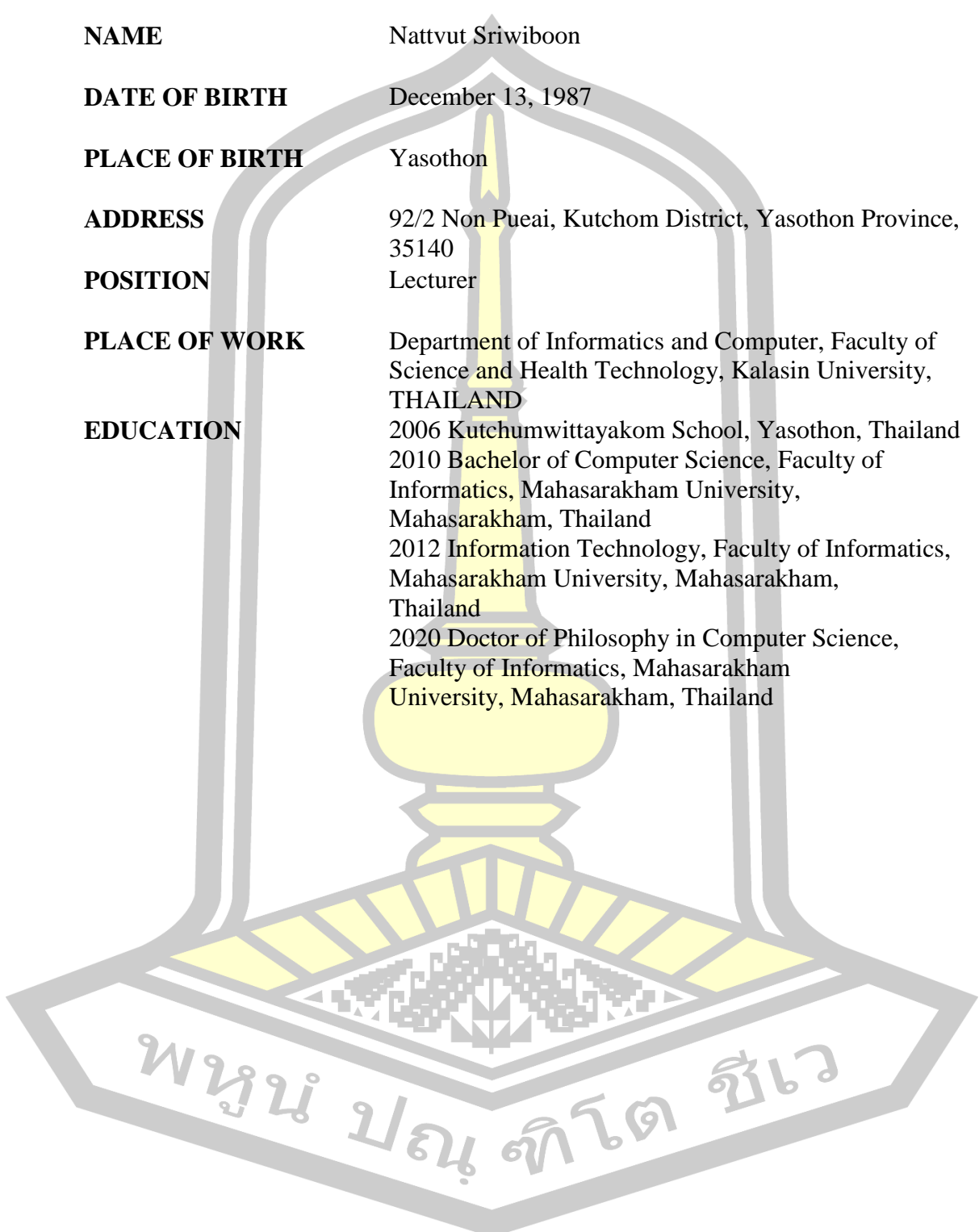
- [29] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attributebased encryption," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2007.
- [30] J. Liang, J. Jiang, H. Duan, K. Li, T. Wan, and J. Wu, "When HTTPS Meets CDN: A Case of Authentication in Delegated Service," in *Proceedings of 2014 IEEE Symposium on Security and Privacy*, San Jose, CA, 2014, pp. 67-82.
- [31] V. Jacobson, M. Mosko, D. Smetters, and J. Garcia-Luna-Aceves, "Content-centric networking," Palo Alto Research Center, Whitepaper, January 2007.
- [32] A. Afanasyev. (1 March 2020). *NDN Named-Based Access Control*. Available: <https://github.com/named-data/name-based-access-control>
- [33] Y. Yu, "Usable Security For Named Data Networking," Computer Science, UCLA, 2016.
- [34] H. Zhang, Y. Li, Z. Zhang, A. Afanasyev, and L. Zhang, "NDN Host Model," *ACM SIGCOMM Computer Communication Review*, vol. 48, pp. 1-7, July 2018.
- [35] Z. Zhang, Y. Yu, H. Zhang, E. Newberry, S. Mastorakis, Y. Li, *et al.*, "An Overview of Security Support in Named Data Networking," Technical Report NDN-0057 Revision 3, 20 June 2018.
- [36] FIU. (1 March 2020). *NDN Packet Format Specification 0.3 documentation*. Available: <https://named-data.net/doc/NDN-packet-spec/current/interest.html>
- [37] FIU. (10 March 2019). *NDN Certificate Format Version 2.0*. Available: <https://named-data.net/doc/ndn-cxx/current/specs/certificate-format.html>
- [38] Y. Yu, A. Afanasyev, and D. Clark, "Schematizing Trust in Named Data Networking," in *Proceedings of ACM Conference on Information-Centric Networking (ICN)*, San Francisco, CA, USA, 30 September-October 2015, pp. 1-10.
- [39] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP," IETF, RFC 2560, June 1999.
- [40] Z. Zhang, Y. Yu, A. Afanasyev, and L. Zhang, "NDN Certificate Management Protocol (NDNCERT)," UCLA, Technical Report NDN-0050, 29 April 2017.
- [41] FIU. (January 2020). *NDN Packet Format Specification 0.2.1 documentation*. Available: <https://named-data.net/doc/NDN-packet-spec/0.2.1/signature.html>
- [42] FIU. (1 January 2020). *Named Data Networking Forwarding Daemon (NFD) 0.7.0 documentation*. Available: <https://named-data.net/doc/NFD/current/INSTALL.html>
- [43] D. Pesavento. (1 January 2020). *NDN Certificate Management Protocol (NDNCERT)*. Available: <https://github.com/named-data/ndncert/blob/master/README.md>
- [44] CAIDA. (1 January 2020). *NFD: Issue Your Own NDN Certificates*. Available: <https://named-data.net/2016/07/28/nfd-issue-ndn-certificates/>
- [45] W. Shang, Z. Wang, A. Afanasyev, J. Burke, and L. Zhang, "Breaking out of the Cloud: Local Trust Management and Rendezvous in Named Data Networking of Things," in *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation*, PA, USA, 18 - 21 April 2017 pp. 3-13.

- [46] FIU. (1 March 2020). *Signed Interest*. Available: <https://named-data.net/doc/ndn-cxx/current/specs/signed-interest.html>
- [47] R. Sandhu and P. Samarati, "Access Control: Principles and Practice," *Communications Magazine, IEEE*, 1994.
- [48] R. Shirey, "Internet Security Glossary, Version 2," IETF, RFC 4949, August 2007.
- [49] Y. Tseng and C. Fan, "FGAC-NDN: Fine-Grained Access Control for Named Data Networks," *Journal of IEEE Transactions on Network and Service Management*, vol. 16, pp. 143 - 152, 08 August 2018.
- [50] V. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, *et al.*, "Guide to Attribute Based Access Control (ABAC) Definition and Considerations," U.S. Department of Commerce, NIST Special Publication 800-162, January 2014.
- [51] G. Acs, M. Conti, P. Gasti, C. Ghali, and G. Tsudik, "Cache Privacy in Named-Data Networking," in *Proceedings of IEEE 33rd International Conference on Distributed Computing Systems*, 2013, pp. 41-51.
- [52] A. Chaabane, E. D. Cristofaro, M. A. Kaafar, and E. Uzun, "Privacy in Content-Oriented Networking: Threats and Countermeasures," *ACM SIGCOMM Computer Communication Review*, vol. 3, pp. 26-33, July 2013.
- [53] P. Gasti and G. Tsudik, "Content-Centric and Named-Data Networking Security: The Good, The Bad and The Rest," in *Proceedings of IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, Washington, DC, 2018, pp. 1-6.
- [54] U. Hengartner and P. Steenkiste, "Exploiting Hierarchical Identity-Based Encryption for Access Control to Pervasive Computing Information," in *Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, Washington, DC, September 2005, pp. 384-396.
- [55] S. Jahid, P. Mittal, and N. Borisov, "EASiER: Encryption-based Access Control in Social Networks with Efficient Revocation," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2011)*, Hong Kong, China, March 2011, pp. 411-415.
- [56] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proceedings of CRYPTO 84 on Advances in cryptology*, Springer-Verlag, New York, 1985, pp. 47-53.
- [57] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proceedings of EUROCRYPT*, Springer-Verlag, New York, 1998, pp. 127-144.
- [58] A. Arnbak, H. Asghari, M. Eeten, and N. Eijk, "Security Collapse in the HTTPS Market," *Journal of ACM Queue*, vol. 12, pp. 1-15, September 2014.
- [59] Z. Durumeric, J. Kasten, M. Bailey, and J. Halderman, "Analysis of the HTTPS certificate ecosystem," in *Proceedings of IMC '13 Proceedings of the 2013 conference on Internet measurement conference*, New York, USA, 2013, pp. 291-304.
- [60] CISA. (1 October 2019). *OpenSSL 'Heartbleed' vulnerability (CVE-2014-0160)*. Available: <https://www.us-cert.gov/ncas/alerts/TA14-098A>.

- [61] E. G. AbdAllah, H. S. Hassanein, and M. Zulkernine, "A Survey of Security Attacks in Information-Centric Networking," *IEEE Communications Surveys & Tutorials*, vol. 17, pp. 1441-1454, 14 January 2015.
- [62] S. Chen and F. Mizero, "A Survey on Security in Named Data Networking," *arXiv:1512.04127v1*, 13 December 2015.
- [63] W. Ding, Z. Yan, and R. H. Deng, "A Survey on Future Internet Security Architectures," *IEEE Access*, vol. 4, pp. 4374-4393, 29 July 2016.
- [64] R. Tourani, T. Mick, S. Misra, and G. Panwar, "Security, Privacy, and Access Control in Information-Centric Networking: A Survey," *arXiv:1603.03409v2*, 29 September 2016.
- [65] G. Acs, M. Conti, P. Gasti, C. Ghali, and G. Tsudik, "Privacy-Aware Caching in Information-Centric Networking," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, pp. 313-328, 2019.
- [66] A. Mohaisen, X. Zhang, M. Schuchard, H. Xie, and Y. Kim, "Protecting Access Privacy of Cached Contents in Information Centric Networks," in *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, Hangzhou, China, 2013, pp. 1-6.
- [67] Quotes. (1 October 2019). *Perfect Forward Secrecy*. Available: Available from: <https://www.perfectforwardsecrecy.com/>
- [68] A. Brusilovsky, I. Faynberg, Z. Zeltsan, and S. Patel, "Password-Authenticated Key (PAK) Diffie-Hellman Exchange," IETF, RFC 5683, February 2010.
- [69] E. Yoon and K. Yoo, "A Practical Convertible Authenticated Encryption Scheme with Message Linkages and Forward Secrecy," in *Proceedings of 14th IEEE International Conference on Computational Science and Engineering*, Dalian, 2011, pp. 339-342.
- [70] M. D. Green and I. Miers, "Forward Secure Asynchronous Messaging from Puncturable Encryption," in *Proceedings of IEEE Symposium on Security and Privacy*, San Jose, CA, 2015, pp. 305-320.
- [71] K. Fu, S. Kamara, and T. Kohno, "Key regression: Enabling efficient key distribution for secure distributed storage," *Computer Science Department Faculty Publication Series*, pp. 1-35, 2006.
- [72] T. Liang and B. Zhang, "Enabling Off-the-Grid Communication for Existing Applications: A Case Study of Email Access," in *Proceedings of the International Conference on Communications Workshops (ICC Workshops)*, Kansas City, MO, USA, 2018.
- [73] T. Chuachan, K. Djemame, and S. Puangprongpitag, "Solving MTU Mismatch and Broadcast Overhead of NDN over Link-layer Networks," *International Journal of Networked and Distributed Computing*, vol. 8, pp. 67-75, 2020.
- [74] J. Bethencourt. (1 March 2020). *CPABE-SETUP*. Available: <http://hms.isi.jhu.edu/acsc/cpabe/cpabe-setup.html>
- [75] J. Klensin, "Role of the Domain Name System (DNS)," IETF, RFC 3467, February 2003.
- [76] D. Naor, A. Shenhav, and A. Wool, "Toward Securing Untrusted Storage Without Public-Key Operations," in *Proceedings of ACM Workshop on Storage Security and Survivability (StorageSS'05)*, ACM, 2005, pp. 51-56.

- [77] FIU. (1 January 2020). *ndn-cxx: NDN C++ library with eXperimental eXtensions 0.7.0 documentation*. Available: <https://named-data.net/doc/ndn-cxx/current/INSTALL.html>
- [78] Z. Zhang. (1 March 2020). *Attribute-based Access Control over Named Data Networking*. Available: <https://github.com/Zhiyi-Zhang/NAC-ABE>
- [79] J. Ahrenholz, "Comparison of CORE Network Emulation Platforms," in *Proceedings of MILCOM*, California, USA, November 2010, pp. 166-171.
- [80] D. Management. (1 January 2020). *Public key cryptography using discrete logarithms. Part 1: Diffie-Hellman key exchange*. Available: <https://www.di-mgt.com.au/public-key-crypto-discrete-logs-1-diffie-hellman.html>
- [81] B. Lynn. (1 March 2020). *The Pairing-Based Cryptography (PBC) library*. Available: <https://crypto.stanford.edu/pbc/>
- [82] R. Alvarez, C. Caballero-Gil, J. Santonja, and A. Zamora, "Algorithms for Lightweight Key Exchange " *Sensors*, vol. 7, pp. 1-14, 2017.
- [83] K. A. Nagaty, "A public key cryptosystem and signature scheme based on numerical series," *SN Applied Sciences*, vol. 2, pp. 1-14, 2020.
- [84] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," in *Proceedings of IEEE Symposium on Security and Privacy (SP '07)*, Berkeley, CA, 2007, pp. 321-334.
- [85] C. Ghali, G. Tsudik, and C. A. Wood, "(The Futility of) Data Privacy in Content-Centric Networking," in *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*, ACM, 2016, pp. 143-152.
- [86] C. Ghali, G. Tsudik, and C. A. Wood, "When encryption is not enough: privacy attacks in content-centric networking," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ACM, 2017, pp. 1-10.
- [87] E. Barker and Q. Dang, "Recommendation for Key Management," U.S. Department of Commerce, NIST Special Publication 800-57 Part 3, Revision 1, 2015.
- [88] D. P. Arjunwadkar, "Introduction of NDN with Comparison to Current Internet Architecture based on TCP/IP," *International Journal of Computer Applications* vol. 105, pp. 31-35, November 2014.
- [89] A. Afanasyev, J. Burke, T. Refaei, L. Wang, B. Zhang, and L. Zhang, "A Brief Introduction to Named Data Networking," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*, Los Angeles, CA, 2018, pp. 1-6.

BIOGRAPHY



NAME	Nattvut Sriwiboon
DATE OF BIRTH	December 13, 1987
PLACE OF BIRTH	Yasothon
ADDRESS	92/2 Non Pueai, Kutthom District, Yasothon Province, 35140
POSITION	Lecturer
PLACE OF WORK	Department of Informatics and Computer, Faculty of Science and Health Technology, Kalasin University, THAILAND
EDUCATION	2006 Kutthumwittayakom School, Yasothon, Thailand 2010 Bachelor of Computer Science, Faculty of Informatics, Mahasarakham University, Mahasarakham, Thailand 2012 Information Technology, Faculty of Informatics, Mahasarakham University, Mahasarakham, Thailand 2020 Doctor of Philosophy in Computer Science, Faculty of Informatics, Mahasarakham University, Mahasarakham, Thailand