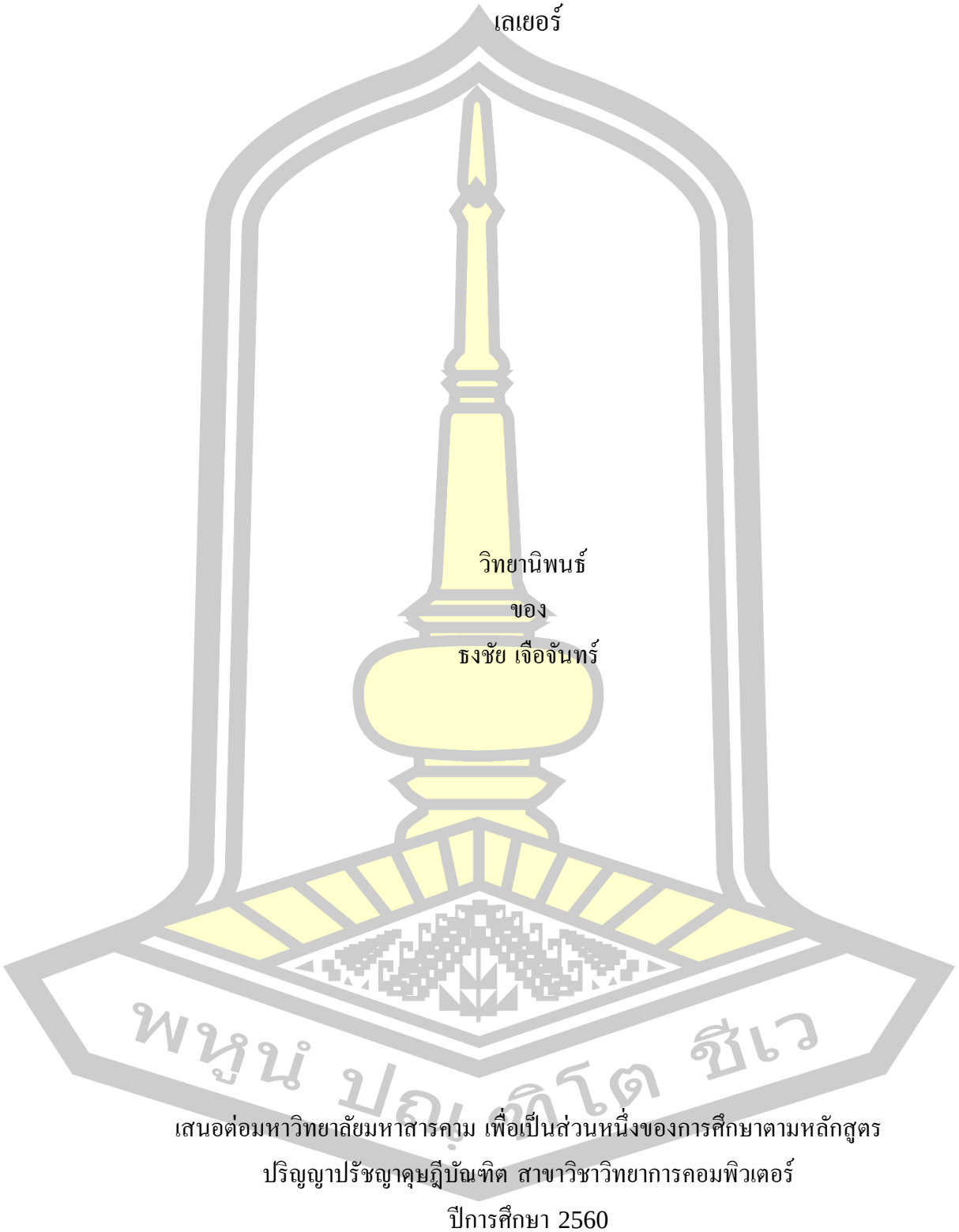
Solving MTU Mismatch and Broadcast Overhead of NDN over Link-layer Networks

Thongchai Chuachan

A Thesis Submitted in Partial Fulfillment of Requirements for
degree of Doctor of Philosophy in Computer Science
Academic Year 2017

การแก้ปัญหาเอ็มทียูไม่ตรงกันและบรอดคาสท์โอเวอร์เฮดของเครือข่ายเนมดาต้าบนลิงก์เลเยอร์



วิทยานิพนธ์

ของ

ธงชัย เจือจันทร์

เสนอต่อมหาวิทยาลัยมหาสารคาม เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาปรัชญาดุษฎีบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

ปีการศึกษา 2560

Solving MTU Mismatch and Broadcast Overhead of NDN over Link-layer Networks

Thongchai Chuachan

A Thesis Submitted in Partial Fulfillment of Requirements

for Doctor of Philosophy (Computer Science)

Academic Year 2017

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

Examining Committee

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

.....Advisor
(Somnuk Puangpronpitag , Ph.D.)

.....Committee
(Asst. Prof. Chatklaw Jareanpon , Ph.D.)

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

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

.....
(Asst. Prof. Sujin Butdisuwan , Ph.D.)
Dean of the Faculty of The Faculty of Informatics

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

Day.....Month...Year.....

TITLE Solving MTU Mismatch and Broadcast Overhead of NDN over Link-layer Networks

AUTHOR Thongchai Chuachan

ADVISORS Somnuk Puangpronpitag , Ph.D.

DEGREE Doctor of Philosophy **MAJOR** Computer Science

UNIVERSITY Mahasarakham University **YEAR** 2017

ABSTRACT

Named Data Networking (NDN) is a clean-state network architecture promising a fidelity data-centric communication. In particular, NDN over link-layer networks would reduce the overheads of TCP/IP, and enhance the efficiency of data distribution. However, several unsolved issues in NDN over the link-layer networks still damage NDN networks, such as broadcast overhead and MTU mismatch. In this dissertation, we have therefore designed and implemented an NDN Neighbourhood Discovery Protocol, named NDN-NDP, to enable a unicast data transmission over the link-layer. Furthermore, our NDN-NDP has included a negotiation mechanism to fix the MTU mismatch issue. Through CORE emulation, we have demonstrated that our NDN-NDP outperforms the previously proposed NDN link-layer technologies in terms of broadcast overhead, network throughput, delay and a number of unsatisfied *interest*. Our NDN-NDP can decrease broadcast overhead, delay and a number of unsatisfied *interest* approximately 99%, 46% and 11% respectively. In addition, our NDN-NDP increases network throughput around 22%.

Keyword : Named Data Networking, NDN Link-Layer Networks, Adaptive MTU, NDN-NDP

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).

Thongchai Chuachan

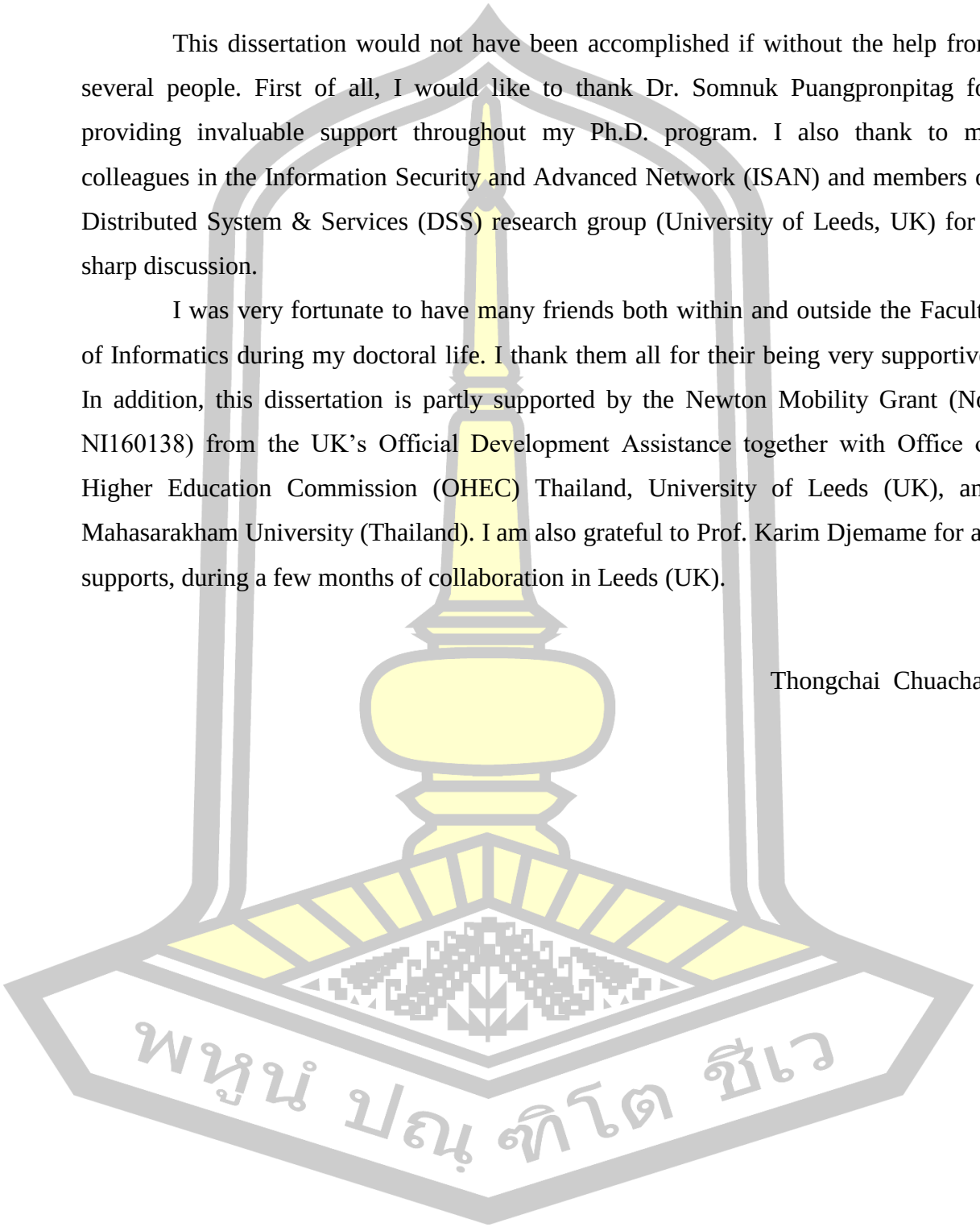


TABLE OF CONTENTS

Page

CHAPTER 1

INTRODUCTION

1.1 Introduction

Named Data Networking (NDN) [1, 2] proposes a data centric network model respecting one of the future Information Centric Network (ICN) architecture, defined by the Internet Research Task Force (IRTF) [3]. NDN changes the basic communication model. By using a data name instead of a destination host, data in NDN can be retrieved from anywhere in the network. Two types of packets, namely interest and data, have been used to retrieve packets (identified by names) in NDN. For NDN deployment, it can be implemented on top of the current TCP/IP protocol stack, or deployed directly over a link-layer network without the TCP/IP protocol stack. NDN over the TCP/IP stack would make NDN possible while the old Internet is still running. However, NDN over the link-layer will decrease overhead.

Several protocols for NDN over the link-layer have been proposed, such as NDNLP [4], FIGOA [5] and BEF [6]. However, they still face two major challenges, namely broadcast overhead and Maximum Transmission Unit (MTU) [7] mismatch. First, the previously proposed NDN link-layer protocols have mainly relied on a broadcast scheme that could drastically increase network overhead. A mechanism to create and manage unicast faces (aka. network interfaces) has not yet been proposed. So, it is impossible to deliver packets in a unicast mode. Second, NDN over link-layer networks may also support a heterogeneous network environment. In such an environment, MTU mismatch is a serious problem, causing transmission failure.

In this dissertation, a neighborhood discovery protocol for NDN, named NDN-NDP, has been proposed to solve the previously described problems. The NDN-NDP focuses on creating and managing a unicast link-layer face to reduce the number of broadcast packets. In addition, an adaptive MTU (*aMTU*) has also been designed into NDN-NDP to solve the MTU mismatch problem. Our NDN-NDP has been implemented by extending Named Forwarding Daemon (NFD) modules [8]. A performance evaluation has been done using a Common Open Research Emulator (CORE) [9]. Experimental results have demonstrated that NDN-NDP can reduce delay and the number of unsatisfied interest packets. Furthermore, NDN-NDP can increase network throughput. So, NDN-NDP effectively enhances the NDN over link-layer networks.

1.2 Objectives

Objectives in this dissertation focuses on an enhancement of NDN over the link-layer network. There are two main goals in this dissertation, included to reduce the number of broadcast packets and to design NDN-NDP for NDN over the link-layer networks.

1. To analyze the problems of NDN over the link-layer network
2. To enable a unicast transmission mode through a design and development of a neighborhood discovery protocol for NDN
3. To design an adaptive MTU to solve MTU mismatch problem in NDN over the link-layer network

4. To evaluate our NDN-NDP with a rigorous statistical analysis

1.3 Contribution of this work

As described earlier, there are several challenges to NDN over the link-layer network. For contribution of this work, we can summarize as follows:

1. An extensive analysis of the future NDN architecture over the link-layer has been analyzed to identify key problems.
2. A unicast transmission mode in NDN over a link-layer network has been provided by developing NDN-NDP protocol.
3. The MTU mismatch problem has been solved by proposing a novel *aMTU* algorithm.
4. NFD has been extended to enable NDN-NDP protocol and *aMTU* for NDN over the link-layer network.
5. The performance evaluation of NDN-NDP protocol has been done through CORE emulator.

1.4 Scope

While NDN supports a lot of transport services, this work is only focused on NDN over a link-layer network. For previously proposed solutions of NDN over link-layer networks, there are some initial proposals, such as FIGOA, NDNLP, and BEF. However, all of them have not been focused on a particular MTU mismatch and broadcast problem. So, a neighborhood discovery protocol is needed to solve these remaining problems. The goal of this dissertation is to solve the broadcast overhead and MTU mismatch problem for NDN over the link-layer networks.

1.5 Dissertation Outline

CHAPTER 2 reviews some networking trends and critical problems in modern networking. It is followed by NDN concepts and mechanisms, NDN forwarder process, and NDN over the link-layer network. The last section of the CHAPTER 2 describes the details of previous solutions.

CHAPTER 3 presents research methodology used for this dissertation. Performance evaluation techniques have been compared. In particular, this chapter justifies why a network emulation technique has been chosen as a measurement tool. CORE is then given. CORE emulation construction, including emulated nodes, network management, and emulation parameters are included in this chapter.

CHAPTER 4 proposes a new design of an NDN-NDP protocol. The design goals of NDN-NDP are to reduce broadcast packets and to solve MTU mismatch problem. In addition, an overview of security issues of our design is discussed.

CHAPTER 5 presents experimental results of our NDN-NDP, using performance metrics defined in CHAPTER 4. Furthermore, the comparative analysis between our NDN-NDP and other existing NDN link-layer proposals is given.

CHAPTER 6 summarizes this dissertation, and suggests future work.

1.6 Abbreviations and Acronyms

ACK	Acknowledgement
ARP	Address Resolution Protocol
BEF	Begin End Fragmentation
BGP	Border Gateway Protocol
CORE	Common Open Research Emulator
CCN	Content Centric Networking
CDN	Content Distribution Networks
CS	Content Store
DHCP	Dynamic Host Control Protocol
FIB	Forwarding Information Base
FIGOA	Secure Fragmentation for Content-Centric Networks
HTTP	Hyper Text Transfer Protocol
ICN	Information Centric Networks
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IOT	Internet of Thing
IP	Internet Protocol
Ipv4	Internet Protocol Version 4
Ipv6	Internet Protocol Version 6
IRTF	Internet Research Task Force
LFBL	Learning First Broadcast Later
MTU	Maximum Transmission Unit
NDNLP	Link Layer Protocol for NDN
NDN	Named Data Networking
NFD	Named Forwarding Daemon
NACK	Negative Acknowledgement
NDP	Neighborhood Discovery Protocol
NDN-NDP	Neighborhood Discovery Protocol for NDN
NS3	Network Simulator 3
OLSR	Optimized Linked State Routing
PIT	Pending Interest Table
SUT	System Under Test
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

CHAPTER 2

BACKGROUND AND RELATED WORK

This chapter briefly introduces background and related work of NDN over the link-layer networks. In addition, the relevant literatures have been described into specialized details of the literature.

2.1 Networking Trends

This section provides the background of the networking trends and evolution of the Internet. A new network architecture is then presented and justified for a possibility of the future network architecture.

2.1.1 The Connected Machines

Cisco Virtual Networking Index (VNI) [10] has shown that IP traffic will reach an annual run rate of 3.3 ZB per year. Moreover, things are more connected to the Internet than the human. This TCP/IP [11] protocol stack tends to be used by a large amount of connected machines. These machines require a high-level of network requirements (such as mobility, robustness and trustworthiness). For example, the minimum speed of Vehicular Networks ranges from ~14 to 47 mph. In such high mobility networks, TCP/IP, as shown in [12], is not well-suited for future Internet connections.

2.1.2 Evolution of the Internet

The Internet has evolved from a small-scale network to share resources between two end points. By 1960s, DARPA [13] had granted multiple universities and private companies to research a network communication. Khan and Cerf [14] had finally developed an IP, which had increased data traffic tremendously. Since IP had been developed, network applications have increasingly been merged into business productivities. Especially, applications on HTTP are beneficial, and may be a new narrow waist of the network [15]. The network traffic in current Internet thus is filled with applications using IP.

Alternatively, network hardware has also been improved for a better performance. It can be used to distribute a large amount of information, and can be forwarded in a complex network infrastructure. Although network hardware is currently equipped with high performance, it cannot be provided in a large amount of user demands.

There are quite a few research proposals for today's network characteristics (such as Information Centric Networks (ICN) [16], Middleboxes [17], and Software Defined Network (SDN) [18]). Given a network characteristic, Popa et al. [15] has shown that applications are relied on HTTP to receive contents in terms of what by using named data. The named data almost provide for modern network devices, and does not applicable in the IP architecture.

2.1.3 Future Internet Architecture

For the future Internet architecture, there are several network paradigms. Four of the prominent network architectures are contributed in several aspects as follows

1. MobilityFirst [19] is a prominent project for making the future Internet architecture. MobilityFirst architecture tries to create: (1) seamless connections over TCP/IP, (2) no global root of trust, (3) intentional data receipt, (4) robustness network connection, (5) addressing, (6) evolvable networks. MobolityFirst concepts build on the top of the IP address space by using names. This IP address may not be matched with the explosive growth of the mobility networks, and may not be efficient to deliver data in challenging networks (as descried in [20]).

2. NEBULA [21] is another future network architecture to make a flexible and extensible network. This NEBULA proposal attempts to distribute content in cloud computing, consisting of Data Plane and Virtual and Extensible Networking Technique (NVENT). The Data Plane involves in forwarding mechanisms. The NVENT is a control plane providing security policies. By extending the current Internet architecture, there are potential problems to rely on TCP/IP protocol stack.

3. eXpressive [22] Internet Architecture (XIA) is a network paradigm to improve several aspects (such as a security, a long-term evolution of the networks and devices, and a well-defined network interface). XIA envisions three possible characteristics of the future network architecture. The first one is to support diverse communication entities. The second one is to build a fundamental building block of the transmission security. The last one is to build a “narrow waist” (i.e., using an Application Programming Interface (API) as a part of communication mechanisms) networks. XIA tries to re-architecture the Internet with a trustworthy and evolvable Internet.

4. Named Data Networking (NDN) [23] aims to address several issues in the current Internet architecture. For examples, address space, security, mobility and content distribution problems are difficult to handle in current network architecture. A fundamental change of NDN fully transforms point-to-point communications into content distributions. By using a data name, a desired data can be retrieved from any potential sources. The desired data thus can be retrieved from multiple locations. All data in NDN are cryptically signed by a content producer (aka. a server). NDN fundamental changes seem to be the most attractive network architectures, and match with current Internet problems. In this dissertation, NDN has been selected and enhanced in the link-layer network.

5. CICN [24] ports from the CCNx project to enable data delivery in a large scale networks. CICN extends to several sub-projects. By designing a universal data plane, CICN builds the ICN network architecture into the kernel space, which follows IRTF and IETF documents. By using Vector Packet Processing (VPP) and CCNx protocols, CICN can easily deploy in large scale networks.

6. Network of Information (NetInf) [25] is an EU-FP7 project. NetInf principle emphasizes on multi-technology/multi-domain interoperability, and adds architecture elements (such as naming, transport, caching) into a major design. For current network developments, 5 technological advancements have been added in NetInf, including 1) Localised Content Delivery Network (CDN), 2) Event with Large

Crowds, 3) Delay Tolerant Video Distribution, 4) Active Information Objects, and 5) Local Collaboration.

7. Publish-Subscribe Internet Routing Paradigm (PURSUIT) [26] is a publish-subscribe Internet architecture. PURSUIT tries to evaluate the current Internet architecture, and offers a new paradigm of internetworking. By given Publish-Subscribe scheme, information routing has been used through multiple brokers. Brokers are then responsible for forwarding information between data consumers and data producers. In addition, the multicast transmission scheme is preferred in PURSUIT. However, PURSUIT relies on the TCP/IP architecture. It is unreliable in a mobility network, which may downgrade a transmission quality.

8. COntent Mediator architecture for content-aware nETworks (COMET) [27] introduces a global naming scheme, and unifies a content-access protocol. COMET uses selective caching instead of ubiquitous caching, and achieves a higher gain by using selective nodes. A challenging goal in COMET is that which network nodes should be selected. A centrality-driven caching scheme thus has been designed for selecting catcher nodes.

9. GreenICN [28] tries to address ICN networks to operate in a high performance and large scale network. An energy efficiency is one of the most important in this network environment. To archive GreenICN network architecture, a publish/subscribe network has been used to deliver content. This content is identified by name. The GreenICN effort has focused on a disaster and video delivery scenario.

10. Named Function Networking (NFN) [29] is similar to NDN. An interest packet carries at least two names to receive data packets. The NFN is actually extended from CCNx. By using λ-calculus expressions, the NFN claims that the data packet can be responsively retrieved from NFN networks. Not only NFN can receive multiple data by using a single interest, but also reduce computational overhead in its NFN forwarding process. However, a security paradigm in NFN has not been explored.

11. iP Over IcN (POINT) [30] considers to innovate an IP-based ICN, and improves quality of service using IP. Firstly, POINT applies a multicast capability of HTTP as a potential distribution of information. Secondly, a light-weight security protocol improves constrained applications (such as IoT). CoAP over ICN will be able to associate security and privacy requirements. Thirdly, CDN provides a basic content distribution, and promises an effective quality of services. POINT describes that an unnecessary content caching may be wasted in the Internet, and degrades network traffic. POINT proposal can bring a powerful tool in the current TCP/IP Internet architecture but it has not yet considered several problems (such as security and mobility problems).

2.2 Critical Problems in Modern Networking

Modern internetworking has made up of a wide variety of communication languages, called protocols. Standard protocols for today's network include Internet Protocol (IP) [11], Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP), Address Resolution Protocol (ARP), and Dynamic Host Configuration Protocol (DHCP). These protocols work together, and form a communication between two end-hosts. In particular, IP has built for point-to-point communication that was originated from telephone systems.

Intermediate routers intelligently use routing tables to recognize which hops should be forwarded to. Since 1980s, IP has become a crucial protocol.

Nowadays, IP has been extended to use for different purposes. Internet-banking, for example, requires a high level of security. Yet, there are critical issues using Internet-banking. For mobile networks, a mobility speed is ranged from a human walk to a high-speed train. The mobile networking mainly maintains routing tables for making connection between two end-users. Real-time applications thus cannot be avoided by network disruption due to a rapid change of routing tables. IP has encountered with several potential problems due to the requirements of modern network applications. In this section, the problems are described as follows:

2.2.1 Inefficient protocol functionality in mobility networks

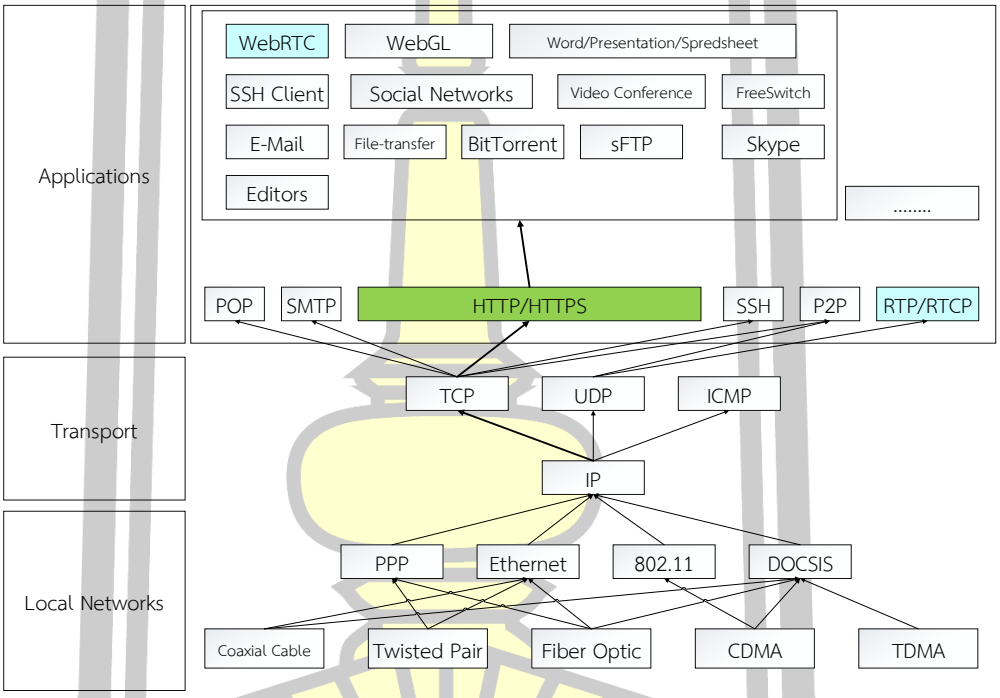


Figure 1 TCP/IP houseglass

IP houseglass, as shown in Figure 1, presents an elegant model of the point-to-point communication. An application layer of the communication network such as HTTP, P2P, and SMTP is increasingly important to carry application data. Moreover, Popa et al. [15] have shown that HTTP is a narrow waist of the Internet, as depicted in the application layer of Figure 1. TCP and UDP places on the top of IP. Port numbers are used to handle data transmission for TCP and UDP. For a waist of the current Internet, IP enables a development of the network hardware and applications without effecting deployments. Network routing is the most important mechanism for providing path between two end-points. Routing protocols such as Border Gateway Protocol (BGP) [31], Optimized Link State Routing (OLSR) [32], and Routing Information Protocol (RIP) [33] are operated for accommodating IP communication. Yet, an explosive growth of mobile devices increases routing overhead. The freshness of routing tables is significant to delivery IP packets properly. For mobility networks, routing issues are still huge challenges. There have been several studies to provide

freshness of routing tables in mobility networks (such as [34-37]). Yet, this issue is still unsolved and left as open-research issues.

While applications and network hardware are still evolved, IP is still a waist of the Interest. In a network disruption like mobility networks. Mobile Ad-hoc Network (MANET) [38] and Multi-path TCP (MPTCP) [39] have been developed to deliver contents efficiently. MANET aims to update routing tables in a mobility network, and reacts quickly to topology change. In contrast to the freshness of the routing table, a stateless is a notorious problem for using IP. MPTCP apparently emerges to solve the stateless problem in IP. By developing sub-flows of a TCP session, MPTCP can efficiently exploit TCP to be a stateful transmission. The details are as follows:

1. Mobile Ad-hoc Network (MANET) [38] deliberately creates a network without infrastructure. Networking devices can form a network topology automatically. Since MANET has been developed, research topics on MANET has extended to various kinds of the specific applications [40-42] (such as Vehicular Ad-Hoc Network (VANET)). However, MANET is still very challenging in IP networking. For example, in high mobility networks, a network connectivity cannot be satisfied by routing mechanisms [12].

2. MPTCP is one of the prominent solutions for improving TCP. It can receive TCP segments from multiple channels. As shown in Figure 2, an additional stack of MPTCP has been inserted between an application layer and IP layer, called subflow. In the IP layer, MPTCP generates TCP segments, and places the segments on the top of all available IPs. However, MPTCP can penalize TCP users, and does not handle a network congestion properly [43].

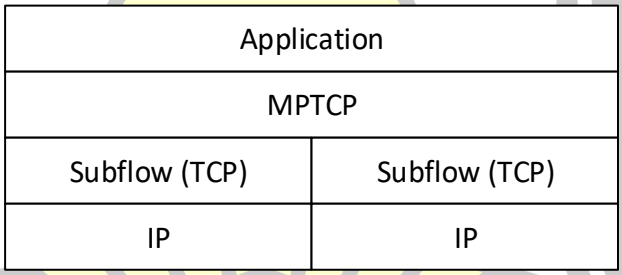


Figure 2 Multipath TCP architecture

2.2.2 Inflexible protocol architecture

Current applications require a versatile pattern of the network communication. Network protocols should be flexible to deliver contents effectively, and must be cached for mitigating content access problems. However, TCP/IP in a modern network infrastructure requires a freshness of routing table to deliver contents as a point-to-point network. Four basic routing components include source IP (*src*), destination IP (*dst*), interface (*iface*) and cost. These components guarantee to deliver data rapidly. In contrast to a flexible network infrastructures, MANETs, VANETs and Wireless Sensor Networks (WSN) demand routing tables to be responsive. Yet, IP cannot efficiently serve this demand. Research topics on IP for flexible network infrastructure have been previously explored, and identify mountainous problems [44].

TCP and UDP are the other complex communication protocols, which are once help Internet from meltdown. In contrast to current user demands, TCP especially requires many functionalities to maintain IP communication.

2.2.3 Lack of security

Transport Layer Security (TLS) [45] has been provided in TCP/IP. TLS encapsulates application data into obscure contents. The content name is a crucial element in TLS. However, there is not possible to design a protocol to deal with all possible attacks. As we have seen, vulnerabilities of the TLS had been revealed from several IT crime cases.

2.2.4 Content Distribution Problems

An explosive growth of communication networks has brought the Internet to provide a large number of data. A major role of the Internet is mainly used for content distributions. There are subsequent issues for distributing a large amount of contents. Firstly, IP is designed for point-to-point communication. This leads TCP/IP to create several sessions, retrieving the same content. Secondly, delay in disruptive has influenced several applications. Currently, Content Distribution Networks (CDNs) [37] have been deployed to mitigate the content access problems. CDNs place cache servers in a global scale, and distribute server contents into the cache servers. To access contents, there are three solutions namely, *URL Rewriting*, *CNAME* and *Domain Hosting*. In *URL Rewriting*, an origin server simply modify URLs of the contents to CDN servers. For *CNAME*, DNS records have been modified to redirect packets to CDN servers. *Domain Hosting* allows CDN providers to control DNS for origin websites.

All desired contents in CDNs have requested by using content names. IP is only used for forwarding contents. Liang et al. [37] have investigated security issues in CDNs, and reported a violation architectural design of the TLS in CDNs. The first security issue is a shared certificate. A content owner must offer a server certificate to a CDN provider. The second one is a custom certificate, which signs origin server-certificates to CDN providers. Such techniques have somehow violated the Public Key Infrastructure (PKI).

2.3 Named-Data Networking

NDN is a future Internet model using an information centric paradigm. It is actually a variant or a proposal of Information Centric Network (ICN) [46] architecture, defined by the IRTF. There are also other proposals of ICN architecture, such as Content Centric Network (CCN) [1, 2], Data-Oriented Network Architecture (DONA) [47], Network of Information (NetInf) [25] and so on. Comparing with the current Internet model, NDN and other proposals of ICN would provide more efficiency for information distribution. Two types of packets, namely *interest* and *data*, have been used in NDN. The *interest* packet is a packet, sent by a consumer (aka., receiver), to request a specific content from a content producer. This requested content is referred by a unique name. To fetch the content, the interest packet follows name-routing paths towards the producer. The *data* packets, carrying the requested content, are then delivered to the consumer. These data packets are originated from

the producer, and then may be cached in any intermediate NDN forwarders (ie., routers) in a reverse path of the interest packet.

Figure 3 is a couple of the NDN and TCP/IP network architecture. TCP/IP, as shown in the left, is an agile delivery mechanism by using network routers. TCP/IP is used for simple point-to-point connections between two parties. An above IP layer is related to transport mechanisms (such as TCP and UDP). A lower IP layer are link protocols (such as the link-layer protocols). The upper and lower layer of the IP protocols are gradually increased in efficiencies, numbers and sizes.

For NDN in the right of the Figure 3, a content chunk plays a major role for transmitting NDN *data*. There are three fundamental concepts for NDN. The first concept is to shift from a point-to-point communication to a content distribution.

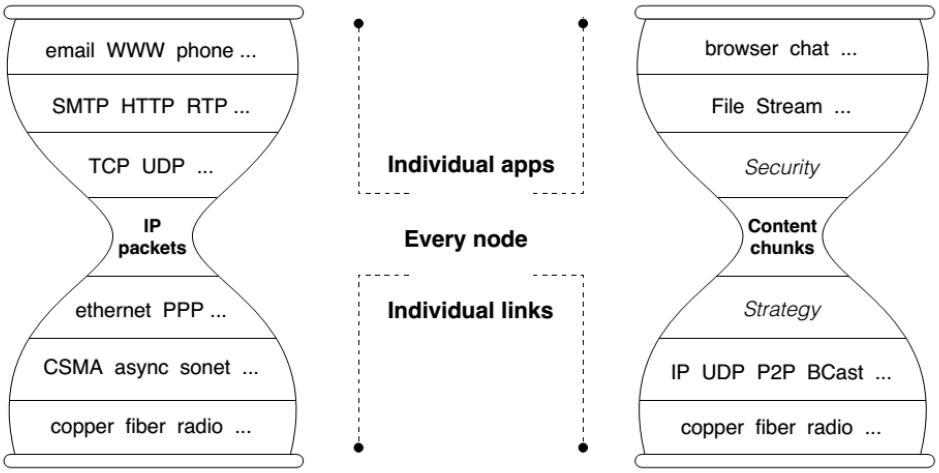


Figure 3 The main building blocks of the NDN architecture
Source : [22]

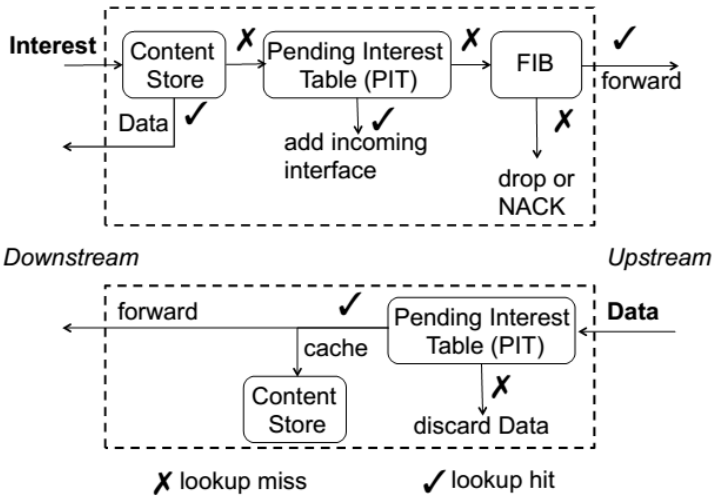


Figure 4 Forwarding process at an NDN node
Source : [22]

2.3.1 Named Forwarding Daemon

Named Forwarding Daemon (NFD) [8] is the process of an NDN forwarder, acting as an intermediate node between consumers and producers. It plays a key role to forward *interest* and *data* packets in NDN networks. As shown in Figure 4, NFD has been implemented with three main components, namely Content Store (CS), Forwarding Information Base (FIB) and Pending Interest Table (PIT). When an incoming *interest* packet arrives, CS is searched for the desired content. If the content exists, NFD then immediately forwards *data* packets of the desired content from CS to the consumer. Otherwise, NFD searches PIT to check whether or not the *interest* packet of the desired content has already been forwarded towards a content producer. If so, the NFD aggregates the *interest* packet into the transient PIT. In case that CS and PIT have been missed, FIB will be used to select routing paths to fetch the desired content. The content producer generates *data* packets, and returns them in a reverse path to the consumer. On the reverse path back to the consumer, the *data* packets may be cached in the CS of the intermediate nodes. In addition, these *data* packets have been digitally signed to ensure data integrity.

2.3.2 NFD Modules

NFD separates six major modules for controlling NDN infrastructure, as follows:

1. Core module: A core NFD module provides common networking manager, and controls NDN dependencies (such as hash computation and DNS resolver). These modules enable NFD to integrate different transmission services to NDN networks.
2. Face System: The Face System in NDN can be placed on the top of different transportation mechanisms (such as Ethernet, TCP and UDP), called face, as described further in Section 2.4.
3. Tables: As described earlier, there are three tables in NDN namely, CS, PIT and FIB. The table system in NFD manages and updates data plane in NDN networks.
4. Forwarding: A forwarding mechanism in NDN interacts with the other modules in NDN for providing paths to reach a destination.
5. Management: A management module primarily exchanges internal states of NFD (such as Forwarding Faces and Tables). For example, an incoming NDN packet from a Face can be updated in PIT for further forwarding.

2.4 Face System

To forward NDN packets, NFD abstracts lower-level network mechanism as faces. So, NDN Face System (FS) [8] is a crucial element to identify a communication link. The NDN community uses the term face instead of interface since NDN packets are not only forwarded over hardware network interfaces, but also exchanged directly with application processes within a machine [1, 2]. An NDN face is identified by its Face ID (FID). It represents as a map of remote Uniform Resource Identifier (URI) to a local URI with some parameters, as shown in Figure 5.

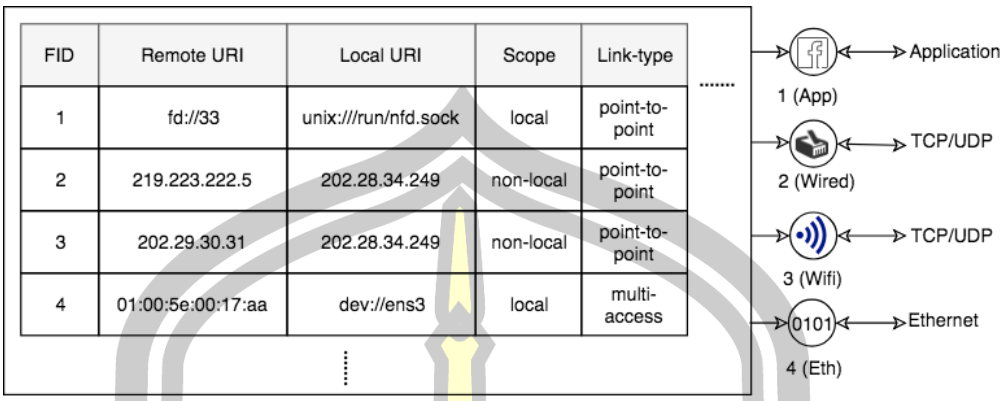


Figure 5 A sampled face fable

Figure 5 represents a sampled face table, maintained at each node. The face ID of 1 is an example of application process faces. The face ID of 2 is a sampled face of NDN over the TCP/IP stack. This face uses the User Datagram Protocol (UDP) as a communication link. It represents a map of remote URI, locating over an IP address of 219.223.222.5, to a local URI, locating over an IP adderss of 202.28.34.249. This face is then mapped to remote named-prefixes by FIB (as shown in Figure 6) to specify routes to fetch some specific contents.

The face ID of 4 is a face for NDN over link-layer networks. It maps a remote URI, "*ether:///[10:00:5e:00:17:aa]*", to a local URI, "*dev://ens3*", which is an Ethernet local *face*. To send and receive *interest* and *data* packets in NDN over an Ethernet link-layer network, a multicast address of *01:00:5E:00:17:AA* is used, as shown from the last sampled face (face ID=4). The address is known as "the default ICN Multicast Address (*ICN-MCAST*)". All NDN-enable devices on the Ethernet link-layer network must join to this multicast address. So, this technique is actually broadcasting to all NDN-enable devices on the link-layer network. This dissertation calls this face as "NDN Link-layer Broadcast Face (*NLBF*)". Deploying the *NLBF* can flood the link-layer network, causing broadcast overhead (further discussed in section 3.2). To avoid the overhead, it is very necessary to design new link-layer unicast *faces* for NDN over link-layer networks.

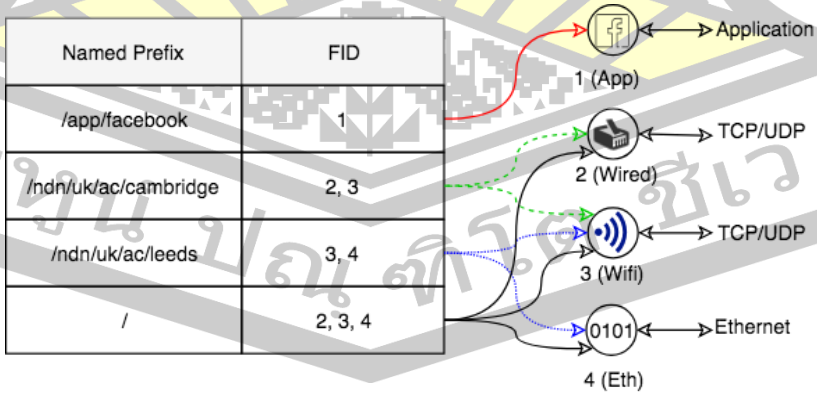


Figure 6 An example of FIB

2.5 NDN Link-layer Networks

To deploy NDN directly over link-layer networks, a Hop-By-Hop Fragmentation and Reassembly (HBH-FR) [48] technique has been used. HBH-FR generates frame-packets according to MTU of the local Network Interface Card (NIC), and broadcast them over LAN.

Deploying NDN directly over link-layer networks without the TCP/IP protocol stack would reduce huge overheads. This overhead cut-off provides an efficiency for data distribution, and would help many modern network applications evolve rapidly. However, the design of NDN over link-layer networks is still at its infant-phase. the link-layer protocol for NDN (NDNLP) [4], FIGOA [5] and BEF [6] are among the initially proposed NDN link-layer protocols. These protocols still suffer some problems, discussed in the next section.

The link-layer networks are the end route of typical networks. There are many types for connecting devices (such as PC desktops, laptops, mobile phones and tablets). NDN-enabled devices would have a wide variety of network characteristics. For example, a mobile network should be robust and tolerant against high lossy network.

2.6 NDN and Link-layer Problems

The current link-layer in NDN simply broadcasts *interest* and *data* packets to all nodes in local networks. By using NDNLP [4], hop-by-hop fragmentation and reassembly (HBH-FR) [48] has been implemented. A NIC's MTU becomes a crucial value to define a frame-packet's size. In addition, the frame-packet's size has been used for different purposes. For example, in high performance networks, JumboFrame [7, 49, 50] offers low overhead, and reduces a CPU load. For a constrained IoT device, they often have a small MTU. In such a real heterogeneous network, frame-packets may be generated by different types of network devices. The broadcast scheme then leads to MTU mismatch and flooding problems.

2.6.1 MTU Mismatch Problems

Maximum Transmission Unit (MTU) [7, 49, 50] is the largest possible payload of frame-packets that can be sent in a particular link-layer network. In general, the current Internet uses a standard Ethernet's MTU size of 1500 bytes, almost universally across networks. However, Ken and Monkul [51, 52] have pointed out that various network devices on the Internet have different MTU sizes, varying from 127 bytes for an IoT [48] to 65820 bytes for a fiber [49]. For example, constrained low-energy links in IoT networks have very small MTUs [50], [51]. The MTU size is only 127 bytes for IEEE 802.15.4-2006 [48]. In such a heterogeneous MTU environment, if a sender tries to transmit a packet too big for the receiver to cope with, the transmission could be failed. This problem is called "MTU mismatch".

To solve this problem for the current Internet, there have been several proposals. For example, *IPv6 Path MTU discovery* [52] has been proposed to find the minimum MTU on the transmission path. Kushalnagar et al. [53] have also proposed an adaptation layer between a link-layer and a network layer to mitigate the problem. However, these solutions could cause overhead and transmission inefficiency.

NDN also aims to support IoT and heterogeneous network devices. Some previous NDN link-layer proposals (such as NDNLP and BEF) uses HBH-FR to

select a frame-packet size according to the MTU of the local NIC. The transmission via these protocols could fail if there is an MTU mismatch between senders and receivers. So, the MTU mismatch is one of the significant problems for NDN over link-layer networks.

Currently, NDN uses HBH-FR to selects a frame-packet's size. It is collected from NICs. An NDN forwarder then slice NDN packets into frame-packets to send to their destination. The receiver then does a full reassembly in order to process packet information. For a state-of-art on link-layer networks for NDN, Secure Fragmentation for Content-Centric Networks (FIGOA) [5] is tried to find a minimum MTU (μMTU). It is a similar concept of *IPv6 Path MTU discovery*. However, the μMTU decrease network efficiency.

2.6.2 Broadcast Overhead

To send and receive interest and data packets in NDN over a link-layer network, the broadcast face (i.e., *NLBF*) is deployed. Broadcasting on the link-layer help simplify content distribution. This technique is also useful for reducing management of remote destination Medium Access Control (MAC) addresses. However, broadcasting could severely flood NDN local devices, cause network and computational overheads, shorten the nodes' battery-life. So, to avoid the broadcast overhead, a unicast scheme for NDN over link-layer networks should be designed and implemented. To implement a unicast face, remote destination MAC addresses must be learned and mapped. Hence, neighborhood discovery mechanisms for NDN link-layers must be designed.

There are some initial solutions to mitigate broadcast overhead in NDN over the link-layer networks. In [54]–[56], they try to reduce a number of broadcast packets by using a unicast face. On Broadcast-based Self-Learning in Named Data Networking (*Broadcast-based Self-Learning*) [54] and Listen First, Broadcast Later: Topology-Agnostic Forwarding under High Dynamics (*LFBL*) [56] learn a destination MAC address from incoming broadcast packets, and then create a unicast *face* for such destination MAC address. Meisel et al. [55] have brought attention toward broadcast overhead by creating a unicast face in IoT networks. Even though such mechanisms effectively leverage broadcast overhead in NDN over the link-layer networks, they do not emphasize on MTU mismatch problem. This could cause a transmission failure.

2.6.3 Lacking of Neighborhood Discovery

NDN principles allow packets to overlay over multiple network functions. For the link-layer networks, NDN broadcasts frame-packets in LANs. To enable neighborhood discovery such ARP and NDP, NDN users need to overlay over TCP or UDP.

A neighborhood discovery is crucial to provide a unicast communication link. It can enhance NDN over link-layer networks in several aspects (such as reducing broadcast overhead). However, NDN over link-layer networks is still lacking of the neighborhood discovery protocol.

2.7 Previous Solutions

2.7.1 NDNLP

A Link Protocol for NDN (NDNLP) [4] is the most widely used link-layer protocol for NDN. *NDNLP* creates frame-packets by using a default *ICN-MCAST*. Each NDN packets then can transfer without IP layer. As shown in Figure 7, NDNLP has replaced an IP layer, and has operated on the top of the link-layer. In doing so, several related IP protocols are no longer necessary (such as Address Resolution Protocol (ARP), TCP and UDP).

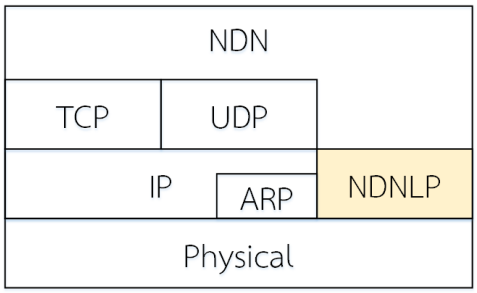


Figure 7 NDNLP layer

Figure 8 illustrates operations of NDNLP, and describes a transmission of a *data* packet from a router *A* to *B*. To do so, A firstly finds a NIC's MTU, and slices the *data* packet into multiple frame-packets. After that, the frame-packets will transfer to *ICN-MCAST*. *B* actively filters and collects incoming the frame-packets. For the last fragment index, *B* starts a reassembly process of the frame-packets, and passes a *data* packet into NFD.

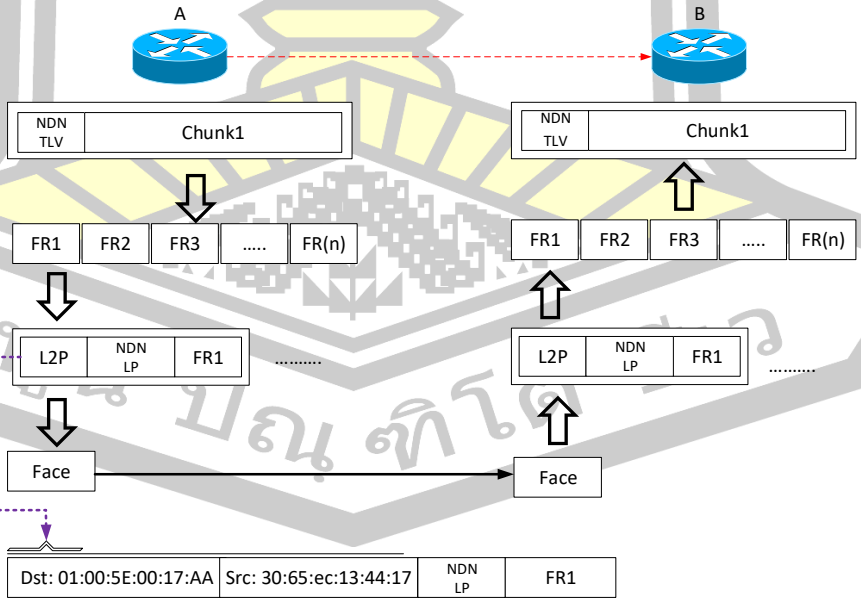


Figure 8 An NDNLP design

2.7.2 FIGOA

Secure Fragmentation for Content-Centric Networks (FIGOA) [5] proposal is attempted to design a fragmentation and reassembly process. FIGOA has referred to Mogul [51, 53, 54] MTU issues, and has explored several aspects of NDN frame-packet transmission. Firstly, an *interest* and *data* fragmentation is unavoidable in NDN. Secondly, a minimal MTU of the commination link is beneficial for a fragmentation mechanism. Lastly, intermediate reassembly and cut-through forwarding of the NDN fragments have been considered to handle the NDN fragments.

Figure 9 presents FIGOA procedures, including minimum MTU (*μMTU*) process. A consumer *A* sends an *interest* packet, and updates the *μMTU* to an NDN producer. The NDN producer then break contents into *data* packets to fit *μMTU*. After that, the NDN packet will be forwarded to the consumer. The *data* packet has also been signed for a delayed authentication [55]. In doing so, intermediate NDN forwarders can proceed two possible schemes. The first scheme is a cut-through forwarding. It is a transmission *data* packets toward consumers without the reassembly process. The second scheme is a common hop-by-hop reassembly.

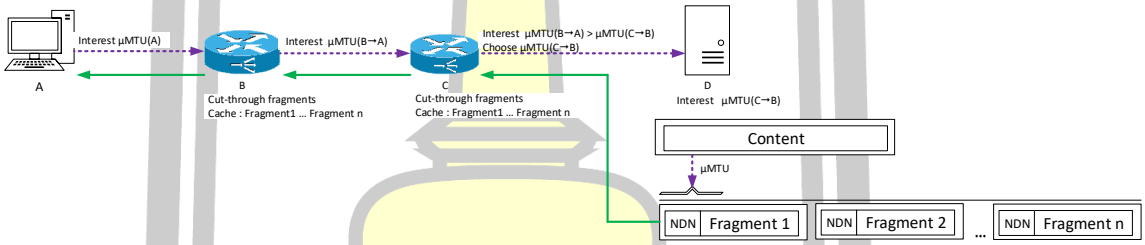


Figure 9 Fragmentation scheme of FIGOA

2.7.3 BEF

Hop by Hop Fragmentation (BEF) [6] proposes a fragmentation and reassembly mechanism. BEF adapts PPP multilink protocol in order to assert incoming packet orders, and uses flag numbers to handle NDN frame-packets. BEF flags consist of *B*, *E*, *BE* and *I*. *B* flag indicates a begin frame-packet, and should be followed by other frame-packets. *E* flag indicates the end of the frame packet. *BE* flag indicates a single frame-packet transmission mode. *I* flag means an idle frame packet, and does not insert any payload. All BEF frame-packets must add *FragSeqNo*, *FragLength* and *FragData* to facilitate a reassembly operation at a receiver.

2.7.4 LFBL

Listen First Broadcast Later (LFBL) [56] learns a network topology by listening broadcast packets. A name of the data packet has been used to design a forwarding path in highly dynamic wireless networks. Network nodes in LFBL actively maintain distances toward destinations by using a simple distance table, including a sequence number, a distance and a variance to destination. A physical mobility of the network node and a logical mobility of data can be managed to

enhance a network connectivity for multi-hop wireless networks. However, LFBL efforts explicitly generate broadcast packets in wireless networks. Network overhead has been occurred, and may downgrade network quality between two end-points.

2.7.5 MAC Address-to-face Mapping

Kietzmann et al. [57] processes to use FIB in the link-layer to reduce network overhead, and improves performance gains for a local resource of network devices. Especially in wireless networks, broadcast packets cannot be handled by some wireless technologies (such as 802.11 and 802.11.5.4). Furthermore, data redundancy in a link-layer network can be increased by using NDN broadcast scheme. MAC address-to-face mapping suggests that an interest and data packet should be sent by using a unicast transmission mode. Kietzmann et al. have shown that the unicast transmission mode potentially reduces the network overhead.

2.7.6 On-Broadcast Self-Learning

Broadcast self-learning [58], proposed by Shi et al., is a technique to learn broadcast *interest* packets. This technique creates a reverse path in PIT with a destination MAC address. A returned *data* packet then uses a unicast transmission mode. The broadcast self-learning has been applied to use in Ethernet switches maintaining three operations. The first operation is to create FIB for NDN forwarding. The second operation recovers physical and logical link-failures. The third operation diverts *interest* packets to a nearest cache in an NDN router to off-load the internet traffic. These three operations have been maintained in a forwarding table by simply listening broadcast packets. High dynamic networks in NDN has been improved by the On-Broadcast Self-Learning, and efficiently delivers data packets in mobility networks.

2.7.7 NDN-NIC

Named-based Filtering on Network Interface Card (NDN-NIC) [59], proposed by Shi et al., has tried to add a name-based filtering for network interface cards. An *interest* and *data* packet in the link-layer is currently distributed into local networks, and processes in a user-space program (i.e. NDN forwarder). NDN forwarders may aggressively consume power because irrelevant packets cause CPU to be awaken. To process many packets in the user-space program, CPU overhead may be increased by processing unnecessary packets. NDN-NIC mitigates power consumption and CPU overhead by adding Bloom filters (BFs) into NICs. Experimental results of NDN-NIC show a significant gain against the user-space program, and reduce CPU overhead by approximately 95%.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

A cycle of theories and experiments is created to explain experimental results in certain conditions. A precise construction and measurement require a rigorously scientific method, which is reproducible by research communities. This dissertation aims to evaluate NDN-NDP and *aMTU*. In doing so, a rigorous statistical analysis and a strong performance evaluation must be considered in order to produce precise values of the experimental results. Four evaluation techniques have been studied. According to Puangpronpitag [60], an evaluation technique has been applied to this dissertation. In this chapter, we first describe choices of performance evaluation techniques, especially in a network emulation. Evaluation techniques report in a section 3.2 Section 3.3 describes Common Open Research Emulator (CORE) for gathering experimental results. Section 3.5 presents an analysis of the experimental results, confidence intervals, and justification. In section 3.4, validation results of CORE emulator have been provided.

3.2 Performance Evaluation Techniques

Four evaluation techniques have been studied, consisting of an analytical modelling, simulation, testbed and emulation. A benchmarking process for our proposed model and previously proposed solution have been provided. The benchmarking process of this research follow a System Under Test (SUT) as described by Bouckaert et al. [60]. To succeed the SUT, evaluation techniques must be discussed.

3.3 Analytical Modelling

An analytical modelling provides a methodical function to explore theoretical problems. The methodical function can be done using an analytical tool. To study scientifically, a set of assumptions are used to construct a model, and run to understand the system. As shown in Figure 10, a real-world problem formulates a real model for a practical use of the situation. A mathematical model then constructs to analyze by using a computer model that generates stochastic results. Experimental results must be rigorously analyzed to conclude an experiment for the real world problem. This cycle in Figure 10 generally done in a complex scenario.

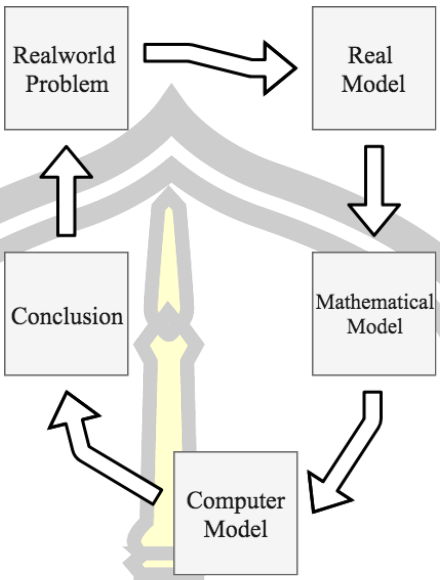


Figure 10 A cycle of the analytical modelling

According to Puangpronpitag [60], the analytical modelling may not be a realistic technique to evaluate NDN-NDP for the following reasons:

1. The analytical modelling is too simplification, which may be inaccurate in real world situation.
2. A network interaction in some critical scenarios may be required. The analytical modelling, however, ignores such an interaction.
3. The analytical modelling reduces precise results in terms of energy consumption, memory, battery and processing power.

Through an analysis of the analytical modelling indicate that it is not suitable to study our NDN-NDP. A network simulation method seems to be attractive, and presents a fewer difficulties than the analytical modelling.

3.4 Network Simulation

A network simulation has been widely used in communication networks. It is easy to simulate WAN networks. Network simulation tools such as ns-2 [61], ns-3 [62] and OMNeT++ [63] are potential in both wired and wireless evaluation techniques. Such tools generally perform event-based stochastic processes, as described by Millman et al. [64].

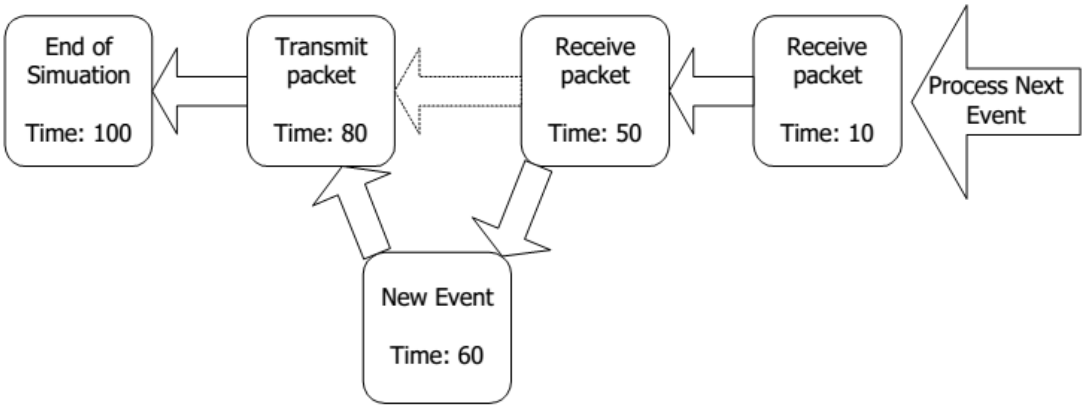


Figure 11 An event-based process in simulator
Source: [60]

Researchers in computer network commonly accept the network simulation as a key tool to explore new network paradigms. Furthermore, the network simulation is easy to improve and to validate existing network protocols. Once a simulation has been done with promising results, such a simulation model will be implemented for real world availability.

To evaluate through network simulation, discrete events in the network simulation help researchers understand protocol behaviors. Figure 11 carries a full sequence of the discrete event system. It maintains the set, called Future Event List (FEL), as follows.

1. Initialized: simulators insert events into FEL.
2. Event Loop: an event loop fetches and inserts events from FEL into a packet processing procedure, and repeats for all events in FEL.
3. Packet Processing: packets follow a simulator network stack, and researcher tasks. All packets are fetched by timestamp orders. The timestamp orders commonly ensure to fetch into an event loop without affecting from prior events.
4. Statistical Report: if all packets have been proceeded, the simulators then generate statistical reports at the end of simulation.

Even though a network simulation has been evaluated with discrete events, there are some drawbacks [64]. Firstly, simulation reports are described in terms of the estimated means and variances. Such reports statistically rely on Gausian distribution. However, the Gausian distribution is commonly used in a real system. In addition, experimental results explain via Central Limit theorem. Secondly, the computer simulation indeed raises a random variable. It is inefficient to claim estimated means and variances. Thirdly, while NDN-NDP involves in an MTU, network simulators do not support a variable size of MTUs. The computer simulation then potentially produces misleading results in this dissertation context.

For experimenting NDN networks, NdnSim [65, 66] plays as a key role in evaluating NDN networks. However, the link-layer in NdnSim does not carry a realistic traffic.

3.5 Testbed Network

A testbed network involves in hardware and software administration. Nodes in the network have been connected with real hardware. Experimental results on the testbed network may give a high accuracy because the real hardware are used. In Figure 12 and Figure 13, for example, an NDN research community has been collaborated to provide a global NDN testbed. In such a testbed environment, NDN protocols must be overlaid over TCP and UDP in order to route NDN packets over the Internet. According to Puangpronpitag [60], several environment parameters (such as workload) can be disrupted by unpredictable parameters in the Internet. Uncontrollable experiment parameters can cause problems to experimental results. The testbed network may be an unreliable solution for experimenting NDN-NDP.

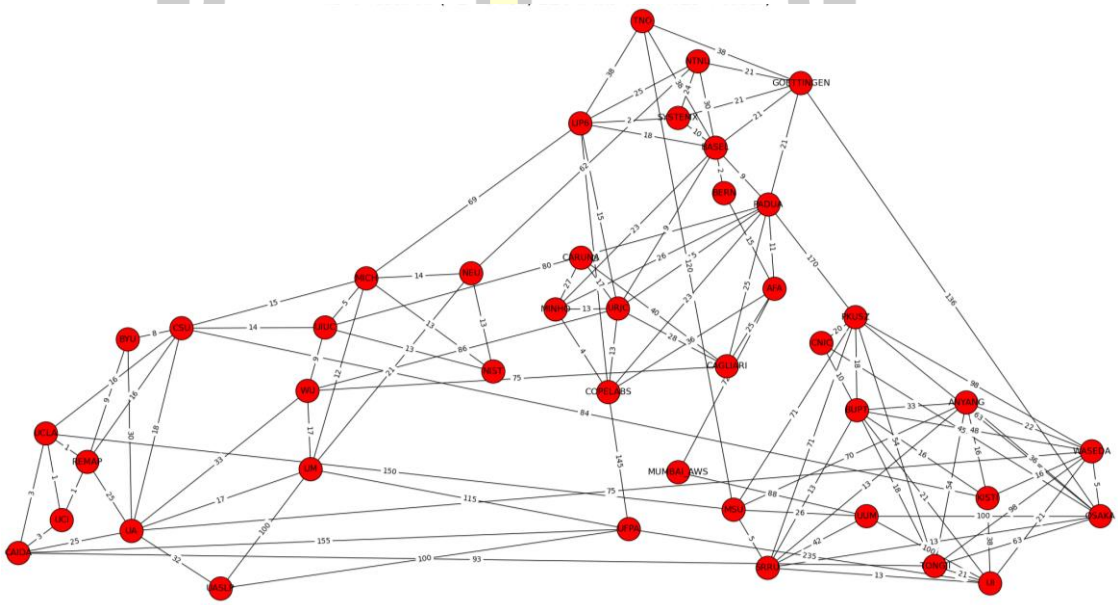


Figure 12 A global NDN testbed for experimenting NDN networks

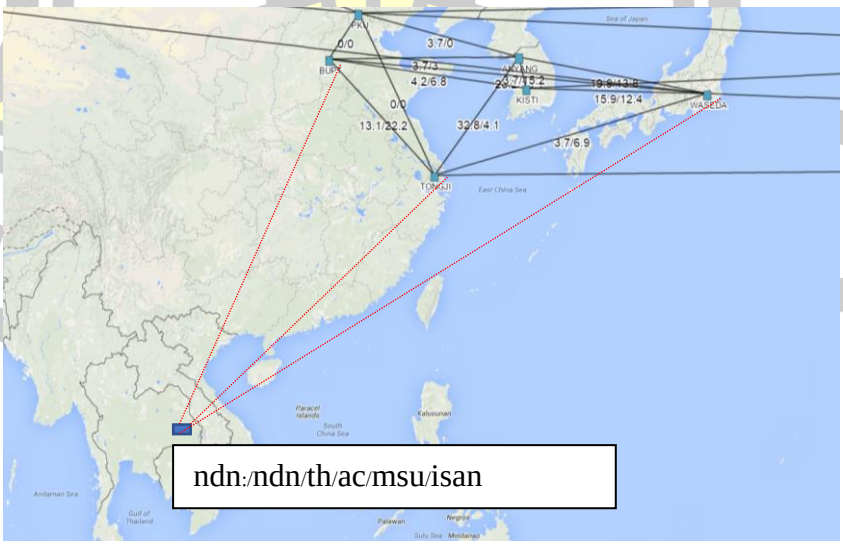


Figure 13 An NDN node of the ISAN research group on the NDN testbed

3.6 Network Emulation

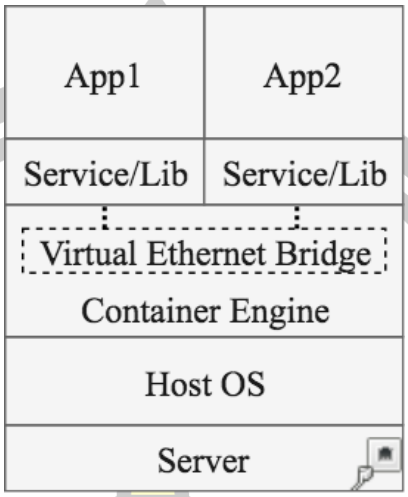


Figure 14 An emulation structure

An emulation technique creates network nodes using virtualization technologies (such as Kernel-based Virtual Machine (KVM) [67] and Linux Container (LXC) [68]). The network emulators provide a realistic network connection. It is managed by a real operating system. In Figure 14, an emulator node is virtualized in both operating systems and network interfaces.

Experimental results from a network emulator could be created with a high fidelity and correctness. Moreover, the emulation technique is cheaper and more controllable than the network testbed.

Emulation tools (such as Common Open Research Emulator (CORE) [9], Integrated Multiprotocol Network Emulator/Simulator (IMUNES) [69], Cloonix [70], and Mininet [71]) are based on virtualization technologies.

In this dissertation, emulation tools have been investigated to elect as a tool for experimenting NDN-NDP. Cloonix, Mininet and Netkit are difficult to form testbed nodes because these tools are based on command-line interfaces. CORE is fully supported to forward NDN packets over link-layer networks. Figure 15 illustrates a simple CORE interface consisting of *n2* and *eth0*. The *n2* is directly connected with a real network interface *eth1*. Through these CORE settings, we can create an effective network scenario for evaluation our design with previously proposed solutions.

Figure 16 indicates an example of NDN connection to the NDN testbed. A circle (1) presents a network interface with address-less IP. A circle (2) demonstrates an NDN ping command to */ndn/edu/ucal* through */ndn/th/msu/isan* router. A circle (3) presents the ping results to */ndn/edu/ucal*. The results have indicated that CORE is fully supported NDN over link-layer networks. So, CORE is selected to evaluate the proposed solutions of this dissertation.

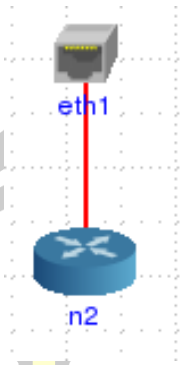


Figure 15 A simple CORE network scenario

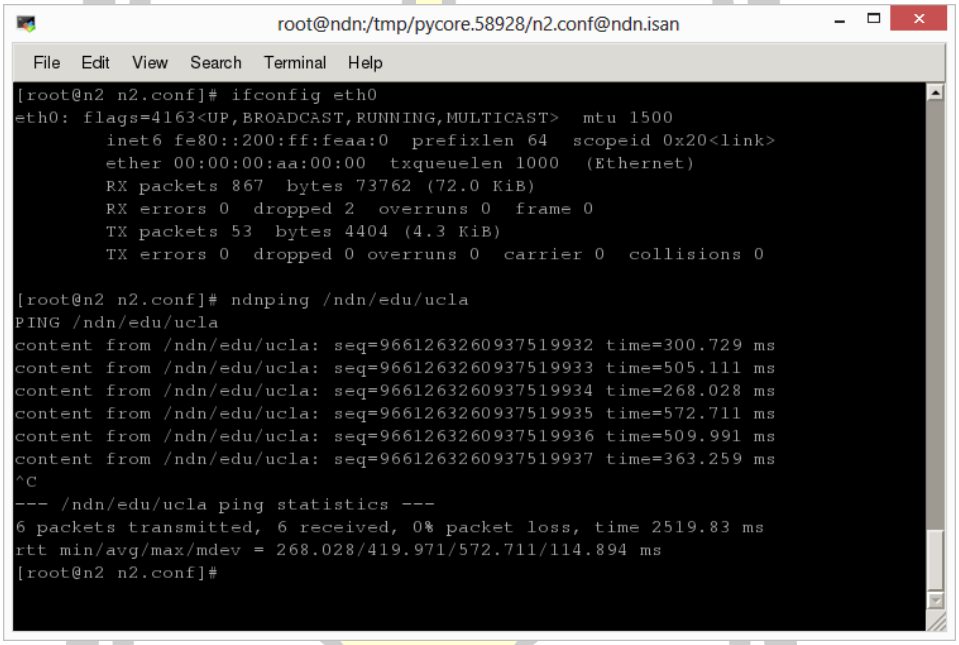


Figure 16 NFD over a link-layer network

3.7 CORE Emulator

CORE emulator has been developed by the U.S. Naval Research Laboratory [72], and aims at making a high fidelity of a network emulation tool. CORE architecture is similar to common network emulators, as described in the previous section. Figure 17 presents CORE architecture. In this architecture, LXC plays a key role to separate CORE nodes. CORE services are used to manage bridge networks, and form a virtual network topology. CORE runs on Linux platform (such as CentOS and Ubuntu) with a simple GUI.

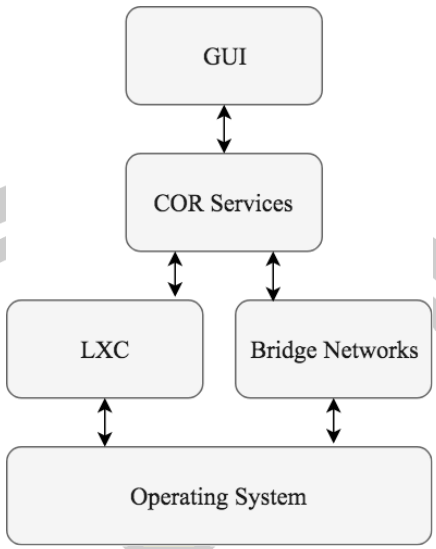


Figure 17 CORE architecture

3.8 CORE Validation

A validation process verifies a correctness of the emulation implementation. In this dissertation, CORE forms virtual nodes, attached into a bridge network. Its link-layer protocol is Ethernet. To validate the CORE emulator, a simple network topology has been created, as shown in Figure 18. A link-speed for the emulated nodes run with the 10/100/1000 NIC. There are three nodes in this simple topology, including a consumer, a producer and an NDN forwarder. An NDN forwarder is located between the producer and consumer. Theoretically, it should be 100Mbps for the link-speed. Through a validation process, experimental results of this validation should be valid and strong for a statistical analysis.

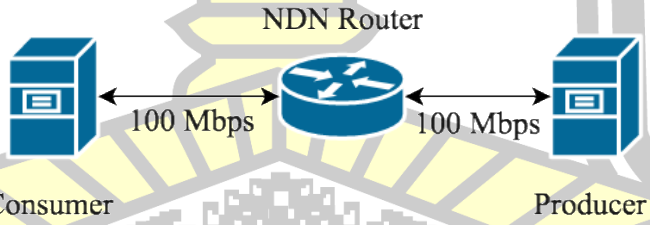


Figure 18 A validation topology for measuring CORE emulator

Figure 19 demonstrates a performance comparison of interest and data transmission using real hardware and CORE emulator. An x-axis is a total run of the experiment. For y-axis indicates a network throughput in Kbps. The real hardware shows the network throughput approximately 94.02±0.08 Kbps. For CORE, it provides bandwidth around 79.95±1.74 Kbps. Such results have suggested that CORE emulator slightly decrease bandwidth in comparison with the real hardware. Indeed, CORE processes under the same hardware that use the same CPU to process packets. Yet, the results of CORE emulator are still steady and reliable to use in this dissertation.

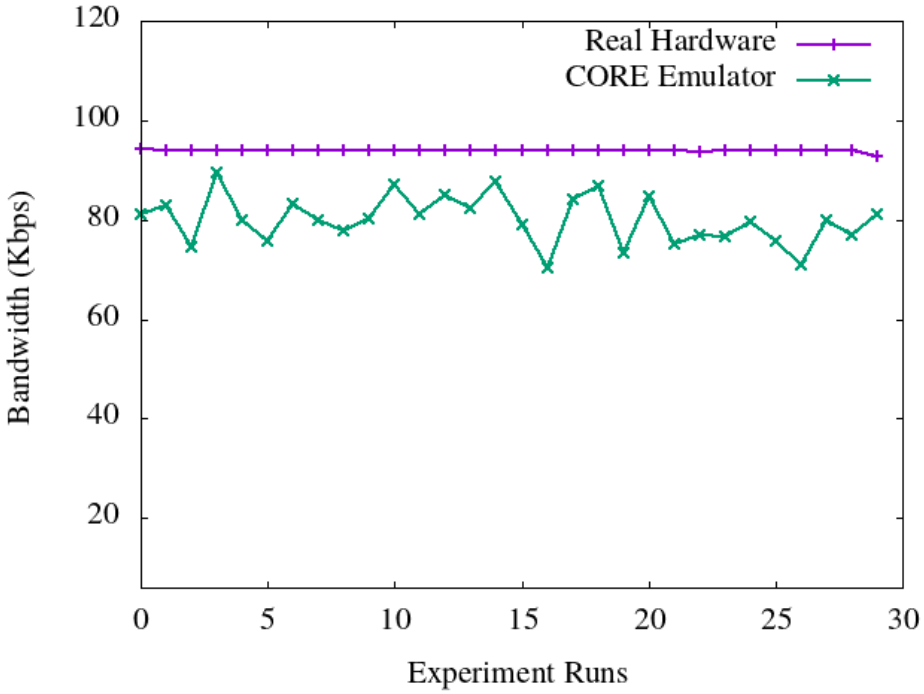


Figure 19 Validation results of real hardware and CORE emulator

3.9 Result Analysis and Confidence Interval

In this dissertation, each experiment is run for 10 minutes. Moreover, each experiment has been repeatedly run for 30 times. Experimental results of NDN-NDP and NDNLP would be generalized, and produces a sufficient amount to make a precise decision. To take a confident result, an average value from the experimental results is marked with *Confidence Interval* of 95%. The results can be traced from our emulated network to calculate as follows:

1. An averaged results $\bar{x}$ can be calculated as:

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

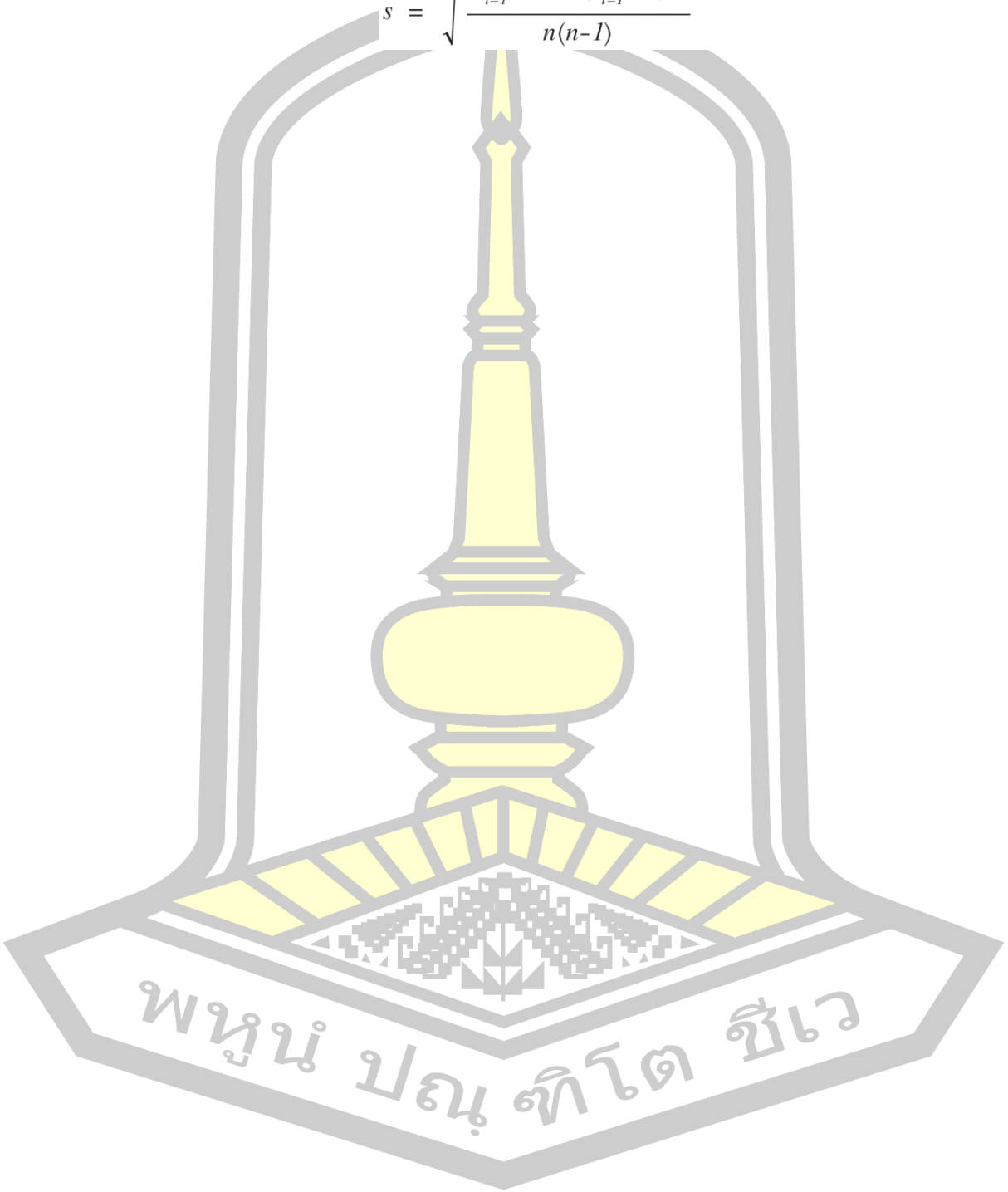
where n is a number of runs, and xi is the result of each run.

2. To mark $\bar{x}$ with *Confidence Interval*, *Significance Level* ($\propto$) of 95% *Confident Level* is 0.05. So, the averaged 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:

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



CHAPTER 4

OUR PROTOCOL DESIGN

4.1 Introduction

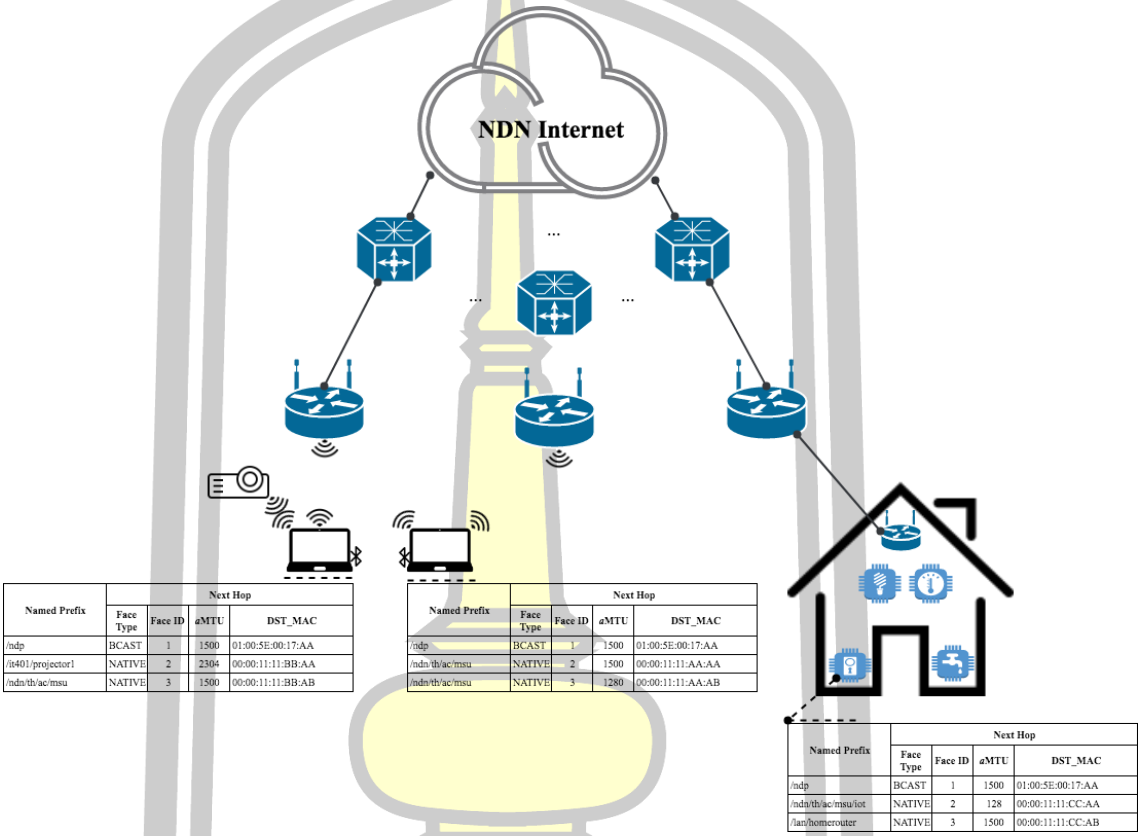


Figure 20 A design overview

In this chapter, our design and implementation to improve NDN over a link-layer network have been explained. As mentioned in chapter 2, the MTU mismatch and broadcast overhead can cause problems to NDN over the link-layer networks. Furthermore, there is no explicit solution to develop a neighborhood discovery protocol for these link-layer networks.

For the design of TCP/IP Internet architecture, neighborhood discovery protocols are widely used, namely Address Resolution Protocol (ARP) for IP version 4 and Neighborhood Discovery Protocol (NDP) for IP version 6. ARP and NDP play an important role to discover their neighbors for unicast communication. In NDN, HBH-FR is a main proposal for NDN over the link-layer networks. Subsequently, there are research proposals (such as NDNLP, BEF and FIGOA) to standardize the link-layer networks. However, these proposals have still faced the described problems.

This dissertation then proposes a novel NDN Neighborhood Discovery Protocol for NDN link-layer, and name it “NDN-NDP”. Our NDN-NDP includes the following mechanisms, namely NDN Link-layer Unicast Face (*NLUF*), NDN-NDP operations,

and adaptive MTU (*aMTU*). As show in Figure 20, an over view of network topology with an NDN-NDP table. NDN-NDP helps create a *NLUF*, and produces the NDN-NDP table for making the unicast transmission. Following sections in this chapter provide NLUF, NDN-NDP operation, *aMTU*, and implementation.

4.2 NDN Link-layer Unicast Face

In NDNLP, *NDN Link-layer Broadcast Face* (*NLBF*) is an only choice for communicating over NDN link-layer networks [4]. This *NLBF* does not support unicast communication. So, we propose a new NDN Link-layer Unicast Face (*NLUF*) that supports unicast transmission over link-layer networks. Not only *NLUF* can reduce the broadcast overhead, but also it can fix the MTU mismatch problem. The details of creating and deploying *NLUF* are discussed as follows.

1. First of all, an adaptive MTU (*aMTU*) table is created and maintained at each node, to record an *aMTU* size of each face (in the Face Table), as shown in the Figure 21 and Figure 22. After that, the face MTU of each *NLUF* is set to *aMTU* to avoid the MTU mismatch. The details of an adaptive MTU selection is further discussed in Section 4.4.

2. In order to support unicast communication over link-layer networks, we have to map a destination named prefix to a destination unicast MAC address. To do so, we use *NLUFs* and FIB as follows. At each NDN node, an *NLUF* is created after learning the destination MAC addresses. This *NLUF* maps a destination unicast MAC address with a link-layer local face in the Face Table (as shown in Figure 21). After that, *NLUF* may be mapped further to named prefixes in FIB (as shown in Figure 23).

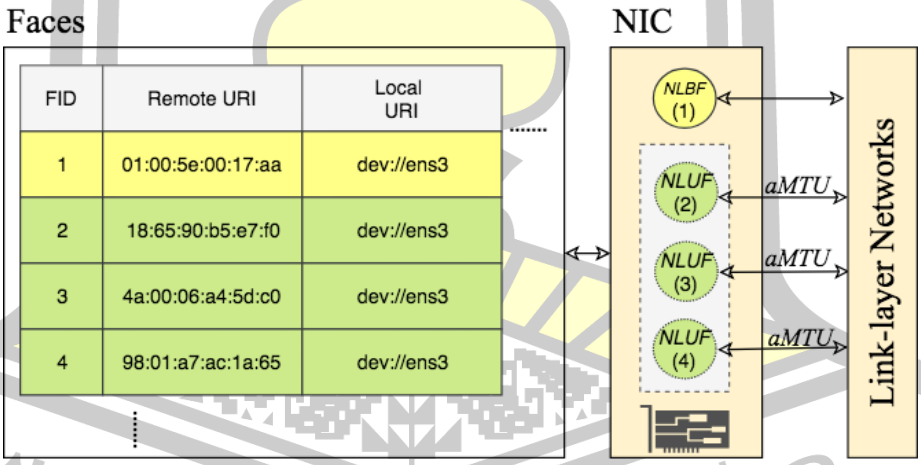


Figure 21 A face table

FID	MTU
1	9000
2	9000
3	2300
4	1280

Figure 22 An adaptive MTU table

Named Prefix	FID
/lab/isan/movie1	2
/	3, 4

Figure 23 FIB

Examples are shown in Figure 21 to Figure 23. The face, with FID=2, is an *NLUF*. It maps a destination MAC address of *18:65:90:b5:e7:f0* to an Ethernet local face (*dev://ens3*), as shown in the face table (in Figure 21). This face uses a very big aMTU size of 9000 bytes (as shown in the adaptive MTU table, in Figure 22) to transfer a movie. In FIB (shown in Figure 23), this *NLUF* is further mapped to a destination named-prefix of *"/lab/isan/movie1"*.

The *NLUF*, with FID=4, is mapped from an Ethernet local face (*dev://ens3*) to a destination MAC address of *4a:00:06:a4:5d:c0*, which is the MAC address of a forwarder (router). The MTU for this *NLUF* is 1500 bytes. In FIB, this *NLUF* is further mapped to a destination named prefix of "/". which is the default named prefix. For any desired named-prefixes with no route (in FIB) to data packets, this *NLUF* (with FID=4) will be chosen to forward an interest packet to the forwarder. The forwarder then forwards the interest packet further to fetch the contents from outside LAN.

4.3 NDN-NDP Operations

NDN-NDP is deployed to learn the destination MAC addresses, and map a destination named prefix to destination MAC addresses. The overall operations of NDN-NDP have been illustrated in Figure 24. From the figure, there are several end-user devices on a link-layer network, including node *A* and node *B*. Node *A* and *B* have 9000 and 1500 bytes of MTUs respectively. Node *A* is an NDN forwarder. The NDN-NDP operations can be explained as the following steps:

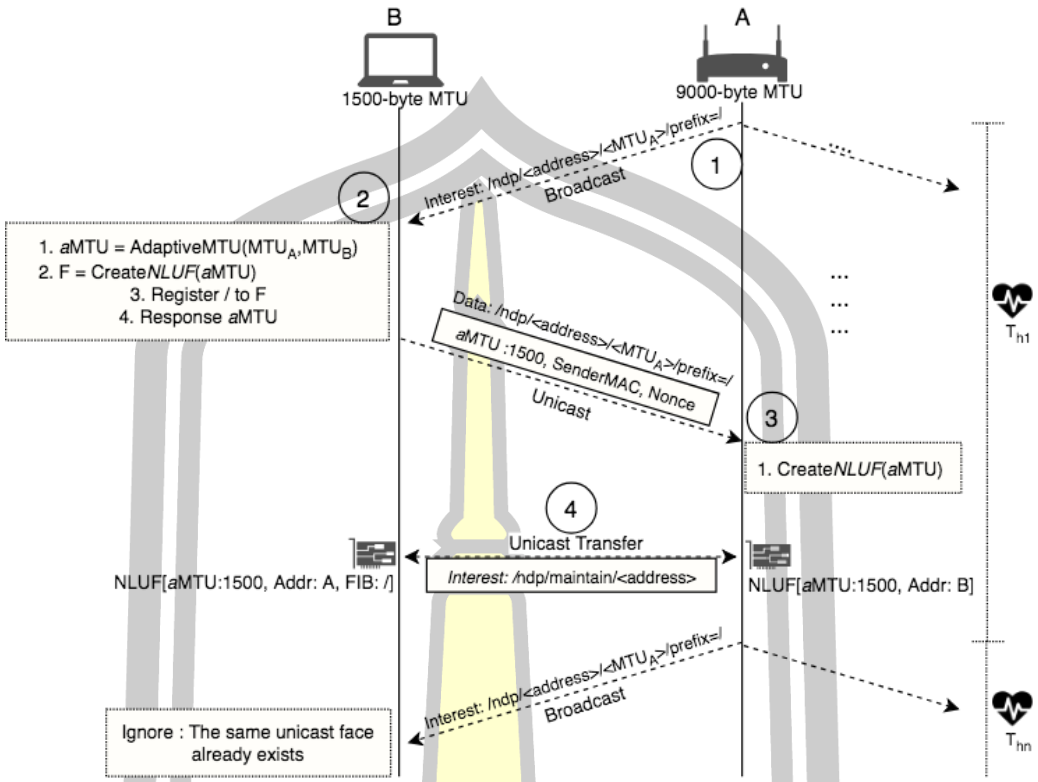


Figure 24 NDN-NDP operations

1. Each node, for example node A, periodically broadcasts a special interest packet, named Neighbor Discovery Interest (*NDI*), to all NDN local devices on the same link-layer network at every *Heartbeat Interval* (*Th*). The "*name*" field of this *NDI* consists of an NDN-NDP identifier, a device's MAC address, an interface MTU, and an optional named prefix (as shown in the *step-1* of the Figure 24). For example, an *NDI* name is */ndp/C0:C1:C0:13:FB:64/9000/prefix=/. /ndp* is an NDN-NDP identifier. *C0:C1:C0:13:FB:64* presents the MAC address of node *A*. It is followed by the MTU of node *A*, which is 9000 bytes. At the end, *"prefix=/"* is the default named prefix, showing that node *A* can act as a forwarder to fetch the Internet contents by forwarding *interest* packets further. If node *A* is a producer, the prefix will show the named prefix of node *A*, for example "*prefix=/ndn/uk/ac/leeds/nodeA*". However, if node *A* is a consumer, this prefix part will not exist.

2. *NDI* from node *A* will arrive at all other NDN nodes in its link-layer network. Each node that received *NDI* from node *A* then responds as follows. For instance, node *B*, after receiving *NDI* from node *A*, will choose an adaptive MTU, *aMTU* from the minimum between its MTU and node *A*'s MTU. From this example, *aMTU* is 1500 bytes. Node *B* can then create an *NLUF* to node *A* with the MTU size equal to *aMTU*. After creating the *NLUF* corresponding to the *NDI* of node *A*, node *B* will ignore all incoming *NDIs* from node *A*. Yet, it will deploy and maintain the created *NLUF* for communicating with node *A*.

3. If a named prefix (in this example, "/") is attached in *NDI*, node *B* will map the *NLUF* to the named prefix in *FIB*.

4. Furthermore, node *B* responds by sending a Neighbor Reply Data (*NRD*) packet back to node *A* via the *NLUF*. The name field of *NRD* is the same as the

received *NDI*. This *NRD* contains two important information for node *A*: *aMTU* and the MAC address of node *B*. *NRD* must also be digitally signed in order to be validated in the next step.

5. After receiving *NRD* from node *B* and validating its legitimacy, node *A* creates a new *NLUF* to node *B*, as shown in the *step-3* of the Figure 24. This *NLUF* has its MTU size equal to *aMTU*.

6. After *NLUFs* between node *A* and *B* have been successfully created, the NDN communication between these two nodes over the link-layer network can be done in the unicast mode. An idle *NLUF* for a period of time will be destroyed, assuming that the link has been disconnected.

7. To keep *NLUFs* alive, both *A* and *B* maintain their *NLUFs* by sending a special interest message (called *"NLUF maintenance message"*) to each other for every Th. Without receiving the *NLUF* maintenance message for a certain period of time, the *NLUF* record will be deleted.

4.4 Adaptive MTU

To solve the MTU mismatch, this dissertation proposes to use the minimal hop-by-hop MTU between two nodes as an adaptive MTU (*aMTU*) of their *faces*. By exchanging *NDI* and *NRD* between two nodes in the NDN-NDP operations (as described in section 4.3), the minimum MTU between them is selected as *aMTU* for HBH-FR. Figure 25 illustrates an example of *aMTU*. According to the figure, there are three nodes, including *A*, *B* and *C*. Node *A* and *C* supports jumbo-frame [19] MTU of 9000 bytes. Node *B* is an IoT device, merely supporting MTU of 127 bytes [50], [51]. According to NDN-NDP operations, node *A* propagates an *NDI* packet, carrying a device's MTU. Node *B* then compares the received MTU with its local NIC's MTU, and creates an *NLUF* to node *A* with an *aMTU* of 127 bytes. Node *B* then responses the *aMTU* value back to inform node *A* via an *NRD*. Node *A* finally creates a new *NLUF* with the *aMTU* of 127 bytes to node *B*. For node *A* and node *C* operations, *NLUFs* will be created by using the same operations as node *A* and node *B*, and vice versa.

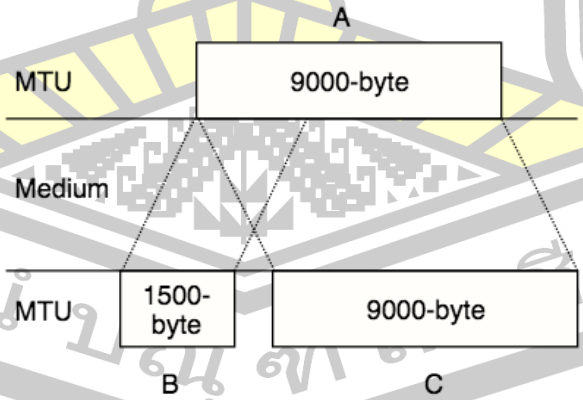


Figure 25 An example of adaptive MTUs

4.5 NLUF Security

To avoid a similar ARP-spoof attack in NDN over link-layer networks, a security vulnerability in a classical Ethernet has been explored. From the literature

[75], ARP-spoof can deviate a communication path to capture and manipulate sensitive information. It finally leads to Denied of Service (DoS) attack, replay attack and Man-In-The-Middle (MITM) attack. To protect against the similar problem, NDN over link-layer networks needs a verification of interest and data packets before making a unicast transmission. So, we have defined procedures to secure the *NLUF* as follows:

1. Signature: The digital signature in an *NRD* packet must be verified before taking any actions.
2. Verification: The incoming *NRD* packet must be corresponded to the outgoing *NDI* packet. This is done by using a nonce.
3. Capturing attacks: Unsolicited *NRD* packets have been analyzed to quarantine a risky behavior of neighbor nodes.
4. Dropping attacks: By capturing and analyzing unsolicited *NRD* packets, we use a threshold (*T*), which is adjustable according to network environment, to recognize undesired behaviors, and drop all *NRD* packets received from attackers.

Even with the above security procedures, other in-depth security issues would be further studied. It is not in the scope of this dissertation, but should be done as future work.

4.6 NDN-NDP Packet Format

NDN-NDP defines a format of the *interest* and *data* packet, as shown in Figure 26. In the left of the figure, it is our *NDI* packet format. For the right figure, our *NRD* is presented. To complete our previously described NDN operations, the *NDI* inserts */ndp* as the first named prefix of the *NDI* packet to identify NDN-NDP protocol. It is followed by a *DeviceType* field to identify a basis link-layer protocol. *LinkAddress* is later added to keep a MAC address of the *NDI* sender. An optional named prefix in the special parameters section will be inserted if the *NDI* sender can be acted as an NDN forwarder. For the last *Nonce* field, is used to check with an incoming *NRD* that is needed to correspond to the *NDI* packet.

For *NRD* packet format, a neighbor, who has received an *NDI* packet, calculates a received MTU by using our *aMTU*, and assigns the calculated MTU's value into *NLUF*. The *NRD* named prefix is copied from an incoming *NDI*. *NRD* adds its MAC address and *aMTU* into *NRD* payload. Then, *interest Nonce* and *NRD* signature will be inserted.

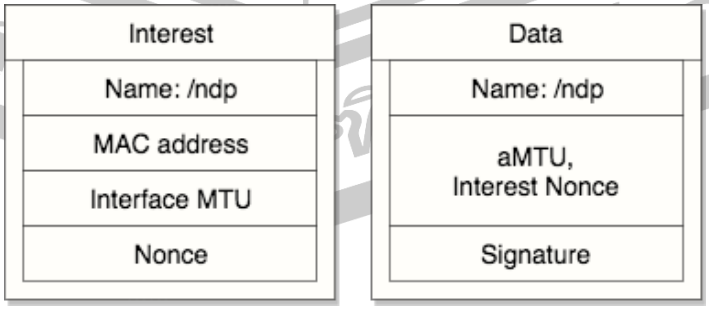


Figure 26 NDN-NDP packet format

4.7 Implementation

We have built our NDN-NDP protocol by extending NFD-0.5.0 [8]. Implementation has been done to handle NDN-NDP operations, *aMTU* selection and *NLUF* management. An identification of NDN link-layer is defined as *0x8624*. The *NLUF* management is implemented in the *NFD Ethernet factory*. It connects with the *NFD Ethernet transport* for multiplexing frame-packets. Finally, validation tests on our implementation have been done extensively.

CHAPTER 5

PERFORMANCE EVALUATION

5.1 Introduction

In the previous chapter, a new NDN-NDP protocol have been proposed and developed. The new NDN-NDP could cut of overhead, and solve MTU mismatch problem. This chapter compares NDN-NDP with a state-of-art link-layer protocol for NDN, called NDNLP. To investigate the performance of our protocol, an emulation technique was deployed to compare with NDNLP.

The CORE emulator is used to emulate our experimental testbed. To do so, we have designed a simple linear topology. In our experiments, we compared our NDN-NDP with NDNLP, which is the most widely deployed NDN link-layer protocol. So, for each MTU setting, the experiments have been run twice: one for NDN-NDP and the other one for NDNLP. Each experiment was run for 10 minutes. Moreover, each experiment was repeatedly run for 30 times. Experimental results were averaged and quoted from the 30 runs with respect to confidence interval of 95%.

The experimental results were collected over 20GB including throughput, delay, the number of unsatisfied interest and the number of broadcast packets. The throughput indicated the number of delivered data packets per second. The delay reflected a period of time to get data packets. The number of unsatisfied interest packets presented the number of failures to deliver data packets.

5.2 Network Scenario

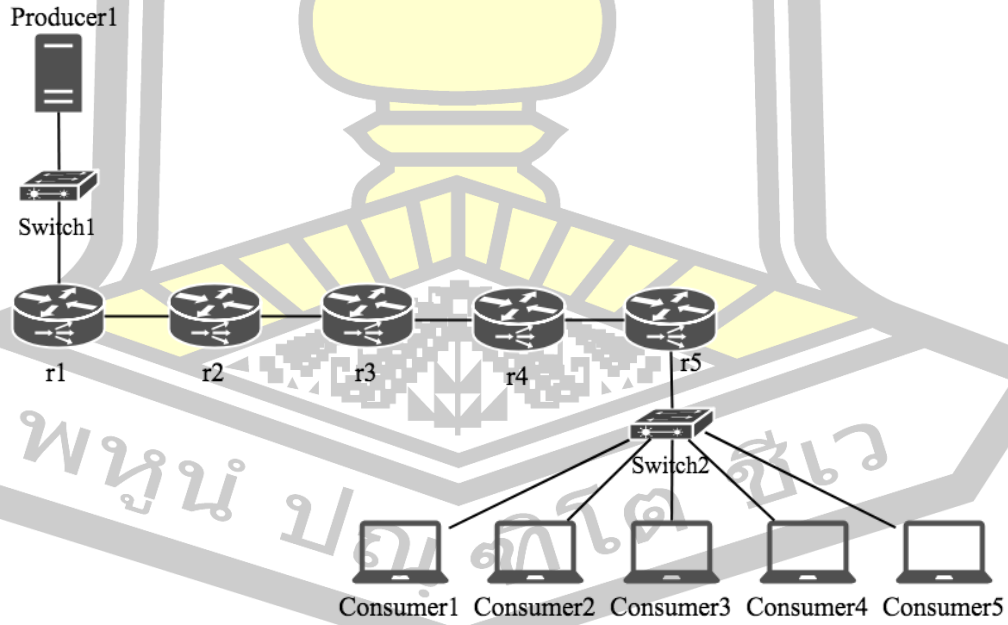


Figure 27 Network scenario

Figure 27 shows our experimental network scenario using a simple linear topology. An NDN producer, five NDN forwarders, and five consumers are connected on the topology. *Producer1* provides NDN contents using *ndnperf-server* [73]. Five

NDN consumers were connected at the *r5* forwarder. They simultaneously retrieved data packets from *Producer1* by sending *interest* packets through the forwarders to consume all available bandwidth.

5.3 Emulation Parameters

We controlled CORE nodes to initiate parameters. CORE created a collection of network nodes. By using network bridges, the network nodes have been connected. In terms of hardware, we run experiments on *Intel(R) Xeon(R) CPUs E3-1220 V2 @ 3.10GHz*. We used 100Mbps NICs. For a basis OS, we emulated network nodes on *Linux Ubuntu 16.04*. The experimental parameters are described as follows:

1. NDN data packets: NDN data packets with the size of 8,192 bytes have been randomly generated by an *ndnperf-server* [73]. The lifetime of these packets have been set to 10,000 ms to ensure 10 seconds of caching. These packet sizes and lifetime of NDN *data* packets were the default values, used by the *ndnperf-server*.
2. Heartbeat Interval (*Th*): Th has been set to 3 seconds for all experiments.
3. MTU: Experiments were run in three different MTU settings to test the parameter sensitivity. The first setting used the MTU size of 1500 bytes for all nodes (a producer, five consumers and five forwarders in the scenario). The second setting used the MTU size of 9000 bytes for all nodes, and the third setting used a random MTU size (from this size list: 1492, 1500, 4392, 9000, or 65820 bytes) for each node. The MTU of 1500 bytes represented a standard Ethernet. The MTU of 9000 bytes represented a jumbo-frame. The random MTU size of 1492 bytes or 4352 bytes or 9000 bytes or 65820 bytes represented the MTU of different standards.

5.4 Performance Metrics

The number of broadcast packets, network throughput, delay and the number of unsatisfied *interest* have been used as our performance metrics. Each metric can be described as follows:

1. The number of broadcast packets: Broadcast packets in an NDN link-layer network are actually the multicast packets, sending over the default *ICN-MCAST* (MAC Address *01:00:5E:00:17:AA*). By inspecting the destination MAC address *01:00:5E:00:17:AA*, all incoming *interest* and *data* broadcast packets will be counted.
2. Network throughput: Network throughput in this dissertation is the rate of successful NDN *data* packet delivery over the experimental NDN link-layer. It is represented in kilobit per second (Kbps). We use *ndnperf* [73] to report the averaged network throughput of our five consumers.
3. Delay: Delay in this dissertation is the averaged time for NDN data packets travelling towards the consumers. It is reported in millisecond (ms).
4. The number of unsatisfied *interests*: The number of unsatisfied *interests* can be used to measure forwarding performance. Outgoing interest packets with no returned *data* packet are counted.

5.5 Experimental Results

The experimental results for each metric are discussed in the following sub-sections.

5.5.1 The Number of Broadcast Packets

In our experiments, the number of broadcast *interest* packets and the number of broadcast *data* packets between NDN-NDP and NDNLP were compared for three different MTU settings. The comparative results are given in Table 1. In summary, by proposing unicast *faces*, NDN-NDP can reduce the number of broadcast packets dramatically in comparison to NDNLP. This reduction would mitigate the broadcast overhead, and finally improve overall efficiency.

Table 1 Comparing the number of broadcast packets between NDN-NDP and NDNLP

MTU settings	1500 bytes		9000 bytes		Random	
Link-layer protocol	NDN-NDP	NDNLP	NDN-NDP	NDNLP	NDN-NDP	NDNLP
#of broadcast interest packets	580±33	262,309± 23,486	453±33	1,176,167± 77,333	501±32	303,699± 24,308
#of broadcast data packets	3±7	265,712± 23,710	3±8	1,315,513± 84,882	3±7	309,421± 24,280

5.5.2 Network Throughput

Network throughput is potentially influenced by MTU sizes. The larger the MTU of a connection, the more data that can be transmitted in a single frame-packet. Figure 28 compares the network throughput between NDN-NDP and NDNLP for different MTU settings. For the 1500-byte MTU setting, NDN-NDP and NDNLP have gained throughput of 20,307±584 Kbps and 17,266±415 Kbps respectively. Also, for the 9000-byte MTU setting, NDN-NDP and NDNLP have shown throughput of 36,702±1,076 Kbps and 28,189±389 Kbps respectively. The experimental results of the network throughput have demonstrated that the bigger MTU can effectively gain more throughput than the smaller. For the random-size MTU setting, NDN-NDP and NDNLP give network throughput of 23,363±427 Kbps and 16,950±368 Kbps respectively. Overall results have illustrated that NDN-NDP gains more network throughput than NDNLP. This success is because unicast *faces* of NDN-NDP utilize the link-layer network more efficiently than the multicast *faces* of NDNLP.

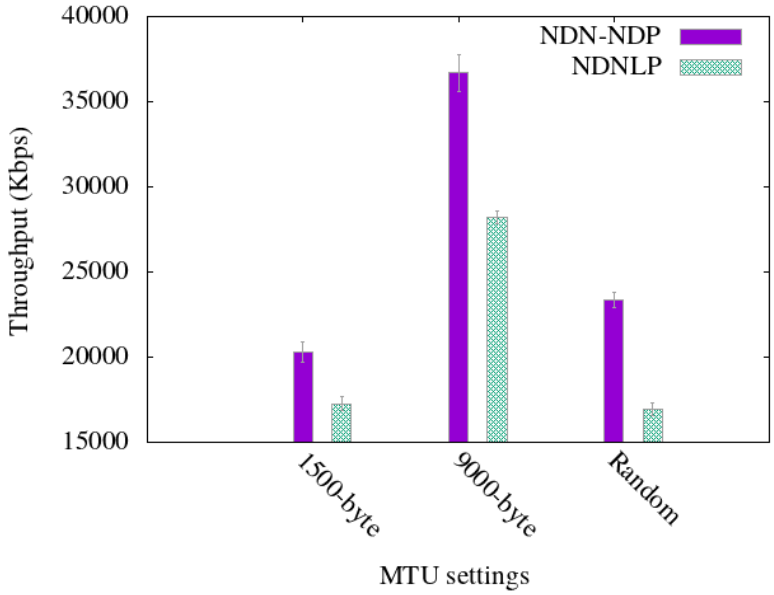


Figure 28 A network throughput comparison of NDN-NDP and NDNLP

5.5.3 Delay

The comparative delay between NDN-NDP and NDNLP for different MTU settings is illustrated in Figure 29. For all MTU settings, NDN-NDP consistently incurred less delay than NDNLP. The results of NDN-NDP comparing to NDNLP were 125±24 ms and 238±55 ms respectively for the 1500-byte MTU setting, 55±4 ms and 73±8 ms respectively for the 9000-byte MTU setting, and 108±63 ms and 225±86 ms respectively for the random-size MTU setting. Therefore, the overall experimental results have pointed that NDN-NDP can reduce delay, which effectively enhance NDN over link-layer networks.

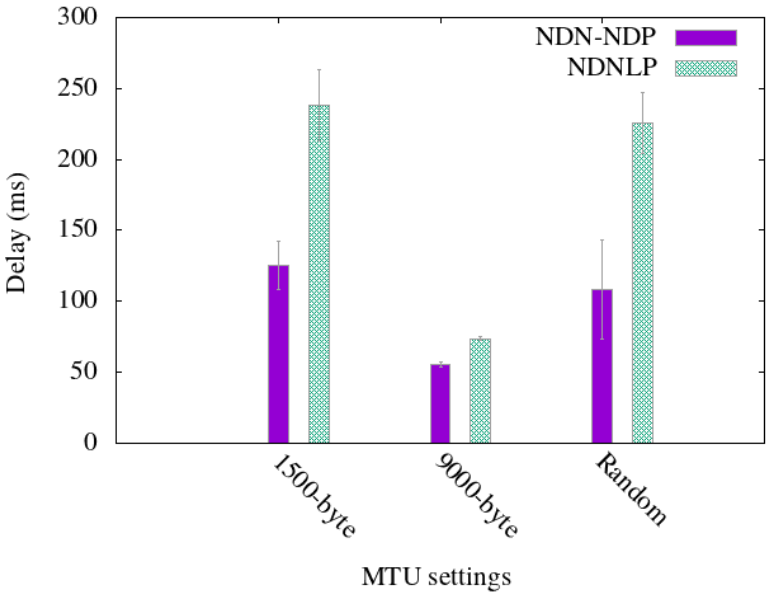


Figure 29 A comparison of transmission delay among different MTU settings

5.5.4 The Number of Unsatisfied *Interest*

NDN unsatisfied *interests* are the *interest* packets that fail to retrieve NDN data packets. From the experiments, the number of unsatisfied *interests* of NDN-NDP and NDNLP were comparatively counted. As shown in Figure 30, the number of unsatisfied *interests* of NDN-NDP is 323±14, 311±5 and 346±13 for the 1500-byte, 9000-byte and the random-size MTU settings respectively, while the number of unsatisfied *interests* of NDNLP is 357±2, 389±2 and 362±2 for the same MTU settings. So, the overall number of NDN-NDP unsatisfied *interests* is less than the number of NDNLP unsatisfied *interests*. So, NDN-NDP can also outperform NDNLP for this metric.

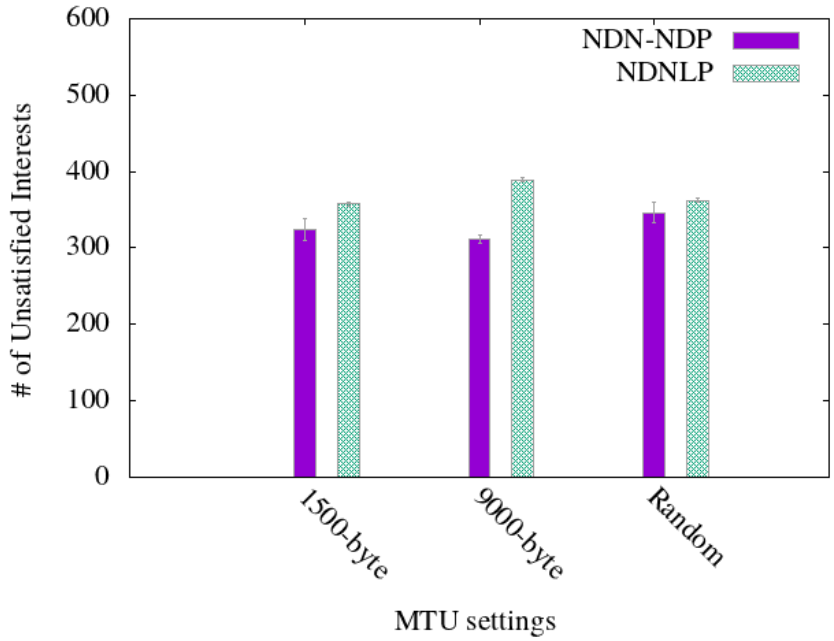


Figure 30 A performance comparison of the number of unsatisfied interest metric

5.6 Security Consideration of NDN-NDP

An NDN-NDP is an ARP-like procedure. A face creates upon receiving neighbor announcements, and negotiates to create a virtual unicast *face* with an *aMTU*. It is possible to attack by launching *interest* spoofing. An attacker simply announces a bogus *interest* packet, and mimics NDN-NDP *data* packets. Therefore, the *interest* spoofing is risky to NDN consumers by capturing, manipulating and blocking NDN frame-packets. To mitigate the link-layer security in using NDN-NDP, a mandatory authentication of the *data* packet is required. A subsequent link-layer policies should be also applied to secure NDN-NDP. A *data* packet of the NDN-NDP packet must be corresponded to an *interest* packet. The attacker is then unable to forge an unauthorized *data* packet in the NDN link-layer.

5.7 Comparing NDN-NDP with other proposals in the literature

5.7.1 NDNLP

NDNLP is the most widely deployed link-layer protocol for NDN. NDNLP relies on multicasting over *ICN-MCAST* group (MAC Address: *01:00:5E:00:17:AA*), which is technically broadcasting to all NDN devices on the link-layer network. NDNLP suffers from both broadcast overhead and MTU mismatch problems, while our NDN-NDP has been designed to avoid the MTU mismatch, and mitigate the broadcast overhead. Moreover, as shown from the experimental results, NDN-NDP can outperform NDNLP.

5.7.2 BEF

BEF proposes a fragmentation and reassembly mechanism. BEF adapts PPP multilink protocol [74] in order to assert incoming packet orders, and uses flag numbers to handle NDN frame packets.

In this dissertation, BEF should be used to compare with our NDN-NDP. However, BEF is based on the CCNx forwarder while our NDN-NDP uses an NDN forwarder. Through functionality analysis, MTU mismatch is the BEF weakness, and does not adapt to complex link-layer networks. These problems can be enhanced by our NDN-NDP with aMTU algorithm.

5.7.3 FIGOA

FIGOA uses a minimal MTU discovery (*µMTU*) technique to mitigate MTU mismatch in NDN networks. The *µMTU* must be discovered to define the maximum size of *interest* and *data* packets to prevent intermediate fragmentation. This idea is similar to the path MTU discovery for IP version 6. By using the minimum path MTU, FIGOA is not vulnerable to the MTU mismatch. However, we argue that *µMTU* of FIGOA would be inefficient in comparison with *aMTU* of our NDN-NDP. The *aMTU* of each hop can be bigger than *µMTU* of the whole end-to-end path. So, the bigger MTU would utilize bandwidth more efficiently.

5.7.4 LFBL

LFBL leans a network topology by listening broadcast packets, and is suitable for highly dynamic wireless networks. Network nodes in LFBL actively maintain distances toward destinations. A simple distance-table has been used to store three values for any known destinations, including a sequence number, a distance and a variance to destination. A physical and logical mobility of *data* packets can be easily managed. However, LFBL efforts explicitly generate broadcast packets. Network overhead has been occurred, and downgrade network quality between local end-points. Such problems can be illuminated by using our NDN-NDP.

5.7.5 MAC address-to-face mapping

MAC address-to-face mapping argues to use FIB in the link-layer to create a unicast *face*. By reducing broadcast packets, MAC address-to-face mapping can more efficiently handle NDN packets in wireless technologies (such as 802.11 and 802.11.5.4), and shows a potential gain from using unicast *faces*. Yet, a heterogeneous

network environment causes the MAC address-to-face mapping from MTU-mismatch problem.

5.7.6 Broadcast Self-learning

Broadcast self-learning is similar to a common mechanism in NDN link-layer. The broadcast self-learning leans a broadcast *interest*, and creates a reverse path in PIT with a destination MAC. A returned *data* packet then uses a unicast transmission. The broadcast self-learning has been applied to use in Ethernet switches maintaining three operations. The first one is to create FIB for NDN forwarding process. The second operation recovers physical and logical link-failures. The third operation diverts *interest* packets to a nearest cache in an NDN router to off-load the internet traffic. However, the broadcast self-learning scheme does not negotiate a frame-packet's size for NIC's MTUs. An MTU-mismatch problem becomes a potential issue.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Summary and Discussion

Named Data Networking (NDN) is very significant to shape the future internet architecture. It is a variant or a proposal of Information Centric Network (ICN) architecture, defined by the IRTF. NDN would provide more efficiency for information distribution. Currently, NDN can distribute data over different types of communication links, such TCP/IP protocol stack and link-layer networks.

By deploying NDN directly on top of link-layer networks instead of TCP/IP protocol stack would remove a huge overhead because some operations of the TCP/IP protocol stacks are cancelled. This overhead cut-off would enhance an efficiency for designing modern network applications. However, NDN link-layer protocols are still at their initial phase. Furthermore, there are few solutions to use a unicast transmission mode in NDN over the link-layer networks. Furthermore, to deploy NDN over the top of link-layer networks, it is risky for a sender to send NDN packets to their neighbors, who cannot be received due to MTU mismatch. Data transmission could be failed from this MTU mismatch problem.

Current NDN over the link-layer networks employs NDNLP to handle NDN frame-packets. By using *ICN-MCAST* address, NDNLP simply broadcasts the NDN frame-packets into the link-layer networks. This simple broadcast mode causes NDN over the link-layer networks to encounter a broadcast overhead problem.

In this dissertation goals these two MTU mismatch and broadcast overhead problems have been investigated and improved. We firstly designed a new neighborhood discovery protocol for NDN (NDN-NDP) to enable a unicast transmission mode. This unicast transmission mode could reduce a number of broadcast packets, and could solve the broadcast overhead problem. In addition, we have proposed an adaptive MTU (*aMTU*) to solve the MTU mismatch problem in NDN over a link-layer network. We have extended our NDN-NDP and *aMTU* into NFD (aka. the most widely used NDN forwarder).

We have deployed a network emulation technique (using CORE emulator as a tool) for our experiments and performance evaluation. For research methodology used in this dissertation, we have found that the analytical modelling and simulation delivered less accurate results than the other techniques because they do not obtain data from a real system. Similarly, a testbed technique is too limited to control network environments. An emulation could carry accurate results by obtaining data from the real system, and can control testbed parameters. Thus, the emulation technique inherently yields the most reliable results.

While an emulation technique is possible to evaluate a network testbed in a single machine, a rigorous and systematic experiment does not a trivial task. There are several parameter settings to be done. In this dissertation, we have rigorously constructed an emulation scenario to study NDN-NDP and NDNLP behaviors. For evaluation criteria, we have established an evaluation scenario to collect results, including the number of broadcast packets, network throughput, delay and the number of unsatisfied *interest*.

Experimental results of this dissertation presents a comparison between NDN-NDP and NDNLP. We first enable a native NDN over link-layer networks. The experimental results have illustrated that: 1) NDN-NDP can cut off the number of broadcast *interest* packets and the number of broadcast *data* packets approximately 99%; 2) Our NDN-NDP increases network throughput around 22% in comparison with NDNLP; 3) NDN-NDP consistently incurred less delay than NDNLP around 46%; 4) The number of unsatisfied *interest* of NDN-NDP occurs less than NDNLP approximately 11%. In summary, our NDN-NDP can efficiently enhance NDN over the link-layer networks.

6.2 Dissertation Achievement

This dissertation has evaluated the previous proposals of NDN over link-layer networks, and proposed a new design of NDN-NDP successfully. The major contribution of the research can be described as follows:

1. We analyzed NDN over the link-layer network, we have found performance criteria to evaluate our NDN-NDP.
2. We solved an MTU mismatch and broadcast problems for NDN over a link-layer network.
3. Our NDN-NDP is a protocol to map between MAC addresses and named data for enabling a unicast transmission mode.
4. We have found NDN-NDP can reduce the number of broadcast packets.
5. Our aMTU can solve the MTU mismatch problem in NDN over a link-layer network.

6.3 Future Work

Even our proposed NDN link-layer protocol can solve both MTU mismatch and broadcast overhead problems, some parts of our work should be extended to use in a real world system. For example, our NDN-NDP can be used in other network area (such as IoT and MANET) for reducing broadcast overhead, delay and the number of unsatisfied *interest*.

A promising results in this dissertation would solve NDN over the link-layer problem in several aspects. However, following remain issues should be investigated to improve our model:

1. Mobility Network Scenario: During working on this dissertation, we have found that a mobility network scenario should be further investigated to understand the delay in an unpredictable movement of NDN nodes.
2. Link-layer Protocols: In this dissertation, our experiment was relied on Ethernet. An experiment on other link-layers (such as IEEE 802.11, IEEE 802.15.4, FDDI and PPP protocol) should be tested to confirm our design and implementation.
3. Heartbeat Interval: In our NDN-NDP have placed a heartbeat interval into our NDN-NDP protocol. It still have a broadcast overhead that is generated by the NDN-NDP itself. A further design of NDN-NDP should remove this interval.

REFERENCES

REFERENCES

[1] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking Named Content," in *Proceedings of CoNEXT* Rome, Italy, 1 - 4 December 2009, pp. 1–12.

[2] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking Named Content," *Communications of the ACM*, vol. 55, pp. 117-124, January 2012.

[3] D. Kutscher, B. Ohlman, and D. Oran. (2018, 5 June). *Information-Centric Networking Research Group*. Available: https://irtf.org/icnrg

[4] J. Shi and B. Zhang, "NDNLP: A Link Protocol for NDN," University of Arizona, NDN Technical Report, NDN-0006, July 2012.

[5] C. Ghali, A. Narayanan, D. Oran, and G. Tsudik, "Secure Fragmentation for Content-Centric Networks," in *Proceedings of IEEE International Symposium on Network Computing and Applications*, Cambridge, MA, USA, 28 - 30 September 2015, pp. 47-56.

[6] M. Mosko and C. Tschudin, "ICN "Begin-End" Hop by Hop Fragmentation draft-mosko-icnrg-beginendfragment-02," IETF, Internet Draft 02, 13 December 2016.

[7] EC2. (5 June 2018). *Network Maximum Transmission Unit (MTU) for Your EC2 Instance*. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/network_mtu.html

[8] NFD, "NFD Developer's Guide," NDN Technical Report NDN-0021, 3 February 2015.

[9] J. Ahrenholz, "Comparison of CORE Network Emulation Platforms," in *Proceedings of MILCOM*, San Jose, California, USA, November 2010, pp. 166 - 171.

[10] Cisco System. (16 September 2016). *Cisco Visual Networking Index*. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html

[11] "Internet Protocol," IETF, RFC 791, September 1981.

[12] G. Grassi, D. Pesavento, G. Pau, R. Vuyyuru, R. Wakikawa, and L. Zhang, "VANET via Named Data Networking," in *Proceedings of INFOCOM*, Toronto, Canada, May 2014.

[13] DARPA. (July 2018). *Defense Advanced Research Projects Agency*. Available: https://www.darpa.mil/

[14] V. G. Cerf and R. E. Kahn, "A Protocol for Packet Network Intercommunication," *IEEE Transactions on Communication*, vol. 5, pp. 637-648, May 1974.

[15] L. Popa, A. Ghodsi, and I. Stoica, "HTTP as the narrow waist of the future internet," in *Proceedings of SIGCOMM*, Monterey, California, 20 - 21 October 2010, pp. 1-6.

[16] S. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, *et al.*, "Less Pain, Most of the Gain: Incrementally Deployable ICN," in *Proceedings of SIGCOMM*, Hong Kong, China, 12 - 16 August 2013, pp. 147-158.

[17] J. Sherry, P. X. Gao, S. Basu, A. Panda, A. Krishnamurthy, C. Maciocco, *et al.*, "Rollback-Recovery for Middleboxes," in *Proceedings of SIGCOMM*, London, United Kingdom, 17-21 August 2015, pp. 227-240.

[18] E. Haleplidis, K. Pentikousis, S. Denazis, J. Salim, and D. Meyer, "Software-

Defined Networking (SDN): Layers and Architecture Terminology," IETF, RFC 7426, January 2015.

[19] I. Seskar, K. Nagaraja, S. Nelson, and D. Raychaudhuri, "MobilityFirst Future Internet Architecture Project," in *Proceedings of Asian Internet Engineering Conference*, Bangkok, Thailand, 09 - 11 November 2011.

[20] J. Lee, B. Kusy, T. Azim, B. Shihada, and P. Levis, "Whirlpool routing for mobility," in *Proceedings of MobiHoc*, Chicago, Illinois, USA, 20 - 24 September 2010, pp. 131-140.

[21] T. Anderson, K. Birman, R. Broberg, M. Caesar, D. Comer, C. Cotton, *et al.*, "NEBULA - A Future Internet That Supports Trustworthy Cloud Computing," in *NEBULA White Paper*, Computer Science Department, Cornell University November 2010.

[22] D. Naylor, M. Mukurjee, P. Agyapong, R. Grandl, R. Kang, and M. Machado, "XIA: Architecting a More Trustworthy and Evolvable Internet," *ACM SIGCOMM Computer Communication Review*, vol. 44, July 2014.

[23] 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.

[24] L. Muscariello, J. Augé, M. Papalini, M. Sardara, and A. Compagno. (12 March 2018). *Cicn*. Available: https://wiki.fd.io/view/Cicn

[25] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, "Network of Information (NetInf) – An information-centric networking architecture," *Computer Communications*, vol. 36, pp. 721 - 735, 27 January 2013.

[26] S. Tarkoma, M. Ain, and K. Visala, "The Publish/Subscribe Internet Routing Paradigm (PSIRP): Designing the Future Internet Architecture," in *Proceedings of Towards the Future Internet - A European Research Perspective*, May 2009, pp. 102-111.

[27] G. García-de-Blas, A. Beben, F. J. Ramón-Salguero, A. Maeso, I. Psaras, G. Pavlou, *et al.*, "COMET: Content mediator architecture for content-aware networks," in *Proceedings of Future Network & Mobile Summit*, Warsaw, Poland, 15-17 June 2011, pp. 1-8.

[28] EU-Japan FP7 GreenICN. (25 May 2016). *Architecture and Applications of Green Information Centric Networking*. Available: https://cordis.europa.eu/project/rcn/109060_en.html

[29] M. Sifalakis, B. Kohler, C. Scherb, and C. Tschudin, "An Information Centric Network for Computing the Distribution of Computations," in *Proceedings of ACM Conference on Information-Centric Networking*, Paris, France, 24 - 26 September 2014, pp. 137-146.

[30] G. Xylomenos, Y. Thomas, X. Vasilakos, M. Georgiades, A. Phinikarides, I. Doumanis, *et al.*, "IP over ICN Goes Live," in *Proceedings of European Conference on Networks and Communications*, Ljubljana, Slovenia, 18-21 June 2018.

[31] Y. Rekhter and T. Li, "A Border Gateway Protocol 4," IETF, RFC 1771, March 1995.

[32] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," IETF, RFC 3626, October 2003.

[33] R. Atkinson and M. Fanto, "RIPv2 Cryptographic Authentication," IETF, RFC

4822, February 2007.
[34] J. Gettys and K. Nichols, "Bufferbloat: Dark Buffers in the Internet," *ACM Queue*, vol. 9, 29 November 2011.
[35] V. Jacobson, "A Conversation with Van Jacobson," *ACM Queue*, vol. 7, pp. 9-16, 23 February 2009.
[36] B. Zhang, "NDN Routing: Similarity and Differences from IP," presented at the ICN WG, The Chinese University of Hong Kong, Sha Tin, Hong Kong, 11 August 2013.
[37] 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 IEEE Symposium on Security and Privacy*, San Jose, CA, USA, 18-21 May 2014, pp. 67-82.
[38] S. Corson and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," IETF, RFC 2501, January 1999.
[39] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses," RFC 6824, January 2013.
[40] Cisco System, "Cisco Wireless Mesh Networking," in *Enterprise Mobility 4.1 Design Guide*, ed, 23 February 2009.
[41] "IEEE Draft Standard for Wireless Access in Vehicular Environments (WAVE)," IEEE P1609.4/D9, 17 August 2010.
[42] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," vol. 52, pp. 2292–2330, 22 August 2008.
[43] R. Khalili, N. Gast, M. Popovic, and J. Y. Le Boudec, "MPTCP Is Not Pareto-Optimal: Performance Issues and a Possible Solution," *IEEE/ACM Transactions on Networking* vol. 21, pp. 1651 - 1665, 20 August 2013.
[44] D. Medhi and K. Ramasamy, *Network Routing: Algorithms, Protocols, and Architectures*, 2 ed.: Morgan Kaufmann, 2007.
[45] R. Canetti, P. Chown, T. Elgamal, P. Eronen, A. Gangolli, K. Hickman, *et al.*, "The Transport Layer Security (TLS) Protocol Version 1.2," IETF, RFC 5246, August 2008.
[46] I. Psaras, D. Corujo, D. Saucez, T. Schmidt, and M. Waehlisch, "Information-Centric Networking (ICN) Research Challenges," IETF, RFC 7927, July 2016.
[47] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, *et al.*, "A Data-oriented (and Beyond) Network Architecture," in *Proceedings of SIGCOMM*, Kyoto, Japan, 27 - 31 August 2007, pp. 181–192.
[48] A. Afanasyev, J. Shi, L. Wang, B. Zhang, and L. Zhang, "Packet Fragmentation in NDN: Why NDN Uses Hop-By-Hop Fragmentation," UCLA, NDN Technical Report NDN-0032, 21 March 2015.
[49] A. P. Iyer, G. Deshpande, E. Rozner, A. Bhartia, and L. Qiu, "Fast Resilient Jumbo Frames in Wireless LANs," in *Proceedings of IEEE IWQoS*, Charleston, SC, US, 15 July 2009.
[50] D. Murray, T. Koziniec, K. Lee, and M. Dixon, "Large MTUs and Internet Performance," in *Proceedings of International Conference on High Performance Switching and Routing*, Belgrade, Serbia, 24-27 June 2012, pp. 82-87.
[51] J. Mogul and S. Deering, "Path MTU Discovery," IETF, RFC 1191, November 1990.

[52] J. McCann, S. Deering, and J. Mogul, "Path MTU Discovery for IP version 6," IETF, RFC 8201, July 2017.
[53] C. Kent and J. Mogul, "Fragmentation Considered Harmful," in *Proceedings of SIGCOMM*, Stowe, Vermont, US, August, 1987.
[54] C. Kent and J. Mogul, "Fragmentation considered harmful," vol. 25, pp. 75-87, January 1995.
[55] G. Tsudik, "Datagram authentication in internet gateways: implications of fragmentation and dynamic routing," *IEEE Journal on Selected Areas in Communications*, vol. 7, pp. 499-504, May 1989.
[56] M. Meisel, V. Pappas, and L. Zhang, "Listen First, Broadcast Later: Topology-Agnostic Forwarding under High Dynamics," in *Proceedings of international technology alliance in network and information science*, London, United Kingdom, September 2010, pp. 1-8.
[57] P. Kietzmann, C. Gündoğan, T. C. Schmidt, O. Hahm, and M. Wählisch, "The Need for a Name to MAC Address Mapping in NDN: Towards Quantifying the Resource Gain," in *Proceedings of ACM Conference on Information-Centric Networking*, New York, NY, USA, 2017, pp. 36–42.
[58] J. Shi, E. Newberry, and B. Zhang, "On Broadcast-based Self-Learning in Named Data Networking," in *Proceedings of IFIP Networking Conference*, Stockholm, Sweden, 12-16 June 2017.
[59] J. Shi, T. Liang, H. Wu, B. Liu, and B. Zhang, "NDN-NIC: Name-based Filtering on Network Interface Card," in *Proceedings of ACM Conference on Information-Centric Networking*, Kyoto, Japan, 26 - 28 September 2016, pp. 40-49.
[60] S. Puangpronpitag, "Design and Performance Evaluation of Multicast Congestion Control for the Internet," PhD Thesis, University of Leeds, Leeds, UK, 2003.
[61] Ns2 Team. (6 Feb 2015). *The Network Simulator - ns-2*. Available: https://www.isi.edu/nsnam/ns
[62] Ns3 Team. (15 Feb 2015). *Network Simulator 3*. Available: https://www.nsnam.org/
[63] OMNeT++ Team. (4 March 2018). *OMNeT++*. Available: https://www.omnetpp.org/
[64] E. Millman, D. Arora, and S. Neville, "STARS: A Framework for Statistically Rigorous Simulation-Based Network Research," in *Proceedings of Advanced Information Networking and Applications*, Biopolis, Spain, 22-25 March 2011, pp. 733-739.
[65] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM 2.0: A new version of the NDN simulator for NS-3," UCLA, Technical Report NDN-0028, 2015.
[66] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the Evolution of ndnSIM: an Open-Source Simulator for NDN Experimentation," *ACM Computer Communication Review*, vol. 47, July 2017.
[67] KVM. (13 October 2015). *Kernel Virtual Machine*. Available: https://www.linux-kvm.org/page/Main_Page
[68] LXC. (12 Feb 2016). *Linux Containers*. Available: https://linuxcontainers.org/
[69] G. Cetušić, M. Mikuc, D. Salopek, V. Vasić, and M. Zec. (17 May 2017). *Integrated Multiprotocol Network Emulator/Simulator*. Available: http://imunes.net/

[70] Cloonix. (16 Jan 2016). *Cloonix*. Available: http://clownix.net/
[71] Mininet. (16 Jan 2016). *Mininet: An Instant Virtual Network on your Laptop*. Available: http://mininet.org/
[72] NRL Lab. (March 2018). *U.S. Naval Research Laboratory*. Available: https://www.nrl.navy.mil/
[73] X. Marchal, T. Cholez, and O. Festor, "Server-side Performance Evaluation of NDN," in *Proceedings of ACM Conference on Information-Centric Networking*, Kyoto, Japan, 26-28 September 2016, pp. 148-153.
[74] K. Sklower, B. Lloyd, G. McGregor, D. Carr, and T. Coradetti, "The PPP Multilink Protocol (MP)," IETF, RFC 1990, August 1996.

BIOGRAPHY

NAME	Thongchai Chuachan
DATE OF BIRTH	May 17, 1986
PLACE OF BIRTH	Surin Province
ADDRESS	41 MOO 18, Tael Distruct, Sikhoraphum, Surin Province, Thailand, 23110
POSITION	Lecturer
PLACE OF WORK	Department of Computer Science, Faculty of Science and Technology, Surindra Rajabhat University, 186 Moo 1, Surin-Prasat Road, Nokmuang District, Muang Surin, Surin, Thailand, 32000
EDUCATION	2005 Taelsiriwittaya School, Surin, Thailand 2009 Bachelor of Computer Science, Faculty of Informatics, Mahasarakham University, Mahasarakham, Thailand 2011 Information Technology, Faculty of Informatics, Mahasarakham University, Mahasarakham, Thailand 2018 Doctor of Philosophy in Computer Science, Faculty of Informatics, Mahasarakham University, Mahasarakham, Thailand