NTNU – Trondheim
Norwegian University of
Science and Technology

Identity-Based Trust Model in Named Data Networking

Haakon Garseg Mørk

Submission date:	May 2015
Responsible professor:	Stig F. Mjølsnes, ITEM
Co-supervisor:	N/A

Norwegian University of Science and Technology
Department of Telematics

Abstract

Abstract goes here...

Acknowledgments

Contents

List of Figures

Listings

List of Tables

List of Acronyms

AES Advanced Encryption Standard.

ARP Address Resolution Protocol.

BDH Bilinear Diffie-Hellman Problem.

CCN Content Centric Networking.

CEK Content-Encryption Key.

CGM Continuous Glucose Monitor.

CIA Confidentiality, Integrity and Availability.

CS Content Store.

DBDH Decisional Bilinear Diffie-Hellman Problem.

DDoS Distributed Denial of Service.

DNS Domain Name System.

DOI Digital Object Identifier.

DoS Denial of Service.

FIA Future Internet Architecture.

FIB Forwarding Information Base.

HLR Home Location Register.

HTTPS Hypertext Transport Protocol Secure.

IBC Identity-Based Cryptography.

IBE Identity-Based Encryption.

IBS Identity-Based Signature.

ICN Information-Centric Networking.

ID Identity.

IoT Internet of Things.

IP Internet Protocol.

IPsec Internet Protocol Security.

IRTF Internet Research Task Force.

KB Kilobyte.

LAN Local Area Network.

MITM Man In The Middle.

MPK Master Public Key.

MSK Master Secret Key.

NDN Named Data Networking.

ndn-cxx Named Data Networking C++ library with eXperimental eXtension.

NFC Near Field Communication.

NFD Named Data Networking Forwarding Daemon.

NSF National Science Foundation.

NTNU Norwegian University of Science and Technology.

OS Operating System.

PARC Palo Alto Research Center.

PBC Pairing-Based Cryptosystems.

PIT Pending Interest Table.

PKG Private Key Generator.

PKI Public Key Infrastructure.

PyNDN2 Named Data Networking Client Library in Python.

RIB Routing Information Base.

SDSI Simple Distributed Security Infrastructure.

SHA1 Secure Hash Algorithm 1.

SSN Social Security Number.

TCP Transmission Control Protocol.

TMPK Temporary Master Public Key.

TTP Trusted Third Party.

UCLA University of California, Los Angeles.

UDP User Datagram Protocol.

URI Uniform Resource Identifier.

URL Uniform Resource Locator.

Chapter 1
Introduction

1.1 Motivation

The translation from name to address and location is a fundamental problem to all networks. Named Data Networking (NDN) is a proposal for content-centric discovery and routing approach to networking going on at the University of California, Los Angeles (UCLA), which is part of the inspiration and a contact point for this work.

In general, the name to address resolution can either be maintained by a catalogue lookup service, such as Domain Name System (DNS) (Internet) and Home Location Register (HLR) (mobile networks), or resolved on-the-fly by a protocol on request, such as Address Resolution Protocol (ARP) (Local Area Network (LAN)). There has been a tremendous amount of work done on the naming problem in distributed systems, some became big failures (e.g. X.500) others such as the web Uniform Resource Locator (URL)s are very successful. Bringing things even further, the Digital Object Identifier (DOI) system is a Uniform Resource Identifier (URI) directed at the content/object itself rather than a location. Very much related to the name/address problem is the information security problem of efficient and practical public key distribution, which remain unsolved in practice, even though a significant number of digital certificate and verification protocols and schemes have been proposed, and systems tested over the last two decades. One notable and early theoretical proposal is Rivest and Lampson Simple Distributed Security Infrastructure (SDSI) proposal [RL96], and subsequent work, that may be revisited for applicable to named data networking.

1.2 Problem and Scope

When designing a new network protocol for the future Internet, one of the most significant changes should be that it is designed for security. Trust management plays a big part in security, and thus we cannot design trust management on known Internet Protocol (IP) failures such as X.500. Public Key Infrastructure (PKI) is a tough challenge to solve and the solution is probably not the one or the other, but rather case specific.

I address the trust management issue in a though sensor device network, e.g. a health sensor network. By using the Named Data Networking Forwarding Daemon (NFD) I will implement my proposal for such a sensor network.

1.3 Methodology

First I find in what context the Public Key System can be applicable, then I design the application flow in sequence diagrams. Based on how NDN is designed, I try to implement the proposed design and see where to redesign the application proposal for achieving minimal communication overhead, security (i.e. Confidentiality, Integrity and Availability (CIA)) and usability.

1.4 Outline

This paper will first introduce one of the proposed protocols for the future Internet, NDN. I will explain the architecture of NDN as well as some related work regarding my application proposal. The application modules will be explain in detail and implementation choices will be discussed. At last I will present the results of the implementation and my conclusion of the trust model.

Chapter 2
Background

"We model the future on the past. Sometimes that's a mistake."

— Van Jacobsen, *SIGCOMM 2001*

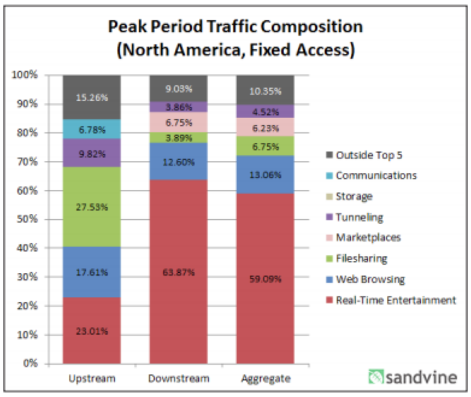
This chapter will give a brief overlook of the motivation for Information-Centric Networking (ICN), as well as explaining more details of how ICN protocols, such as Content Centric Networking (CCN) and NDN, works. It will also contain a quick summary of related works.

2.1 Motivation for Information-Centric Networking

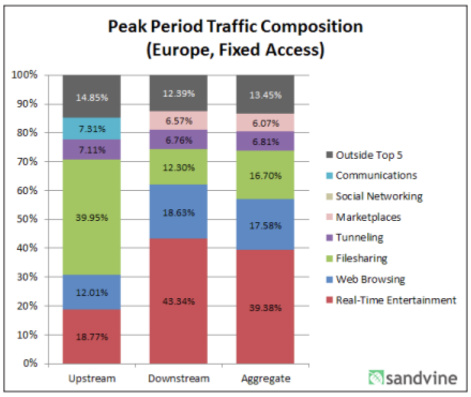
When Internet was created in the 1960's, the researchers where inspired by the existing communication network, the telecommunication network. Because it was not natural and logical to think that people would download content, but rather send and receive short messages and instructions, the point-to-point communication model was a logical architecture. As Internet have developed, the traffic has increased enormously over the past few years. In the Global Internet Phenomena Report 1H2014 done by Sandvine [San14], close to 64% of all IP traffic in North America was Real-Time Entertainment streaming. In Figure 2.1 it can easily be seen that most of the traffic is content, and not communication. With this in mind, the IP architecture does not provide an efficient transport model for what we are actually using the network for.

Also, the IP network was not designed with security in mind. Most of the protocols related to Internet have been designed and deployed mainly with the goal of functionality to work. Internet Protocol Security (IPsec) is a very good example of work trying to patch up security flaws in Internet.

A node in a IP network does not know WHAT goes through, but rather the packet's endpoints, i.e. WHERE it goes and WHERE it comes from. This makes every node dumb, hence the network is designed for redundancy when it comes to content download.



(a) North America



(b) Europe

Figure 2.1: (a) Peak Period Aggregate Traffic Composition - North America, Fixed Access [San14]. (b) Peak Period Aggregate Traffic Composition - Europe, Fixed Access [San14].

Transmission Control Protocol (TCP)/IP not designed for broadcast and therefore wireless connection is not as easy as it can be.

These design failures are some the reasons why the research for Future Internet began. ICN [ADI+12] is a concept developed under this research. It is built upon delivery of content, rather than the point-to-point model we previously have seen. ICNs goal is to build an infrastructure of a new Internet that can achieve efficient, secure and reliable distribution of content. In 2012 Internet Research Task Force (IRTF) established ICN working group.

2.2 Content Centric Network & Named Data Network

The first network protocol purposed for ICN, CCN, was presented by Van Jacobsen at Google Talk in 2006. He, amongst other contributors of CCN, has been working on developing the Internet as we know since the early start. Jacobsen has contributed to TCP/IP, i.e. with his flow control algorithms and TCP header compression.

CCN focuses on naming content, instead of naming IP-addresses. The research project is lead by Palo Alto Research Center (PARC).

A branch of CCN is the NDN [ZAB$^+$14] research project started in 2010. One of the biggest contributers is UCLA, with Lixia Zhang in the lead. The NDN project is also one of few projects selected by National Science Foundation (NSF) Future Internet Architecture (FIA) program [Fou].

2.3 NDN Architecture

Since the knowledge of how NDN works is not disseminated amongst computer scientists, it is essential for this paper to describe how it works. This section will describe the basic architecture of NDN [ASZ$^+$15] and compare some solutions with the equivalent solutions in IP.

2.3.1 Names

In the NDN network there is no strict rules for a Name. This means that a network node only routes an Interest based on longest prefix match. Naming is left to the application design, thus it can be customized for the applications best purpose. However the network assumes hierarchical structured names, hence it will perform better with a hierarchical name design.

For the network to perform even better, the Interest can append some Selectors that can help the network to decide which Data to retrieve and where to route. With Selectors a partially known name can successfully retrieve the right Data. E.g. when a user want to download the newest version of some content, lets say "/ndn/no/ntnu/haakon/file/<version?>", but do not know which version is the newest, the user can append a ChildSelector to choose the greatest version.

When designing applications for the NDN network, one might learn from DNS and Operating System (OS) naming.

2.3.2 Packets

There is two types of packets in NDN; *Interest packet* and the corresponding answer, i.e. the *Data packet*, illustrated in Figure 2.2.

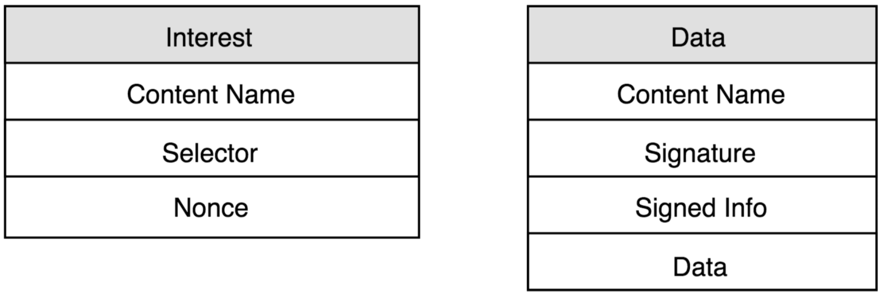


Figure 2.2: Interest packet and Data packet

The Interest packet specifies a Content Name. The name can have a hierarchical structure and signatures can be added after the URI, e.g. "/ndn/no/ntnu/haakon/-file/1/<signature>". An Interest can also contain a set of different Selectors to specify original requirements for the Data response. Some of the Selector fields are:

– KeyLocator - can be used to specify where the Public Key for the signature can be found.

– Exclude - can be used to specify a list or a range of names that should be excluded from the Name. I.e. if the Name is "/ndn/no/ntnu" and the Exclude contains "/item", the returned Data does not contain "/ndn/no/ntnu/item".

– MustBeFresh - if True, a node cannot answer with a Data packet where the FreshnessPeriod has expired.

– ChildSelector - can be used to select either the leftmost (least) or the rightmost (greatest) child, e.g. content version.

– Min/MaxSuffixComponents - refers to name components that occur in the matching Data beyond the prefix.

The Nonce field sets automatically. This is used to uniquely identify an Interest and prevent looping in the network.

The Data packet it a response to the Interest packet, and contains the Content Name and the Content itself. It also has a MetaInfo field that is used to specify the FreshnessPeriod (milliseconds), ContentType and FinalBlockId. Now when somebody requests a file "/ndn/no/ntnu/haakon/file/1" with an Interest, the response will have the same name, but also containing the file.

Because only a Data packet can exists if there is a corresponding Interest, NDN is pull-based. Hence unsolicited Data packets will be thrown away, i.e. there is no content in the network, that is not requested from someone. This reduces unwanted traffic compared to User Datagram Protocol (UDP) in IP.

2.3.3 Network Node

If we look at an existing model of an IP node Figure 2.3 and compare it to a NDN node Figure 2.4, we see that they look much the same. However, the logic behind a NDN node is a bit more complex, thus lead to more knowledge about WHAT content the node has to offer. To understand this, the following entities in a NDN node should be understood:

1. Face - A term used for generalization of different interfaces, e.g. physical like Ethernet, or overlay like TCP and UDP. A Face can also be a UNIX-domain socket for communication with a local application.

2. Pending Interest Table (PIT) - All pending or recently satisfied Interests are stored here, together with the incoming and outgoing Face. If a new incoming Interest matches a entry in the PIT, the incoming Face will be added to the entry.

3. Content Store (CS) - When a node receives a Data packet that has a corresponding entry in the PIT, it stores the Data packet as long as possible in the CS.

4. Forwarding Information Base (FIB) - Here a forwarding strategy is stored for each Name prefix. When a node forwards an Interest, it will do a longest prefix lookup in the FIB and send the Interest further to the best matching Face.

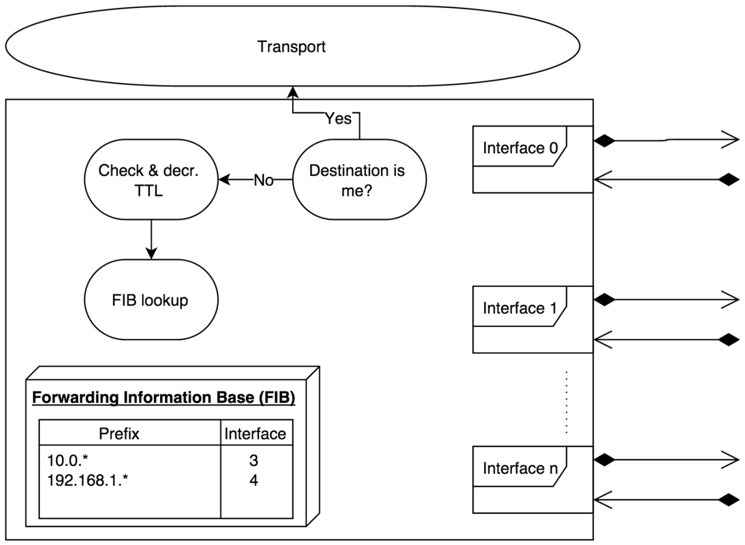


Figure 2.3: Model of IP node. A packets enters the node through an Face. The node decides whether the packet is for the node itself, or passes it further to next node, found in the FIB.

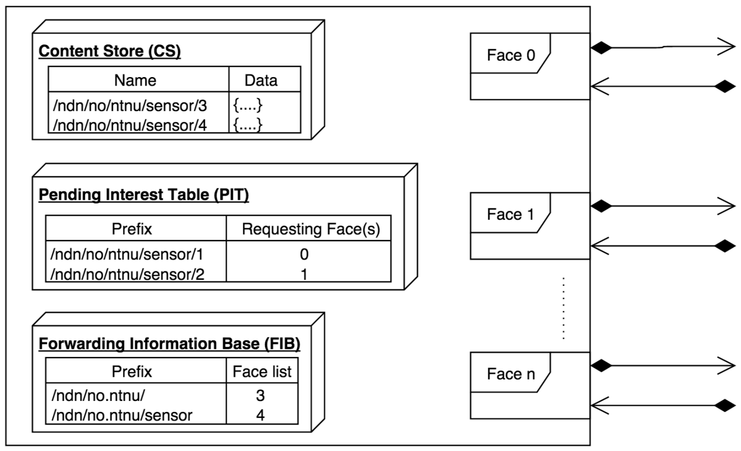


Figure 2.4: Model of NDN node. A packet enters through an Face. The node checks whether the Interest is already queried in the PIT, or stored in the CS, or passes it further to next node, found in the FIB.

In contrary to an IP node, a NDN node knows WHAT content comes through itself. Since all content is associated with a Name, a NDN node can know

WHAT is requested, but not satisfied (i.e. PIT)

WHAT has been satisfied earlier, and still available (i.e. CS)

Unicast (TcpFace, UdpFace) vs Multicast (MulticastUdp, Ethernet) This leads to multicast by nature. In Figure 2.5 we can see that the network does not nearly have to send equal amount of traffic if every node is a NDN node.

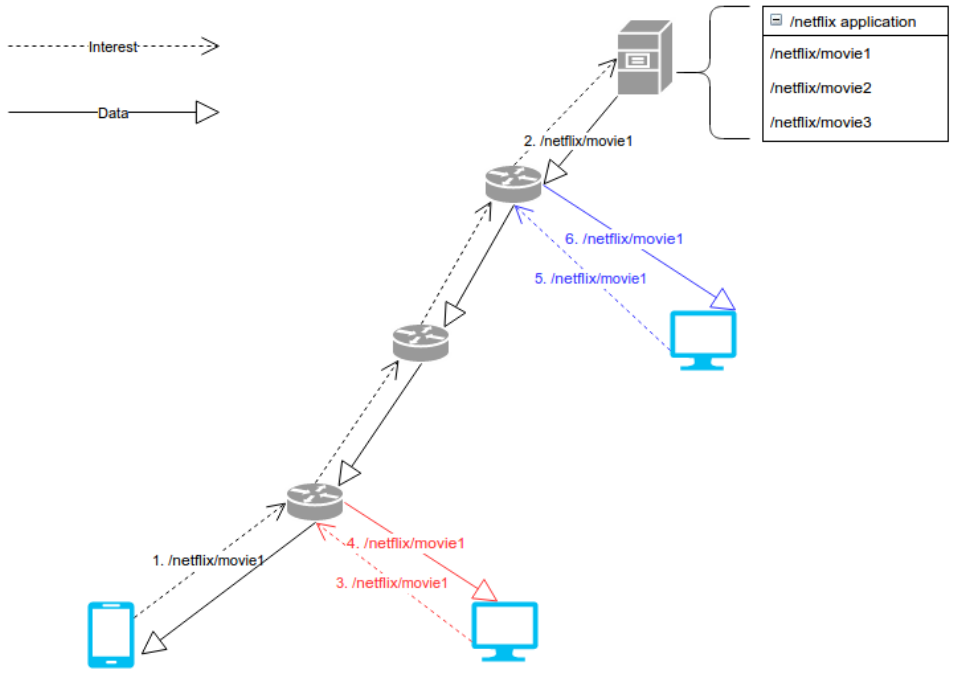


Figure 2.5: Multicast in NDN.

2.3.4 Incoming Interest

In Figure 2.6 we see an incoming Interest. Incoming Interest through a Face. The node checks the PIT for pending or recently satisfied Interests. If there is no match, the node will do a lookup in CS to see if a corresponding Data packet is cached. If there is a match in the PIT it will only add the Face to the PIT entry. If there is a match in the CS the node will return the Data. If there is no match in either the PIT or the CS the node will make a new PIT entry and do a longest prefix match lookup in the FIB to decide which Face(s) to forward the Interest. How to forward a Interest; routing strategy. A strategy per PIT entry. I.e. whether, when, and where to forward Interest.

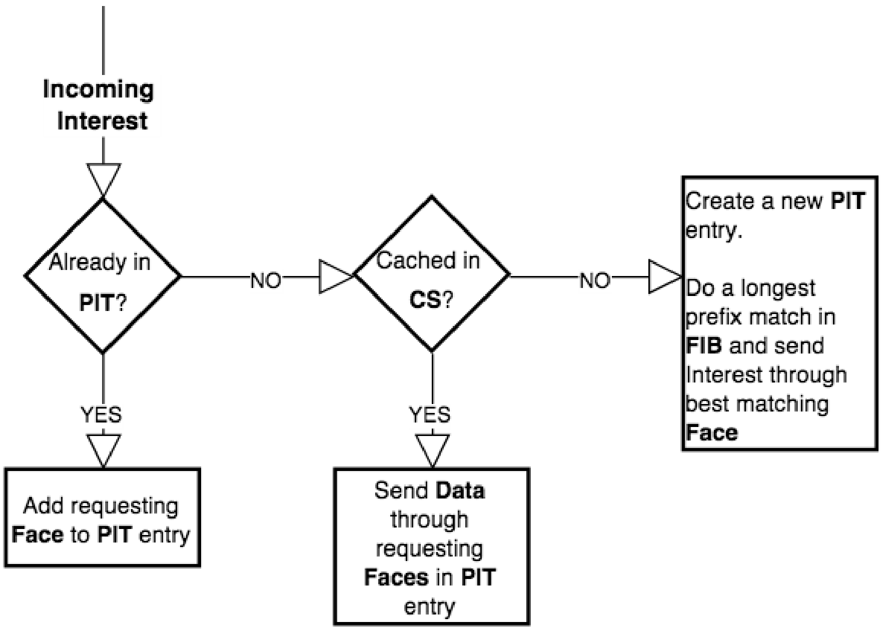

Incoming Interest
Already in PIT?
NO
Cached in CS?
NO
Create a new PIT entry.
Do a longest prefix match in FIB and send Interest through best matching Face
YES
Add requesting Face to PIT entry
YES
Send Data through requesting Faces in PIT entry

Figure 2.6: Decision tree for a NDN node when receiving an Interest.

2.3.5 Incoming Data

In Figure 2.7 we see incoming Data. The node will check the PIT for an entry, if match it will forward the data to all the Faces registered in the PIT entry. Check data from local applications first cached in CS, if not, store in CS send data to all requesters (i.e. all Faces stored in the PIT entry).

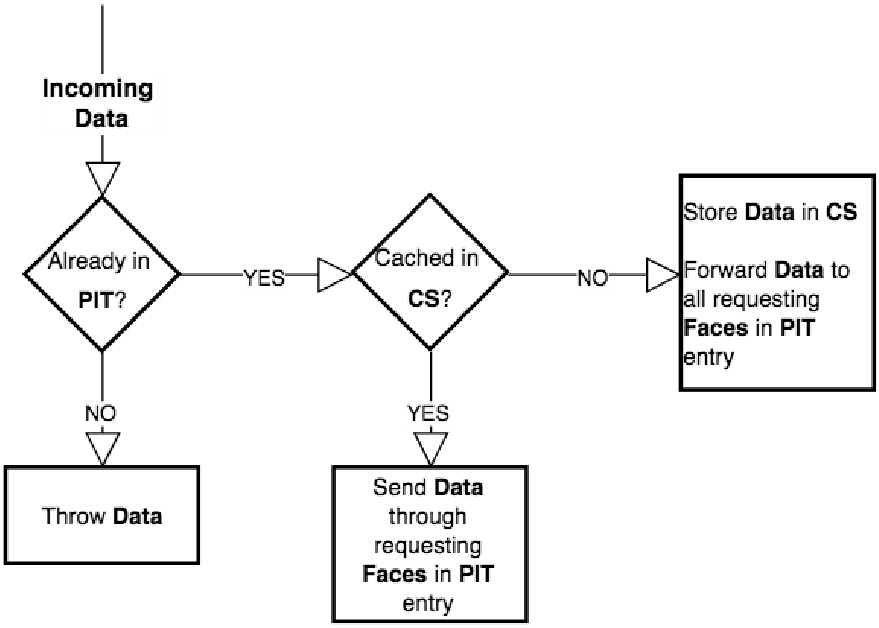

Incoming Data
Already in PIT?
YES
NO
Cached in CS?
NO
YES
Throw Data
Send Data through requesting Faces in PIT entry
Store Data in CS
Forward Data to all requesting Faces in PIT entry

Figure 2.7: Decision tree for a NDN node when receiving Data.

2.3.6 Security

The security aspects of NDN network protocol is one of the most important reasons why Secure data VS secure channel. The network demands that every packet delivered from application layer is signed by the application for integrity and authenticity.

Based on the nature of this architecture, NDN facilitates the practice of anonymity in the network. In a Tor network [DMS], each node participating in a circuit only knows the two neighboring nodes. Only a "Global passive adversary" that can monitor the whole network is able to decide the whole packet path, hence know *who* is requesting and *who* is responding. Since the packet format (subsection 2.3.2) in NDN has no source or destination specific field as in a IP packet, the privacy of the network is more similar to a Tor network. If a packet is captured at any arbitrary point of its path, the only information an adversary will get, is the two nodes between the packet capture and the data name. Unless monitoring a complete network, it should be close to impossible to track packets. However because of the semantic naming there are some issues related to privacy as it easily can be seen in the Name *what* the content contains in many cases. Also since signing of each Interest is required by the sender, some privacy information might leak. DiBenedetto et al. try to address these problems in [DGTU12] with an approach that use existing solutions from the Tor network.

2.4 Attacks

Paolo Gasti et al. identifies several Denial of Service (DoS) attacks on NDN in their paper about DoS & Distributed Denial of Service (DDoS) in NDN [GTUZ13]. Other works have been done related to DoS in NDN [WCZ$^+$14, SNO13, CCGT13]

In [LZZ$^+$15] Zhang et al. proposes an extension of the NDN protocol for addressing the access problem of cached data in nodes. The NDN network is also potentially susceptible to content poisoning attacks which Ghali et al. addresses in [GTU14].

2.5 Related work

Synchronization application over NDN called iSync [FAC14]. A synchronization application build by the NDN team is ChronoSync [ZA13].

Amadeo et al. [ACM14] proposes a solution for reliable retrieval of data from different wireless producers which can answer to the same Interest packet. This is highly applicable to a sensor network where you want to communicate with closest sensor, e.g. the light in *this* room. In [ASLF14] Abid et al. simulate data aggregation in wireless sensor networks. Burke et al. addresses efficient and secure Sensing over NDN in [BGNT14]

There is done some research with Identity-Based Cryptography (IBC) in NDN. In [ZCX$^+$11] Zhang et al. proposes a hybrid scheme with traditional PKI and IBC.

Chapter 3

File Synchronization over Named Data Network

This chapter will present some of the work done in conjunction with this thesis. Explaining what the file synchronization application is built upon and its purpose over the NDN network.

3.1 File Synchronization Module

Since NDN provides multicast in network layer as explained in Figure 2.5, we do not have to think of network load in the same way as in IP. One of the benefits is that we can distribute lots of data, to many people. In our case, synchronization of file(s) to a large group of nodes. For example we want to be able to distribute a list of public keys to a large set of nodes that wants to have every public key up-to-date. Lets say there is a list owner, e.g. a trusted third party. When the list of public keys gets updated, caused by for instance a key revocation or a key initialization, every node should immediately synchronize with this new list. Now since NDN is pull based (section 2.3), there is no way for the list owner to send the list to everyone. To achieve the goal of synchronization, the NDN-team has developed an application over NDN, ChronoSync.

3.2 ChronoSync

ChronoSync is a distributed (server-less) synchronization application developed by the NDN-team. Each node in a ChronoSync application broadcasts its sync state in a Sync Interest (e.g. /ndn/sync/fileSyncFolder/<username>/<state>). When a node receives a Sync Interest, it will inspect the state of the interest, and compare with its own state. Each node holds a state tree that is used to detect new and outdated states. If the incoming interests state is equal to the receiving node, the node has no reason to do anything, as the system is in a *stable state*. If not, the receiving node has to find out whether the incoming interest is 1) a state the node itself has been in, or if its 2) a new state. In case of 1), the receiving node has new data and should provide the new content as a response to the incoming interest. In case of 2), the receiving node should send out a Recovery Interest for the new state.

1. *Sync Interest* is an interest a participating node sends out to discover new data.
2. *Sync Data* is a response to 1), if another participating node has new data.
3. *Recovery Interest* is an interest sent out if a node discovers that another node has a newer state.
4. *Recovery Data* is a response to 3).

ChronoSync is only taking care of data discovery, and leaves other logic to the application. Such logic can be for instance; what should happen when a new participant enters the room. Should all history be download?

As mentioned in subsection 2.3.1, the NDN protocol leaves the naming rules up to each application to decide. ChronoSync have to forms of rules in this matter.

ChronoSync is explained in detail here [ZA13].

3.3 File Watcher

The application should trigger synchronization when files that are watched is changed, or when a file is added/removed. A library that makes it possible to watch files in an OS is Watchdog [Pyt].

Chapter 4 Key Infrastructure

This chapter will present the concept of Identity-Based Encryption (IBE) and Identity-Based Signature (IBS), and why it is highly applicable to use this type of cryptography in NDN. Then the possibilities to use the file synchronization module to do key distribution and revocation will be introduced.

4.1 Identity-Based Cryptography

IBE was first proposed by Shamir [Sha84] in 1984. The concept of IBE builds upon every user having an Identity (ID) that is used as the public key. This ID can be anything, i.e. email, phone number, Social Security Number (SSN), or a Name (subsection 2.3.1). This eliminates the need of certificates. Shamir did propose a scheme for IBS, but not a scheme for IBE. The IBE implementation remained unsolved until 2001, when Dan Boneh and Matthew K. Franklin proposed [BF01]. However the scheme has only been shown to be secure with a random oracles model [Wat04], hence less practical.

IBE is based upon performing cryptography with a publicly know ID. Since the ID can be practically anything it is highly applicable for NDN where the ID can be a Name ("/ndn/no/ntnu/haakon"). Hence the Name becomes the public key.

There is a Trusted Third Party (TTP) in IBE that is called Private Key Generator (PKG). The PKGs task is to produce a private key that corresponds to a given ID and provide

1. `Setup()` generates a key pair, Master Public Key (MPK) and Master Secret Key (MSK). These keys are used by only the PKG to extracting private keys, encryption and decryption.

2. `Extract(MPK, MSK, ID)` generates a Private Key from a given ID.

3. `Encrypt(MPK, ID, message)` encrypts the message.
4. `Decrypt(MPK, private key, cipher)` decrypts the cipher generated from the encryption.
5. `Signing(MPK, private key, message)` signs a hash digest of the message (e.g. Secure Hash Algorithm 1 (SHA1)).
6. `Verify(MPK, ID, message, signature)` verifies the signature.

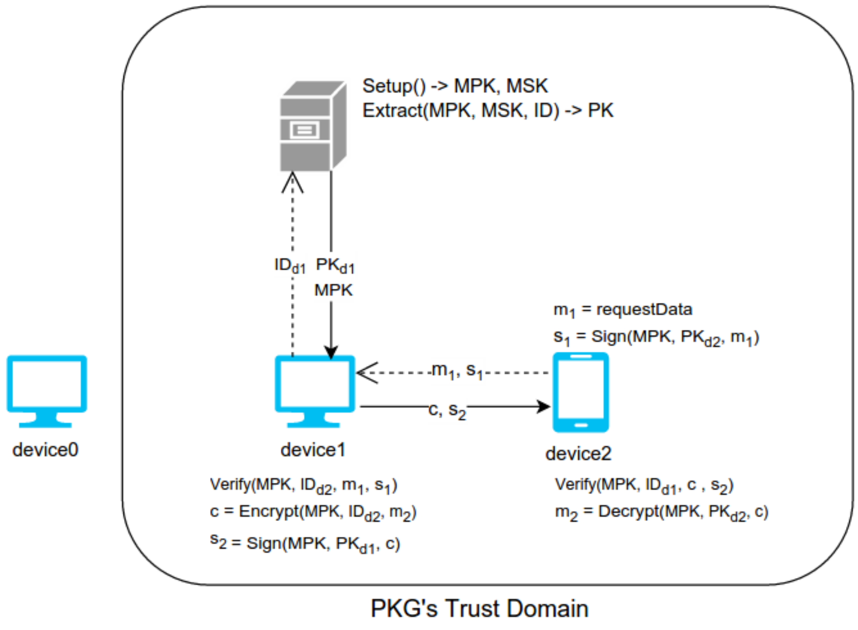

Setup() -> MPK, MSK
Extract(MPK, MSK, ID) -> PK
ID_{d1}
PK_{d1}
MPK
m_1 = requestData
s_1 = Sign(MPK, PK_{d2}, m_1)
m_1, s_1
c, s_2
device0
device1
device2
Verify(MPK, ID_{d2}, m_1, s_1)
c = Encrypt(MPK, ID_{d2}, m_2)
s_2 = Sign(MPK, PK_{d1}, c)
Verify(MPK, ID_{d1}, c , s_2)
m_2 = Decrypt(MPK, PK_{d2}, c)
PKG's Trust Domain

Figure 4.1: Methods of an IBC systems illustrated in action.

To encrypt a message with IBE, the user encrypts a Content-Encryption Key (CEK) with the recipients ID. The user encrypts the message using the CEK together with symmetric encryption [AMS09, section 2.2.2], and sends both the encrypted CEK and the encryted content to the requester.

Some drawbacks related to IBE are listed below:

1. If PKG is compromised. Adversary has private key to all nodes that used the compromised PKG
2. PKG can read and write messages related to the node, because it has all private keys, i.e. Man In The Middle (MITM).

3. PKG and the requesting node has to establish a secure channel.

4.2 Identity-Based Cryptography - Secureness

Boneh and Franklins IBE scheme is only secure when using random oracles. A random oracle is like a “black box” that outputs truly random numbers. Typically random oracles are used when a hash function does not fulfill the mathematical requirements of a proof. A system is in the *random oracle model* when the system is fully secure and every hash function is replaced by a random oracle.

In the *standard model* hash functions are sufficient to achieve a fully secure proof. Since true randomness can be hard for every device to create, we tend to wanting to make cryptographic schemes in the standard model.

Boneh and Boyen proposed a fully secure scheme in the standard model [BB04]. However it is not efficient.

First practical scheme was [Wat04]. But as David Naccache states in his paper [Nac05], Waters’ scheme without random oracles introduces too large public parameters (164Kilobyte (KB)!). Naccache proves that he was able to construct a practical and fully secure scheme in the standard model based on the Decisional Bilinear Diffie-Hellman Problem (DBDH) assumption. The scheme is a modification of Waters’ scheme, but with public parameters of just a few KB size.

Brent Waters created a fully secure IBE system with short parameters under simple assumption in 2009 [Wat09].

To understand the mathematical assumptions for IBE, the reader should take a look at [BF01, section 3] for details about bilinear maps and Bilinear Diffie-Hellman Problem (BDH).

4.3 Key Distribution

Instead of in PKI where each public key is signed by a certificate authority and the generated certificate is sent as a response in Hypertext Transport Protocol Secure (HTTPS) then validated by the the client, I want to make the certificate authority obsolete by distributing every ID (public key) issued by the PKG. This can be done through an IDSync application built upon the application presented in section 3.1. In Figure 4.2 we see that the PKG multicasts the ID list to all devices that have joined the domain. Each device can verify the integrity and authenticity of the sync state Data and be sure that the ID list surely originates from its own PKG.

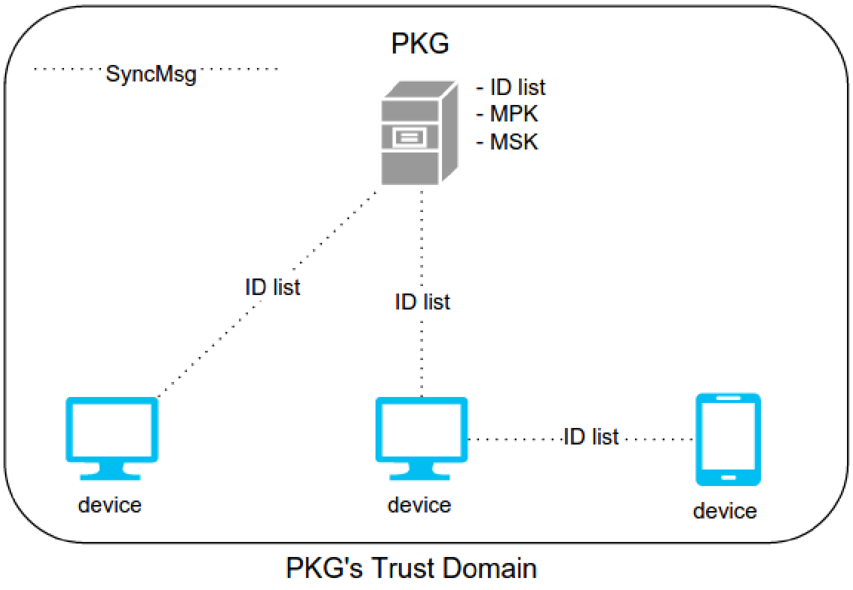


Figure 4.2: IDSync with tree devices and a PKG.

4.4 Key Revocation

Few alternatives to revocation scheme in IBE. Key Revocation in IBE [BGK12] One suggestion is to allocate private keys with the public key combined with some sort of date, e.g. month-year or just year. In this alternative a user has to renew its private key each time the date changes, i.e. either the month or the year depending on the date format. The problem with the revocation solution is that it is cumbersome for the PKG.

Key revocation becomes obsolete with the IDSync distribution solution.

Chapter 5 Application

In this chapter the sensor application will be presented. Several sequence diagrams will be explained, as well as concepts needed to understand the flow of the application.

5.1 Health Sensors

There is an ongoing discussion of when the health technology revolution will come to human bodies now that Internet of Things (IoT) have become so popular. By revolution, I mean sensors placed in the human body. Sensors that can read your blood pressure, heart rate and measure insulin levels. Sensors that can detect whether your body is missing out of a substance, or if its poisoned. There is no limit for what can be done. Everything that should be measured, will be measured by sensors integrated in the human body. But who will be able to read the data? Or perform instructions to the sensors/devices? There is some major privacy issues related to this discussion, and there problems that needs to be solved.

In 2011, Jerome Radcliffe discovered that his insulin pump easily could be hacked [Rad]. Basically the pump would take instructions from anyone and do anything, with no questions asked. This is a worst case scenario when it comes to hacking medical devices attached to a human.

For this matter I propose a health sensor system that is built upon NDN with IBC ensuring a secure and locked environment. First, let me introduce you to The Stig. He has developed diabetes and as everybody else that do not have diabetes, he does not want to manually monitor his glucose levels and adjust the insulin pump at every meal. He has injected a Continuous Glucose Monitor (CGM) to monitor his glucose levels and report to the insulin pump, automatically. In addition to his diabetes, he has a heart disease which forces him to monitor his heart rate at any given time. In Figure 5.1 we can see The Stig with all his sensors and devices. The CGM reports periodically to the insulin pump, and all sensors reports to The Stig's mobile so that The Stig can watch what is going on.

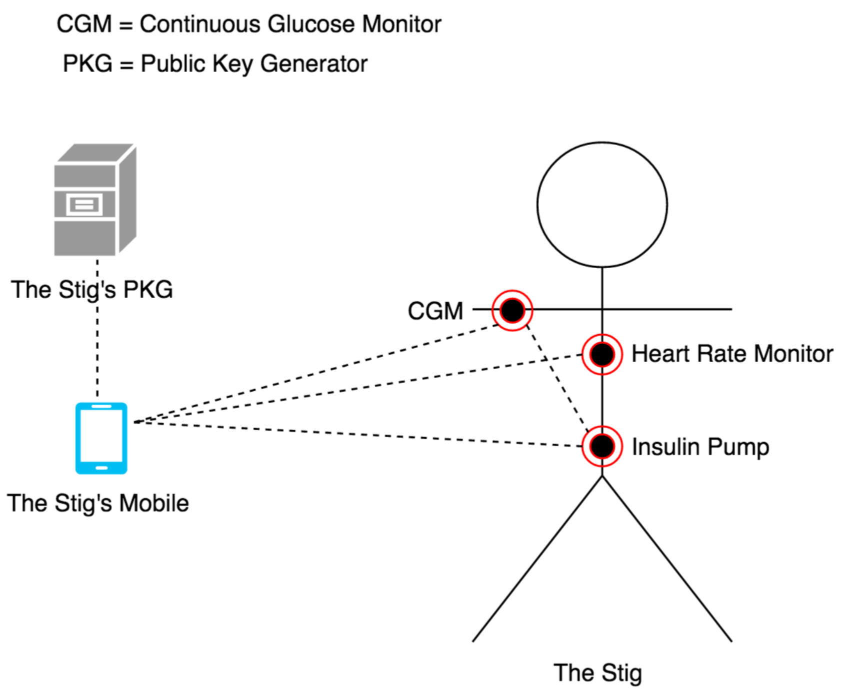


Figure 5.1: Health Sensor System

5.2 Health Sensor System

To have a secure system, it needs to be established trust between the sensors and the devices. There needs to be integrity controls, confidentiality protection and access control. In the following sections, I will describe the protocols suggested for achieving the mentioned goals.

5.2.1 Rendezvous Authentication

One of the best solutions for authentication of an identity in cryptography is rendezvous authentication, the concept of meeting face-to-face for authenticating who you are talking to. In IoT, we have in most cases the advantage of identifying devices in a physical matter. This means that it is possible to authenticate devices, such as sensors. Typically, this kind of authentication will rely on 1) manually inspection and 2) digital connection, e.g. through Near Field Communication (NFC). In the proposed system, I assume that this type of authentication is achieved in a secure matter and do not discuss whether how this should be done.

5.2.2 Initialization

When The Stig is setting up his health sensor system, first he wants to configure the PKG. Any type of computer can play the role of the PKG and The Stig has chosen his home server, from now “the PKG”. The PKG creates two key pairs that is used to do IBE and IBS. Second, he wants his mobile device, from now “the mobile”, to be a part of the PKGs trust domain, and further add all of the other devices and sensors, from now “device(s)”.

1. First device, e.g. a mobile, connects to the PKG through physical connection as showed in Figure 5.2.

2. Additional devices can connect through physical connection via another authorized device, or the PKG itself.

3. Data pulling from existing nodes in network Figure 5.3

In Figure 5.2 the device acts as the mobile. The ID of every device has been given the term Name, this is due to NDN. Hence ID and Name is the same. To be able to communicate securely under initialization, the device have to create a temporary MPK and MSK and then extract a temporary Private Key for its Name. At first, the device has to act as a PKG for itself to ensure confidentiality, and the trust between the device and the real PKG is based on the concept explained in subsection 5.2.1. The device sends an Interest appending the temporary MPK and the Name to the PKG asking to join the PKGs domain. The PKG encrypts a CEK with the temporary MPK and the Name received from the device. Then the PKG extracts the Private Key for the device (this will be the key belonging to the PKGs trust domain) and uses the CEK to do a symmetric Advanced Encryption Standard (AES) encryption on the Private Key. The Data packet response to the initialization Interest will contain the identity-based encrypted CEK, the symmetric encrypted Private Key and the PKGs MPK. To finish the initialization protocol, the device decrypts the CEK, throws away the temporary keys, and finally decrypts the Private Key. The device has established a trust with its PKG and can verify other devices in this domain.

Now that the mobile is authenticated, devices can connect to the mobile through e.g. NFC for initialization. This results in a rendezvous authentication between the device and the mobile, and if the mobile is given the authorities to perform initialization (subsection 5.4.1), the new device has joined the PKGs trust domain.

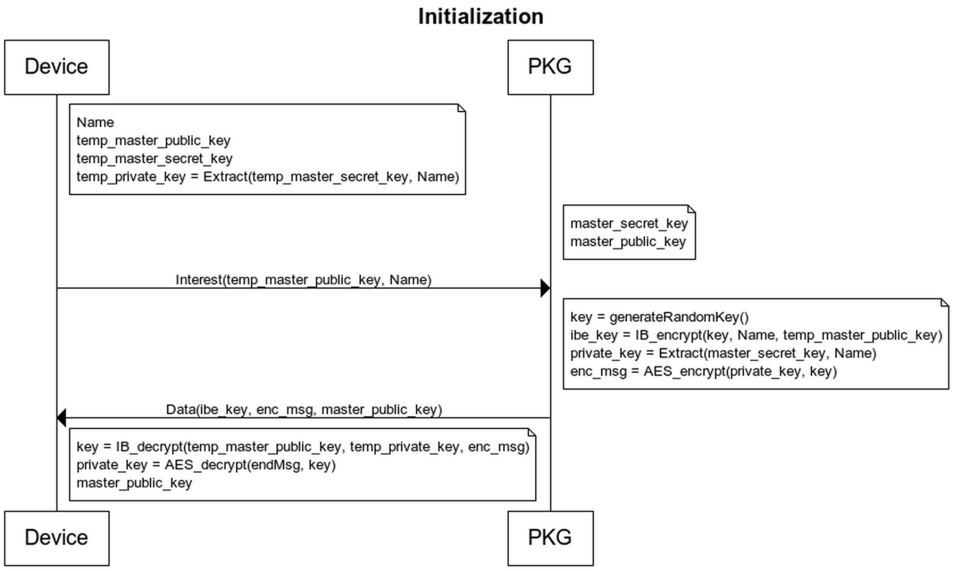

Initialization
Device
PKG
Name
temp_master_public_key
temp_master_secret_key
temp_private_key = Extract(temp_master_secret_key, Name)
master_secret_key
master_public_key
Interest(temp_master_public_key, Name)
key = generateRandomKey()
ibe_key = IB_encrypt(key, Name, temp_master_public_key)
private_key = Extract(master_secret_key, Name)
enc_msg = AES_encrypt(private_key, key)
Data(ibe_key, enc_msg, master_public_key)
key = IB_decrypt(temp_master_public_key, temp_private_key, enc_msg)
private_key = AES_decrypt(endMsg, key)
master_public_key
Device
PKG

Figure 5.2: Initialization IBE

5.2.3 Data Pull

As illustrated in Figure 5.1, the devices has now joined the PKGs trust domain and are ready to communicate. This flow is illustrated in Figure 5.3. By signing the Interest and appending the Name, the receiving device can verify that the requester is a part of the same trust domain, and also be able to encrypt the sensor data (if needed) with the requesters Name. The requesting device receives the Data and verifies the signature and decrypts the sensor data.

5.2.4 IDSync

The File Synchronization Module (section 3.1) can be used to distribute a fresh list of every device that is a member of the trust domain. This synchronization will be controlled by the PKG, hence every device can verify each synchronization message.

5.3 Confidentiality, Integrity and Availability

Confidentiality The encryption can be achieved by encrypting with the Name of the requester. As explained in the sequence diagrams presented in the above sections, each Interest appends the requesters Name, hence all packets can be encrypted, and thus the confidentiality in the system is achieved.

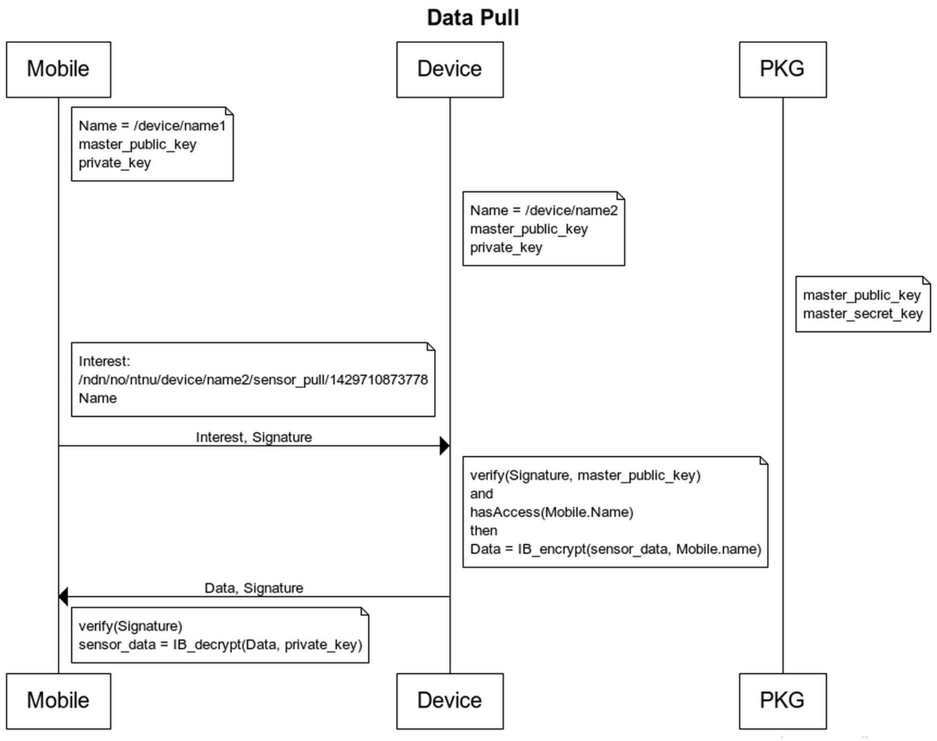


Figure 5.3: Mobile performing a data pull from a device in the network.

Integrity and Authenticity Each device will obtain a private key allocated by its superior PKG, as explained in section 4.1. With the concept from subsection 5.2.1 together with the PKGs MPK, you can trust that the device is authorized for the PKGs trust domain. Hence all signed packets can be verified by anyone with the MPK. Every Interest has a timestamp attached to the Name (e.g. `/ndn/no/ntnu/device/name2/sensorPull/1429710873778`), i.e. milliseconds from `UTC1970-01-0100:00:00`, that is used for protection against replay attack.

Availability This is a harder problem to solve. The network is purely wireless, hence vulnerable to jamming.

5.4 Trust Model

5.4.1 Access Control

When a device retrieves an Interest for its sensor data, there should be a authorization mechanism.

Capability based approach to IoT access control [GPR12].

Least privilege access

Chapter 6
Implementation and Testing

This chapter will first introduce the most significant frameworks that must be installed to be able to run NDN applications. Then the various application modules design and implementation choices will be explained. It will also be presented how these applications will be tested.

6.1 Installing Named Data Networking Protocol

There are several programs that is required to install for experimenting in a NDN environment. Installation guides can be found at the Github project [NT15].

Named Data Networking C++ library with eXperimental eXtension (ndn-cxx) is a implementation of NDN primitives. It is a fundamental framework that NDN application requires.

NFD [AMY+] is a network forwarder and also in the core implementation of NDN. The major modules implemented in NFD is:

– Core - Common services shared between the different NFD modules (such as hash, DNS resolver, face monitoring etc.).

– Faces - Generalization of different interfaces, explained in subsection 2.3.3.

– Tables - PIT, CS, FIB, explained in subsection 2.3.3.

– Forwarding - Packet processing.

– Management - Enables users/programs to interact with the NFD forwarder state.

– Routing Information Base (RIB) Management - Managing routing protocols and application prefix registration.

6.1.1 PyNDN2

The work done in this thesis is written in Python, hence the Named Data Networking Client Library in Python (PyNDN2) is used. This is an easy to use implementation of NDN and comes with example code.

Because the NDN protocol require signing of Data packets (subsection 2.3.6) some new implementation in the PyNDN2 source code was necessary to be able to sign and verify with IBS.

I added the `python/pyndn/sha256_with_ibswaters_signature.py` file that follows the pattern of the existing RSA Signature (`python/pyndn/sha256_with_rsa_signature.py`) and is of type `Signature`. Some small additions in the `python/pyndn/encoding/tlv_0_1_1_wire_format.py` and the `python/pyndn/encoding/tlv/tlv.py` is added so PyNDN2 recognizes the signature when the Data packet is encoded and decoded.

6.2 Installing Identity-Based Cryptography

To be able to run IBC there is a Pairing-Based Cryptosystems (PBC) [Ben07] that needs to be installed. Further I use a Python implementation [AAG$^+$13] of IBC.

The code in [Mø15, identityBasedCrypto.py] implements two IBE schemes and one IBS scheme.

Waters05 [Nac05] that is a variant of Brent Waters IBE scheme [Wat04], but with smaller key size, hence more practical.

Waters09 [Wat09] that is also a fully secure implementation of IBE scheme.

Waters [Wat04] that is a implementation of IBS scheme.

6.2.1 Charm - A Framework for Rapidly Prototyping Cryptosystems

The Charm framework [AAG$^+$13] implements several IBE and IBS schemes in Python. Some small modifications had to be done in the Waters-IBS [Wat04] implementation in Charm. Not really something worth mentioning, yet explained due to the concept of academic reproduction. In `charm/schemes/pksig/pksig_waters.py` there is a global variable, i.e. `waters`, that is used throughout all the methods in `pksig_waters.py`. The problem is that this variable is declared in the `setup()`, which is only called at PKG (section 4.1), and not by another device that do not play the role of a PKG. And thus, the declaration of `waters` must be moved to the `__init__()` in `pksig_waters.py`.

6.3 IDSync Module - Implementation

Application goal explained in section 3.1

FileSync code [Mø15, fileSync.py]

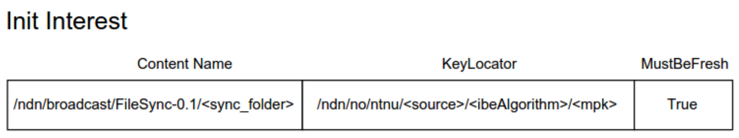
FileSync is a python application that will synchronize all files in a specified path, with all participants within the synchronization room.

6.3.1 Packet Design

Init Interest - The MPK as well as the joining nodes Name is added in the KeyLocator. See Figure 6.1.

Sync Interest - Figure 6.2

Sync Data - Figure 6.2



Init Interest
Content Name
KeyLocator
MustBeFresh
/ndn/broadcast/FileSync-0.1/<sync_folder>
/ndn/no/ntnu/<source>/<ibeAlgorithm>/<mpk>
True

Figure 6.1: Initialization Interest for joining a synchronization folder.

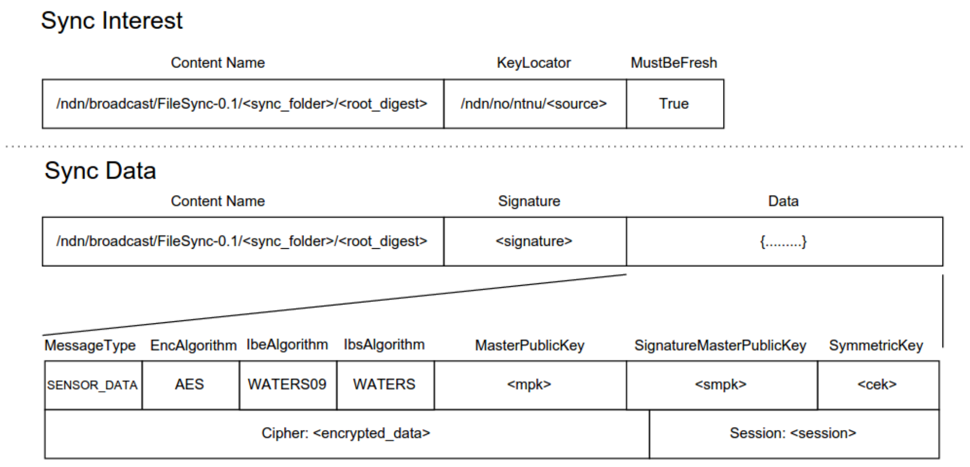


Figure 6.2: Sync Interest and Data

6.4 Sensor Application - Implementation

Application flow explained in chapter 5

Device code [Mø15, device.py]

PKG code [Mø15, publicKeyGenerator.py]

Main code [Mø15, application.py]

6.4.1 Packet Design

Initialization packets have the structure presented in Figure 6.3. Initially the idea was to have the *Temporary Master Public Key (TMPK)* appended to the Content Name. However I experienced a problem where the Init Data never arrived at destination node. After some research in `ndn-cxx` documentation I found that the packets have a `MAX_NDN_PACKET_SIZE` of 8800 bytes and the Init Data exceeded this limit and reached 8904 bytes. Because the TMPK is approximately 2KB and was appended to the Content Name in the Interest, the Data response off course had to have the same Content Name, hence 2KB overhead in the Name. The TMPK can as easily be appended to the KeyLocator Name, hence the Data response can be 2KB less, resulting to a 6866 bytes Init Data packet.

Sensor packets have the structure presented in Figure 6.4.

Init Interest - The initialization Interest seen in Figure 6.3 KeyLocator can be of type Name. As described in the NDN Packet Format [NT], generally this field can be used to specify where to download the certificate used to sign the Interest. However, in our trust model we use this field to publish the requesters Name, i.e. the requesters public key. This is very useful when using IBE and IBS.

Init Data - The Data response the the initialization Interest contains data with a structure defined in [Mø15, messageBuf.proto] and illustrated in Figure 6.3.

Sensor Interest - As in the initialization Interest the KeyLocator field is used to define the requesters Name. Figure 6.4

Sensor Data - The Data response to the Sensor Interest uses the same structure as the initialization Data. It is illustrated in Figure 6.4

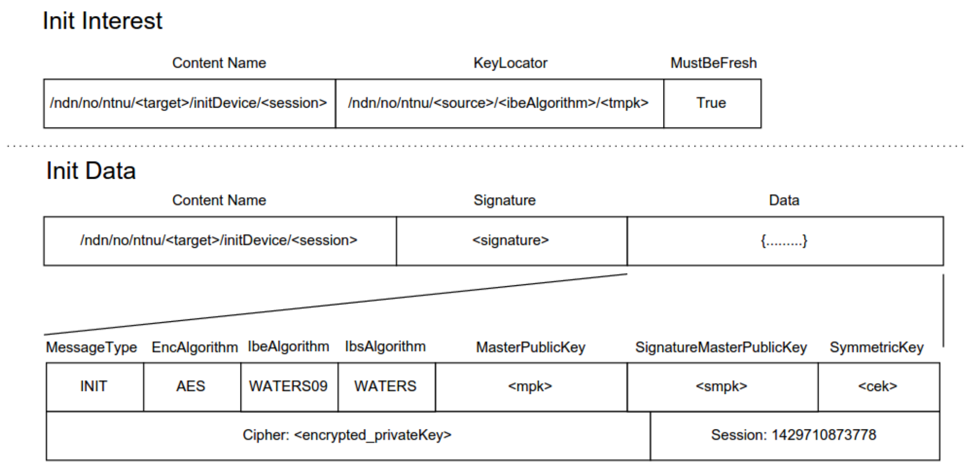


Figure 6.3: Initialization Interest and Data

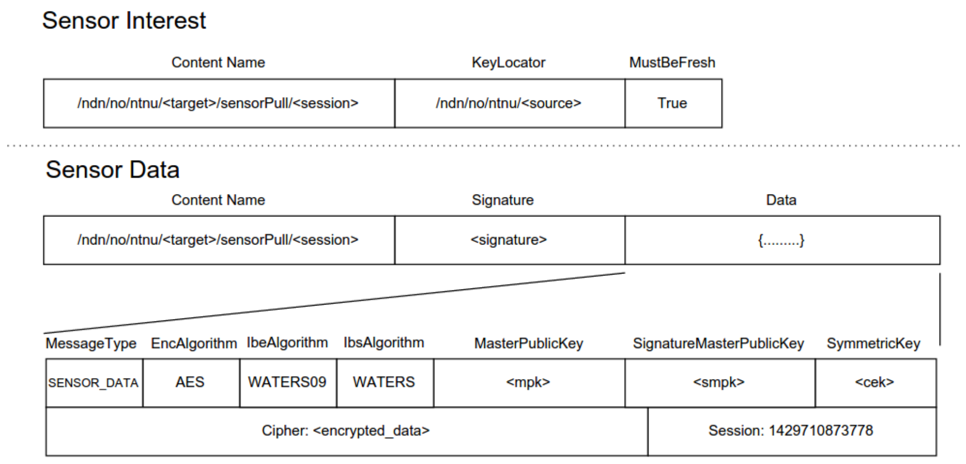


Figure 6.4: Sensor Interest and Data

6.5 Testing

At first I wanted to test the application with several Raspberry Pi's to simulate a sensor network. However this is not possible with the Charm framework as it is not compatible with ARM processors.

The health sensor system is tested over several computers installed in the lab at Norwegian University of Science and Technology (NTNU).

Encryption / decryption time

Signing / verification time

Power consumption

Init Protocol time

Sensor Data Protocol time

Chapter 7
Results

Results from the application described in chapter 3

7.1 Sensor Application with Public Key Infrastructure

7.2 Sensor Application with Identity-Based Cryptography

7.3 Key Distribution

7.4 Key Revocation

Chapter 8
Discussion

In this chapter it will be discussed

8.1 Identity-Based Cryptography]

One suggestion has been to add a monthly timestamp [BGK12] [LV09] to the name, but the the PKG has to renew private keys for everybody each month. This solution scales very badly. With the PKGs Name Sync application, every user will be notified when a identity is revoked. There is no use for periodically checking names.

find better cite

8.1.1 Trusting Trusted Third Party in Identity-Based Cryptography

Issues of trusting the PKG, i.e. a TTP. If the PKG is compromised by an adversary, the adversary will retrieve all private keys corresponding to all IDs that used the compromised PKG. Do every ID trust the PKG? Suspicion of MITM, where the PKG is the adversary, will always be a problem for a user. Secure channel for private key exchange.

8.2 Health Sensor System

Not tested on with real sensors, hence I cannot conclude with anything regarding the computational power of such devices, nor the life time of the battery when performing IBE.

8.2.1 Sync

The sync application makes it possible for users to know who has a valid public key within the PKGs domain. One drawback with the IDSync solution is that for the sender to be 100% sure that the message is encrypted with the latest ID, the sender has to rely on that it has reveiced the latest sync state available from the PKG.

Likewise when a receiver verifies a signature, it has to rely on the same principle to be able to know if the belonging ID is still valid.

8.2.2 Attacks

In [SA99] Frank Stajano and Ross Anderson mentions possible DoS attacks, suck as radio jamming and battery exhaustion. All applications that are relying on some sort of crucial information derived from a Sync application section 3.1 are vulnerable to DoS.

8.3 Other Use Cases

1. Home Automation Systems
2.

Chapter 9

Conclusion and Future Work

9.1 Future Work

Chapter 10

NDN node 23 - NTNU

10.1 NDN node 23 - NTNU

Node 23 in NDN testbed

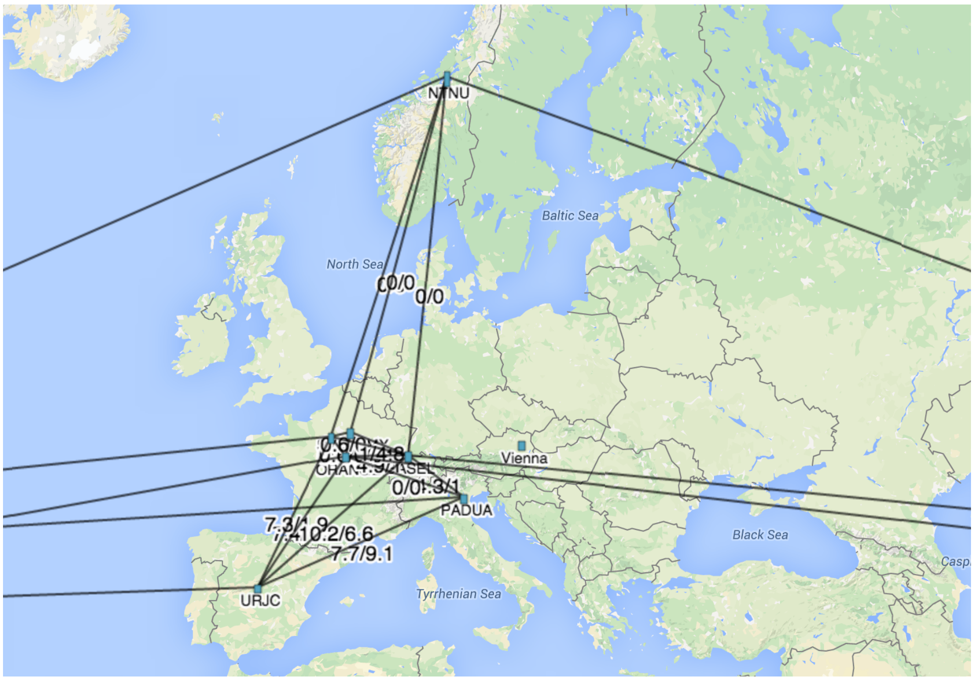


Figure 10.1: NDN Map

References

[AAG$^+$13] Akinyele, Joseph A., Garman, Christina, Miers, Ian, Pagano, Matthew W., Rushanan, Michael, Green, Matthew, Rubin, and Aviel D. Charm: a framework for rapidly prototyping cryptosystems. 3(2):111–128, 2013.

[ACM14] Marica Amadeo, Claudia Campolo, and Antonella Molinaro. Multi-source data retrieval in iot via named data networking. In *1st International Conference on Information-Centric Networking, ICN'14, Paris, France, September 24-26, 2014*, pages 67–76, 2014.

[ADI$^+$12] Bengt Ahlgren, Christian Dannewitz, Claudio Imbrenda, Dirk Kutscher, and Börje Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(7):26–36, 2012.

[AMS09] G. Appenzeller, L. Martin, and M. Schertler. Identity-Based Encryption Architecture and Supporting Data Structures. RFC 5408, RFC Editor, January 2009.

[AMY$^+$] Alexander Afanasyev, Ilya Moiseenko, Yingdi Yu, Wentao Shang, Lixia Zhang, Junxiao Shi, Yi Huang, Jerald Paul Abraham, Beichuan Zhang, Steve DiBenedetto, Chengyu Fan, Christos Papadopoulos, Davide Pesavento, Giulio Grassi, Giovanni Pau, Hang Zhang, Tian Song, Haowei Yuan, Hila Ben Abraham, Patrick Crowley, Syed Obaid Amin, Vince Lehman, Minsheng Zhang, and Lan Wang. Named data networking forwarding daemon. http://named-data.net/doc/NFD/current/overview.html. [Online; accessed 08-May-2015].

[ASLF14] Younes Abidy, Bilel Saadallahy, Abdelkader Lahmadi, and Olivier Festor. Named data aggregation in wireless sensor networks. In *2014 IEEE Network Operations and Management Symposium, NOMS 2014, Krakow, Poland, May 5-9, 2014*, pages 1–8, 2014.

[ASZ$^+$15] Alexander Afanasyev, Junxiao Shi, Beichuan Zhang, Lixia Zhang, Ilya Moiseenko, Yingdi Yu, Wentao Shang, Yi Huang, Jerald PaulAbraham, Steve DiBenedetto, Chengyu Fan, Christos Papadopoulos, Davide Pesavento, Giulio Grassi, Giovanni Pau, Hang Zhang, Tian Song, Haowei Yuan, Hila Ben Abraham, Patrick Crowley, Syed Obaid Amin, Vince Lehman, and Lan Wang. Nfd developer's guide, ndn-0021. Technical report, February 2015. Revision 3.

[BB04] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, pages 443–459, 2004.

[Ben07] LYNN Ben. *On the implementation of pairing-based cryptosystems.* PhD thesis, PhD thesis, Stanford University, 2007. https://crypto. stanford. edu/pbc/thesis. pdf, 2007.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 213–229, 2001.

[BGK12] Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. *IACR Cryptology ePrint Archive*, 2012:52, 2012.

[BGNT14] Jeff Burke, Paolo Gasti, Naveen Nathan, and Gene Tsudik. Secure sensing over named data networking. In *2014 IEEE 13th International Symposium on Network Computing and Applications, NCA 2014, Cambridge, MA, USA, 21-23 August, 2014*, pages 175–180, 2014.

[CCGT13] Alberto Compagno, Mauro Conti, Paolo Gasti, and Gene Tsudik. Poseidon: Mitigating interest flooding ddos attacks in named data networking. *CoRR*, abs/1303.4823, 2013.

[DGTU12] Steve DiBenedetto, Paolo Gasti, Gene Tsudik, and Ersin Uzun. Andana: Anonymous named data networking application. In *19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, February 5-8, 2012*, 2012.

[DMS] Roger Dingledine, Nick Mathewson, and Paul F. Syverson.

[FAC14] Wenliang Fu, Hila Ben Abraham, and Patrick Crowley. isync: a high performance and scalable data synchronihttps://github.com/named-datazation protocol for named data networking. In *1st International Conference on Information-Centric Networking, ICN'14, Paris, France, September 24-26, 2014*, pages 181–182, 2014.

[Fou] National Science Foundation. Future internet architecture project. [Online; accessed 5-May-2015].

[GPR12] Sergio Gusmeroli, Salvatore Piccione, and Domenico Rotondi. Iot access control issues: A capability based approach. In *Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, IMIS 2012, Palermo, Italy, July 4-6, 2012*, pages 787–792, 2012.

[GTU14] Cesar Ghali, Gene Tsudik, and Ersin Uzun. Network-layer trust in named-data networking. *Computer Communication Review*, 44(5):12–19, 2014.

[GTUZ13] Paolo Gasti, Gene Tsudik, Ersin Uzun, and Lixia Zhang. Dos and ddos in named data networking. In *22nd International Conference on Computer Communication and Networks, ICCCN 2013, Nassau, Bahamas, July 30 - Aug. 2, 2013*, pages 1–7, 2013.

[LV09] Benoît Libert and Damien Vergnaud. Adaptive-id secure revocable identity-based encryption. In *Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings*, pages 1–15, 2009.

[LZZ+15] Qi Li, Xinwen Zhang, Qingji Zheng, Ravi Sandhu, and Xiaoming Fu. LIVE: lightweight integrity verification and content access control for named data networking. *IEEE Transactions on Information Forensics and Security*, 10(2):308–320, 2015.

[Mø15] Haakon Garseg Mørk. Master thesis work. https://github.com/haakonmo/master-thesis-ndn/tree/master/master-thesis-work, 2015.

[Nac05] David Naccache. Secure and *Practical* identity-based encryption. *IACR Cryptology ePrint Archive*, 2005:369, 2005.

[NT] NDN-TEAM. Ndn packet format. http://named-data.net/doc/ndn-tlv/. [Online; accessed 02-May-2015].

[NT15] NDN-TEAM. Named data. https://github.com/named-data, 2015.

[Pyt] Python. Watchdog. https://pypi.python.org/pypi/watchdog. [Online; accessed 24-April-2015].

[Rad] Jerome Radcliffe. Hacking medical devices for fun and insulin: Breaking the human scada system. Black Hat Conference 2011.

[RL96] Ronald L Rivest and Butler Lampson. Sdsi-a simple distributed security infrastructure. Crypto, 1996.

[SA99] Frank Stajano and Ross J. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Security Protocols, 7th International Workshop, Cambridge, UK, April 19-21, 1999, Proceedings*, pages 172–194, 1999.

[San14] Intelligent Broadbands Networks Sandvine. Global internet phenomena report 1h 2014, 2014.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, pages 47–53, 1984.

[SNO13] Won So, Ashok Narayanan, and David Oran. Named data networking on a router: Fast and dos-resistant forwarding with hash tables. In *Symposium on Architecture for Networking and Communications Systems, ANCS '13, San Jose, CA, USA, October 21-22, 2013*, pages 215–225, 2013.

[Wat04] Brent R. Waters. Efficient identity-based encryption without random oracles. *IACR Cryptology ePrint Archive*, 2004:180, 2004.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, pages 619–636, 2009.

[WCZ$^+$14] Kai Wang, Jia Chen, Huachun Zhou, Yajuan Qin, and Hongke Zhang. Modeling denial-of-service against pending interest table in named data networking. *Int. J. Communication Systems*, 27(12):4355–4368, 2014.

[ZA13] Zhenkai Zhu and Alexander Afanasyev. Let's chronosync: Decentralized dataset state synchronization in named data networking. In *2013 21st IEEE International Conference on Network Protocols, ICNP 2013, Göttingen, Germany, October 7-10, 2013*, pages 1–10, 2013.

[ZAB$^+$14] Lixia Zhang, Alexander Afanasyev, Jeff Burke, Van Jacobson, kc claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *Computer Communication Review*, 44(3):66–73, 2014.

[ZCX$^+$11] Xinwen Zhang, Katharine Chang, Huijun Xiong, Yonggang Wen, Guangyu Shi, and Guoqiang Wang. Towards name-based trust and security for content-centric network. In *Proceedings of the 19th annual IEEE International Conference on Network Protocols, ICNP 2011, Vancouver, BC, Canada, October 17-20, 2011*, pages 1–6, 2011.

Appendix A

Code