A Security Monitoring Plane for Information Centric Networking: application to Named Data Networking

Ngoc Tan Nguyen

THÈSE

en vue de l'obtention du grade de

DOCTEUR DE L'UNIVERSITÉ DE TECHNOLOGIE DE TROYES

Spécialité : OPTIMISATION ET SÛRETÉ DES SYSTÈMES

A Security Monitoring Plane for Information Centric Networking: application to Named Data Networking

présenté par

Ngoc Tan NGUYEN

Soutenance envisagée le 3 Juillet 2018
devant le jury constitué des personnalités ci-dessous

JURY

Mr. Christian F. TSCHUDIN	Professor	Rapporteur
Mr. Patrick BAS	Directeur de Recherche	Rapporteur
Mrs. Sandrine VATON	Professeure	Examinateur
Mr. Ludovic MÉ	ARP	Examinateur
Mr. Guillaume DOYEN	Maître de conférences	Directeur de thèse
Mr. Rémi COGRANNE	Maître de conférences	Directeur de thèse

Acknowledgements

This Ph.D. thesis has been carried out within the team of Science and Technologies for Risks Management (STMR) at the University of Technology of Troyes (UTT) under the co-supervision of M. Guillaume DOYEN and M. Rémi COGRANNE.

First of all, I would like to express my deepest gratitude to my supervisors, M. Guillaume DOYEN, and M. Rémi COGRANNE. You are the coolest supervisors who are very responsible. Moreover, you are always ready to push me beyond my own limit, so that I can achieve better results than I could. And yet, you have always been supporting me and providing me with proper guidance. It is an honor to work with you two.

I would like to express my sincere thanks to Professor Christian F. TSCHUDIN, M. Patrick BAS, Professeure Sandrine VATON and M. Ludovic MÉ for taking your time and agreeing to be a jury member for my thesis defense. The valuable remarks provided by the respectful experts have helped in improving the quality of this manuscript.

I gratefully acknowledge the French National Research Agency (ANR), colleagues from DOCTOR project and UTT for providing me with financial and technical support during my work. I want to express my thanks to secretaries of ROSAS department, Bernadette ANDRÉ, Véronique BANSE, and secretaries of UTT's doctoral school, Thésèse KAZARIAN, Pascale DENIS, Isabelle LECLERCQ for their support during my time working in UTT.

I want to express my gratefulness for all of my friends in Troyes. Five years of living far from hometown, studying abroad and doing the Ph.D. were not always easy for me. However, they became easier and more joyful with all of you by my side. Thank you all for the all the meals that we had, all the event that we participated, all the board games that we played, all the trips that we had together and all the support that you gave me.

Last but not least is my greatest thank to my family: my Dad, my Mom, my sister, my brother, my sister-in-law, and my two lovely nieces for your boundless love and support. Although we only talk on Skype at the weekend, you have always been reminding me to do my best no matter what. And that is the most precious encouragements that I need. Without you all, I couldn't be what I am today.

Ngoc Tan NGUYEN

To my Mom, my Dad, my sisters, brother and my nieces
for their boundless love, support and encouragement.

Abstract

The current architecture of the Internet has been designed to connect remote hosts. But the evolution of its usage, which is now similar to that of a global platform for content distribution undermines its original communication model. In order to bring consistency between the Internet's architecture with its use, new content-oriented network architectures have been proposed, and these are now ready to be implemented. The issues of their management, deployment, and security now arise as locks essential to lift for Internet operators. In this thesis, we propose a security monitoring plan for *Named Data Networking* (NDN), the most advanced architecture which also benefits from a functional implementation. In this context, we have characterized the most important NDN attacks - *Interest Flooding Attack* (IFA) and *Content Poisoning Attack* (CPA) - under real deployment conditions. These results have led to the development of micro-detector-based attack detection solutions leveraging hypothesis testing theory. The approach allows the design of an optimal (AUMP) test capable of providing a desired *Probability of False Alarms* (PFA) by maximizing the detection power. We have integrated these micro-detectors into a security monitoring plan to detect abnormal changes and correlate them through a Bayesian network, which can identify events impacting security in an NDN node. This proposal has been validated by simulation and experimentation on IFA and CPA attacks.

Keywords: Computer networks – Security measures, Statistical hypothesis testing, Anomaly detection (Computer security), Information-Centric Networking, Named Data Networking

Résumé

L'architecture de l'Internet a été conçue pour connecter des hôtes distants. Mais l'évolution de son usage, qui s'apparente à celui d'une plate-forme mondiale pour la distribution de contenu met à mal son modèle de communication originale. Afin de mettre en cohérence l'architecture de l'Internet et son usage, de nouvelles architectures réseaux orientées contenu ont été proposées et celles-ci sont prêtes à être mises en oeuvre. Les questions de leur gestion, déploiement et sécurité se posent alors comme des verrous indispensables à lever pour les opérateurs de l'Internet. Dans cette thèse, nous proposons un plan de surveillance de la sécurité pour *Named Data Networking* (NDN), l'architecture la plus aboutie et bénéficiant d'une implémentation fonctionnelle. Dans le déploiement réel, nous avons caractérisé les attaques NDN les plus importantes - *Interest Flooding Attack* (IFA) et *Content Poisoning Attack* (CPA). Ces résultats ont permis de concevoir des micro-détecteurs qui reposent sur

la théorie des tests d'hypothèses. L'approche permet de concevoir un test optimal (AUMP) capable d'assurer une *Probabilité de Fausses Alarmes* (PFA) désirée en maximisant la puissance de détection. Nous avons intégré ces micro-détecteurs dans un plan de surveillance de la sécurité permettant de détecter des changements anormaux et les corréler par le réseau Bayésien, qui permet d'identifier les événements de sécurité dans un noeud NDN. Cette solution a été validée par simulation et expérimentation sur les attaques IFA et CPA.

Mots clé: Réseaux d'ordinateurs – Mesures de sûreté, Tests d'hypothèses (statistique), Détection des anomalies (informatique), Réseaux orientées contenu, Named Data Networking

Contents

List of Figures

List of Tables

List of Abbreviations

ANDaNA	**A**nonymous **N**amed **Da**ta **N**etworking **A**pplication
ANFIS	**A**daptive **N**euro-**F**uzzy **I**nference **S**ystem
API	**A**pplication **P**rogramming **I**nterface
AR	**A**nonymizing **R**outer
AS	**A**utonomous **S**ystem
AUMP	**A**ssymptotically **U**niformly **M**ost **P**owerful
BN	**B**ayesian **N**etwork
BNC	**B**ayesian **N**etwork **C**lassifier
CCN	**C**ontent **C**entric **N**etwork
CDN	**C**ontent **D**istribution **N**etwork
CLT	**C**entral **L**imit **T**heorem
CPA	**C**ontent **P**oisoning **A**ttack
CPD	**C**onditional **P**robability **D**istribution
CPT	**C**onditional **P**robability **T**able
CS	**C**ontent **S**tore
CUSUM	**CU**mulative **SUM**
DAG	**D**irected **A**cyclic **G**raph
DNS	**D**omain **N**ame **S**ervice
DONA	**D**ata **O**riented **N**etwork **A**rchitecture
DoS	**D**enial **o**f **S**ervice
eGW	**e**gress **G**ate**W**ay
Eq.	**Eq**uation
ETSI	**E**uropean **T**elecommunications **S**tandards **I**nstitute
EWMA	**E**xponentially **W**eighted **M**oving **A**verage
EU	**E**uropean **U**nion
FAQ	**F**requently **A**sked **Q**uestions
FIB	**F**orwarding **I**nformation **B**ase
FId	**F**orwarding **Id**entifier
FN	**F**orwarding **N**ode
FP7	Seventh **F**ramework **P**rogramme
GDSF	**G**reedy **D**ual-**S**ize **F**requency
GLRT	**G**eneralized **L**ikelihood **R**atio **T**est

HSkip	**H**ierarchical **Skip**net
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
HTTPS	**H**yper**T**ext **T**ransfer **P**rotocol **S**ecure
ICN	**I**nformation **C**entric **N**etwork
ICNRG	**I**nformation **C**entric **N**etwork **R**esearch **G**roup
IFA	**I**nterest **F**looding **A**ttack
iGW	**i**ngress **G**ate**W**ay
IoT	**I**nternet **o**f **T**hings
IP	**I**nternet **P**rotocol
ISP	**I**nternet **S**ervice **P**rovider
JT	**J**unction **T**ree
LFU	**L**east-**F**requently **U**sed
LNB	**L**ate **N**ame **B**inding
LRU	**L**east-**R**ecently **U**sed
LRT	**L**ikelihood **R**atio **T**est
LSA	**L**ink **S**tate **A**nnouncement
MDHT	**M**ulti-level **D**istributed **H**ash **T**able
MEC	**M**ulti-access **E**dge **C**omputing
ML	**M**aximum **L**ikelihood
MMT	**M**ontimage **M**onitoring **T**ool
NACK	**N**egative **ACK**knowledge
NDO	**N**amed **D**ata **Object**
NDN	**N**amed **D**ata **N**etworking
NetInf	**Net**work of **Inf**ormation
NFD	**N**DN **F**orwarding **D**aemon
NLSR	**N**amed-data **L**ink **S**tate **R**outing
NRS	**N**ame **R**esolution **S**ervice
OLS	**O**rdinary **L**east **S**quare
PCA	**P**rincipal **C**omponent **A**nalysis
PDF	**P**robability **D**ensity **F**unction
PER	**P**IT **E**xpiration **R**ate
PFA	**P**robability of **F**alse **A**larm
PIT	**P**ending of **I**nterest **T**able
PKI	**P**ublic **K**ey **I**nfrastructure
PLA	**P**acket **L**evel **A**uthentication
PoP	**P**oint-**o**f-**P**resence
POR	**P**IT **O**ccupancy **R**ate
PSIRP	**P**ublish **S**ubscribe **I**nternet **R**outing **P**aradigm
PURSUIT	**PU**blish **SU**bscribe **I**nternet **T**echnology

RENE	**RE**ndezvous **NE**twork
RId	**R**endezvous **Id**entifier
ROC	**R**eceiver **O**perational **C**haracteristic
RFC	**R**equest **F**or **C**omment
RH	**R**esolution **H**andler
RL2FA	**R**un **L**ength **to F**alse-**A**larm
RN	**R**endezvous **N**ode
RT	**R**egistration **T**able
SAIL	**S**calable and **A**daptive **I**nternet so**L**ution
SCIC	**S**elf-**C**ertifying **I**nterest/**C**ontent
SEM	**S**ecurity **E**vent **M**anagement
SId	**S**cope **Id**entifier
SIEM	**S**ecurity **I**nformation and **E**vent **M**anagement
SIM	**S**ecurity **I**nformation **M**anagement
SMP	**S**ecurity **M**onitoring **P**lane
SNMP	**S**imple **N**etwork **M**anagement **P**rotocol
TDM	**T**hreshold-based **D**etection and **M**itigation
TM	**T**opology **M**anagement
TTL	**T**ime **T**o **L**ive
UMP	**U**niformly **M**ost **P**owerful
URI	**U**niform **R**esources **I**dentifier
VNI	**V**isual **N**etworking **I**ndex
VPN	**V**irtual **P**rivate **N**etwork

List of Notations

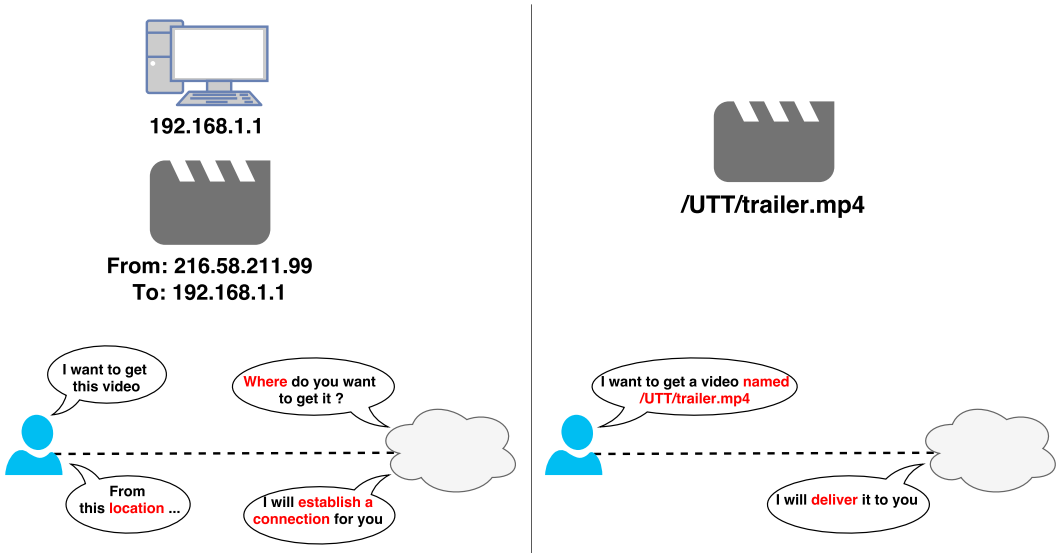
$\mathcal{H}_0$	null hypothesis, i.e. no attack case
$\mathcal{H}_1$	alternative hypothesis, i.e. attack case
i_t	number of incoming *Interests* at an instant t of a router's face
d_t	number of outgoing *Data* packets at an instant t of a router's face
ℓ_t	measured ratio of packet loss
$\mathbb{E}(x)$	expectation of a random variable x
N_{a_t}	number of malicious *Interests* received at an instant t
$\mathbb{P}_i(E)$	the probability of an event E under hypothesis $\mathcal{H}_i$
$\lfloor h \rfloor$	the "floor" under h, i.e. the greatest integer less than or equal to h
$cov(x, y)$	covariance between random variables x and y
$\boldsymbol{A}'$ or $\boldsymbol{A}^T$	transpose of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{(i,j)}$	elements on row i_{th} and column j_{th} of matrix $\boldsymbol{A}$
$\widetilde{x}$	estimated value of the variable x
$\boldsymbol{I}_N$	Identity matrix of size N
α	prescribed false alarm probability
β_δ	the detection power of the test δ
σ_x^2	variance of the random variable x

Introduction

Context

The current Internet architecture was designed decades ago when the primary interest was to connect distant systems. At that time, the communication model mainly focuses on *where* the systems are. As time goes by, the Internet and its usage have significantly evolved. Computers and devices have increased in number thanks to the more affordable price. Their processing speed and memory capacity have also been greatly improved, and so does the network links' speed. Users now can join on the Internet more easily to access an enormous pool of contents and services. People now care about *what* content or services they access and no longer *where* it is located. As such, the original communication model no longer fits the Internet's primary usage. Moreover, the Internet was originally conceived without any intentional considerations for possibly emerging problems such as security, management, and quality of service. Therefore, as more issues arise, the architecture becomes more burdensome with patches to deal with such issues.

As the *what* becomes the primary interest, the content should now become the nucleus of the network. A clean-slate approach to the future Internet following such a direction has drawn the attention of the research community, resulting in several *Information-Centric Networking* (ICN) architectures. Among these proposals, the *Named Data Networking* (NDN) is the most promising one and is in the most advanced stage among ICN proposals. Figure 1 illustrate the primary differences between NDN and IP. NDN is based on content-naming, i.e., identifying content objects with names (e.g., /UTT/trailer.mp4) instead of identifying hosts with IP address. It uses the content-based routing paradigm, meaning that network elements will route a content object by using its name. NDN also deploys in-network caching to improve the delivery performance for popular content and associates signature to each content in order to ensure integrity and security of communications.



(A) IP network (B) Information-Centric Networking

FIGURE 1: Communication model of (A) Internet Protocol (IP) network and (B) Information-Centric Networking

Problem statement

After a decade of research and development, the NDN protocol is now mature enough with a functional *NDN Forwarding Daemon* (NFD) implementation. As a result, research efforts in NDN community have shifted to the management, deployment and security considerations. By envisaging the security requirement in the design, NDN intrinsically mitigates several attacks inherited from IP network. Nevertheless, novel features and router's components in NDN also exposes the network to new security flaws, making NDN intrinsic security insufficient. Many security threats have been discovered in NDN. In order to be adopted and reliably deployed by Internet Service Providers (ISPs) in their operation infrastructure, the security of NDN must be ensured, and thus, a global monitoring plane for the security of NDN is essential.

The NDN *Security Monitoring Plane* (SMP) must be able to address existing security threats in NDN efficiently. Among discovered NDN attacks, *Interest Flooding Attack* (IFA) and *Content Poisoning Attack* (CPA) are considered as the most important ones. The two attacks have drawn the attention of the NDN community. However, previous works on these topics share several common drawbacks. First, most of the previous works evaluate their proposals with results from a simulated environment, thus questioning the actual performance in the real deployment context. Moreover, a majority of existing works do not provide reliable detection. More specifically, they cannot guarantee a prescribed rate of false alarms which waste network resources

and accidentally penalize legitimate clients. Also, the detection threshold either requires a non-trivial learning phase or is based on authors' experience, leading to rigid and unreliable performance. Furthermore, existing works in NDN security usually focus on a particular NDN attack. As such, the envisaged SMP must also be generic, i.e., can tackle other discovered threats, as well as potential attacks revealed in the future. Such a generic security solution has not been proposed so far in the literature.

Contributions

The objective of this thesis is to propose a *Security Monitoring Plane* (SMP) for the real deployment of NDN that can address existing security flaws efficiently and allow considering further threats not yet revealed to date. The proposed plane consists of reliable micro detectors that are based on hypothesis testing theory. We argue that an unreliable detection will lead to a lot of false alarms. Once being associated with a countermeasure, they can cause instability, wasting the network resources and penalizing legitimate users, which is unacceptable in a real deployment context. As such, we deliberately prioritize developing a reliable detection in terms of guaranteeing a prescribed rate of false alarms, rather than rushing prematurely to a mitigation solution for the envisaged SMP. As use-cases for the SMP proposal, we first characterize the most significant NDN security attacks - IFA and CPA - under real deployment of NDN. Such an investigation has never been done in any previous works. We identify attack parameters and highlight the impacts on important network entities.

As IFA can be featured by a single metric, we hence address it with a micro detector using hypothesis testing theory. The methodology allows designing an optimal *Asymptotically Uniformly Most Powerful* (AUMP) test guaranteeing a prescribed *Probability of False-Alarms* (PFA) while maximizing the detection power. The resulted micro detector is extended to a sequential version to enhance its accuracy. We assess the overall performance of the micro detector, using data from both simulations and real experimentations.

For attacks that impact several aspects of NDN node status such as CPA, as well as unrevealed potential attacks, we propose an SMP with an exhaustive set of metrics based on a thorough analysis of NFD pipelines. Each metric is associated with a micro detector to catch any abnormal change under the constraint of a prescribed PFA. Afterward, a correlation engine based on a Bayesian Network is proposed to combine micro detector alarms and thus to identify any abnormal security events in an NDN node. To validate our proposal, two CPA scenarios have been considered

in a real testbed. The collected data demonstrate the capability of our solution to detect the CPA accurately with various rates.

Document Structure

The rest of this document is organized as follows. Chapter 1 familiarizes the concept of ICN by providing an overview of ICN architectures. We introduce ICN's key concepts and focus on the detail of NDN architecture as it is the most promising one. Other ICN architectures are also introduced and compared with NDN such that one can capture the differences between those approaches implementing ICN's key concepts. We also present the current state of ICN deployment efforts from different projects across all over the world.

Chapter 2 zoom in state of the art on NDN security, especially the data plane's security in NDN. In this chapter, we propose a taxonomy to classify NDN attacks, introducing attacks' principle and remarkable existing works on the issues. The survey is summarized with an assessment of NDN attacks according to selected attributes. Based on the assessment results, we argue for the choice of IFA and CPA as two use-cases for our study.

Chapter 3 concentrates on featuring IFA and CPA in NDN real deployment. The chapter first motivates the need of considering NDN security threats in real conditions and introduces common features in our testbeds deployment. For each attack, we present scenarios in which the attack can be launched successfully, specify the experiment settings and explain the featuring results with data collected from our testbed.

Chapter 4 present our design of the micro detector, step by step, with hypothesis testing theory, using IFA as a use-case for evaluation. The chapter first introduces the basic framework of hypothesis testing theory and formally defines the IFA detection problem. We then propose an optimal detector in an ideal case when the parameter is known, followed by a test when the parameter is unknown. The detection scheme is extended to a sequential version to enhanced the accuracy. We assess the overall performance of the micro detector. Data from both simulations and real experimentations insist that those results show "sharpness of theoretical findings" and the gain in performance of sequential detection.

Chapter 5 presents the proposed SMP for NDN security. We first provide some background for the chapter, including an overview of network management plane and an introduction to Bayesian networks. We then present our proposal for an SMP of NDN after motivating for its need. An exhaustive set of metrics is proposed

based on a thorough analysis of NFD pipelines. Each metric is associated with a micro detector that can capture abnormal shifts from the metric's normal behavior with a prescribed PFA. Alarms from micro detectors are correlated by a Bayesian-network-based engine to identify any abnormal security events in an NDN node. By leveraging CPA data performed on a real NDN testbed, we provide numerical results that demonstrate the relevance and the efficiency of the proposed approach.

Chapter 1

Information-Centric Networking Architectures - State of the Art

Contents

Since it was designed decades ago, the Internet and its usage have significantly evolved. As such, the original host-centric communication model no longer fits its primary usage of distributing content. *Information-Centric Networking* (ICN) is a clean-slate approach to the future Internet utilizing the novel information-centric

communication model. This chapter is an introduction to ICN's fundamental concepts and proposed architectures. We first motivate the development of ICN, followed by an introduction of ICN key features and their differences from the current IP network's. Afterward, we present *Named Data Networking* (NDN) – the most promising ICN proposal – with details. Other ICN architectures are also introduced, including *Data Oriented Network Architecture*, *Publish Subscribe Internet Technology*, *Network of Information*. The proposals above are then compared to highlight their differences between when implementing ICN's key concepts. After a decade of research and development, ICN proposals have achieved a certain level of technical maturity to be considered for real deployment. To help envision the activeness in this aspect, the chapter also presents the current state of ICN deployment efforts from different projects across all over the world.

1.1 Motivations of Information-Centric Networking

Decades ago, the Internet was initially designed to connect and exchange data between a limited number of stationary machines over long distance links, with well-established trust relationships. Such an original demand has evolved tremendously. Nowadays, it is much easier to connect to the Internet thanks to the wireless connection and smaller, cheaper mobile devices. According to the Cisco *Visual Networking Index* (VNI) Forecast[1], by the year 2021, there will be 4.6 billion global Internet users (58% world population), with 27.1 billion networked devices and connections. Moreover, users can also access a variety of applications, services and especially data on the Internet without much concerns about their physical location. Such an enormous growth of the Internet has risen new demands that were not envisaged for the existing architecture at the first place, such as support for data distribution, mobility, security, trust. New patches have been added to the network to cope with these emerging requirements, making it burdensome and a more complex system. Motivated by such a necessity, many research efforts have addressed the future Internet architectures from a clean-slate perspective, resulting in ICN designs that are based on the information-centric communication model.

1.2 Information-Centric Networking Overview

In this section, we introduce key concepts of ICN architectures, including information naming, information delivery, intrinsic security and mobility [6]. To highlight

[1]See www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni

the novelty of ICN, each ICN feature will be compared with its counterpart in the current IP network.

1.2.1 Information Naming

The primary difference between ICN and the current Internet is that ICN names information (data) objects instead of hosts. The current Internet is based on a host-centric communication model, i.e., the host is the network primitive and is identified by an IP address. When a user wants to get data, he must know the IP address of the server where the data is located. Such a communication model fits well in the network's early stage whose primary purpose is to connect a limited number of data sources and hosts. As time goes by, the Internet has significantly evolved. Devices now have more processing power and capacity, as well as connectivity, allowing them to access to an enormous amount of services. The Internet now becomes an environment mainly for distributing data. As a result, the host-centric communication model no longer fits in the Internet primary usage. Hence, it gets more challenging to manage the network and more complicated for users to know the optimal location to retrieve the data, hence preventing the network from reaching its delivery potential.

On the other hand, ICN uses the information-centric communication model, i.e., the data object is identified with a content name, hence called a *Named Data Object* (NDO). Communication is now initiated by using the content name. By naming data objects independently of their location, ICN architectures make the data request, and the data retrieval occur in a name-based manner. Instead of specifying the address of the content source, users only need to indicate the data name[2]. The responsibility of finding the best source for requested data is now delegated to the network. As such, any copies of an NDO are considered equivalent and can be used to answer the user's request.

In general, ICN has two types of naming schemes, namely flat and hierarchical [7]. The flat naming scheme identifies an NDO with a randomly-looking bit string. The most common way to compute flat name is to use a cryptographic hash function, for instance, on the data itself. On the other hand, the hierarchical name has a structure similar to the *Uniform Resources Identifier* (URI) [8], which consists of many name components. Each name component can be human-readable or randomly-looking bit strings.

[2]The terms "subscriber", "requester" and "user" are used interchangeably to mention the entity that requests for an NDO. Similarly, the terms "publisher", "content source" and "provider" mention the one that publishes an NDO

1.2.2 Information Delivery and In-network Caching

The current Internet lacks natural mechanisms to deliver a massive amount of data efficiently, as well as to handle sudden bursts of traffic. It only focuses on forwarding the received packets as fast as possible without finding any possible optimization for the delivery. Although techniques that help enhance the information delivery (e.g., web cache, multicast) were developed, they are resource-consuming and application-specific. Hence, the network capability is not fully exploited. Also, redundant traffic still traverses on Internet links every day, wasting the bandwidth and diminishing network performance [2, 9]. On the other hand, ICN architectures are natively designed to handle information delivery more efficiently (e.g., reduce redundant traffic, minimize latency) with three functionalities: *name resolution*, *data routing* and *caching*.

Name Resolution and Data Routing

The *name resolution* functionality resolves the requested data name into information of optimal data sources and can aggregate similar requests to reduce unnecessary traffic to the upstream. Meanwhile, the *data routing* is responsible for transferring the NDO from the source back to the user and natively supports multicast, thus can distribute data more efficiently. There are two possible models of these two functionalities:

- **Coupled (or integrated) model**: the request is forwarded directly to a data source, while the NDO follows the reverse path back to the user. Consequently, the network entity responsible for name resolution also decides the path of data routing;
- **Decoupled (or independent) model**: the request is first sent to a dedicated entity for name resolution. This entity will then answer the user with information about data sources. As such, the user can decide the optimal data source that he would use. Another possibility is that the name resolution entity selects the optimal data source and directs it to send the data back to the user.

Caching

ICN architectures utilize in-network caching to improve the delivery performance, especially in term of the delivery delay. Caching in ICN can be categorized into two types [10]:

- **On-path caching**: the network makes use of cached copies along the path taken by the name resolution request. In other words, whenever a network entity receives a request, it prioritizes answering the request with the cached data, if possible, rather than forwarding the request. On-path caching is compatible with both model of name resolution and data routing;
- **Off-path caching**: the network also exploits cached copies located outside the path of the name resolution request. More specifically, with decoupled name resolution and data routing, the name resolution component must know locations of cached copies and handle them as a data provider. Off-path caching is not compatible with the coupled model of name resolution and data routing, since the request and the NDO always take the same path.

1.2.3 Information security

In the beginning, the Internet was designed to operate in a trustworthy environment without any consideration for the information security. The network is sender-driven, i.e., no matter who has sent the packet, the network mainly focuses on forwarding it to the indicated destination. It has no mechanism to reduce redundant or unsolicited traffic (Section 1.2.2), thus allowing attackers to launch *Denial-of-Service* (DoS) attacks against the Internet infrastructure. Besides, attackers can send altered data to end-users (e.g., fake web) and cause damage. Although several security solutions against these behaviors exist, the processing overhead and the end-to-end communication model prevent implementing them deeper into the network, where it would be most effective in avoiding or identifying and stopping attacks [2]. Thus malicious data still gets forwarded.

On the other hand, ICN is receiver-driven, i.e., the network will not forward data to a receiver unless he has explicitly asked for it. As such, network nodes will refuse unsolicited flow and reduce unwanted data. Moreover, ICN architectures intrinsically provide means to verify data integrity. For hierarchical and flat naming scheme, it is the digital signature and self-certifying name, respectively, which are usually computed from the data itself by using the hash function. Moreover, in ICN, the original data sources does not always answer all the requests. NDOs can be cached in network elements and utilized to resolve requests, hence enhancing the availability of information. Furthermore, such communication also decouples the users and the data sources, protecting their privacy during data exchange.

1.2.4 Mobility Support

Nowadays, mobile devices have become very popular. According to the Cisco VNI Forecast, by 2021, there will be nearly 12 billions mobile-connected devices, with 5.5 billion global mobile users. However, our current Internet was initially designed for fixed hosts, not for mobile ones. Communication in the current Internet is end-to-end, i.e., two hosts must successfully establish a connection and maintain it during the data exchange. Thus, the mobility becomes a problem to manage end-to-end connections.

On the other hand, NDOs in ICN are named independently of their location. As such, ICN does not require end-to-end connections for data exchange. The end-to-end connection management becomes unnecessary for ICN, thus simplifying the mobility problem. A mobile user only needs to send new requests for NDOs when he moves to a new location. Meanwhile, when a data source relocates, the network can utilize cached copies or other alternative data sources to respond to user's requests while updating routing information for the data source's new location.

1.3 Named Data Networking

The most promising and popular among ICN architectures – *Named Data Networking* (NDN) [11] is a research project in the Future Internet Architecture Program funded by the U.S. National Science Foundation (NSF). Its root is an earlier project, *Content-Centric Networking* (CCN) [12], which Van Jacobson started at Xerox PARC in 2006[3]. Its basic ideas were described in a Google tech talk of Van Jacobson, long before the first paper describing the NDN architecture was published.

1.3.1 Naming Scheme and Packets

NDN uses a hierarchical naming scheme. NDN content name (or *prefix*) consists of at least one name component and can be presented in the Uniform Resource Locator (URL) form. For instance, an introduction video of UTT can have the name `/utt/videos/utt_intro.mpg`. Such a naming scheme is human-readable and reflects the relationships of data elements. For example, segment 2 of version 3 of a UTT demo video might be named `/utt/videos/utt_intro.mpg/3/2`. Besides, similar requests for the content from `/utt` can also be aggregated, thus facilitating the scalability of the architecture [2]. A naming convention must be agreed by both the content providers and users so that they operate smoothly over NDN content names.

[3] NDN Frequently Asked Questions

Besides, a naming system is necessary to define and allocate top-level names, ensuring the uniqueness of content name between providers. However, not all NDN names need to be globally unique. In some local communication, NDN names can be based on local context.

Communications in NDN uses two types of packets, namely *Interest* and *Data*. The *Interest* carries the user's request for content, while *Data* contains the NDO itself. An *Interest* packet consists of the following fields[4]:

- *Name* (required): indicates the prefix of the requested content. Note that it is unnecessary to specify the full content name;
- *Nonce* (required): a value generated randomly by the issuing user. When combining with *Name* field, *Nonce* allow uniquely identifying an *Interest*. More specifically, two *Interests* having the same *Name* are considered duplicated (or looped) if they also have the same *Nonce*. Otherwise, they are two separate requests for the same content. As such, the router can detect duplicated *Interests* for further processing;
- *InterestLifetime*: specifies the time remaining in milliseconds before the *Interest* times out. *InterestLifetime* is set by the application that issues the *Interest*. When omitted, the default value of *InterestLifetime* is 4 seconds.
- *Selector*: since NDN use longest-prefix-matching, an *Interest* can have several matching *Data*, this field provides additional information to select the only one *Data* to return. *Selector* contains several sub-fields, for example:
  - *MinSuffixComponents, MaxSuffixComponents*: refer respectively to the minimum and the maximum number of name components beyond those in the prefix that are allowed in the matching *Data*. Default value for *MinSuffixComponents* is 0 and for *MaxSuffixComponents* is infinite, meaning that any *Data* whose name starts with the requested prefix is a match;
  - *PublisherPublicKeyLocator*: indicates the prefix of the key that signs the requested *Data*. As such, the user can select answers from a particular publisher;
  - *Exclude*: specifies a list of suffixes (i.e., following name components of the prefix) that the user wants to avoid. For instance, if an *Interest* is expressed for `/utt/img` and *Exclude* defines one name component `/gala`, then any *Data* that has prefix `/utt/img/gala` will not match this *Interest*;
  - *ChildSelector*: an *Interest* can match several *Data* with same prefix's length. The *ChildSelector* bit expresses a preference for the name component right

[4]See http://named-data.net/doc/NDN-TLV/current/interest.html

after the prefix (i.e., child) based on the canonical NDN name component ordering[5]. The bit values of 0 (by default) and 1 correspond, respectively, to the leftmost (least) and the rightmost (greatest) child preference;

– *MustBeFresh*: when this bit is set, the user insists in getting a "fresh" *Data*. If it is absent from an *Interest*, the matching *Data* does not have to be fresh.

Any node receiving the *Interest* and having the requested data can respond with a *Data* packet. A *Data* packet has the following fields[6]:

- *Name*: contains the full name of the NDO carried inside;
- *MetaInfo*: this optional field provides additional information about the *Data*, such as *ContentType* and *FreshnessPeriod*.
  - *ContentType*: reveals the type of information transmitted in the *Data*, e.g., public key, payload;
  - *FreshnessPeriod*: when a *Data* is cached in router, it is marked as "fresh". After its *FreshnessPeriod* has elapsed, the router will mark it as stale, making it less prioritized for answering *Interest*. This field is set by the original producer and its expiration also implies that the producer may have published newer version. When this field is absent or equals to 0, the *Data* will always be fresh;
  - *FinalBlockId*: indicates the identifier of the final block in a sequence of fragments.
- *Content*: contains the requested NDO;
- *Signature*: includes the digital signature and supporting information used to verify the integrity of *Data*. This fields includes two sub-fields:
  - *SignatureInfo* fully describes the signature, signature algorithm (e.g., DigestSha256), and any other relevant information to obtain parent certificate (e.g., the name of the *Data* containing certificate or public key);
  - *SignatureValue* represent actual bits of the signature, computed by taking the hash over all other fields (including *SignatureInfo*), then encrypting it using provider's private key.

1.3.2 Router and Name-based Routing

Name resolution and data routing functionalities in NDN are integrated into the NDN router. NDN leverages the term "*face*" – a generalization of interface. A face

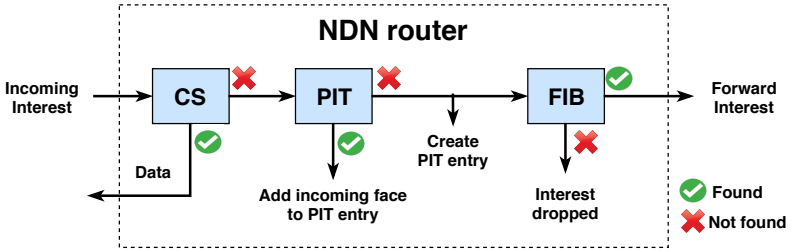
[5]See http://named-data.net/doc/NDN-TLV/current/name.html#name
[6]See http://named-data.net/doc/NDN-TLV/current/data.html

in NDN can refer to a router interface used to communicate either with other nodes or between internal components within a router. An NDN router has three main components:

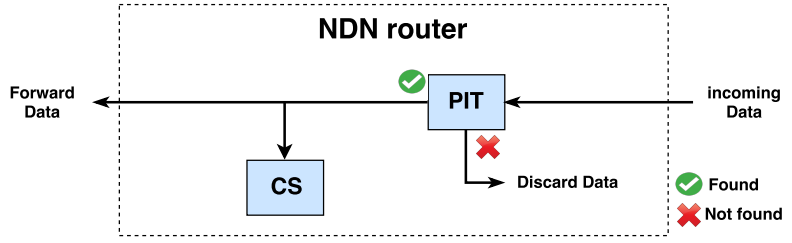
- *Forwarding Information Base (FIB)*: similar to a routing table in IP router, this component contains next hops to send *Interests*. Each FIB entry consists of a name prefix and a list of outgoing faces that can be used to send *Interest* to content providers;
- *Pending Interest Table (PIT)*: keeps tracks of forwarded *Interests*. Each PIT entry consists of a name prefix, a list of incoming faces of pending *Interests* for that prefix and a list of outgoing faces to which aggregated *Interests* are forwarded; This information will then be used to forward *Data* back to users.
- *Content Store (CS)*: a local cache in the router which provides on-path caching to improve the delivery performance in NDN.

NDN Router's Operation

Figure 1.1 illustrates how NDN router operates when it receives an *Interest* and a *Data*. When an *Interest* arrives (Figure 1.1a), the router first looks up in the CS. If



(A) For incoming *Interest*



(B) For incoming *Data*

FIGURE 1.1: NDN router's operation for (A) an incoming *Interest* and (B) an incoming *Data* (modified from [1])

a matching cached copy exists in CS, the router immediately forwards that cached copy to the incoming face of *Interest*. In case there are many matching cached copies found, the *Selector* field (Section 1.3.1) of the *Interest* is used to decide which cached *Data* will be returned. Then, the *Interest* is discarded. If no matching *Data* is found

in CS, the router checks its PIT. If a longest-prefix matching entry is found and the *Interest*'s *Nonce* value is not recorded yet (meaning that this *Interest* is not a loop), the *Interest*'s incoming face will be added to the matching PIT entry. The *Interest* will be discarded because there is already an *Interest* for this *Data* sent upstream. In case there is no matching PIT entry, the router checks the FIB for a longest-prefix match to find an outgoing face. Depending on the router's forwarding strategy, the *Interest* can be, for example, forwarded to the lowest-cost route (i.e., bestroute strategy) or multicasted to several outgoing faces (i.e., multicast strategy). When no matching entry in FIB is found, the router discards the *Interest* [12].

When a *Data* arrives (Figure 1.1b), the router looks up its PIT first. If there is no matching PIT entry, the *Data* is considered unsolicited, and it is discarded. Otherwise, the *Data* is duplicated and forwarded to all incoming faces of the matching PIT entry. The router also decides whether to insert this *Data* into the CS. After forwarding the *Data*, the matching PIT entry is erased immediately, implying that "One *Interest* retrieves at most one *Data* packet." PIT entries that do not receive any matching *Data* after a period will get expired and erased.

1.3.3 Transport

NDN does not have a separate transport layer. Demultiplexing, reliable delivery, and congestion control functions are moved to applications, their supporting libraries and the strategy component in the forwarding plane. Segmenting and reassembling segments are done by using hierarchical names. Information required for transport is in the content name. For example, the name `/utt/videos/utt_intro.mpg/3/2` specifies where to forward *Interests* for that name (`/utt/`), which application should receive them (`/videos/`), and any application-specific information (version 3, segment 2).

NDN is designed so that it can operate on top of unreliable packet delivery services. To ensure the reliability of delivery, unsatisfied PIT entries get expired after a period and are discarded, making the user issue another *Interest* if he still wants to get the content. Caches also help data retransmission during congestion. If a router already caches a *Data* packet before it is dropped at a congestion link, the retransmitted *Interest* will be satisfied by this cached copy, reducing the latency during congestion. Flow control and congestion control no longer depend on end hosts. NDN's forwarding mechanism (Section 1.3.2) helps to reduce significant duplicated *Data* and *Interests*. Each *Interest* has a *Nonce* fields to detect and get rid of duplicate *Interest* (Section 1.3.1). Besides, checking PIT entry before forwarding an *Interest* helps reduce redundant *Interests* for the same content. Checking PIT before forwarding a *Data*, in the same way, helps eradicate unsolicited *Data* traffic. Each FIB entry and

PIT entry record several faces, thus support multipath and provide rich connectivity for nodes in its network.

1.3.4 Intrinsic Security Features

SignatureInfo and *SignatureValue* in *Data* packet provide an intrinsic mechanism for authenticity and integrity. The computation process of *Signature* has been described in Section 1.3.1. Any nodes receiving a *Data* packet can retrieve publisher's public key with information in *SignatureInfo* sub-field and then uses it to decrypt the *SignatureValue*. The result is then compared to the hash over all other *Data*'s fields. This process allows verifying the integrity of the *Data* packet received. Moreover, to verify the authenticity of the content provider, the user must trust the owner of the public key used for signing the *Data*. The hierarchical name structure facilitates building trust relationships. For instance, the content `/fr/utt/stmr/main.html` is signed by the owner of the `/fr/utt/stmr` domain, whose key, in turn, is certified by the owner of the `/fr/utt` domain.

1.4 Other Information-Centric Networking Architectures

1.4.1 Data Oriented Network Architecture

Data Oriented Network Architecture (DONA) [13] is one of the first complete ICN architectures. It replaces hierarchical *Domain Name Service* (DNS) with flat names and deploys an overlay name resolution service that translates its flat names to the corresponding information for data retrieval, while still maintaining IP protocol and routing. DONA is motivated by the following user-relevant issues in the IP network:

- **Persistence:** the user prefers that a data name remain valid as long as the underlying data is still available, even its source relocates;
- **Availability:** a request should be replied as quick as possible with reliability by replication at endpoints. However, IP network establishes a communication to a fixed IP address which is not always the nearest data source;
- **Authenticity:** the user wants to be sure that the data source is trustworthy.

Naming Scheme

Naming scheme will handle authenticity and persistence issues. DONA uses flat names instead of hierarchical names as NDN. Flat names in DONA are organized

around *principals*. Each principal is associated with a public-private key pair managed by an external mechanism. Names in DONA have a form $(\mathbf{P} : \mathbf{L})$. The **P** name component is the hash of the principal's public key, while the **L** component is a label given by the principal so that the data name $(\mathbf{P} : \mathbf{L})$ is globally unique. For static data, the **L** component is usually the hash of the data itself. Only hosts which are authorized by the principal (i.e., they can access to the principal's public key) can provide the data, forming a group of data providers. Since the flat name is not user-friendly, DONA's users require an external mechanism to map flat names to user-friendly names.

Name Resolution

While the name resolution and data routing functionalities in NDN are integrated through name-based routing, DONA utilizes the independent model of the two functionalities. DONA is deployed as an overlay over IP network that handles the name resolution functionality and retains IP protocol for data routing.

Resolution Handler To resolve flat names, DONA introduces a new network entity called *Resolution Handlers* (RHs). At least one logical RH is associated to each *Autonomous System* (AS). RHs are hierarchically organized according to the AS-level relationship (Figure 1.2). Each RH maintains a *Registration Table* (RT) to forward requests for name resolution. Each RT entry includes: (1) the content's flat name; (2) corresponding next-hop information and (3) the distance to the data source. The next-hop information can point to a next-hop RH or the IP address of the data source located in the same AS. The distance to the data source can be the number of RH-hop to the provider or any other metrics. Figure 1.2 is an example of DONA's operation. Name resolution is accomplished by two primary messages: REGISTER message for data registration and FIND message for data retrieval.

Data Registration When a publisher wants to publish its data, it sends a REGISTER message to its local RH (arrow 1). When an RH receives a REGISTER from its children or local users, it will check its RT first. If no matching RT entry is found, the RH will create a new entry for this data. In case a matching RT entry exists and the incoming REGISTER points to a closer data source, the RH will update its matching RT entry with information to the better sources. The RH then update the REGISTER message by appending its distance/cost to the previous-hop RH and forward the message to the parent and peer RHs (arrow 2 - 3). When a REGISTER message comes from peer RHs, the forwarding decision depends on the local policy. By doing this, the higher level RHs will store entries about all registrations from its lower level RHs,

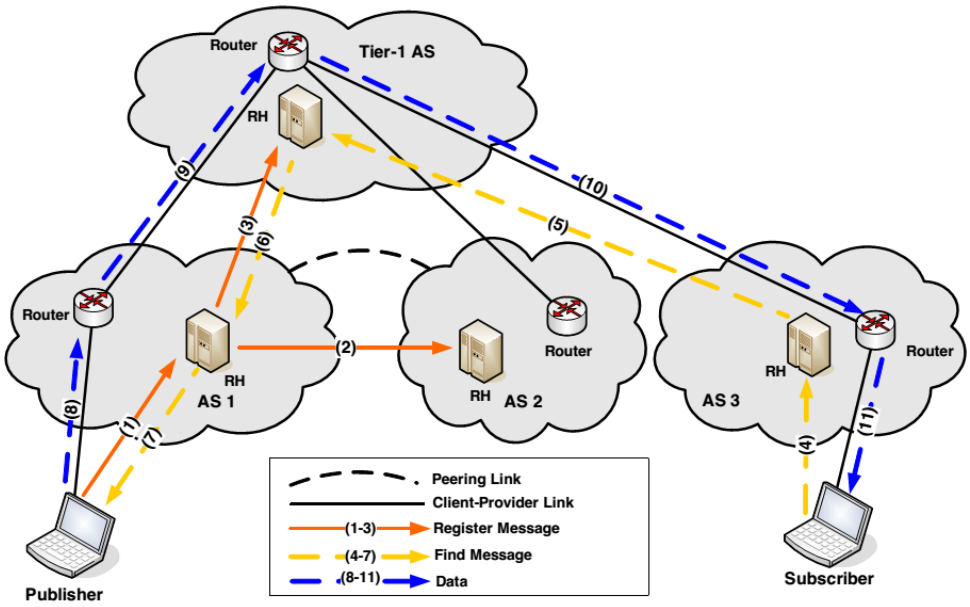


FIGURE 1.2: DONA's name resolution [2]

with the closest location to retrieve the data, ensuring name resolution for all existent data. In [13], the authors argue that the memory and processing requirements of a large ISP can be easily handled within reach of today's technology, hence enabling the feasibility of keeping all registrations in the higher level RHs. The REGISTER message has its lifetime and must be refreshed periodically.

The publisher can register for a specific data by indicating its unique flat name $(\mathbf{P} : \mathbf{L})$ in the REGISTER message. It can also register for a wildcard name $(\mathbf{P} : *)$, meaning that the publisher can serve all data associated with the principal P. There is also UNREGISTER message, informing that the publisher no longer serves some specific data. The REGISTER and UNREGISTER messages enable the mobility for DONA: when a host relocates, it unregisters its data from its local RH and then re-registers its data to the new location's RH.

Data Retrieval FIND message is dedicated to the data retrieval. When an RH receives a FIND (arrow 4), it looks up in its RT for a longest-prefix matching entry and forwards the FIND to the corresponding next-hop. Otherwise, the FIND message will be sent to the parent RH (arrow 5). Unless the requested data does not exist, this progress certainly succeeds (arrow 6 - 7) since the highest RHs know about all data registrations from its children networks via registration forwarding. In case the FIND message reaches the Tier-1 AS and does not find any matching RT entry, the Tier-1 RH returns an error message to the sources of FIND. After the FIND message arrives at the publisher, the data is sent to back to the user, using the IP network (arrow 8-11).

The user can retrieve a particular data by specifying its globally unique name $(\mathbf{P} : \mathbf{L})$ in a FIND. Note that for static data, the **L** component is usually the hash of the data itself. As such, for popular static data that are hosted by several publishers, the user can issues a FIND message for $(* : \mathbf{L})$, implying that he wants to retrieve this particular static data from no matter which provider.

Extensions

Caching Similarly to NDN, RH can be extended to provide on-path caching. To cache an NDO, instead of the IP address of the requesting user, RH replaces the source IP address in the FIND message with its IP address. Consequently, the data undoubtedly go through this RH before going back to the user. The RH has to keep a state before changing the source IP address in the FIND message to forward the data after caching it. When a caching-enabled RH receives a FIND message, it looks up in its cache first. If a cached copy exists, the RH will return it immediately.

Long-term Subscription Long-term subscription can be achieved by adding a *Time-To-Live* (TTL) field to the FIND message. When a publisher receives a FIND message with a TTL field, it creates a record for a long-term subscription. This record consists of the list of subscribers' IP address, subscribed data names and the valid duration of this subscription. Whenever the publisher has an update for data, it checks subscription records and sends data to valid subscribers.

Overloaded Providers Avoidance When a publisher sends REGISTER to the local RH, it also includes information about its current load. Based on such information, RH may adjust the distance for registrations from high-load publishers, so that a better one will be kept in RT. Moreover, RH can store additional information about a secondary data sources in its RT entries. As such, when a publisher is overloaded, RH can forward FIND messages to other data sources.

Intrinsic Security Features

DONA's flat naming scheme provides an intrinsic mechanism to verify the data authenticity and integrity. A user issuing a FIND message will receive the data along with the principal's public key and a signature as meta-data. As the **P** component is the hash of the principal's public key, the user can verify the data authenticity. To check the data integrity, the user can use either the **L** component for static data or the received public key to decrypt the signature.

Moreover, when a publisher wants to register data, he must solve a challenge from the local RH by signing a nonce with the principal's private key and sending it back. The local RH then uses the principal's public key to decrypt the response. If the result matches the issued nonce value, the local RH will accept REGISTER messages from the publisher.

Furthermore, the REGISTER messages are also authenticated. As a part of establishing parent/child/peering relationships between AS, RHs are assumed to exchange their public keys securely. When receiving a REGISTER message, the RH also sign it before forwarding. As such, the other RHs can be sure that the message comes from a legitimate RH.

1.4.2 Publish Subscribe Internet Technology

Funded by the *European Union* (EU) *Seventh Framework Programme* (FP7), the *Publish-Subscribe Internet Routing Paradigm* (PSIRP) [14] and its continuation, the *Publish-Subscribe Internet Technology* (PURSUIT) [15] propose an architecture that completely replaces the IP protocol stack with a publish-subscribe one [2]. The PURSUIT architecture includes three separate functions:

- **Rendezvous function:** matches subscriptions and publications;
- **Topology management function:** directed by the rendezvous function, creates a route from the publisher to the subscriber;
- **Forwarding function:** transfers data by using the route from the topology management function.

Naming Scheme

In contrast with NDN hierarchical name, PURSUIT name is a unique pair of flat identifiers: a *Rendezvous ID* (RId) and a *Scope ID* (SId). The RId is NDO specific and has to be unique within a scope, while the SId denotes the scope to which the NDO belongs. A scope is a defined group of related NDOs. An NDO must have at least one scope and can belong to multiple scopes. NDOs in the same scope can have a common policy, allowing the architecture to limit the reachability of these objects. Besides, SIds can be nested within each other, producing a flexible structure. As a result, a complete PURSUIT name can contain a sequence of SIds and a single RId, i.e., having the form of `<SIdSequence>:<RId>`.

Name Resolution and Data Routing

Name resolution and data routing in PURSUIT are decoupled and demonstrated in Figure 1.3. Name resolution is handled by the rendezvous function via a *Rendezvous Network* (RENE), while data routing is organized by topology management function via *Topology Manager* (TM) nodes, and is executed by forwarding function via *Forwarding Nodes* (FNs).

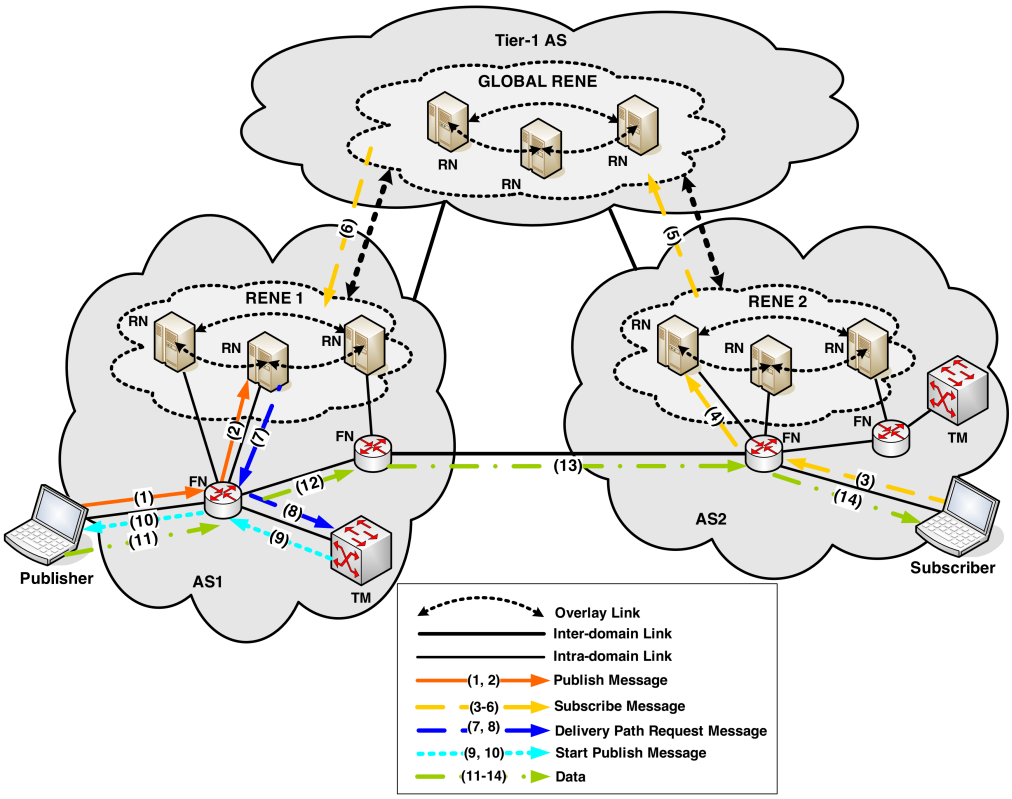


FIGURE 1.3: Name resolution and data routing in PURSUIT [2]

RENEs are organized in a hierarchy, with local RENEs and a global RENE. Similar rendezvous functionality has been used in many distributed systems [16–19]. Each RENE consists of *Rendezvous Nodes* (RNs), and each RN is in charge of a SId. When a publisher wants to advertise an NDO, it sends to its local FN a publication message carrying the RId and SId of the requested NDO (arrow 1). The message is forwarded to the RN responsible for managing the corresponding SId (arrow 2). If a subscriber needs an NDO, it sends a subscription message specifying the RId and SId of the NDO to an appropriate RN (arrows 3-6) which then instructs a TM node to find a delivery path (arrows 7-8).

TM nodes jointly monitor the network topology, detects changes by executing a distributed routing protocol. Once a TM node receives a request from an RN, it will create a path from the publisher to the subscriber and then sends it to the

publisher (arrows 9-10) so that the data delivery can start (arrow 11-14). Each FN assigns a *Forwarding ID* (FId) to each of its outgoing links, and propagates these FIds via the routing protocol. A delivery path includes a series of FIds, encoded using a technique based on Bloom filters [20]. Specifically, a path through the network is encoded by OR-ing the FIds of constituent links. The resulting Bloom filter is included in each data packet. When an FN receives a data packet, it simply verifies the tags of its outgoing links with the Bloom filter in the packet. If any FId matches, then the packet is forwarded over the corresponding link. As such, FNs only have to maintain the FId of its outgoing links.

Caching and Mobility

On-path caching is ineffective in PURSUIT due to the decoupled model of name resolution and data routing functionalities, i.e., the request forwarding path (to an appropriate RN) and the data forwarding path (to the subscriber) are different. PURSUIT can support off-path caching by deploying caches near several RNs. TM nodes can help caches gather NDOs by instructing FNs to forward a copy of NDO to a cache nearby. Caches will operate as publishers, i.e., advertise their available NDOs to RENE.

Mobility in PURSUIT is greatly facilitated by multicast and caching. When a subscriber moves, NDOs can be multicasted to multiple possible locations of the subscriber and can be retrieved from nearby caches after the hand-off. Mobility prediction can be used to reduce hand-off latencies by caching NDOs to the areas where the subscriber is expected to move after the hand-off. The publisher mobility is more burdensome because TM nodes need to update topology information.

Intrinsic Security Features

PURSUIT supports *Packet Level Authentication* (PLA) [21], allowing individual packet's encryption and signing. As a result, PURSUIT assures data integrity and confidentiality. Flat ID also permits self-certifying names for static NDOs by using the hash of the data as RId. Furthermore, paths encoded into Bloom filters [22] can use dynamic FIds, making it infeasible for an attacker to craft Bloom filters or reuse Bloom filters to launch DoS attacks.

1.4.3 Network of Information

The *Network of Information* (NetInf) [23] proposal is a research area of the *Scalable and Adaptive Internet Solutions* (SAIL) project[7] and its predecessor 4WARD [24], funded by the FP7 of EU. NetInf wants to target global-scale communication. A particular feature of NetInf is the generality: it combines elements present in the NDN and PURSUIT approaches and can even operate in a hybrid mode. Such an architecture enables the adaptation to different networks and deployments, hence facilitating a smooth transition from the current Internet.

Naming Scheme and Messages

NetInf uses the *named information* (ni) URI scheme [25] - a general URI scheme developed by the project to foster application development and simplify migration. The structure of NetInf name is described as follow:

```
ni://<Authority>/<Hash_algorithm>;<Hash_value>?<Parameter>
```

- **Authority**: this optional component provides some structure in the content name, thus assisting name-based routing and name resolution process;
- **Hash_algorithm**: indicates the hash algorithm used to create the hash value;
- **Hash_value**: for static data, the hash value is calculated over the NDO associated with the name. In case of dynamic data, this component is the hash of the public key which will be used to decrypt the signed data;
- **Parameter**: a list of optional attributes associated with the object, e.g., content attributes, owner's information.

NetInf names are flat-ish. On the one hand, they are flat because they require an exact match for name comparison and they do not contain any location or organizational information. At the same time, they can be considered hierarchical when used for routing and name resolution since routers can use longest prefix matching to determine how to route a message.

Communications in NetInf are based on the following fundamental messages and responses:

- **GET/ GET-RESP**: the GET message implies a request for an NDO. If a NetInf node has a copy of the requested NDO, it replies with a GET-RESP message;

[7] See http://www.sail-project.eu/

- **PUBLISH/ PUBLISH-RESP**: a user send a PUBLISH message to register its NDO to the network and receive a PUBLISH-RESP message with a status code in return;
- **SEARCH/ SEARCH-RESP**: since NetInf names are flat-ish and not human-readable, users can send a SEARCH message to look up object names which are related to particular keywords. The SEARCH-RESP returns names of NDOs that is considered to match the query keywords.

Name Resolution and Data Routing

Name resolution and data routing in NetInf can be coupled, decoupled, or even hybrid and are illustrated in Figure 1.4.

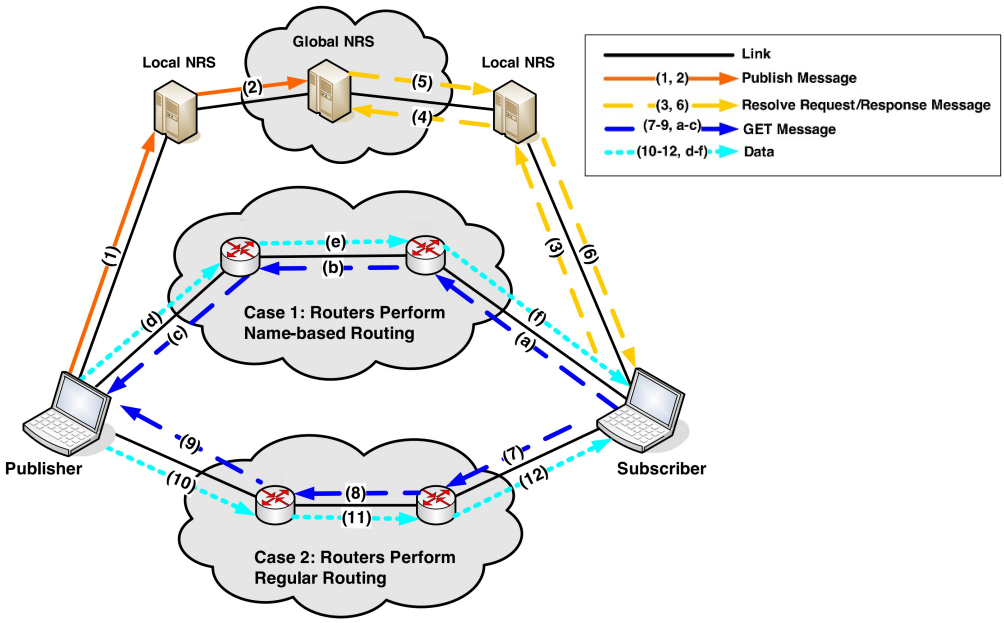


FIGURE 1.4: Name resolution and data routing of NetInf [2]

In the decoupled case, the *Name Resolution Service* (NRS) is responsible for mapping the requested name to locators that can be used to retrieve the corresponding content. The NRS is organized in multiple levels, i.e., the local NRS will handle different names of the same authority, while the global NRS will deal with the `<Authority>` component. NetInf offers multiple options to interconnect separate local NRSes into a global NRS infrastructure, including the *Multi-level Distributed Hash Table* (MDHT) system [26] and the *Hierarchical SkipNet* (HSkip) system [27]. An NDO is advertised to a local NRS by using the PUBLISH message (arrow 1) including the content name, routing hints (i.e., supporting information to get the content), and some metadata. Afterward, the local NRS will aggregate all the publications for the

same authority and sends a PUBLISH message to the global NRS (arrow 2). To retrieve an NDO, the user sends a GET message to its local NRS (arrow 3). The local NRS may consult the global NRS to resolve the authority part if necessary (arrow 4). As a result, the user will receive a locator (arrow 5-6) to the requested content. The user then sends a GET message to the publisher using the obtained locator (arrows a-c), and the publisher will reply with the NDO (arrows d-f).

In the coupled case, the NetInf routers will perform the name-based routing for GET messages. A routing protocol is used to propagate content names and to update the routing tables of routers. When a user sends a GET message, it is forwarded hop-by-hop to a content source, following the routing tables of routers (arrows 7-9). While the GET is being forwarded, it also accumulates routing directions along the path. As such, when a copy is found, the publisher or cache can reverse the accumulated routing directions so that the GET-RESP can reach the user (arrows 10-12).

In a heterogeneous network, it is often not possible to resolve the content name into a locator that is meaningful to the user. In such case, NetInf will operate in hybrid mode, i.e., the NRS will return routing hints when receiving a GET message from a user. A routing hint is a partial locator where more information about the requested content can be found. There are three operational modes for routing hint selection. In the first mode, the NRS returns all of its available routing hints for the requested name and leaves the decision to the user. In the second mode, NRS decides which routing hints will be returned to the user based on its knowledge about the network. In the third mode, the NRS returns a prioritized list of routing hints and leaves the final selection to the user. Eventually, the GET message will arrive at a publisher or a cached copy.

Caching and Mobility Support

For efficient operation in challenging network conditions, NetInf supports three types of caching for different scenarios. The first is on-path caching, i.e., the router caches NDOs in each GET-RESP traversing. Secondly, NetInf supports off-path caching, i.e., a dedicated cache is placed close to one or more NetInf NRS nodes and is treated as a publisher. The NRS can tell the off-path cache which objects to cache based on the content popularity. Finally, NetInf also supports peer caching in NetInf nodes, and on user devices. Peer caches can be on-path or off-path caches. They are beneficial in challenging network conditions, e.g., high mobility network. Moreover, NetInf also considers a hierarchy of caches in which caching servers are located at the root, and local caches are part of the tree. Popular objects can be dynamically migrated to caches that are closer to the user by cache migration policies.

NetInf inherently makes requester mobility simpler since a request for an NDO is location-independent. Therefore, the corresponding object can be delivered from different sources no matter where requesters are located. The request is responded merely by the copies close to its new location. For publisher mobility, routing information and name resolution information need to be updated to the NRS of the new location. The NRS also utilizes *Late Name Binding* (LNB), i.e., the name resolution process may end at a node close to the current are of the moving publisher. As such, when a publisher moves within its current area, it only needs to update its local NRS without changing the routing hints in the global NRS.

Intrinsic Security Features

As described in Section 1.4.3, NetInf naming scheme provides data authority and name-data integrity. For static data, the integrity is checked by verifying if the hash of the data contained in the NDO matches the `<Digest_Value>` part of its name. For dynamic data, the `<Digest_Value>` is the hash of a public key. Only the data signed by the owner of the corresponding private key can be decrypted by this public key. NetInf also supports owner pseudonymity and owner identification. Owner pseudonymity indicates that multiple NDOs belong to the same origin. Users, who trust in an owner pseudonymity, also trust in subsequent content published under it. Owner identification can bind the pseudonym to the owner's real-world identity. This option, however, requires the use of standard *Public Key Infrastructure* (PKI) mechanisms which are not part of the core NetInf architecture. Although these intrinsic security features can protect ICN architectures from issues inherent in IP, they cannot entirely prevent new security threats that are introduced by novel features and components in ICN. Thus, ICN security still requires further investigation.

1.5 Comparison Between Information-Centric Networking Architectures

In this section, we compare ICN proposals mentioned above to summarize and highlight their differences according to important aspects of presented ICN architecture, including naming scheme, name resolution, data routing, caching and security. Between when implementing ICN's key concepts. The result is presented in Table 1.1.

TABLE 1.1: Comparison between ICN architectures (modified from [2])

	Naming scheme	**Name resolution and data routing**	**Caching**	**Intrinsic Security**
NDN	Hierarchical name can be presented in URL form. NDN name can be human-readable and may contain a publisher-specific prefix.	Coupled: NDN routers perform name-based routing for *Interest* and keep states of forwarded *Interest* in the PIT to forward *Data* following the reverse-path.	Support on-path caching: CS in NDN router caches solicited *Data*.	*SignatureInfo* and *SignatureValue* fields in *Data* enable the verification for data authenticity and integrity. Certification chain can follow name hierarchy.
DONA	Flat name have the form of (**P** : **L**). **P** is the hash of principal's public key. **L** uniquely identifies an NDO of the principal.	Decoupled: RHs are responsible for resolving names to IP addresses of the publisher. RHs are organized following AS hierarchy.	Can support on-path caching at RH: RH alters the source IP address of FIND before forwarding so that the NDO is forced to traverse the RH.	REGISTER is challenged before being accepted and is authenticated when being sent between RHs. Users have access to the signature, metadata, and hash of principal's public key to check the data integrity and authenticity.
PURSUIT	Flat name includes a sequence of SIds and a RId. Scopes can be nested, forming a flexible structure and can be organized hierarchically.	Decoupled: RENE matches subscriptions to publications while a TM node selects the path from publisher to subscriber and FNs execute the data delivery.	Can support off-path caching: caches operate and advertise their NDOs as publishers. On-path caching is ineffective due to the separation of name resolution and data routing.	Support packet-level authentication, enabling the verification of data integrity and confidentiality. Flat names are self-certifying. Dynamic FIds can be used to prevent bloom filters re-usage to launch attacks.
NetInf	Flat-ish name with the ni URI scheme. Name components are location-independent and human-unreadable, but also has some structure that allows name aggregation.	(1) Coupled: GET gathers states while being forwarded to the publisher; GET-RESP can reverse the path to reach the subscriber. (2) Decoupled: NRS resolves names to content locators. (3) Hybrid: NRS answers GET with routing hints to lead the subscriber closer to the publisher.	On-path caching: routers cache NDOs in GET-RESP. Off-path caching: dedicated caches are placed close to NRS nodes and operate as publishers.	`<Digest_Value>` components provides data authority and name-data integrity. Can support owner pseudonymity and owner identification.

Naming scheme: the naming scheme is one of the fundamental design choices in each ICN architecture since the name's structure and semantics have a profound impact on all other aspects of the architecture. Most of described ICN architectures use flat names. The advantage of flat name is that it is location-independent and allows self-certification of content and authenticity. However, flat names are hard for aggregating and storing, as well as not user-friendly. A search engine is essential so that users can look for actual content flat names from user-friendly names. On the other hand, the hierarchical name is more user-friendly and easy to aggregate. Also, it usually expresses some semantic binding to the data, thus, reveal information related to the data.

Name resolution and data routing: decoupled model of name resolution and data routing appears in most of presented ICN architectures. Such a model merely changes the name resolution, hence can co-operate with existent network infrastructures and use them for data routing. However, managing an enormous number of content names and keeping name-routing information updated are real challenges, especially for flat names which is hard to be aggregated. On the other hand, the coupled model eliminates the name resolution step, thereby potentially reducing the overall latency and simplifying the whole process.

Caching: on-path caching is used more in the presented ICN proposals since it requires less additional registrations and routing information. Nevertheless, it is advocated that the benefits from the extensive use of caching in ICN are not substantial [28,29]. When caching takes place inside the network, several types of traffic will compete for the same caching space. Cache space management, therefore, becomes crucial for the network. Also, pervasive caching might also present security threats [30].

Intrinsic security: all the presented ICN proposals can verify data integrity and data authenticity. Caching also enhances data availability. Nevertheless, ICN proposals also present new issues. Most ICN approaches rely on cryptographic keys and trusted entities for information-name verification. Besides, by shifting from host-centric to information-centric communication model, ICN also shift the target of attack from host to NDO. Also, since ICN uses in-network caches, cached NDOs will offer attackers many sources to get desired NDOs. They also have more time to extract the information they want since data are not only stored in the providers, but also exist in caches.

1.6 Information-Centric Networking Deployment Status

After years of research and experimentation, ICN's potential has been acknowledged eventually by the research community. It has achieved a certain level of technological maturity, enabling many ongoing research effort in the ICN community, such as caching [10], routing [31], forwarding [32, 33], energy-efficiency [34], name service [35], and *Internet of Things* (IoT) [19, 36–38], and especially the consideration for ICN deployment in reality. To help picture the activeness in this aspect, this section briefly introduces possible deployment options of ICN, along with several ICN deployment efforts from different projects across all over the world.

Over the course of ICN development, multiple deployment efforts have been conducted by international collaborative projects across all over the world. The community pays significant attention to deployment efforts that can provide Internet access in a live environment. The ICN Research Group (ICNRG), chartered by the Internet Engineering Task Force (IETF), proposes to divide ICN deployment efforts into four options, namely *clean-slate ICN*, *ICN-as-a-Slice*, *ICN-as-an-Overlay*, *ICN-as-an-Underlay* [39]. The first option's goal is to renew or replace existing IP routers with ICN-specific counterparts. However, this approach is no longer considered viable choice and thus, has never been trialed.

The second option is related to the concept of Network slicing [40] that allows the network operator to provide dedicated virtual networks with functionality specific to the service or customer over a common network infrastructure with Software-Defined Networking (SDN) / Network Function Virtualization (NFV) techniques. Although this approach is an attractive option for future 5G systems, it still has not be experimented yet because 5G standards are still not set until the mid-2018 timeframe [39].

Majority of existing ICN trial systems belong to either of the last two deployment options, i.e., ICN-as-an-Overlay, ICN-as-an-Underlay. The ICN-as-an-Overlay approach deploys ICN in islands on top of the existing IP-based infrastructure by encapsulating ICN packets inside of IP packets. A strength of this approach is that it utilize the existing IP network, thus allows rapid and probably the easiest deployment of ICN for experimentation and testing. The NDN (Section 1.3) testbed is deployed as an overlay on the public Internet to connect 34 institutions across several continents[8] with their forwarding daemon (NFD). Experimentation using NDN testbed includes real time video-conferencing[41], geo-locating and interfacing to

[8]See: https://named-data.net/ndn-testbed/

consumer applications. In another effort, ICN2020[9] project presents in their deliverable D4.1 progress made in both local and federated testbeds. Especially, their federated testbed is planned to integrate the NDN testbed, the CUTEi testbed [42] and the GEANT testbed[10] to create an overlay deployment of ICN. The FP7 PURSUIT[11] deploys the PSIRP proposal (Section 1.4.2) in its testbed as an Layer 2 VPN-based overlay between several European, Us an Asian sites [43]. The testbed can be utilized for ICN message exchange and video transmissions.

The ICN-as-an-Underlay approach uses an ICN underlay that would integrate with existing IP-based networks by deploying application layer gateways at appropriate locations [39]. Therefore, given islands, e.g., Content Distribution Network (CDN) or Internet of Thing (IoT), can obtain profits from native ICN such as multicast delivery, mobility support, location independence. CableLabs content delivery system, NDN IoT are trials using ICN-as-an-Underlay. In the CableLabs efforts [44], ICN is deployed as an underlay such that CDN server farms can benefit from ICN in-network caching. This ICN-based CDN server farm is then used to answer standard HTTP/IP-based content retrieval request from the general Internet. Although results are not provided, the trial acknowledges the difficulty of completely replacing existing HTTP/IP end user application and related Web infrastructure. In [45], the authors present an experiment with NDN in a wireless IoT scenario consisting of 60 nodes over several buildings in a university campus. The NDN protocol is adapted to run over 6LoWPAN wireless link layers. The NDN-based IoT system demonstrates its benefits of energy consumption, RAM and ROM footprints in comparison with the standard IP-based IoT.

Although it does not correspond to any deployment options proposed in [39], Cisco Hybrid ICN[12][46] is a notable approach that encodes ICN names into IP addresses, thus enabling the deployment of ICN within IP while maintaining all features of ICN communication. ICN information will be carried as payload inside the IP packet[13]. Therefore, standard IP routers can forward packets using IP information, while ICN-aware routers can also understand ICN information, allowing cohabitation of ICN with legacy IP traffic. Cisco has considered ICN as an emerging innovative technology and intents to adopt it for their 5G networks. A related open source effort was kicked off in 2017[14]. The trials intend to show the routing performance efficiency of the Hybrid ICN router over existing IP routers, but results have not been published.

[9] See: http://www.icn2020.org/
[10] See: https://www.geant.org/
[11] See: http://www.fp7-pursuit.eu/PursuitWeb/
[12] See: Cisco, Mobile Video Delivery with Hybrid ICN: IP-integrated ICN Solution for 5G, 2017
[13] See: Cisco, Hybrid ICN: Cisco Announces Important Steps toward Adoption of Information-Centric Networking, 2017.
[14] See: https://wiki.fd.io/view/Cicn

1.7 Conclusion

This chapter provided a state of the art on ICN to familiarize its fundamental concepts and proposals. We motivated the ICN development by indicating changes of the Internet and its usage. We then presented four ICN key features, including naming scheme, name resolution-data routing, caching and information security and highlighted their differences from the current IP network's. We mainly focused on details of NDN protocol – the most promising ICN architecture in the literature. Besides, we introduced other ICN proposals, including DONA, PURSUIT and NetInf, as references to demonstrate various designs of the ICN's key concepts. We summarized the presented ICN architectures with a comparison table regarding the ICN key features mentioned above. As real deployment is an important aspect drawing the attention from the ICN community, we briefly introduced possible ICN deployment options, as well as ICN deployment efforts from different projects across all over the world. Despite the activeness in deployment efforts, ICN and especially NDN will hardly be adopted in ISP operating infrastructure without considering its security. Therefore, in the next chapter, we will investigate the security of NDN in more detail.

Chapter 2

Named Data Networking Security Attacks - State of the Art

Contents

As stated in the previous chapter, despite deployment efforts, NDN can hardly be adopted by ISP without considering its security. Before addressing a solution for NDN security, one must be aware of its current status, discovered threats and existing works on related topics. Therefore, this chapter will provide a state of the art on NDN security. At first, we propose a taxonomy to classify NDN attacks and to organize the presentation in this chapter. Then, for each type of attack, we introduce its principle, provide a survey of its related works and identify major limitations. At the end of the chapter, we evaluate presented attacks accordingly to selected attributes to summarize their severity. Based on such evaluation, we select a set of crucial NDN attacks that need be prioritized when further considering NDN security.

2.1 Existing Security Surveys and Proposed Taxonomy

Since NDN was proposed, many security issues have been discovered and addressed. Several surveys on NDN security are available, and each of them categorizes NDN security attacks according to different criteria. In [47], the authors classify NDN security attacks based on the mechanism leveraged to carry them out, e.g., flooding, forced computation, cache manipulation. Countermeasures on DoS attacks in NDN are also presented and grouped according to counter-measures mechanisms. The survey, however, only focuses on DoS-type attacks. In [48], the authors propose a taxonomy for ICN security attacks based on the features exploited, e.g., naming, routing, caching. Nevertheless, attacks' principles are presented generically, and countermeasures are also missing in this survey. In [49], the authors cover a wider range of aspects when addressing not only security attacks but also privacy and access control in ICN. The survey also provides taxonomies for classifying related works in each topic covered.

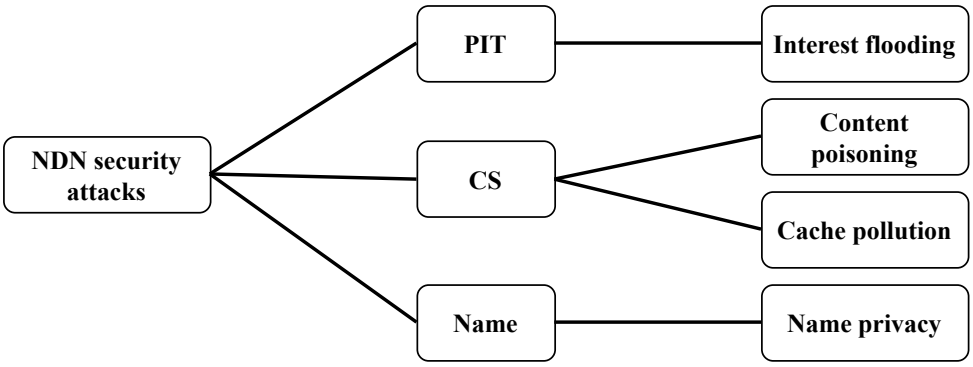


FIGURE 2.1: Proposed taxonomy for NDN security attacks

Having considered the pros and cons of the existing survey on NDN security, we come up with a taxonomy for NDN security attacks as illustrated in Figure 2.1. Our taxonomy divides NDN attacks into three categories, based on the mainly exploited NDN component: (1) *PIT* attacks, (2) *CS* attacks and (3) *Name* attacks. One should note that, although FIB is also an NDN component, we exclude it from our taxonomy. We argue that this component belongs to the control plane which is hardly be compromised in practice, especially when NDN has a secure routing protocol [3, 50]. Thus, we are more interested in the security of components related to data plane where end-users involve and thus, can cause more impacts to the network. In addition, we deliberately decide not to consider the access control, as the authors of [49], since the majority of previous works on this topic involves encryption and key management, which does not align with our main interest of data plane security. Furthermore, we aim at addressing security threats in NDN with a statistical method, hence, collecting data is essential. Therefore, we focus on attacks that

are introduced with a detail description and demonstrated with either simulations or experiments in the literature. For each of them, we will (1) explain the attack principle and (2) introduce remarkable previous works. More details can be found in existing security surveys [47–49].

Related work: Named-data Link State Routing Protocol

As stated in the previous section, the FIB is hardly compromised in practice due to the existence of the NDN secure routing protocol. In this section, we briefly introduce Named Data Link State routing protocol (NLSR) [3, 50] - a link-state routing protocol that authenticates *Link State Announcement* (LSA) exchanged between NDN routers. The trust in NLSR is organized according to a hierarchy with five levels: root, site, operator, router and NLSR process. Each level's key is certificated by the higher level, i.e., NLSR process's key is certified by router's key and so on, as illustrated in Figure 2.2. The trust model is rooted at the network administrator, who owns the root key. The *root key packet* is signed by itself. Moreover, the last component of key names is always the hash of the key itself to ensure the binding between the key and the key name. These key packets are distributed through key repositories so that a router can fetch the necessary keys for the verification process.

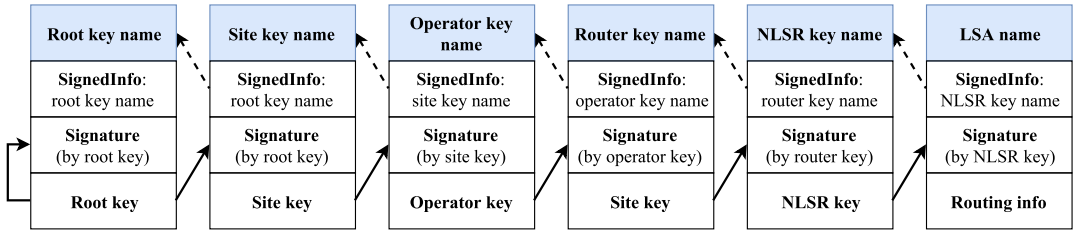


FIGURE 2.2: NLSR's signing and verification chain (modified from [3])

LSA is considered like a *Data* packet with *Name* and *Signature* fields. As such, a router must send an *Interest* to collect routing info. When an LSA is generated, it will be signed by the NLSR process key whose name is inserted in *SignedInfo* field of the LSA. Before processing an LSA, an NDN router must first verify the packet by retrieving the NLSR key from the key repository thanks to the *SignedInfo* field. The router then checks the origin of the NLSR by requesting the router key indicated in *SignedInfo* field of the NLSR key packet, and so on until reaching the root key. If at any step key fetching is unsuccessful, or the signature is invalid, the LSA is considered illegitimate. Although the verification process passes through many steps, a key's legitimacy will be recorded once it is already verified successfully. Hence, if the router encounters a recorded key name in the *SignedInfo* field, it only needs to verify the signature and does not have to recheck the key's origin.

2.2 Pending Interest Table Attacks

In an NDN router, the PIT keeps states of *Interests* that the router already forwarded. Those PIT states are used later to reverse-path forward *Data* packets. Attacks in this category leverage this vital router component to cause damage. A crucial attack of this type is the *Interest Flooding Attack* (IFA) – a variation of DoS attack in NDN.

2.2.1 Interest Flooding Attack

In NDN, *Interest* is used to express users' requests while *Data* packet contains an NDO (see Section 1.3.1). However, because NDN router does not accept unsolicited *Data* (see Section 1.3.2), sending *Data* without precedent *Interest* hardly causes any harm to the router. On the other hand, the attacker can flood both the PIT and content providers by sending a large number of *Interests* in a short period, thus preventing them from handling legitimate requests. Hence, it is called "interest flooding." It is identified by the NDN community as one of the most significant security attacks[1].

Based on the type of content requested, we can identify three types of *Interest* flooding attacks [51]: (1) existing (static), (2) dynamic and (3) non-existent content. Since NDN aggregates *Interest* for the same content, it is obvious that these attacks must issue *Interests* for different content names. The first type requires knowledge about existing contents in the network. However, this attack is much less efficient since in-network caches, and NDN's forwarding mechanism already provides a built-in countermeasure against it [11,51]. After an initial wave of *Interest* flooding, copies of the requested content will be stored in caches. Further *Interests* for the same static contents will be satisfied by cached copies, without being forwarded further.

In the second case, the dynamic content is generated by the provider on demand. As a result, caching dynamic content is not useful, and *Interests* for dynamic content are usually forwarded to the provider. Such a case puts more load on the content provider since the dynamic content can hardly be pre-computed. Furthermore, generating dynamic contents might be expensive, the content provider may get overloaded due to significant signature computations. Besides, sending a large number of *Interests* also consumes bandwidth and router storage.

In the third scenario, benefits of caching are lost since attacker continuously sends *Interests* for different non-existent names. These names can be entirely forged by the attacker or composed of a valid prefix and a random suffix [52]. In the latter case, *Interests* are routed to specific content providers, following the longest-prefix matching in FIB since the prefix exists. Although content providers can ignore such

[1] See NDN Frequently Asked Questions (FAQ).

Interests, they still take up space in routers' PIT until expiration, preventing legitimate *Interests* from being processed. Compared to two other types of IFA, this type is more dangerous because it impacts not only content providers but also the network infrastructure through router's PIT. Also, it requires no knowledge of existing content and exploits forged content names, thus can be launched with ease.

Previous works on IFA can be divided based on their approaches into two categories, namely *detection - mitigation* or *protocol enhancement*. The first approach proposes a separate solution that can detect or mitigates IFA, while the second approach suggests modifying some features in the existing NDN protocol.

Detection - Mitigation Approach

Previous works in this category can be divided according to their granularity into three sub-category, namely *per-face* [53–55], *per-prefix* [56–59] and *hybrid* [1, 60, 61]. The per-face approach addresses the router's faces that are under attack. Meanwhile, the per-prefix approach targets the content prefix used to launch IFA. Solutions of the hybrid approach can tackle IFA on both the router's face and the malicious prefixes of *Interests*.

Per-face Approach Based on the *token bucket* algorithm in packet-switched networks, Afanasyev et al. [53] propose three countermeasures against IFA. Among them, *satisfaction-based push back* is the most effective method. This approach leverages two NDN-specific features: (1) one *Interest* is satisfied by at most one *Data* and (2) *Data* is forwarded on the reverse path of the *Interest*. The router monitors the number of *Data* received and forwarded *Interests*, respectively denoted as D_i and I_i, as well as maintains an up-to-date value of the *Interest* satisfaction ratio (D_i / I_i) for each router's face i to determine which face is under attack. The router also performs a push back periodically, i.e., announce an *Interest limit* to the downstream node on the malicious face. The announced limit on a given face is calculated based on the face's satisfaction ratio and the limit received from the upstream. When receiving announcements from the neighbors, the router will change the *Interest limit* of its faces. However, the authors state that, because a router can have several upstream faces, it may not be entirely clear which limit from which upstream should be used to calculate an announed limit for a given downstream. In their actual implementation, they modify the FIB to overcome this problem. The authors, however, does not describe clearly how the database of FIB will change to adapt their countermeasure.

Compagno et al. propose *Poseidon* [54] – a detection and pushback countermeasure based on periodical calculation of ρ_i and ω_i for each router face i. The former is the number of bytes used to store, in the PIT, *Interests* coming from for face i. The

former represents the reciprocal satisfaction ratio, i.e., the ratio between the number of incoming *Interests* I_i and the number of outgoing *Data* D_i given a face i. Their corresponding threshold Ω_i and P_i are not constant and can change over time in order to adapt to different conditions of the network. An attack is detected when both ω_i and ρ_i exceed their thresholds[2]. Once an attack is detected, the router restricts the *Interest* sending rate of the involved face and issues an alert message in a *Data*. Since *Data* packet can be signed, receiving router can verify the validity of the alert message. When a router receives an alert message and the last alert message already expires[3], it will decrease their thresholds Ω_i and P_i by a factor s. As the message moving towards the attack sources, the statistics ρ_i and ω_i will get smaller due to the weak footprint. Therefore, decreasing the thresholds helps the router distinguish better the malicious face from others. As such, the attack is pushed back eventually toward its source. If no attack is detected for a predefined amount of time, the thresholds are restored to their initial values. Poseidon's approach has simpler computation than [53]. However, the authors do not indicate how they choose the values for s and the initial values for Ω_i and P_i.

Per-prefix Approach Dai et al. [56] exploit PIT entries to trace back malicious *Interests* to its source. This idea is not feasible in IP network since an IP router does not keep states in its memory when it forwards a packet. When the PIT size exceeds a pre-defined threshold, the router triggers the *Interest* traceback. In this countermeasure, the router under attack generates spoofed *Data* to resolve *Interests* that have not been satisfied for a long time. By looking up the PIT, spoofed *Data* will be forwarded back to the previous node, and eventually arrives at the source of malicious *Interests*. When the spoofed *Data* packet arrives at the edge router directly connected to the attacker, the router will limit the sending rate of this face, mitigating the DoS attack. Since rate limiting only occurs at the edge router, the proposal may be ineffective if the edge router is compromised. Besides, the authors do not clarify how to identify *Interests* that should be traced back and how to choose the PIT size's threshold.

Inheriting the concept of proof-of-work [62], Li et al. propose *Interest Cash* [57] – an application-based solution against IFA for dynamically generated content. The purpose is to increase the user's computational cost to issue an *Interest*. To send some *Interests*, the user must first request and retrieve from the desired content provider a meta-puzzle composing of pieces of information so that the user can generate and solve the actual puzzle, which is a usual cryptography-related problem. The solution is considered as the "cash" to send *Interest* and will be appended to the *Interest*'s

[2]Note that ω_i is the reciprocal satisfaction ratio. Thus when IFA occurs, this value tends to increase.

[3]The latter is to avoid the case when several alert messages arrive at the router at the same time. In such case, the router should not decrease its thresholds several times consequently.

content name. The content provider can verify the solution without too much computational cost (since it already knows the puzzle's solution) and will send *Data* if the cash is valid. A meta-puzzle can be saved on user's storage so that it can be used to find the cash for later *Data*. The meta-puzzle will be updated in two cases either after the expiration time or when the content provider receives too many or too few valid requests. The mechanism will not cause significant impact to a legitimate user. However, for the attacker who sends a large number of *Interests*, it will suffer a heavy burden. The attack can still be distributed to many hosts, but its severity is still mitigated greatly since the total number of requests is limited.

You et al. [58] propose a method based on entropy [63] to detect IFA. The router will aggregate PIT entries into groups, according to the corresponding name prefixes, and calculate the entropy for each group. If there is no IFA, a group of PIT entries should have a stable entropy. Otherwise, the entropy and the PIT usage will increase. The number of prefix levels used to aggregate PIT entries is proposed to change dynamically so that the attacker cannot predict the threshold of entropy value. When an attack is detected, and the PIT usage varies from $75 - 95\%$, the router decreases the lifetime of PIT entries belonging to the group with the highest entropy value. When the PIT usage is even higher, PIT entries of the highest entropy group will be dropped.

Hybrid Approach Wang et al. [60] propose a countermeasure based on fuzzy-logic and a pushback cooperation of routers. The router periodically monitors two metrics for detection: (1) *PIT Occupancy Rate* (POR) and (2) *PIT Expiration Rate* (PER). The former represents the ratio between the current number and the maximum number of PIT entries, while the latter means the ratio between the instantaneous number of expired PIT entries and the number of pending PIT entries. Real-time POR and PER values are inputted to fuzzy inference rules to decide if they are abnormal. Upstream routers are more likely to detect the attack since they aggregate more traffic than the edge routers, especially when the attack is distributed. If an attack is detected, the mitigation is triggered. The router will listen to alert or pushback messages. The alert message raises the attention for a router, telling it to start identifying malicious prefixes (characterized by a high number of expired PIT entries) and malicious faces (which have the most incoming *Interests* with the identified malicious prefixes). If a malicious prefix is identified, this prefix will be notified to other routers on malicious faces by a pushback message. When a pushback message arrives an edge router, it limits the incoming rate of all *Interest* packets for the malicious prefix. However, it may require a substantial amount of training data to get good fuzzy rules.

In [1], Tang et al. propose a two-phase approach that first determines the face

under attack and then identifies the malicious prefixes on the involved face. In the first phase, the router calculates periodically the Exponentially Weighted Moving Average (EWMA) number of incoming *Interests* $\hat{I}_i$ and outgoing *Data* $\hat{D}_i$ of a given face i, as well as the *Relative Strength Index*[4] $RSI_i = \hat{I}_i/(\hat{I}_i + \hat{D}_i)$. Usually, RSI should be about 50%, implying that one *Interest* should have one *Data* returned. If RSI overcomes a pre-defined threshold, the next phase should be triggered for the involved face. The router records incoming *Interests* and expired *Interests* on the reported face for each *Interest*'s prefix and calculate the expired ratio. If the expired ratio of a name prefix exceeds a threshold, the prefix is considered abnormal. The router can limit propagation for *Interests* requesting the abnormal prefix while maintaining the forwarding for other prefixes of *Interests* on the involved face.

Protocol Enhancement Approach

In [64, 65], Yi et al. propose to use *Negative ACKnowledgment* (NACK) packet to help routers take the initiative when there is a problem with *Interest* forwarding. When a PIT entry expires, or a router cannot forward an *Interest*, a *NACK* is sent to the downstream node. This packet carries the content name of original *Interest*, plus an error code. There are three types of error codes: (1) *Duplicate*; (2) *Congestion* or (3) *NoData*. *Duplicate* error code indicates that the upstream router has received and forwarded a duplicated *Interest* (i.e., having the same **Name** and **Nonce** fields as ones recorded in the corresponding PIT entry, see Section 1.3.1) and still waits for Data. The *Congestion* code indicates that the *Interest* cannot be forwarded due to the congestion on the outgoing link. Finally, *NoData* code informs the downstream node that the router cannot retrieve any *Data* to satisfy the *Interest* packet for some reason (e.g. find no path in FIB or PIT entry times out). In the second and the third cases, the sending router also removes the corresponding PIT entry to release its resources. When a router receives a *NACK* packet with *Congestion* or *No Data* code, it can try forwarding the *Interest* to other available upstream links if possible, as long as the corresponding PIT entry is still valid. In case there are no links available, another *NACK* will be issued to the next downstream nodes.

Remarks on Interest Flooding Attack

The majority of existing solutions on IFA requires routers to maintain some statistics to detect either the faces or the prefixes under attack. However, the authors do not clearly state how they choose the detection threshold. A poorly defined threshold may lead to rigid and unreliable detection, resulting in lots of false alarms that

[4] In fact, RSI and satisfaction ratio are equivalent to each other because they are all ratios calculated from I_i, D_i, U_i and $I_i = D_i + U_i$ for a given face i and a given measurement interval.

waste network resources or accidentally penalize legitimate client. Moreover, many of proposed solutions leverage routers' collaboration to mitigate IFA, making them dependent on each other. Once a router is compromised, it can disrupt other routers by sending false announcements. Furthermore, most of the previous works evaluate their proposals with a simulated environment, thus leaving their performance in the real deployment unknown.

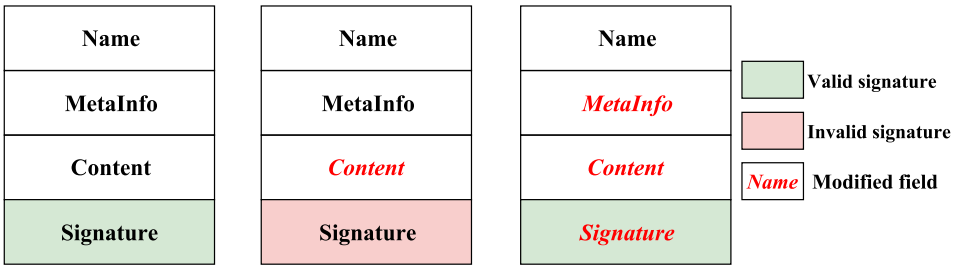
2.3 Content Store Attacks

Caching is an important feature in NDN. Ubiquitous in-network caches are deployed through routers' CS to increase the availability of popular contents in the network. As a result, *Interests* can be resolved by cached copies on the path, hence improving the delivery performance. Attacks in this class disrupt the advantage of NDN caching system. *Content Poisoning Attack* (CPA) and *cache pollution attack* are two typical attacks in this category.

2.3.1 Content Poisoning Attack

In CPA, an attacker will inject malicious *Data* into router's cache, in order to resolve legitimate *Interests*. Such attack leverages in-network caches in NDN to spread malicious *Data* to as many users as possible. To increase the effect scale, the attacker would forge fake content for a popular content name (which can be figured out by other NDN security attacks, e.g., cache snooping). To preserve the longevity of the attack, the attacker can set the freshness field of fake content to the maximum value. Malicious *Data* still carries a valid name, but its content was modified. Differences between malicious *Data* from the legitimate one are illustrated in Figure 2.3. There are two types of malicious *Data* [51]:

- **Corrupted *Data***: the attacker simply corrupts the *Content* field without further modifications on other fields. Since the *Data* is modified, its signature becomes invalid. This type of malicious *Data* can be easily created. Since users and network entities are not enforced to verify the *Data* signature, this type of bad *Data* mostly affects naive ones who neglect the signature verification;
- **Fake *Data***: the malicious provider not only alter the *Content* field but also has the valid key to make a valid signature. Information to retrieve the malicious provider's key is attached in *MetaInfo* field. Although it is impossible for user and network entity to recognize such a malicious *Data*, this kind of bad *Data* requires the attacker to have a valid key for the name prefix it publishes.



(A) Legitimate *Data* (B) Corrupted *Data* (C) Fake *Data*

FIGURE 2.3: (A) Legitimate *Data* and two types of malicious *Data*: (B) corrupted and (C) fake *Data*

To date, proposed solutions to detect and mitigate CPA are restricted in number and can be divided into three groups, namely (1) *Interest enhancement*, (2) *verification improvement*, and (3) *feedback-based*.

Interest Enhancement Approach

Approaches in this group propose to include additional information in the *Interest* to enhance the binding between a content object and its name. In [51], Gasti et al. were the first to discuss the content poisoning attack. The authors propose to use *Self-Certifying Interests/Content* (SCIC), i.e., the user includes the hash of the desired content in the *Interest*. This strongly binds the content name with the corresponding object. Upon receiving a *Data*, a router can compare its hash to one from the *Interest* to verify the *Data*. However, this approach requires the client to obtain the expected content's hash beforehand (e.g., through a mapping service), thus increasing the retrieval latency. Besides, retrieving wrong hash also lead to incorrect content. The approach is more suitable to static content since it is impractical to compute hashes for dynamically generated content beforehand.

In [66], the authors point out that CPA is partly caused by the *Interest* ambiguity, i.e., the only *Name* and *Nonce* fields in *Interest* are mandatory while other fields (e.g., *Selectors*) are optional (Section 1.3.1). Therefore, the authors propose to clarify this ambiguity by an *Interest-Key Binding* (IKB) rule, i.e., *Interest* has to specify the key digest of the desired publisher and the *Data* is required to include the provider's public key in its *KeyLocator* field. As such, the content now binds with the publisher's public key. Routers only forward *Data* packets if the key digest of pending *Interest* matches the *Data*'s public key digest. Nevertheless, the proposal requires that users obtain the provider's public key in advance. Besides, routers still have to retrieve the key and compute the hash to verify all *Data* packets, which is impractical at line speed. Moreover, routers only verify key, not the content. Therefore content can still be corrupted.

Verification Improvement Approach

Routers can mitigate CPA by verifying *Data* before forwarding. However, in reality, such a straightforward solution is impractical due to the expensive computation cost at line speed [51, 67]. Therefore, the main goal of the second approach is to improve the verification performance on routers, by changing router's verification routine while maintaining its resilience against CPA. Inspired by *check before storing* proposal in [68], Kim et al. propose in [69], that a router caches all the *Data* it forwards, but only verifies their signatures when there is a cache-hit. Successfully verified *Data* are forwarded without further verification. Otherwise, *Data* packets are removed from the cache. As such, it significantly reduces the routers' verification load. Moreover, *Data* with cache-hits is more likely popular. Therefore, it still maintains verification for popular content. However, this solution only solves the problem locally. Malicious clients can still re-issue other malicious *Data* to insert them in a cache or increase the verification load on the router by sending *Interests* to create false cache-hit.

In [51], Gasti et al. also propose that the verification load can be distributed evenly among routers in the same organization according to a defined rule. The router can also verify *Data* probabilistically as proposed in [70]. When a router finds out a corrupted *Data*, it drops the *Data* and sends a warning message to its neighbors. However, such an approach requires trust between routers. Once a router is compromised, it can send a false warning to mislead other routers. Besides, if the rule distributing the verification load is revealed, the attacker can generate *Data* to enforce verification on a specific router.

Feedback-based Approach

When a user receives a *Data*, he usually verifies its signature. As such, the user now also shares the verification load. If the signature is invalid or *Data* is undesired, he can indicate that *Data* in the *Exclude* field of the *Interest* (Section 1.3.1) to avoid it. Solutions in this group exploit this *Exclude* field to mitigate CPA. In [71], DiBenedetto et al. propose a forwarding strategy driven by user's feedback for NDN. After verifying malicious *Data*, users will send a *Report* packet to the upstream node under a reserved namespace. This namespace is only valid between each pair of nodes. The malicious *Data* and its signing key are also included in the *Report* so that the upstream router can verify the *Data* again. If the *Data* is truly poisoned, the router will drive path to another provider, in order to avoid the source of malicious content. Nodes that send a wrong *Report* will be blocked.

In [67], the authors propose a *content ranking* solution, i.e., routers ranks its

cached *Data* based on users' exclusions. The proposed ranking function considers three factors for a cached *Data*: (1) number of exclusions, (2) exclusion recentness (the newer exclusion, the more it weights) and (3) number of exclusion's incoming faces. The ranking value varies from 0 to 1. Contents with higher ranking value will be more prioritized in caching and more likely to be returned to users. This solution requires no change to NDN architecture. Similarly, the authors in [72] also rank a content based on exclusions from users. Besides, their ranking function also considers the content popularity and the credibility of the nodes that send the content. However, a typical drawback of solutions in this group is that they rely on feedback from users which are more likely to be compromised. Malicious users can either exclude legitimate *Data* or provide, on purpose, incorrect verification information to sabotage.

Remarks on Content Poisoning Attack

Besides their weaknesses, previous works on CPA share some common drawbacks. First is the inconsistency in CPA's impact evaluation. Since their authors usually couple such an evaluation with their proposed solutions, the understanding of this phenomenon is partial and biased towards emphasizing the proposed solutions. Therefore, the results are neither re-usable nor comparable. Secondly, simulation scenarios rely on a one-shot attack in which clients often stop when receiving legitimate *Data*, while CPA is more likely going to operate as a flow, hence leaving blind spots about the phenomenon. Thirdly, most of the previous works over-estimate the CPA with unrealistic behaviors. For example, they do not enable the use of *Exclude* field to avoid malicious *Data* or consider pre-polluted caches with an impractically high percentage of malicious *Data*. Although there are explanations on the attack scenario and how malicious *Data* is inserted in caches, they are insufficient to explain why CPA can achieve such high percentage of malicious *Data* in caches.

2.3.2 Cache Pollution Attack

In cache pollution, the attacker forces caches to store content irregularly to degrade the caching's benefits for legitimate users. This attack has been widely studied in IP, especially for web-caching [73]. There are two classes of cache pollution, namely locality disruption, and false locality:

- **Locality disruption** (or *random request*): attacker continuously issues requests for random contents to flatten contents' popularity in caches. Contents will be ranked evenly in the cache replacement policy. Thus, popular contents are less likely to be found in caches, hence decreasing the cache hit ratio. Legitimate

Interests must be forwarded up to the content provider, increasing the delivery delay;

- **False locality** (or *unpopular request*): the attacker sends requests for a set of particular contents to create artificial popularity. Such an attack can be associated with other attacks, e.g., CPA (Section 2.3.1), to promote contents deliberately. As such, the cache prioritizes these contents and removes truly popular ones. Similar to local disruption, requests for truly popular content are thus forwarded to the original provider, increasing the delivery time. The attacker can figure out which contents are unpopular by side channels or collaborates with a provider to fetch unpopular content.

Existing works on cache pollution attack can be divided based on their approach into two groups: *caching decision* or *detection - mitigation*.

Caching Decision Approach

Approaches in this group address the cache pollution attack by modifying the default caching decision in NDN. Karami et al. [74] proposed a cache replacement policy based on *Adaptive Neuro-Fuzzy Inference System* (ANFIS). First, an ANFIS structure is built according to characteristics of cached *Data*, such as duration in the cache, request frequency. Given a cached *Data*, the ANFIS structure will return a goodness value between 0 and 1. Low goodness value indicates the artificial popularity. More specifically, 0 means false-locality, 0.5 means locality-disruption and 1 indicates a valid *Data*. The system continuously evaluates the goodness of *Data* packets that have been cached beyond a predefined period and applies cache replacement policy only to *Data* with high goodness values.

Xie et al. [75] propose an add-on proactive solution to make NDN caching more resilient to cache pollution attacks without the need of preceding detection. It does not require coordination among routers and can adapt to various existing cache policies. An essential component of CacheShield is the *shielding function* – a function that calculates the caching probability for each solicited *Data*. To prevent the attacker from predicting the caching decision, the authors propose to use a probabilistic function as the shielding function. For instance, the authors consider a logistic function $P(t_i) = 1/\left[1 + e^{(p-t_i)/q}\right]$ in which t_i is the number of times a content C_i has been requested, p and q are the function's parameter. As t_i increases, implying that C_i is a popular content, the shielding function will yield a higher caching probability for C_i. When the router receives a solicited *Data*, it executes the shielding function. If the function returns *true*, the *Data* is cached. If the function returns $false$ and the content is requested for the first time, the content name is recorded, and its counter

t_i is initiated. If the function return $false$ and the content name is already recorded, t_i is increased. Due to the characteristics of the shielding function, the proposal is effective against locality disruption attack and not very useful against false locality one. Although calculating the shielding function might be simple, this approach requires the router to record content names and their corresponding counter t_i, adding the storage overhead for the router, especially when the number of content names increases. Also, a common disadvantage of approaches in this group is that they run continuously, consuming router's resources even when no attack occurs.

Detection - Mitigation Approach

As its name suggests, previous works in this group focus on detecting and mitigating the cache pollution. In [76], Guo et al. analyze ISPs' *Point-of-Presence* (PoP) networks with real-world Internet measurement datasets and show that *Interests* for popular *Data* will go through many different router-level paths within the PoP. If such a path diversity does not hold for a cached *Data*, it is unlikely to be popular and is exploited by an attacker for polluting the cache. Based on this remark, the authors design an algorithm for detecting and mitigating the pollution.

Conti et al. [77] propose a lightweight detection against cache pollution attack based on machine learning. First, the algorithm randomly selects a subset S of *Data* as a reference sample set instead of checking all *Data*, reducing the computational cost. For each *Data* $D_i \in S$, the router calculate a probability value $p_i = n_i / \sum_{j \in S} n_j$, where n_i is the number of times D_i has resolved an *Interest* in the measurement interval. In the learning phase, router will gather samples for *Data* $D_i \in S$ to compute a detection threshold τ using Knuth's algorithm [78]. In the detection phase, the router collects samples and compute a detection statistic δ_m that depends on p_i, n_i and the sample size. If $\delta_m > \tau$, then the router is considered to be under attack. A drawback of this approach is that the threshold τ is calculated on-line only in the learning phase and is applied directly to the detection phase without further tuning. Besides, the evaluation results are based on an assumption that the attacker's *Interests* follow a uniform distribution, which is simplistic and not compatible with real conditions of the router.

Remarks on Cache Pollution Attack

Similarly to CPA, a majority of existing works on cache pollution attack is evaluated only with simulation results, thus questioning their performance in reality. Besides, there is no consensus on the evaluation scenario for NDN caching attacks in the scientific community so far [79], making the results incomparable. Moreover, most of

them can only target a single type of cache pollution, thus requiring router running complementary solutions to prevent cache pollution.

2.4 Name Attacks

The fact that hierarchical content names are indicated explicitly in *Interest* can reveal information about the communication to a third-party. In addition, as NDN deploys ubiquitous in-network caching, content objects now can stay longer in the network to resolve similar requests from users, providing more time for the attacker to extract information from content names, even when the communication has already ended. Name attacks leverage those flaws of content name and cached *Data* to gain knowledge of users' demands and content, violating the privacy.

2.4.1 Timing Attack

NDN deploys in-network cache to reduce the latency of content retrieval. As a result, cached content has a shorter delay than one retrieved from other sources. Timing attack exploits such a delay difference to distinguish a cache hit and a cache miss. A cache hit indicates that the content is recently requested by another client, and a cache miss means that the content is not requested recently, or has been removed from the target cache. Figure 2.4 illustrates the first step of timing attack, i.e., estimating cache hit's latency [4]. First, the attacker sends an *Interest* to insert a content into the cache. Then, the attacker requests for the same content, and measures the cache latency d_c. The attacker can repeat the second step many times and take the mean value for a better estimation.

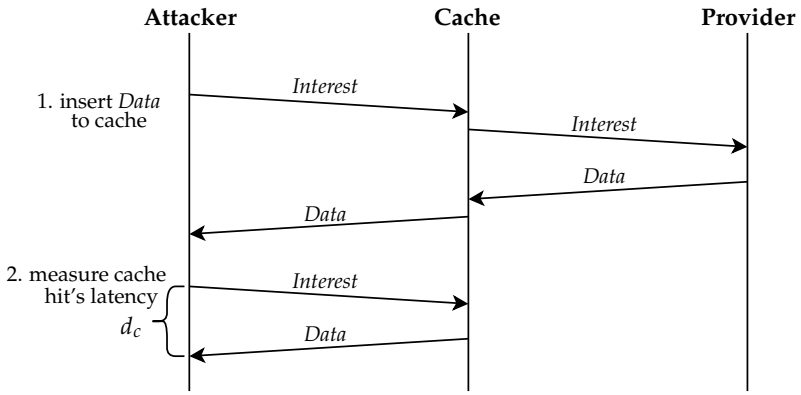


FIGURE 2.4: Cache hit's latency estimation [4]

Afterward, to check whether a target content was requested recently, the attacker sends an *Interest* for this content and measure the latency $d_{request}$. If $d_{request} \approx d_c$, the attacker can infer that the content was requested recently, revealing neighbor

clients' privacy. Otherwise, it was returned from another location. Most of the previous works on timing attack use a scenario in which the attacker targets the attacker's edge router. Indeed, targeting a farther router is more complicated because the attacker cannot consider all impacts outside (e.g., attacker hardly know the number of hops to the target cache). Timing attack can be extended to locate the client, as demonstrated in [80] or help attack gain knowledge to prepare subsequent attacks.

Asc et al. [81] propose to deceive the latency estimation by delaying cached *Data*. The authors argue that privacy-sensitive content are usually unpopular and only cached *Data* of these contents needs to be delayed. Therefore, the router will only apply a random delay for a cached *Data* for the first *k Interests*. When a content is requested more than *k* times, it is considered popular and will not be delayed. This solution requires tracking the number of *Interests* for each content and undermines the user's experience for the first *k Interests*

In [82, 83], Mohaisen et al. also has a similar approach of adding delay to deceive the timing attack. When an edge router fetches a content for the first time, it records the retrieval latency. The first *Interest* for a cached content from a new user (i.e., who has not requested for that content) will be delayed to mimic the retrieval time from the original provider. By adding delay to returned *Data*, the attacker cannot distinguish between cached or not cached *Data*. Following *Interests* for the same content from the same user will usually be replied without adding delay. The solution requires the edge router to track the number of *Interests* that a user sent for each content and the delay to add to each face. To share the storage overhead, the authors propose to delegate the former to the access points connected to the edge router. The access point flags the *Interest* from a new user to notify the router. The router only delays the cached *Data* for the flagged *Interests*. However, the main drawback of this approach is to eliminate the positive effect of caching on content retrieval delay.

2.4.2 Name Privacy

Possible actions exploiting NDN name privacy are various. First, NDN *Interest* packet expresses a user's demand for specific content. By monitoring the content name in *Interests* (e.g., inspecting a cache with timing attack, see Section 2.4.1), one can know the trends in proximate users' requests and popular contents [84]. In addition, NDN uses hierarchical naming scheme, and NDN names are usually human-readable. Therefore, the content name may have some semantic binding to the content and its provider. Moreover, NDN pervasive caching may keep traces of communications. Using the longest-prefix matching mechanism in NDN and the *Interest*'s *Exclude* field, attackers can iteratively send *Interest* to discover which contents are kept in cache [4, 30]. As such, attackers can capture content names, gain knowledge

about communications, even when it already ended. In case the attack can compromise a router, it can also black hole *Interests* for some specific contents in its watchlist [85].

Arianfar et al. [86] propose to mix blocks of a *target* file (i.e., the file that a user wants to retrieve) with blocks of *cover* file (i.e., the file used to mix with a target file to conceal it). The goal is to force the attacker to perform huge computations in order to understand the contents it gets. The content provider is responsible for selecting and mixing cover files with target files. The selected target and cover file are divided into blocks. The content provider then mixes two or more of blocks, resulting in chunks – the actual data objects that are published. The mixing function can be reversed by the user once necessary chunks have been received. The content names for all files, cover or target, are known to all parties. The name convention of blocks or chunks is agreed between users and the provider. As such, users know how to compute the chunk's name using the correct hash function. To fetch the content, the user must request necessary chunks and reconstruct the original file by himself. Although the attacker can snoop cache to retrieve chunks, it cannot know which blocks are used to compose those chunks. In addition, since all names of blocks and chunks are hash values, the attack cannot know which blocks are being requested. However, this approach requires a secure side channel for delivering meta information to the users who want to retrieve the whole content. The approach is only promising with a substantial storage infrastructure for all published chunks. Besides, it is only suitable for static content, not for dynamic content because of high computation cost.

Chaabane et al. [85] propose to use bloom filter to replace NDN hierarchical name. Instead of issuing the plain-text content name, a user computes a matching Bloom filter[5] $HB = (B_1, B_2, \ldots, B_n)$ for content retrieval, where B_i is the Bloom filter of i first components of the content name. The router's data structure and the matching rule also have to adapt to the Bloom filter of the content name. FIB, PIT, and cache will now store bloom filter in each entry instead of the prefix. In addition, this approach also reduces the size of router data structure. However, using Bloom filter requires periodical resetting and has false positives, which may lead to delivering wrong content or incorrect forwarding *Interests*.

Anonymous Named Data Networking Application (ANDaNA) [87, 88] is an onion

[5] A Bloom filter for representing a set $S = \{x_1, x_2, \ldots, x_n\}$ of n elements is described by an array of m bits, initially all set to 0. A Bloom filter uses k independent hash functions $h_1, \ldots, h_k$ that yield an output in range $\{1, \ldots, m\}$. Each hash function h will map an element x to a position uniformly over the range $\{1, \ldots, m\}$. For each element $x \in S$, the bits $h_i(x)$ are set to 1 for $1 \leq i \leq k$. A position can be set to 1 multiple times, but only the first change has an effect. To check if an item y is in S, we check whether all $h_i(y)$ are set to 1. If not, then certainly y is not a member of S. If all $h_i(y)$ are set to 1, we assume that y is in S. A Bloom filter may yield a false positive, where it suggests that an element y is in S even though it is not. For many applications, false positives may be acceptable as long as their probability is sufficiently small [22].

routing [89] overlay network for NDN, that prevents adversaries from linking users with the content they retrieved. In ANDaNA, before transmission, the user will set up an *ephemeral circuit* by choosing a pair of *Anonymizing Routers* (ARs) and securely give them two symmetric keys used for encrypting packets in the circuit. The router close to user's side is called *entry AR*, and the other is called *exit AR*. A *Interest* is encrypted with two layers and sent to the entry router. Each router of the circuit will decrypt one layer and forward the request to next hop. When a corresponding *Data* arrives, the *exit AR* encrypts the whole set of information consisting of *(content, name, signature)*, under the key provided by the user. The ciphertext is treated as payload for the new *Data* packet which is then signed and sent to *entry AR*. The latter removes the packet's signature and name, then encrypts the remaining ciphertext again using the second symmetric key provided by the user. The *entry AR* also signs and forwards the ciphertext as a normal *Data* packet, but with the original encrypted name. After decrypting the payload, the user discards signatures created by the two ARs and verifies the one from the content provider.

In [30, 90], Lauinger et al. propose an idea of *selective caching*. The authors argue that most traffic is probably not privacy-sensitive. Thus, applying solutions uniformly to all traffic is not necessary and may lead to undesirable impact since it reduces network performance. Such trade-off could be lessened if those solutions are applied only to privacy-sensitive traffic. Hence, the privacy/performance trade-off is now transformed to a problem of classifying sensitive traffic. The paper proposes that privacy-sensitive has specific properties: low locality (i.e., not requested by many users) and instantaneous popularity (i.e., requested during a short period). A content with low popularity is much easier to link to a specific user and reveals more specific information. The popularity of a content is defined in term of the number of individuals requesting the content in a specific time and location. However, protecting privacy will require a much more sophisticated definition of popularity. Moreover, globally deploying traffic classification might be difficult in practice. One of the main research challenges is how to evaluate popularity correctly without consuming too many resources for monitoring and recording purposes.

2.5 Summary on Named Data Networking Security Attacks

In this section, we evaluate attacks presented in this chapter accordingly to selected attributes in order to summarize and select a set of prioritized attacks to address further in the following chapters. We first introduce attributes used to assess attacks, as well as the values for grading the impact on each attribute. When considering the information security, the CIA triad [91] is usually mentioned. Its name is an acronym

representing three principles in the information security, including:

- **Confidentiality** (or privacy) indicates that entities should be given sufficient privilege to perform their duties and no more. Confidentiality impact is related to unauthorized access to data or discloses of identity information. Grading values are: *H* (high) - the attacker directly monitors the communication and disclose the information; *M* (medium) - data/information is disclosed indirectly after the communication ends; *L* (low) - little data/information is leaked; *Blank* - no impact;

- **Integrity** implies that data must not be changed until reaching the destination. Violating this characteristic implies that the attack changes the content delivered to users. Grading values are: *H* - the attack can modifies *Data* regardless of the prefix; *M* - the attack can modifies *Data* of a specific prefix; *L* - the attack modifies a few *Data* packet; *Blank* - no impact;

- **Availability** ensures that data and resources are always available for authorized entities. Attacks that prevent users from retrieving content violate this characteristic. Grading values for this attribute are: *H* - users cannot retrieve most of content; *M* - user can fetch most content with delay or fetch incorrect content; *L* - a few *Data* packets cannot be fetched or fetched with unease; *Blank* - no impact.

We also consider an additional attribute when evaluating attacks' severity:

- **Effect scale** refers to the range of entities impacted by the attack. Grading values for this attribute are: *H* - the attack affects multiple routers; *M* - the attack affects a router and its connected users; *L* - the attack affects a specific entity.

The four previous attributes have been evaluated in [48] using the the OWASP risk rating[6] (for confidentiality, integrity, availability) and the severity assessment by Symantec[7] (for effect scale). In addition to select a set of attacks for further study, we assess two following attributes:

- **Number of previous works** represents the ease with which one can find existing works on the involved attack. This also partly represents the attention given to an attack. Grading values for this attribute are: *H* - vast number of previous works available; *M* - medium number of previous works available; *L* - only a handful of previous works are available;

[6] See https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology.
[7] See https://www.symantec.com/content/en/us/about/media/securityintelligence/SSR-Severity-Assesment.pdf.

- **Replicability** refers to the ease with which one can reproduce an attack for further study. As stated in previous sections, the majority of previous works evaluated their proposals with simulation environment, hence leaving the performance in reality unknown. To study an attack in reality, one must be able to reproduce it. Therefore, a detail demonstration on how to perform the attack does matter. Grading values for this attribute are: *H* - detail attack protocol available and can be reproduced; *M* - attack protocols are lacks of detail to demonstration; *L* - only attack principle available.

TABLE 2.1: Overview of NDN attacks

	Confidentiality impact	Integrity impact	Availability impact	Effect scale	Number of previous works	Replicability
Interest flooding			H	H	H	H
Time analysis	M			M	L	H
Name privacy	M			M	M	L
Content poisoning		M	M	H	M	M
Cache pollution			M	M	M	M

Table 2.1 summarizes our assessment on NDN attacks according to aforementioned attributes. One can note that most of the attacks in NDN do not affect any specific user. Since NDN does not have host-identifiers, attacks must target routers to cause damage, indirectly affecting proximate users. Regarding the number of previous works, the table shows that most of the research effort focuses on IFA. It also has high replicability, facilitating further studies in the real deployment context. Besides, the table shows that name privacy, CPA and cache pollution attacks also draw the attention of the NDN community, but to a lesser extent. The name privacy attack, however, lacks replicability and thus is not selected. Regarding content poisoning and cache pollution, the two attacks have similars grading values. However, the CPA exhibits the impact on a larger scale and more attribute (integrity). In short, we deliberately select IFA and CPA as two NDN attacks to study further in the following chapters. The former is easy to implement while the latter is NDN-specific and have been neither fully described nor comprehensively characterized.

2.6 Conclusion

This chapter provided a state of the art on NDN security attacks. We first had a look at existing surveys on NDN security and identified the scope of attacks that we want to address in this chapter which is data-plane security threats. Based on existing surveys, we presented our taxonomy, including PIT, CS and privacy attacks. For each attack, we explained its fundamental idea and introduced remarkable existing works. To compare attacks and summarize their severity, we evaluated them

based on the impact on confidentiality, integrity, availability, effect scale, number of previous works and replicability. Based on such evaluation, we selected IFA and CPA as important NDN attacks to be prioritized in following chapters. Beside their significant impacts on aspects of availability and effect scale, they received the most attention from the NDN community as well as a decent capability of replication for further study.

Remarks on previous works indicated three common limitations that need to be tackled. First, most of the previous works evaluate their proposals with results from a simulated environment, thus questioning the actual performance in the real deployment context. Secondly, a majority of existing works do not provide reliable detection. More specifically, they cannot guarantee a prescribed false-alarm rate. Undesired false alarms can waste network resources or accidentally penalize legitimate clients. Also, the detection threshold either requires a non-trivial learning phase or is based on authors' experience, leading to rigid and unreliable performance. Finally, existing works usually focus on a particular NDN attack. The literature lacks a generic security solution that can address both revealed security threats and potential ones not yet revealed to date. These weaknesses will be addressed in each of the following chapters. In the next chapter, we will address the first identified drawbacks by featuring IFA and CPA in NDN real deployment.

Chapter 3

Characterization of Crucial Attacks in Real Deployment of Named Data Networking

Contents

In this chapter, we address the first limitation identified, which is the extensive use of simulated results in existing works (see Section 2.6), by considering NDN crucial attacks in the real deployment. We first argue for the need of featuring NDN security attacks in realistic conditions. We then introduce common deployment features of our NDN testbeds for experimentation on the two use-cases attacks – IFA and CPA. For each attack, we describe scenarios that successfully launch the attack

in our testbed, introduce environment setup for the experimentation in detail, followed by remarks on the attack impacts based on a thorough analysis of collected data.

3.1 Necessity of Attacks Characterization in Real Deployment

As mentioned in Chapter 2, previous works mostly evaluate their proposals with results from the simulated environment. Such extensive use of simulation was reasonable in the early stage of NDN. Nowadays, NDN has achieved a certain level of technological maturity. Its deployment efforts in the real environment have become more active and are currently receiving significant attention from the community (see Section 1.6). Therefore, exhaustive characterization of NDN security attacks in the real environment is essential such that one can fully understand its impacts in reality and prepare for the further investigation. Moreover, A fully detailed attack protocol in realistic conditions is indispensable since it not only proves the reality of the security threat but also facilitates the replicability of attack experimentation, thus encouraging the research in the community. In addition, experimentations on NDN attacks must take into account practical behaviors of involved entities (e.g., legitimate client, attacker's client, malicious server, router), as well as the dynamic nature of network traffic.

3.2 Common Deployment of Testbeds

Our experiments for each attack are conducted on a use-case topology which consists of several clients, intermediate routers with NFD installed and servers / NDN providers. The deployment of the use-case topology is depicted in Figure 3.1. We use the cloud operating system OpenStack[1] to control large pools of computing, storage, and networking resources throughout several physical hardware. For each node in the topology, a virtual machine (VM) is created, following the template configuration of OpenStack with Ubuntu as the operating system, in which corresponding applications are installed. These VMs are connected to a virtual network provided by OpenStack Neutron, the OpenStack networking service.

A conventional deployment, i.e., installing dedicated hardware with integrated software, is unaffordable to deploy NDN nodes since it would be costly, difficult to manage and time-consuming. Thus, we leverage Docker[2] as a container-based virtualization framework of network functions. VMs that emulate NDN nodes will

[1] See: www.openstack.org.
[2] See: www.docker.com.

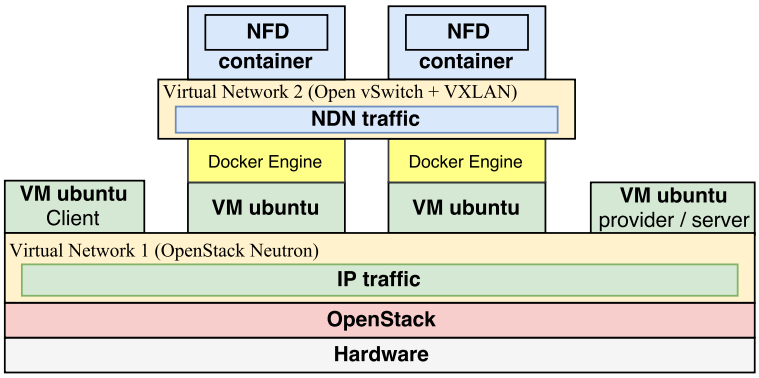


FIGURE 3.1: Common deployment of testbeds

host a Docker container in which NFD is installed. To allow these containers to communicate across different VMs, we deployed OpenVSwitch (OVS)[3] in each VM as a transport framework for all data from one container to another. In such an architecture, NDN traffic is configured as an overlay which encapsulates all *Interest* and *Data* packets in IP/UDP channels and transports through a VxLAN tunnel.

3.3 Interest Flooding Attack Characterization

As described in the literature, the principle of IFA is to occupy the PIT with an excessive amount of *Interests* for non-existent contents that cannot be resolved (see Section 2.2.1) As such, the PIT eventually gets full and stops handling new incoming *Interests*. In this section, we investigate the feasibility of IFA given the existence of *Negative ACKnowledgement* (*NACK*) packet and highlight the impact of IFA in the real deployment of NDN.

3.3.1 Limitations of *NACK* Packet

In the effort to mitigate IFA, the authors of [64, 65] introduce the *NACK* packet to extend the forwarding mechanism of NDN. Although the *NACK* packet is already presented as a proposal against IFA in Section 2.2.1, this section focuses on how it is actually implemented in NFD and its limitations. When an NDN router can neither satisfy nor forward an *Interest*, a *NACK* is sent to downstream routers. A *NACK* also carries an error code to notify why the router cannot satisfy the *Interest*. Currently, the *NACK* packet is already implemented in the NFD (v0.5.1) [5] with three error codes defined as follows:

- *Congestion NACK* indicates that there is congestion on the outgoing links;

[3] See: openvswitch.org.

- *Duplicate NACK* implies that the *Interest* is considered as a loop, i.e., having the same *Name* and *Nonce* fields (see Section 1.3.1) as ones in the corresponding PIT entry;
- *NoRoute NACK* notifies that there is no eligible route available to forward the *Interest*.

More specifically, a route is considered *eligible* for an *Interest* if it matches the two following conditions:

- The corresponding face is up and different from the *Interest*'s incoming face;
- Forwarding to this route does not violate special prefixes (e.g., `/localhost`, `/localhop`).

Upon receiving a *NACK*, a router can do one of the following [5]:

- Retransmit the *Interest* with the same or different upstream nodes;
- If some but not all upstream nodes returned *NACK*, the router might want to wait for the response (*Data* or *NACK*) from more upstream nodes and do nothing by the meantime;
- When all pending upstream nodes returned *NACKs*, the router gives up and forward the *NACK* to downstream nodes. In case the error codes from upstream *NACKs* are different, the least severe reason is passed to downstream nodes. The order of error codes' severity is defined as: *Congestion* < *Duplicate* < *NoRoute*.

As such, the *NACK* has many benefits on NDN architecture. With the error code, downstream routers can adapt their sending rate and forwarding strategy so that the upstream nodes do not get overload. In addition, routers can release PIT resources much faster, rather than waiting for the *Interest* lifetime expiration. Moreover, in case a router receives an *Interest* for non-existing content, it has no matching route in FIB to forward the *Interest* and thus will send back a *NACK* packet with *NoRoute* error code to notify the downstream. Hence, *NACK* prevents non-existing *Interest* of IFA from being forwarded further, hence occupying routers' resource.

Although *NACK* is an effective mechanism to mitigate IFA, there are still several limitations that make it an incomplete solution against the PIT overload and especially IFA. First, the *NACK* mechanism is based on the *Interest* prefix to mitigate the overload. If an attacker can send *Interests* with different prefixes, the router must issue a *NACK* for each prefix. Such a reaction can be burdensome. In this case, it

could be more efficient to limit the sending rate of the face under attack. Secondly, using *NACK* means that a router's forwarding strategy depends on its upstream. A malicious upstream hence can add a delay to its response to postpone the issue of *NACK* to downstream, thus enabling a vulnerability for IFA. Finally, dealing with a non-existing content name may be time-consuming, especially when several routes are available because the router may try all available routes before giving up and sending a *NACK* to downstream.

3.3.2 Investigated Scenario of NDN Coupled with IP Application

As mentioned in Section 1.6, research efforts in ICN deployment is very active and having significant attention. In our IFA featuring, we investigate a scenario where an *Internet Service Provider* (ISP) couples an NDN network to the existing IP network in order to provide the *HyperText Transfer Protocol* (HTTP) service to users, as illustrated in Figure 3.2. The web service is selected as an illustrative example of IP applications since it is the most popular on the Internet. Addressing such a service would be a relevant step toward the integration of NDN into the existing networks. In this scenario, an NDN island is deployed inside the ISP's core network to leverage the benefits of its caching system and low-latency data delivery for a substantial part of traffic.

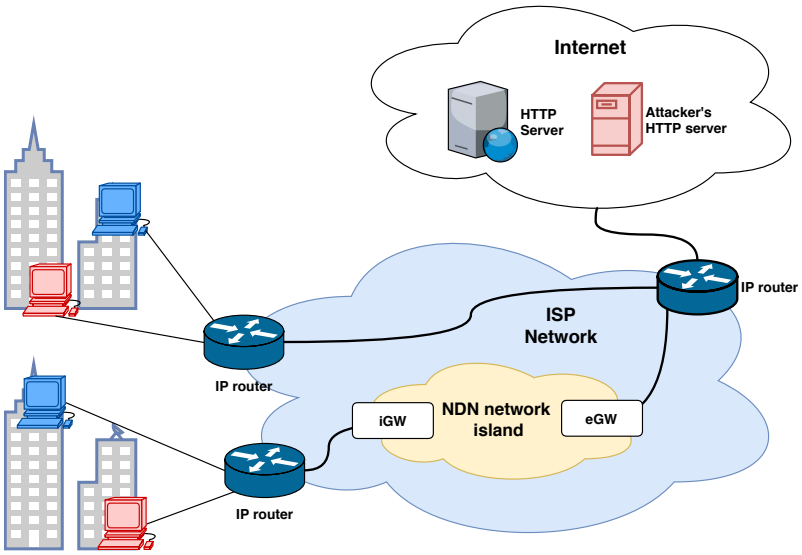


FIGURE 3.2: An illustrative example of NDN coupled with IP stack for HTTP application

As the current Internet and web users do not implement NDN, their IP traffic will be forwarded to dedicated gateways [92] which translate HTTP traffic to NDN traffic and vice versa. Two types of gateways are required: *ingress gateway* (iGW) and *egress gateway* (eGW). The operation of *iGW* and *eGW* is briefly demonstrated

in Figure 3.3. When an HTTP request arrives at *iGW* (arrow 1), it is translated into *Interests* and injected in the NDN network. *Interests* that cannot be satisfied by NDN routers' cache will reach the *eGW*. The *eGW* checks *Interest*'s name for fragmentation and retrieves remaining chunks of the HTTP request if needed (arrows 3,4). After that, the *eGW* reconstructs the original HTTP request and sends it to the corresponding HTTP server (arrow 5). When an HTTP response arrives (arrow 6), the *eGW* converts it into *Data* packets and sends them into the NDN network, reaching the *iGW* (arrow 7). The first *Data* is considered as a response to the *Interest* from the arrow (2). The *eGW* also includes information about fragmentation in the *Data* name, so that the *iGW* can retrieve remaining chunks of the HTTP response (arrows 8, 9). Afterward, the *iGW* reconstructs the HTTP response and delivers it to the client. All of these operations and the existence of the NDN island is entirely transparent to network users. Hence the users can still experience the benefits from NDN without any adaptation effort from their side.

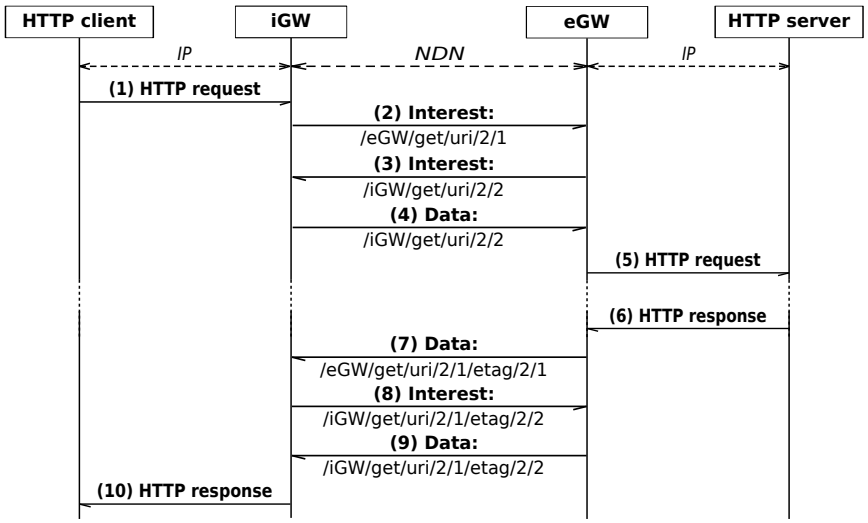


FIGURE 3.3: Interaction between ingress and egress gateway

3.3.3 Proposed Attack Scenario in the Real Deployment Context

In our use-case of NDN coupled with IP (Figure 3.2), since clients are deployed in IP network, the attacker cannot control as many aspects of IFA as they can in the NDN native environment. Moreover, with the existence of *NACK* in the current NDN implementation, the IFA mechanism in the literature is no longer feasible because *Interests* for non-existing content cannot be forwarded, but instead, are responded with a *NoRoute NACK*. Nevertheless, Figure 3.3 reveals an exploitable flaw to corrupt the NDN network: after a long enough delay between the HTTP request and the corresponding HTTP response (arrows 5,6), the first *Interest* (arrow 2) expires. Hence, the first *Data* (arrow 7) is considered unsolicited and rejected by the *iGW*. As such, only

the third attack scenario identified in [93] enables an attacker to implement an IFA in the operational context presented above. More precisely, by leveraging a botnet or an equivalent means, the attacker can own the control of multiple web users and a malicious web server on the Internet (see Figure 3.2). Attacker-controlled users will browse for the website hosted on the malicious server. With the existence of *NACK* packets, the attack relies on intentionally adding a large delay to the response from the malicious server so that *Interests* exchanged in the NDN core network will linger in the router as long as possible. Moreover, by using a malicious server and delaying its response, the attacker can bypass the protection mechanism against IFA of the *NACK* packet. Furthermore, since malicious requests occupy NDN routers during the attack, legitimate users will suffer longer delays when accessing a website. Since the *Interest* packet will automatically expire and release resources for the router, it is hard to shut down a router when the attacker does not have the control over *Interests* in NDN. Therefore, such a scenario is likely to slow down the network rather than completely deny the service.

3.3.4 Experimental Setup

Use-case topology

To feature IFA's impacts, we deploy the topology depicted in Figure 3.4, using the tools described in Section 3.2. The NDN network consists of four nodes with NFD installed (v0.5.1 with *NACK* implemented). The *iGW* connects to web users and the *eGW* connects to the Internet. The *iGW* also plays the role of an HTTP proxy for clients. On the other side of the NDN network, we deploy a malicious server which connects to the *eGW* and runs an Apache HTTP server that collaborates with the attacker to perform IFA.

Data collection tool

To collect the data from our experiments, the Montimage Monitoring Tool[4](MMT) probe has been used. A plugin for this probe was developed to interpret NDN protocols for both native (NDN only) and overlay (NDN/IP stack) cases. The plugin uses TLV-based signatures and can extract the values of all NDN protocol fields as well as perform basic statistics. This extracted metadata allows monitoring the NDN traffic, differentiating NDN packet types, as well as conducting performance and security analysis of the communication between NDN nodes.

[4]See: www.doctor-project.org

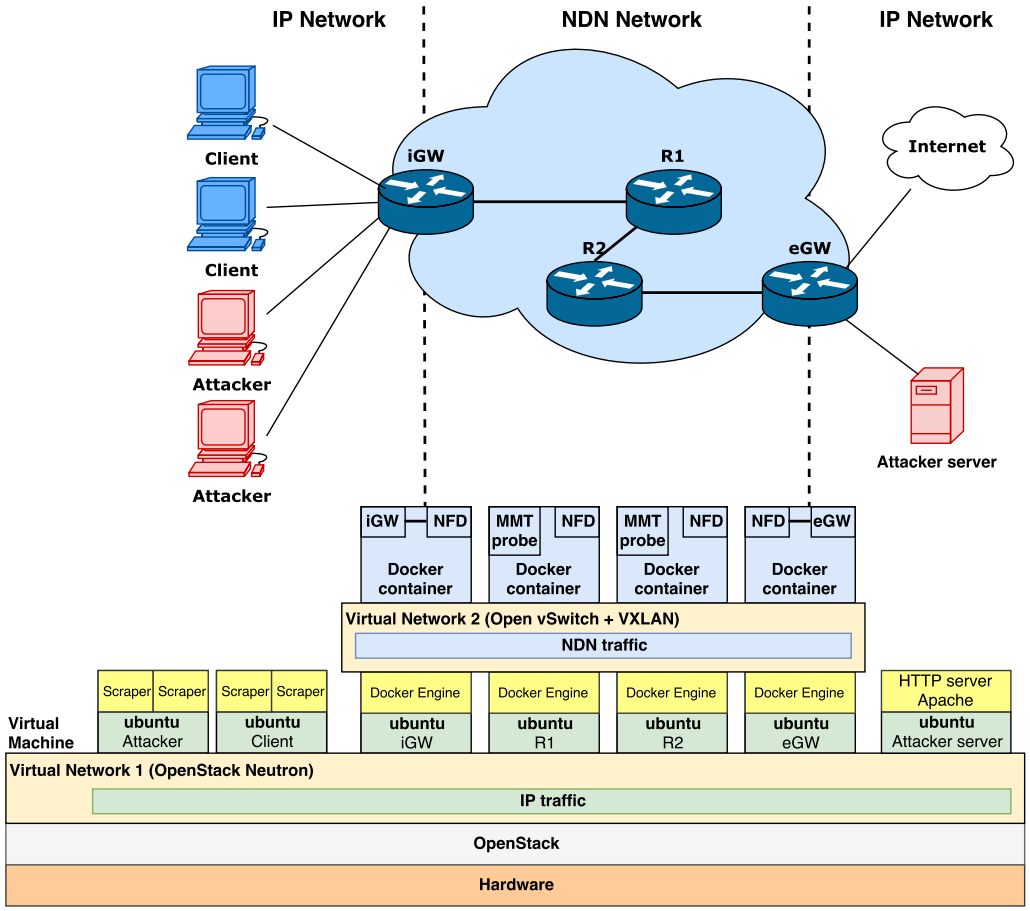


FIGURE 3.4: Use-case topology and testbed architecture for IFA

Web User Emulator

To generate web traffic, we developed a web user emulator based on Jaunt API[5] - a Java library for web-scraping that allows retrieving objects in a web page such as images, CSS, and javascript. Such an emulator will mimic the behavior of a web

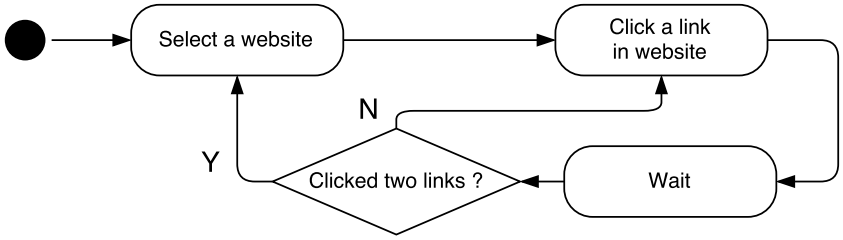


FIGURE 3.5: Flowchart for web user emulator's behavior

user. The emulator behavior flowchart is illustrated in Figure 3.5. To start browsing, a user enters the website to the browser's address bar and wait for it to fully loaded. The more users access a website, the more popular the website become. To reverse-engineer this process, the emulator will first select a website randomly from

[5]See: jaunt-api.com

a given list of popular sites, based on a Zipf distribution - a well-known distribution to model contents popularity [94]. Next, the emulator retrieves the website's object list, then check for objects' existence on the Internet. If an object exists and does not require *HTTP Secure* (HTTPS) connection, the emulator passes the HTTP request for that object toward the *iGW* to retrieve it. After loading the whole object list, the emulator waits before selecting a link randomly on the website to browse. This behavior emulates web users' action of reading and clicking when navigating through a web page. The waiting time is drawn from the exponential distribution - a popular distribution when modeling the interval between user's requests [95]. The same process of loading and waiting is repeated one more time for a second randomly selected link. The emulator then selects another website in the given list and repeats the whole process. The resulted traffic is expected to be as close as possible to one generated by real web users.

Experiment Scenario

Each VM can run several emulators at a time. One can change the amount of traffic generated by modifying: (1) the number of emulators; (2) the number of threads given to each emulator (the more threads, were given, the quicker to load the object list) and (3) the average emulated browsing time. Each experiment lasts for 30 minutes, including 15 minutes without IFA followed by 15 minutes with IFA traffic. In all experiments, 30 emulators, each with 2 threads, are launched on two VMs to generate legitimate traffic. Their average browsing time is set to $10s$. Such a configuration for legitimate traffic generates, on average, approximately 80 HTTP requests/s.

The list given to the emulator includes 90 websites chosen among the most popular ones. Using more websites would be unnecessary because sites located in the tail of the Zipf distribution have very low probability to be selected. The probe generates one sample after each 4-second interval. Each sample consists of the number of incoming *Interests* and outgoing *Data* for each interface of the router. Such a configuration for probes helps reduce the noise in packet loss rate, minimize unnecessary management effort under the constraints of experiment resources.

On the attacker's side, bad (or malicious) emulators requests only objects from the malicious server's site. We vary the attack payload by changing the number of malicious emulators and number of threads given to each emulator. We set up a very small value to their average browsing time ($1ms$) because bad emulators want to send as many HTTP requests as possible and do not take time to browse the site. In the following, the attack power will be presented in term of bad HTTP requests/s for easy understanding. Each attack setup is run for 10 times in order to increase the amount of data and, hence, reducing the statistical spread. The malicious server's

TABLE 3.1: IFA experiments' constant parameters

Constant	Value
Number of legitimate emulators	30
Number of threads/legitimate emulator	2
Legitimate emulator's average waiting time	$10s$
Attacker emulator's average waiting time	$1ms$
Malicious server's delay	$\mathcal{U}(4.9, 5.1)$ s
Number of websites in list	90
MMT-probe sampling interval	$4s$
NFD version	v0.5.1, latest until October 16^{th}, 2017

delay is a random value that follows a uniform distribution $\mathcal{U}(4.9, 5.1)s$. We do not use a constant value to emulate the dynamic latency in the real condition[6]. The hosted website contains lots of objects[7]. Therefore, requested objects are more diverse, and sessions to the malicious server last longer, prolonging their loads on NDN network. Constant parameters are summarized in Table 3.1.

3.3.5 Attack Phenomenon

Under IFA, one can expect that the NDN network is occupied severely, and users will experience a longer delay when loading and browsing a website. As described in Section 3.3.4, a website is selected randomly based on Zipf distribution. As a result, those located at the tail of content popularity are less likely to be picked, hence contribute less to the density presented above. To provide an overall view of IFA effect on the whole website list with various delays, it is proposed to record the latency 20 times for each website in the list under different attack scenarios. No click will be emulated since randomly selected links may add more delay to the record, causing inconsistency between various sites. Websites that take too long to retrieve ($>$ 300s) are considered unreachable. Figure 3.6 illustrates the average delay under attack as a function of average delay without attack. This figure helps visualize the severity of the increased delay that users suffer when IFA happens. Each point represents the measurement of an individual website. For readability, the black dash line shows the equation $y = x$ so that one can see easily the increase of delay due to

[6] Delays larger than 5.1s were tested. In such cases, malicious emulators fail to establish an HTTP connection to the attacker's server. A possible reason is that *NACK* has been triggered by such long delay, releasing PIT entries, thus *iGW* and *eGW* cannot interact successfully. Delays lower than 4.9 were also tested. However, they are not sufficient to flood NDN routers. Keep in mind that the default *Interest* lifetime of 4s in NDN network cannot be modified by the attacker located in IP network. As such, the attacker must tune the malicious server's delay accordingly, rather than using an arbitrary high delay.

[7] To be precise, the malicious web's HTML is 60KB and the full website including objects is 1.6MB. Thus, an HTML request will result in more than one *Data* packet of 4KB in NDN, increasing the router's processing. With today's web pages and their content size (larger than our website), the proposed IFA's effectiveness could be considerably higher.

the attack. Solid lines show the results of an affine regression of these measurements, using least squares estimator, for each attack scenario. One can note that the attack is still successful even when it has low power (green line). This trend, however, is not significant since it is quite close to the neutral line. When the attack power increases, the delay gets worse, as indicated by the increased in trends' slope.

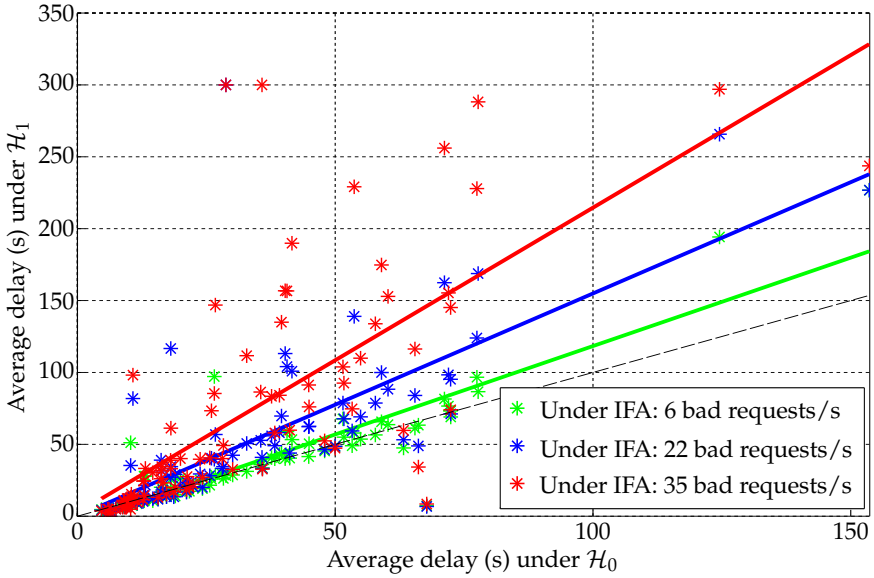


FIGURE 3.6: Attack effect on increasing the delay of each individual website, under different attack setups

We conclude that despite the implementation of the *NACK* packet, IFA is still feasible. By proposing a practical attack scenario in the real deployment of NDN couple with IP network, we have demonstrated that IFA succeeded in degrading the user's experience (i.e., web loading delay).

3.4 Content Poisoning Attack Characterization

Although CPA is also identified as an important attack in NDN, it does not receive as much attention from the community as IFA (see Table 2.1). In CPA, *Interests* from legitimate users will be matched by *Data* packets that are forged by the malicious provider and inserted in router caches. In the literature, the bad *Data* is mostly assumed to be injected by a compromised router. We argue that taking control a router, though possible, is hard to achieve in reality. Therefore, in this section, we describe three CPA scenarios that leverage the provider-user collaboration and an unsolicited provider to inject bad *Data*. We then conduct a set of experiments with various values for different attack parameters and highlight the impacts on important network entities. Afterward, we present the *Principal Component Analysis* (PCA) result to disclose correlations of metrics and parameters.

The work in this section is the result of our collaboration with colleagues from LORIA, University of Lorraine, Nancy, France, and especially Ph.D. student Xavier Marchal. We are the ones who propose scenarios that leverage multicast and bestroute forwarding strategy while our colleagues come up with the idea of the unsolicited provider. We share the workload of running experiments. After UTT finished the illustration of results and the PCA, both partners jointly analyze the whole collected data. The work in this section has been published as a full paper in the international conference, see [96].

3.4.1 Investigated Topology and Entities' Behavior

To study the CPA's impact, we implement the topology illustrated in Figure 3.7, using the tools presented in Section 3.2. We believe that our topology, together with behaviors of all implied components, is sufficient to achieve the main purpose of proving the feasibility and featuring CPA impacts for the following reasons. First, it exhibits the general role of the different nodes / functions involved in CPA and reflects a typical network operator structure: an access router R1 where both good and bad clients are connected, a core router R2 and an access router R3 that provides the access and caching system to the legitimate provider. This topology enables us to highlight the effect of CPA for all involved entities.

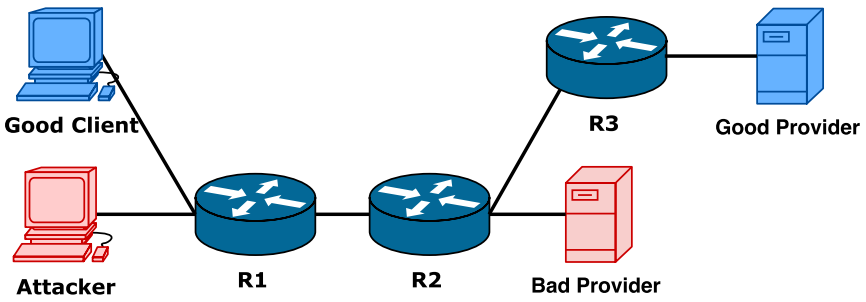


FIGURE 3.7: Use-case topology for Content Poisoning Attack

Secondly, the user behavior is as close as possible from reality. We assume that clients always issue *Interest* packets for fresh and latest *Data* of the content thanks to *MustBeFresh* and *ChildSelector* fields in the *Interest* packet (see Section 1.3.1). Legitimate users send requests over the full range of content and only accept good *Data*. When receiving bad *Data*, they re-issue another *Interest* whose *Exclude* field is set to avoid receiving the same bad *Data* once again. Bad clients also request for the whole content popularity, but for the names of poisoned content, they act oppositely by excluding good *Data*, thus favoring the dissemination of bad provider's *Data*.

Thirdly, there is only one good provider, who replies to *Interests* for the whole content popularity. Such a case stands for one of the worst cases where legitimate

content availability is limited to one route. Also, the legitimate provider is located farther from R2 than the malicious one, leading to a longer delay in the content delivery. Such a delay for later requests, however, will be significantly reduced once *Data* is cached in R1. Besides, due to the valid registration, we consider that the path towards the legitimate provider always has a lower cost than that towards the bad provider[8]. Also, the good provider will automatically update its *Data* when *Data*'s *FreshnessPeriod* (see Section 1.3.1) expires.

Moreover, note that in a network aiming to optimize the delivery efficiency and support content availability, traditional secure routing is too restrictive and consequently not acknowledged as a realistic deployment case of NDN [71]. Hence, we argue that content providers can publish their contents under some registered prefixes. Such procedure might be simple for the legitimate provider but is not straightforward for an attacker to control several malicious providers/prefixes. Hence our topology only has one bad provider which is located near the core router R2. This bad provider only replies to *Interests* for the contents it targets. To challenge the legitimate provider, we consider that the attacker always selects the most popular contents to poison, given a popularity distribution. Such knowledge can easily be gained, e.g., in case of web traffic through publicly available information about website popularity ranking. Besides, since bad *Data* is useless after being excluded, the bad provider must update its *Data* whenever the current bad *Data* start being excluded, to maintain the attack persistence. Updating a bad *Data* can be simply publishing a newer version component in the content name (see Section 1.3.1). Also, the attacker is likely to set the bad *Data FreshnessPeriod* to a high value so that bad *Data* will linger longer, increasing the number of victims who issue naive requests (i.e., without exclusion for bad *Data*).

It is noteworthy that we do not consider the NDN/IP gateway in this topology. Since IFA is well-studied in the literature, investigating it in a more realistic deployment would bring more novelty. Meanwhile, CPA is an NDN-specific threat that is not well-studied. It is lack of a detailed attack scenario. Thus, CPA must first be studied in a pure NDN environment.

3.4.2 Content Poisoning Attack Scenarios

In this subsection, we describe three attack scenarios highlighting weaknesses in NDN protocol design and implementation that can be exploited to insert bad *Data*, including (1) *unsolicited* provider, (2) *multicast* forwarding and (3) *bestroute* forwarding. Note that we do not consider the case in which the attacker injects the bad

[8]It is noteworthy that the route's cost rather reflects the trust given to a route and does not necessarily depends on the link's delay or the number of hops.

Data through a compromised router (see Section 2.3.1) because we argue that taking control over a network element is hardly possible for a user in a real operating context. Meanwhile, it is more feasible to leverage end-hosts to perform the attack from both the provider and user sides. As a result, our three scenarios consider cases where CPA is carried out by a single unsolicited provider (for scenario 1) or by a collaboration between bad providers and clients coordinated by a single attacker (for scenarios 2 and 3). It is noteworthy that the last two scenarios depend on the router's forwarding strategy, which is not set by the user.

Unsolicited Scenario

Unsolicited scenario exploits an NFD (v0.4.1) implementation flaw which is discovered by our colleagues from LORIA (see [96]). For each incoming *Interest*, NFD keeps track of the content name, incoming and outgoing faces in a corresponding PIT entry. NFD considers a *Data unsolicited* if the *Data* does not correspond to what the NFD expected. More specifically, there are two possibilities: (1) the *Data* does not match any PIT entry or (2) the *Data* does not arrive at the same face to which the corresponding *Interest* has been forwarded. When this implementation flaw is discovered, NFD only checks whether the *Data*'s name matches any existing PIT entry and ignores the second possibility. As a consequence, unsolicited *Data* from faces without a legitimate established route can be considered valid, hence consume PIT entries. Exploiting this flaw, the attacker can deploy an unsolicited provider by taking control of any client that connects to a router along the path between clients and the good provider (R2 in Figure 3.7). The unsolicited provider then sends bad *Data* with random popular content names to R2 so that they can match R2's pending *Interests* and be inserted in the cache.

One should note that the bad provider is blind to R2's pending *Interests*, but can still get information on recently requested contents, for instance, by performing the time analysis attack [4, 82] to spoof the cache. This feature, however, is not investigated in our work because our main focus is to demonstrate the feasibility and to feature the impact of CPA in a real deployment. When an *Interest* arrives at R2, a race condition begins between the good and the bad providers. Only the first matching *Data* arriving at R2 is accepted and resolves the PIT entry, while all the later *Data* are dropped. As a consequence, a malicious *Data* has a higher chance to match an *Interest* for a targeted content if it arrives at R2 during the time window $[t_{receive}; t_{receive} + t_{gpDelay}]$; where $t_{receive}$ is the time when R2 receives the *Interest* and $t_{gpDelay}$ is the delay of corresponding *Data* from the good provider. Since estimating this time window is hardly feasible for the bad provider, it can merely send bad *Data* for targeted contents regularly at a sufficient rate to increase the success rate of the

attack.

Multicast Scenario

Multicast is one of the forwarding strategies available in the current NFD. When a router using this strategy receives an *Interest*, it forwards it to all eligible faces registered in the corresponding FIB entry. A next-hop face is *eligible* as an upstream if it matches the following conditions: (1) the router is not waiting for a response from this face; (2) the scope is not violated and (3) this face is different from incoming faces recorded in the corresponding PIT entry of the *Interest*. While the CPA is carried out by the sole effort from the bad provider in the *unsolicited* scenario, the *multicast* scenario requires collaborating clients to insert bad *Data* in caches. Specifically, bad clients regularly send *Interests* only for the targeted content name but exclude the current copies of good *Data* in order to bypass caches. This forces R1 to forward the requests toward R2, which then forwards them to both the legitimate and bad providers, according to the *multicast* forwarding strategy. Consequently, a *Data* packet is returned by both providers. However, due to the shorter delay, the bad *Data* arrives at R2 first, resolves the corresponding PIT entry and is cached in R2 and then in R1. Meanwhile, because of the longer delay, the legitimate *Data* arrives R2 later and is dropped since the PIT entry has already been resolved.

Bestroute Scenario

Bestroute is the default forwarding strategy used by the current NFD [5]. A router running this strategy forwards the incoming *Interest* to the face with the lowest cost in the corresponding FIB entry. If there are two faces with the same lowest cost, the router uses the first one registered. Once an *Interest* is forwarded, the NDN router will suppress any similar *Interest* with the same *Name*, *Selectors* but different *Nonce* (a random value to avoid *Interest* loop, see Section 1.3.1) if it arrives during a *retransmission suppression interval*. The purpose of such an interval is to prevent a malicious entity from retransmitting too frequently. After this interval, a similar *Interest* received is considered as a valid retransmission and is forwarded to the next lowest-cost face that has not been previously used, hence opening the door for the bad provider to act. When all registered faces in the FIB entry have been used, it is forwarded again to the first-used face. For NFD v0.4.1, the initial interval is $10ms$ and is doubled after each retransmission until it reaches a maximum of $250ms$, meaning that a user cannot retransmit the same *Interest* (same *Name*, *Selectors*, different *Nonce*) more than 4 times per second. Thanks to the collaboration with malicious clients, the

attacker can generate additional similar *Interests*, forcing router R2 to use the other route towards the bad provider, hence pulling bad *Data* to caches in R2 and R1[9].

3.4.3 Experimental setup

We deployed the topology described in Figure 3.7 using NFD on Docker containers, one entity per container. We configured an artificial latency on both the good and bad providers to emulate the delay between a server and its users in the real Internet. In our experiments, all clients and providers are remotely connected to NFD nodes so that caches are only present in NDN routers. The constant values shared by all our experiments are listed in Table 3.2 and most of the values are motivated by [79]. Each experiment lasts 600*s*, with first 300*s* spent without attack followed by 300*s* with CPA. Finally, each experiment is repeated 5 times, and all the curve points depicted subsequently stand for the average of all measurements over the last 300*s* and over the 5 repetitions, bounded with a 95% confidence interval.

TABLE 3.2: CPA Experimental constants

Constant	Value
Number of contents [79]	10000
Good *Data* FreshnessPeriod	90*s*
Bad *Data* FreshnessPeriod	120*s*
Good provider latency to R2	100*ms*
Bad provider latency to R2	10*ms*
Users average *Interest* rate by default	10 Interests/s
Zipf distribution factor	1.5
Maximum number of exclusions	700
NFD version	v0.4.1, latest untill October 4^{th}, 2016

Client Behavior

The good client's and attacker's behavior are implemented using jNDN[10] - an NDN client library written in Java. Apart from behaviors described in Section 3.4.1, the number of *Interests* sent by a client per second will follow a Poisson distribution,

[9] It is noteworthy that when this thesis report is being written, the NDN packet format specification has been updated to version 0.3. In this version, the *Exclude* field of *Interest* has been removed.

However, we have examined with the updated version of NDN library and found out a scenario that succeeding injecting bad *Data*, even without the *Exclude* field. In this scenario, the malicious client know the full content name of *Data* provided by the bad provider (e.g. by an agreement on naming convention). Malicious *Interests* requesting such content name can bypass existing cached *Data* in routers and can only by matched by bad provider. As such, the attacker pulls bad *Data* and the routers will cache it. Meanwhile, the legitimate clients have neither knowledge of content names from the legitimate provider nor the *Exclude* field. The situation gets worse for the legitimate clients because now they cannot avoid malicious *Data* once it is cached.

[10] See: github.com/named-data/jndn

and the requested content name is selected from a Zipf distribution. Such a model has been used by previous works in ICN [97, 98]. Moreover, it is noteworthy that the client cannot exclude NDN names indefinitely, due to the limitation of packet size in NDN. Therefore, it will consider a content unreachable when the *Exclude* size reaches a defined value and will ask this content later with an empty *Exclude* field. After the client receives a legitimate *Data*, it memorizes excluded names to use later when it requests for the same content again. Such behavior is safer and more realistic than the memoryless way (i.e., no memory of previous exclusions) because clients hardly know whether previously received bad *Data* still exist in the network.

Attack Parameters

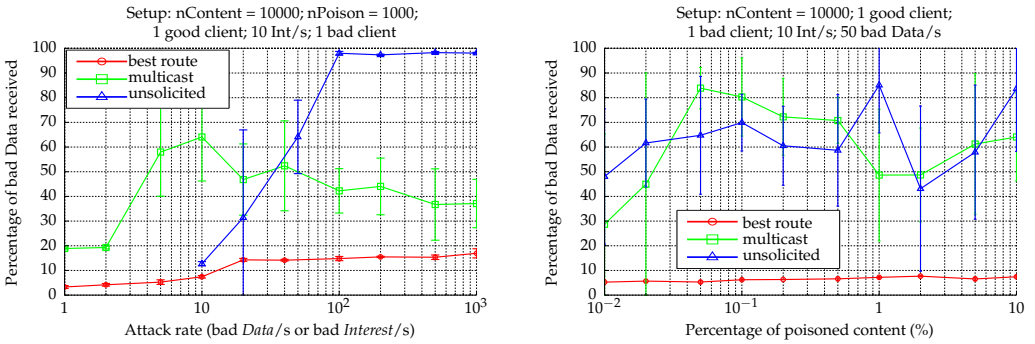
We consider the attack rate as, the main parameter that impacts the attack success. For *unsolicited* scenario, the attack parameter is the number of bad *Data* sent per second by the *unsolicited* provider, varies in the range $[10, 1000]$ following a logarithmic scale and is set to 50 Data/s as a default value. We do not run experiments for the range from 1 to 10 (i.e., less than good Interest rate) since the CPA barely succeeds with such a small attack rate. One should note that for *unsolicited* scenario, the attacker is blind to legitimate *Interests*, hence needs to send *Data* with a high rate in order to opportunistically match those *Interests*. *Multicast* and *bestroute* scenarios have two attack parameters. The first is the average number of bad *Interests* sent per second. Its value varies in the range $[1, 1000]$ following a logarithmic scale and is set to 10 Interests/s as default value, i.e., equals the average *Interest* rate of the good client. The second attack parameter is the fraction of the most popular content poisoned by the attacker. For this parameter, the value varies in the range $[0.01, 10]$ percentage of the content popularity (equivalent to the range $[1, 1000]$ contents), following a logarithmic scale and is set to 1% (100 contents) as a default value.

3.4.4 Attack Phenomenon

In this subsection, we highlight the attack's impact on the client, the provider and routers' caches. For this subsections' figures, blue, green and red colors represent, respectively, *unsolicited*, *multicast* and *bestroute* scenario.

Impacts on the Legitimate Client

Figure 3.8a and Figure 3.8b depict the percentage of bad *Data* that a good client receives as a function of the attack rate and the number of poisoned contents, respectively. It is necessary to clarify that in order to calculate the percentage of bad *Data*, we only take into account *Data* packets of content names that are poisoned.



(A) Attack rate effect (B) Number of poisoned contents effect

FIGURE 3.8: Attack's impacts on the legitimate client

Figure 3.8a shows that in the *bestroute* scenario, the legitimate client suffers the least from CPA. It is because R2 always prioritizes the *Interests* forwarding to the legitimate provider due to its lower cost. Moreover, R2 only uses the other route if a retransmission arrives while waiting for the good provider's response, giving the bad provider a low chance to insert bad *Data*. Figure 3.8a also demonstrates that the damage just slightly increases when the attack rate is higher than that of legitimate traffic and remains almost the same for higher attack rates. Such a phenomenon can be explained by the retransmission suppression interval of NFD *bestroute* forwarding strategy. As mentioned in Section 3.4.2, for each retransmission, the interval will be doubled, eventually to the point that it is larger than the delay of the good provider ($> 100ms$). When it happens, good provider's *Data* will arrive at R2 before an *Interest* is retransmitted to the bad provider, limiting the chance a bad *Data* is inserted in R2's cache. The CPA, however, still succeeds in such scenario (with about 20% of bad *Data* received).

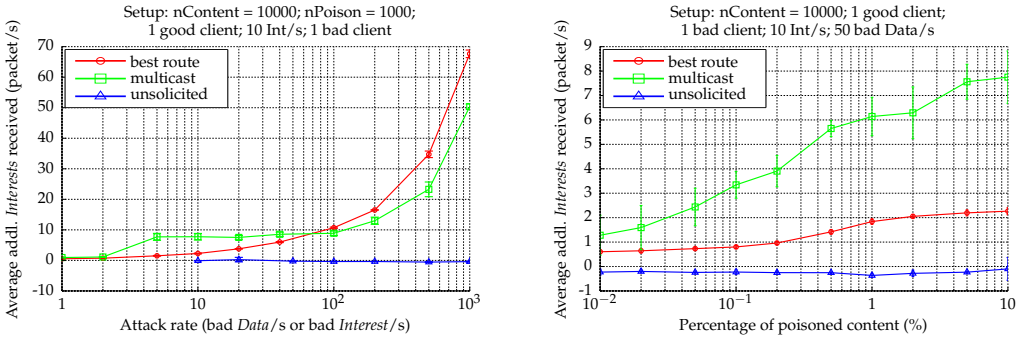
Meanwhile, the *multicast* and *unsolicited* scenarios poison the legitimate client more effectively. Especially for the unsolicited provider with a high attack rate, nearly 100% of *Data* received are bad. Under high attack rate, incoming *Interests* in R2 are mostly matched by fresh and new unsolicited bad *Data*, despite the client's exclusion. On the other hand, the effect on legitimate clients does not exhibit an obvious trend in the multicast scenario, but it is clear that with high attack rate, the effect is more severe than *bestroute* and less severe than *unsolicited* scenario. When the attacker sends too many *Interests*, more good *Data* are pulled to R1 - the caching system located in between the legitimate provider and R2. Such an entity gives good *Data* an advantageous delay as compared with that of the bad provider. Hence, when an *Interest* on R2 is forwarded to both routes, good *Data* at R1 are more likely to arrive at R2 sooner.

Figure 3.8b shows that even when the attacker changes the number of content he

targets, the *bestroute* scenario still maintains its protection against CPA. Also, for *multicast* and *unsolicited* scenarios, the number of target contents does not have a clear impact on the legitimate client. This implies that if an attacker wants to improve the damage on legitimate clients, he should not put much effort into expanding the number of target contents, but rather focus on a few highly popular ones.

Impact on the Content Provider

Figure 3.9a and Figure 3.9b depict the CPA impact on the legitimate provider as a function of the attack rate and the number of targeted content, respectively. The selected metric is the difference $(avgInt_{after} - avgInt_{before})$; where $avgInt_{after}$ and $avgInt_{before}$ are the average number of Interests per second that the provider receives after and before the attack, respectively.



(A) Attack rate effect (B) Number of poisoned contents effect

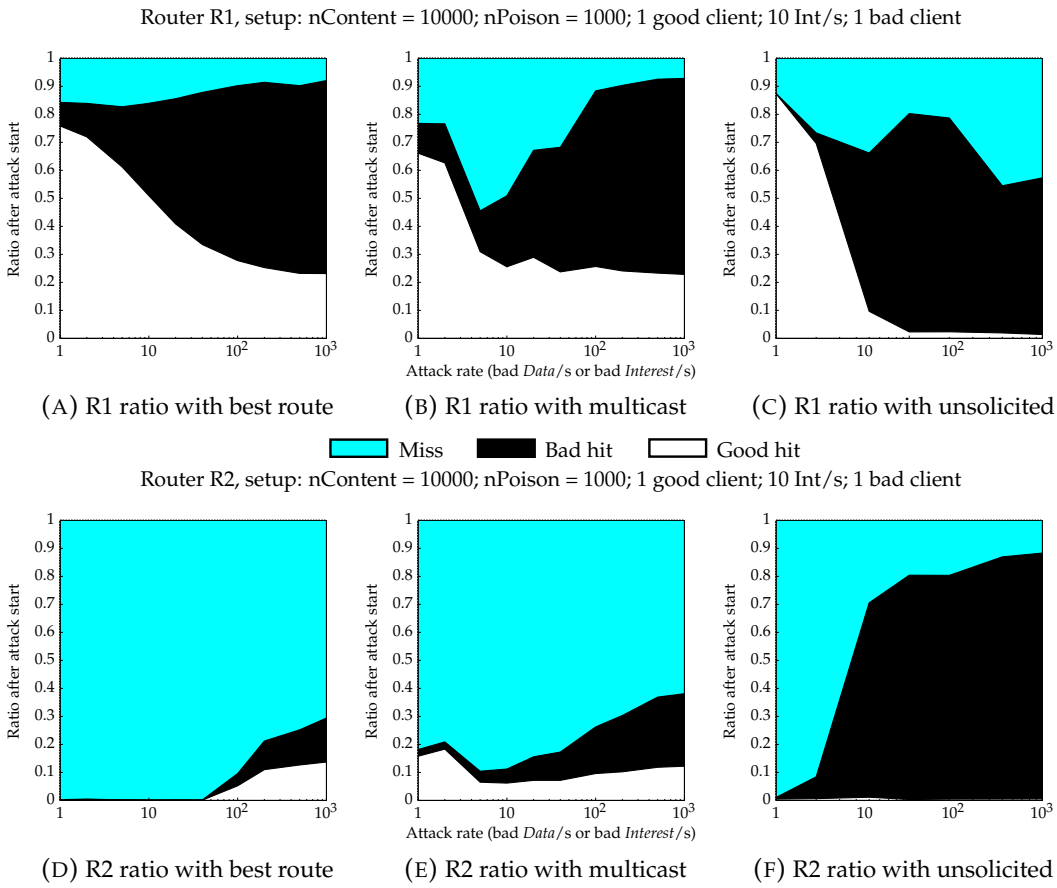
FIGURE 3.9: CPA's impacts on the legitimate content provider

Figure 3.9a and Figure 3.9b exhibit that the *unsolicited* scenario has no impact on the legitimate provider regardless the value of attack parameters. Also, one can notice that the metric in this scenario is near and even slightly below zero, implying that the legitimate provider even received less *Interests* under CPA as compared to that before the attack. It is because most of the *Interests* are already satisfied at R2 by unsolicited fresh *Data*, hence cannot reach the good provider. *Interests* with exclusions can bypass the cached bad *Data* and reach the good provider, even though they will soon be satisfied by newly issued unsolicited *Data* before good *Data* arrives.

On the other hand, Figure 3.9a shows that for the *bestroute* and *multicast* scenarios, the effect on the provider is highly related to the attack rate and has a similar trend. Since most of the *Interests* issued by bad clients will be forwarded to the legitimate provider, it must handle more requests when the attack rate increases in the *bestroute* or *multicast* scenarios. Figure 3.9b shows that the *bestroute* and *multicast* scenarios exhibit a nearly linear growth of the provider's burden with the range of poisoned content. Indeed, when a bad user targets a wider range of content, each

Interests is unlikely to be aggregated in an existing PIT entry, and consequently, more *Interests* are forwarded to the provider.

Impact on Routers' Cache



(A) R1 ratio with best route (B) R1 ratio with multicast (C) R1 ratio with unsolicited

(D) R2 ratio with best route (E) R2 ratio with multicast (F) R2 ratio with unsolicited

FIGURE 3.10: Attack rate effect on caches of core router R2 (a)(b)(c) and access router R1 (d)(e)(f)

Figure 3.10 illustrates the impact of the CPA on the most important routers of our topology: access router R1 and core router R2. Evaluating the effect on caches is important considering the central role of caches to avoid network congestion in the NDN architecture, as well as the high amount of resources dedicated to caching. It is even more critical if we consider that attackers can exploit network caches to maintain and amplify the pollution at a reduced cost. The two rows in Figure 3.10 respectively illustrate measurements for R1 and R2. Each dedicated sub-figure in each row shows the average proportion of good hit (i.e., a cached good *Data* is retrieved), bad hit (i.e., a cached corrupted *Data* is retrieved) and miss (no *Data* is retrieved) in routers' cache according to the attack rate for each attack scenario.

We first consider the access router R1. In our setup, R1 is more prone to cache *Data* as we can observe a very high proportion of good hits at lowest attack rates.

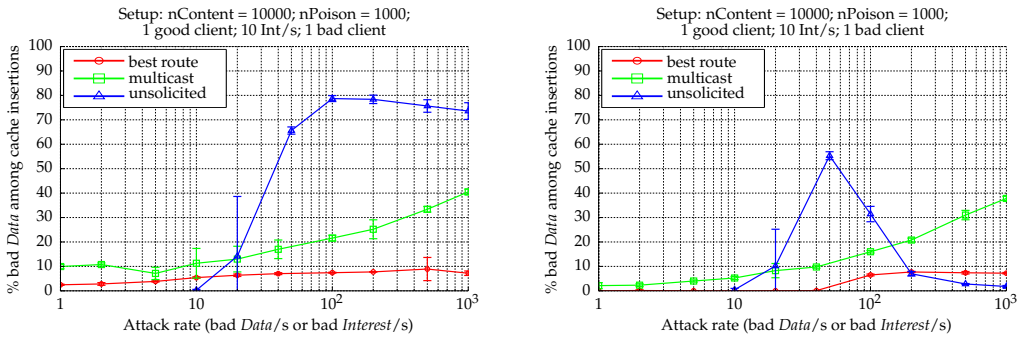
Then, all attacks show a similar trend with the rate of good hits decreasing to the benefit of bad hits when the attack rate increases. This effect is however more progressive for the *bestroute* (Figure 3.10a) than for *multicast* (Figure 3.10b) which only increases the proportion of miss at lowest attack rates, before increasing the bad hits when the attack rate reaches 500 *Interests* per second. In both cases, at the highest attack rates, the proportion of good hits is small ($\simeq$ 20%) while the proportion of bad hits is very high ($\simeq$ 70%), and the miss ratio is under (10%), making both attacks very successful on R1. The trend is made quicker by the unsolicited provider scheme, but the maximum attack rate ends with a different ratio from the other two. There are no more good hits in Figure 3.10c but the cache hits are balanced between 60% of bad hits and 40% of miss. However, lower attack rates can achieve better results for this specific attack with a proportion of bad hits going up to 80% for 100 bad *Data* per second.

Looking at the general aspect of the curves in the second row of Figure 3.10, we can already see that the cache of the core router R2 shows a different behavior from the access router R1 when exposed to the same attacks. One can notice that R2 cache is not useful as we can observe a huge proportion of miss even at the lowest attack rates. It is because, in such a small topology and with our configuration, the access router R1 is sufficient to answer most of *Interests* from clients. Since all *Data* has the same *FreshnessPeriod* (see Section 1.3.1), cached *Data* in R1 and R2 will becomes stale at the same time. Thus, all *Interests* that cannot find fresh *Data* in the access router R1 also will not find fresh ones in cores router R2, leading to a high proportion of miss. Figure 3.10d shows that the *bestroute* attack does not affect R2 cache at low rates and only has a limited impact with only 10% of bad hits with the highest attack rate. This can be explained by the fact that the client will retrieve the majority of polluted *Data* from R1 and consequently, the majority of *Interest* forwarded to R2 by R1 (i.e., after a cache miss on R1) already exclude the names of most of the bad *Data* preventing bad hits in R2's cache. In the case of the *multicast* attack (Figure 3.10e), the effect is globally similar to *bestroute* with a slight increase in magnitude. As the attack rate increase, the proportion of bad hits increases from 2% to 25%. The *multicast* scheme offers better opportunities for the bad provider to answer back with polluted *Data* which can explain this result. Finally, the unsolicited provider attack exhibits a different behavior on R2 in Figure 3.10f. The ratio of bad hits increases rapidly with the attack rate, achieving 80% of bad hits for an attack rate of 20 bad *Data*/s. After that, increasing the attack rate has less impact on the bad hit ratio only increases by 5% with 10 times more aggressive attack. This attack, if very effective in propagating pollution through the cache as the flow of newly generated bad *Data*, easily enters the cache by consuming legitimate *Interests*. The attacker is always one step ahead of the client excluding names also explains the lower proportion of miss.

To conclude on the attack level, we can state that the unsolicited scenario does a better job in polluting the routers in the path toward the client, but its efficiency decreases when it traverses routers. An alternative scenario with the unsolicited provider connected to the access router, reproducing Figure 3.10f closer to the clients, would make it even more difficult for them to obtain good Data. We can also notice that the larger amount of miss on R1 compared to the other scenarios have good chances to end up as bad hits on R2. Regarding the *bestroute* and *multicast* scenarios, they have a less significant impact on the core router R2 but still have a high impact on the good hit depletion on access router R1.

Caching Resources Wasted on Bad Data

Figure 3.11b and Figure 3.11a show the percentage of bad *Data* among cache insertions at router R1 and R2, respectively, representing the cache corruption effect of CPA. They perfectly align to the preceding results. For instance, the maximum efficiency of the *unsolicited* attack at 100 bad *Data* per second matches the attack rate corresponding to the highest ratio of the bad hit of R1 in Figure 3.10c. Similarly, we can notice a peak in attack efficiency at the rate of 50 bad *Data* per second for the *unsolicited* scenario on R2, which matches the start of the flat behavior on Figure 3.10f. The *bestroute* and *multicast* attacks do not show such local maximum of their efficiency. The *bestroute* attack has almost a constant efficiency per *Interest* regarding the attack rate, while the *multicast* attack is even more efficient per *Interest* when the attack is more aggressive. Overall, we conclude that the *unsolicited* attack is the most efficient per packet sent to pollute routers and should quickly get fixed by the NDN community.



(A) Router R1 (B) Router R2

FIGURE 3.11: Resources wasted on routers

3.4.5 Attack Footprint with Principal Component Analysis

Finally, as a follow-up in the analysis of multiple-variable data, we performed a PCA on our overall dataset to reveal correlations of all metrics, parameters and attack scenarios. The dataset contains measurements of 11 metrics from attack scenarios only, without any legitimate traffic. Each metric has been normalized individually, by subtracting the empirical mean and dividing by the standard deviation. We recall that the principal components stands for the eigenvectors of the covariance matrix, and represent the axes onto which the projection of the data reach a maximal variance. In other words, PCA allows building the orthonormal basis with the least mean squared error.

TABLE 3.3: Values of the two firsts principal components with the label of associated metrics.

Location	**Metric**	1^{st} **component**	2^{nd} **component**
Provider	# additional Interests	-0.1618	0.4252
Access router R1	% good hit	-0.3554	-0.24
	% bad hit	0.2976	-0.2521
	% miss	0.1227	0.5727
	Resource waste	0.4018	-0.0401
Core router R2	% good hit	-0.0778	0.3178
	% bad hit	0.3913	-0.144
	% miss	-0.3891	0.1085
	Resource waste	0.3036	-0.1963
Client	% bad Data received	0.3243	0.3626
	# bad Data	0.2731	0.2549

The values of the first two components, that account for 80.5% of the total error, are provided as rows of the Table 3.3. One can remark that the projections are not centered around zero since this analysis has been performed using measurements from attack scenario only, without any legitimate traffic. The table shows that the first component (accounting for 56.5% of the total error) is featured by a high impact on bad hit ratio, resources wasted for the bad hit of both routers, together with a high number and rate of bad *Data* to the good client. As such, this first component represents the main expected impact of the CPA with the injection of bad *Data* in routers' cache.

Meanwhile, the second principal component (accounting for 24% of total error) shows a similar impact on the number of bad *Data* to the client, but a much higher impact on the miss ratio of the access router, a lesser extent on the core router and the additional traffic to the provider. This exhibits the side effect of the CPA that prevents the routers from caching good *Data*, hence creating a higher rate of miss hit and traffic to the legitimate provider. Moreover, it is noteworthy that those three

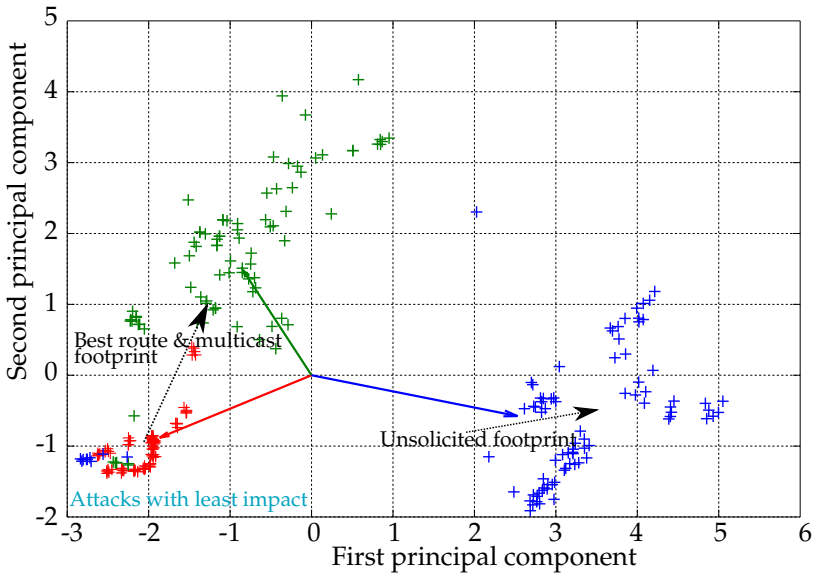


FIGURE 3.12: Projections of the measurements on the two first principle components.

attack scenarios have similar impacts on the number of bad *Data* received by the legitimate client as it is their primary goal.

Figure 3.12 now presents the projection of individual measurements on these first two components. The solid line arrow represents the mean projection of each scenario, and the '+' represent the projections of individual experiments. The black dashed arrows denote the direction when the attack rate increases. The figure clearly shows that the components distinguish the *unsolicited* scenario from the *multicast* and *bestroute* scenarios that exhibit the same operating mode. In the figure, the cyan circle points out the projections of the experiments with least attack impact (lowest attack rate). Similarly, the dashed arrow indicates the direction toward which the results move when the attack strength increases.

The figure also shows that the *unsolicited* scenario has a specific footprint mainly captured by the first principal component. As expected, in this case, the unsolicited *Data* creates a high rate of the bad hit. It also shows that the *bestroute* and *multicast* scenarios have the similar impact when the attack rate increases, mostly featured by the second principal component. Indeed, those scenarios create a higher rate of miss hit as the legitimate clients try to avoid the bad *Data* from caches. On the contrary, the bad client tries to prevent the caching of good *Data*. This also explains the higher number of *Interests* to the provider. Finally, one can remark that the *bestroute* and *multicast* attack scenarios are also, to a lesser extent, characterized by the first principal component. However, the *bestroute* scenario exhibits a much smaller impact.

To conclude for this section, we state that CPA is feasible in reality by demonstrating three attack scenarios that exploit weaknesses in NDN protocol and implementation to insert bad *Data* to routers' cache. Different from IFA which is mostly featured by a singular satisfaction ratio (see Section 2.2.1), experiments show that CPA impacts simultaneously various metrics on different network entities.

3.5 Conclusion

This chapter addresses the literature's limitation of extensively using simulated results (see Section 2.6) by characterizing NDN crucial attacks in the real deployment. Given the technical maturity of NDN protocol with its forwarding daemon (NFD), we argued in this chapter that NDN attacks must be studied in a real deployment context. As such, we reconsider the feasibility and feature the impact of two crucial NDN attacks – IFA and CPA. The IFA is studied in an environment where IP and NDN coexist to deliver the HTTP application to web users. Although the attacker lacks control over *Interests*, and the *NACK* packet is already implemented in NFD, we proposed an IFA scenario that succeeds in degrading user's experience when accessing websites. For CPA, we introduced three attack scenarios that exploit weaknesses in NDN protocol and implementation to inject malicious *Data* in the caching system. Results in our NDN native testbed exhibit CPA's impacts on essential network entities. While IFA can be characterized by the satisfaction ratio (i.e., the number of *Data* over the number of *Interests*) as a majority of related works (see Section 2.2.1), the impact of CPA is more elusive and can hardly be featured by a single metric on a particular entity. Consequently, accurate detection of such attack would require taking into account multiple metrics (e.g. number of cache miss, cache hit) at the same time.

Chapter 4

Design of Micro Detector Using Hypothesis Testing Theory

Contents

The envisaged security solution for NDN requires micro detectors to capture security anomalies. As such, the reliability of these micro detectors significantly impacts the accuracy of the overall proposal. However, existing detections for NDN security threats are mostly unreliable in terms of guaranteeing a prescribed rate of false alarms, leading to waste of network resources or accidental penalizing legitimate clients (see Section 2.6). To overcome such a drawback of existing works, this chapter addresses a micro detector design by leveraging the hypothesis testing theory. Such method allows us to calculate a threshold that guarantees a desired *Probability of False-Alarm* (PFA) and to establish the theoretical detection performance.

Because the IFA can be featured by a single metric (see Chapter 3), it is selected as a use-case for our micro detector. First, we introduce the basic framework of hypothesis testing. Then, we define the detection problem regarding statistical hypothesis and design a *Likelihood Ratio Test* (LRT) for the ideal case when the packet loss rate is known beforehand. Next, we address the realistic case when the loss rate is unknown with a parametric model upon which a *Generalized LRT* (GLRT) is built. The detection accuracy is then enhanced by developing into a sequential version based on the snapshot GLRT. The overall performance is evaluated with numerical results from both simulation and real NDN testbed (see Section 3.3.4). A part of the work in this chapter has been published as full papers in international conferences, see [99, 100].

4.1 Basic Framework of Hypothesis Testing Theory

The method we used to design our detection is based on statistical hypothesis testing theory with *Neyman-Pearson two-criteria* approach that simultaneously aims at guaranteeing a desired false-alarm rate while maximizing the detection power. The input of hypothesis testing is a sample $\mathbf{Z}_N$, $\mathbf{Z}_N \in \mathcal{Z}$ drawn from a probability distribution $\mathcal{P}_\theta$. This sample is a set of N empirical realizations of a random variable z. A *statistical hypothesis* $\mathcal{H}_j$ refers to a set of parameters vectors Θ_j. Each vector θ in this set defines a possible probability distribution $\mathbb{P}_\theta$ of $\mathbf{Z}_N$ [101]:

$$\mathcal{H}_j \;=\; \{\, Z_N \;\sim\; \mathbb{P}_\theta,\; \theta \;\in\; \Theta_j \}.$$

The hypothesis $\mathcal{H}_j$ is called *composite* when Θ_j contains more than one value. Otherwise, it is referred to as a *simple* hypothesis. In the usual case of binary statistical tests, there are two hypotheses: (1) the *null hypothesis* $\mathcal{H}_0$ and (2) the *alternative hypothesis* $\mathcal{H}_1$. $\mathcal{H}_0$ is usually the normal case and $\mathcal{H}_1$ is usually the abnormal case that we want to detect. A *statistical test* δ between two hypotheses $\mathcal{H}_0, \mathcal{H}_1$ is a subjective and measurable mapping from the sample space $\mathcal{Z}$ to the set of hypotheses [101]:

$$\delta :\; \mathcal{Z} \;\rightarrow\; \{\mathcal{H}_0, \mathcal{H}_1\}.$$

In order to design a good statistical test with the Neyman-Pearson approach, there are some key concepts which should be aware of: (1) probability of false alarm, (2) detection power, (3) Likelihood Ratio and (4) the uniformly most powerful test. The set Θ_0 defining $\mathcal{H}_0$ contains many parameters θ_0. For each of these parameters, there is a probability that the test δ rejects the null hypothesis $\mathcal{H}_0$ while it is actually true. The greatest among these probabilities is called the PFA of the test δ, denoted

by:

$$\alpha(\delta) = \sup_{\boldsymbol{\theta_0} \in \boldsymbol{\Theta}_0} \mathbb{P}_{\theta_0}\Big[\delta(\mathbf{Z}_N) = \mathcal{H}_1\Big],$$

Meanwhile, the *detection power* of the test δ, for a parameter $\theta_1 \in \Theta_1$, is the probability that $\mathcal{H}_1$ is detected correctly, denoted by:

$$\beta(\theta_1, \delta) = \mathbb{P}_{\theta_1}\Big[\delta(\mathbf{Z}_N) = \mathcal{H}_1\Big].$$

For a prescribed false alarm probability α_0, we define the class of test $\mathcal{K}_{\alpha_0}$ containing all the tests whose PFA is lower than α_0:

$$\mathcal{K}_{\alpha_0} = \Big\{\delta : \alpha(\delta) \leq \alpha_0\Big\}.$$

A *Uniformly Most Powerful* (UMP) test $\tilde{\delta}$ in the class $\mathcal{K}_{\alpha_0}$ is an universally optimal test regardless of the parameter value, i.e., providing the highest power under all the parameters $\theta_1 \in \Theta_1$:

$$\forall \delta \in \mathcal{K}_{\alpha_0}, \quad \forall \boldsymbol{\theta}_1 \in \boldsymbol{\Theta}_1, \quad \beta(\boldsymbol{\theta}_1; \delta) \leq \beta(\boldsymbol{\theta}_1; \tilde{\delta}).$$

The idea of the Neyman-Pearson approach is to design a test in the class $\mathcal{K}_\alpha$ that can warrant a pre-defined false alarm probability α and maximizes the test power $\beta(\theta_1, \delta)$. In the case of simple hypotheses, according to the Neyman-Pearson lemma [101, Theorem 3.2.1], the most powerful test $\tilde{\delta}$ is the LRT:

$$\tilde{\delta}(Z_N) = \begin{cases} \mathcal{H}_0 & \text{if } \Lambda(Z_N) = \frac{f_1(\mathbf{Z}_N)}{f_0(\mathbf{Z}_N)} < \tau, \\ \mathcal{H}_1 & \text{if } \Lambda(Z_N) \geq \tau \end{cases} \tag{4.1}$$

in which $\Lambda(\mathbf{Z}_N)$ is the likelihood ratio and f_j is the probability density of $\mathbb{P}_{\theta_j}$, j = 0,1. The LRT can be transformed by applying a monotone function to both side of the inequality in (4.1). The threshold τ is the solution of the equation:

$$\mathbb{P}_0\left[\Lambda(Z_N) \geq \tau\right] = \alpha.$$

Meanwhile, in the case of composite hypotheses, the UMP test barely exists in reality. The testing theory for this type of hypotheses is only well-developed for some particular cases which we will introduce later in this chapter.

4.2 Interest Flooding Attack Detection Problem

Sections 4.2–4.4 focus on the local instantaneous IFA detection, i.e., a detection is performed at each instant for a given face of a specific router. For the sake of clarity, the notation is simplified by omitting the index of face and router. In this section, we represent the IFA detection problem in the form of statistical hypotheses and highlight the difficulties of the testing problem. Our detection problem also involves the satisfaction ratio – the statistic used by a majority of previous works on IFA detection. It is noteworthy that we examine the IFA in the same experimental context that we have used to feature its impacts in Section 3.3.

4.2.1 Definitions

Let i_t and d_t, respectively, be the number of incoming *Interests* and outgoing *Data* packets at an instant t measured for each router's face. The ratio (d_t/i_t) is the satisfaction ratio that is widely used in the IFA detection literature. Ideally, for each face, each incoming *Interest* should result in one outgoing *Data*. However, as in any networks, a part of the packets could be lost. Hence, we define

$$\ell_t = 1 - \frac{d_t}{i_t} = \frac{i_t - d_t}{i_t} \tag{4.2}$$

the measured ratio of packet loss (unresolved *Interests*) at an instant t. In addition, following the model proposed in [53, 54], it is assumed that, at an instant t, all *Interests* have the same probability of being unresolved, denoted as $p_t \geq 0$. Under the normal situation, such a probability should correspond to the expectation of measured packet loss rate, i.e., $\mathbb{E}(\ell_t) = p_t$. As a result, the number of *Data* packets received d_t follows a binomial distribution $\mathcal{B}(i_t, 1 - p_t)$ with expectation $\mathbb{E}(d_t) = i_t(1 - p_t)$. When IFA occurs, a significant number of *Interests* is sent to the pirate server which intentionally delays its response, resulting in an abrupt increase of the packet loss rate ℓ_t at instant t.

For the sake of definition, let $i_t^\star$ and N_{a_t}, respectively, be the number of legitimate *Interests* and malicious *Interests* received at an instant t during the attack. Thus, IFA can be characterized by an increase in the number of incoming *Interests*:

$$i_t = i_t^\star + N_{a_t}. \tag{4.3}$$

It is noteworthy that distinguishing legitimate *Interests* $i_t^\star$ from the whole *Interests* flow i_t is impossible in practice. Moreover, because the malicious *Interests* N_{a_t} are responded with delay, the expectation of the instantaneous packet loss rate is increased

as follows:

$$a = \mathbb{E}(\ell_t) - p_t > 0. \tag{4.4}$$

Besides, given that only legitimate *Interests* are responded on time, the expected number of *Data* received at a given face will remain the same, regardless of IFA:

$$\mathbb{E}(d_t) = (1-p_t)i_t^\star = (1-p_t-a)(i_t^\star + N_{a_t}), \tag{4.5}$$

$$\Leftrightarrow a = \frac{(1-p_t)N_{a_t}}{i_t^\star + N_{a_t}} = \frac{(1-p_t)N_{a_t}}{i_t}. \tag{4.6}$$

4.2.2 Detection Problem Statement

According to the definitions in Section 4.2.1, the detection problem against IFA consists in choosing between two hypotheses:

- $\mathcal{H}_0$: "d_t *is consistent with what is expected from* i_t *and* p_t",
- $\mathcal{H}_1$: "d_t *is significantly lower than what is expected from* i_t *and* p_t".

Those two can be written formally as the following statistical hypotheses:

$$\begin{cases} \mathcal{H}_0\ (p_t, N_{a_t} = 0) : d_t \sim \mathcal{B}\left(i_t, 1-p_t\right), \\ \mathcal{H}_1\ (p_t, N_{a_t} > 0) : d_t \sim \mathcal{B}\left(i_t - N_{a_t}, 1-p_t\right). \end{cases} \tag{4.7}$$

Note that for a random variable $x \sim \mathcal{B}(n,p)$, the expectation $\mathbb{E}(x) = np$ and the variance $var(x) = np(1-p)$. We focus on the Neyman-Pearson bi-criteria approach that simultaneously aims at guaranteeing a prescribed PFA while maximizing the detection power.

The hypotheses formulated in Eq. (4.7) highlight two main difficulties of the testing problem. First, the distribution is discrete, which makes it difficult to calculate an exact threshold that can gurantee a desired PFA. Secondly, the attack payload N_{a_t} and the expected packet loss rate p_t are unknown in practice. Designing a test for such hypotheses and establishing its performance are much more difficult than for ones that depend on a single parameter.

4.3 Optimal Likelihood Ratio Test for Known Packet Loss Rate

In this section, we first address the detection problem in the ideal case when the packet loss rate $0 < p_t < 1$ is known beforehand. Results from such a case will provide us an upper bound on the performance that one can expect from any other cases

of this detection problem. We establish a test using a property of one-parameter exponential families. However, the result shows several drawbacks. In an attempt to simplify the test design, we transform the original detection problem to the renowned normal distribution using the *Central Limit Theorem* (CLT) and come up with a theoretical optimal LRT. As this Ph.D. is a continuation of a master degree's traineeship, we reused a part of this traineeship report and presented in this section.

4.3.1 Approach of One-parameter Exponential Families

If we consider each *Interest* packet as a Bernoulli trial with $(1 - p_t)$ chance of success, d_t then follows a binomial distribution with parameter i_t and $(1 - p_t)$. Since p_t is known for each instant t, the LRT does not require measurements from previous instants. Therefore, i_t and d_t are adequate for the test's input. Moreover, the two hypotheses now only depend one parameter – the attack payload N_{a_t}:

$$\begin{cases} \mathcal{H}_0(N_{a_t} = 0) : d_t \sim \mathcal{B}\left(i_t, 1 - p_t\right), \\ \mathcal{H}_1(N_{a_t} > 0) : d_t \sim \mathcal{B}\left(i_t - N_{a_t}, 1 - p_t\right). \end{cases} \tag{4.8}$$

The hypotheses in (4.8) are called *composite hypotheses* because the unknown parameter N_a can take several values from a parameter space. The binomial distribution belongs to the family of the exponential distribution, see [101, Section 2.7]. Therefore, according to [101, Corollary 3.4.1], for a prescribed PFA α_0, there exists a UMP test given by the following decision rule:

$$\delta^\star(d_t) = \begin{cases} \mathcal{H}_0 & \text{if } T(d_t) = \quad d_t \geq h, \\ \mathcal{H}_1 & \text{if} \qquad\quad d_t < h, \end{cases} \tag{4.9}$$

where threshold h is determined by the equation:

$$\begin{aligned} \mathbb{P}_0(d_t < h) &= \alpha_0, \\ \Rightarrow \sum_{j=0}^{\lfloor h \rfloor} \left[\binom{i_t}{j} (1 - p_t)^j p_t^{i_t - j} \right] &= \alpha_0. \end{aligned} \tag{4.10}$$

Eq. (4.10) shows that the threshold h depends on i_t and p_t that changes at each instant t, hence must always be recomputed for each instant. In addition, the threshold h given by Eq. (4.10) is a discrete value, which makes it hard to guarantee the PFA as close as possible to the desired value α_0. Besides, it is hard to exactly evaluate the statistical properties of a test between discrete distribution such as (4.9). In short, though a test such as (4.9) is simple to build, it is hardly usable in practice.

4.3.2 Asymptotic Approach Using Central Limit Theorem

Since the original hypotheses cannot yield a practical test, we transform the original hypotheses to the well-studied distribution by applying the asymptotic approach to our detection problem. The idea of the asymptotic approach is that, instead of designing a test for the original hypotheses, one can develop a test between two hypotheses that are asymptotically equivalent to the originals when a given variable tends to infinity. The result of this approach is an asymptotical test. We use the CLT (see [101, Theorem 11.2.5]) to asymptotically transform the original IFA detection problem. Given a set of independent real-valued random variables of the same distribution and with positive finite variance, the CLT states that their normalized sum converges to a normal distribution as the sample size tends to infinity.

Because the number of packets at a router face is usually enormous, it is reasonable to assume that the value i_t is large enough. Moreover, since each *Interest* packet is considered as a Bernoulli trial with $(1 - p_t)$ chance of success, one has:

$$\mathbb{E}(d_t) = i_t(1 - p_t), \tag{4.11}$$

$$0 < p_t < 1 \Rightarrow 0 < \sigma_{d_t}^2 = i_t p_t(1 - p_t) < \infty. \tag{4.12}$$

Applying the CLT to the hypothesis $\mathcal{H}_0$, we have:

$$\begin{aligned} & \frac{d_t - i_t(1-p_t)}{\sqrt{i_t p_t(1-p_t)}} && \rightsquigarrow \mathcal{N}(0,\, 1), \\ \Rightarrow \quad & d_t - i_t(1 - p_t) && \rightsquigarrow \mathcal{N}\left(0,\, i_t p_t(1 - p_t)\right), \\ \Rightarrow \quad & d_t && \rightsquigarrow \mathcal{N}\left(i_t(1 - p_t),\, i_t p_t(1 - p_t)\right), \end{aligned} \tag{4.13}$$

where $\rightsquigarrow$ represents the convergence in distribution as i_t tends to infinity.

Let r_t be the residual packet loss rate, i.e., the difference between observed and expected loss rates:

$$r_t = \ell_t - p_t = \left(1 - \frac{d_t}{i_t}\right) - p_t. \tag{4.14}$$

Under $\mathcal{H}_0$, the distribution of the residual r_t can be established as follows:

$$\begin{aligned} & d_t && \rightsquigarrow \mathcal{N}\left(i_t(1 - p_t),\, i_t p_t(1 - p_t)\right), \\ \Rightarrow \quad & -\frac{d_t}{i_t} && \rightsquigarrow \mathcal{N}\left(p_t - 1,\, \frac{p_t(1 - p_t)}{i_t}\right), \\ \Rightarrow \quad & l_t = 1 - \frac{d_t}{i_t} && \rightsquigarrow \mathcal{N}\left(p_t,\, \frac{p_t(1 - p_t)}{i_t}\right), \end{aligned} \tag{4.15}$$

$$\Rightarrow \quad r_t = l_t - p_t \quad \rightsquigarrow \mathcal{N}\left(0,\, \frac{p_t(1 - p_t)}{i_t}\right). \tag{4.16}$$

Let σ_t^2 be the variance of r_t under $\mathcal{H}_0$:

$$\sigma_t^2 = \frac{p_t(1-p_t)}{i_t} > 0. \tag{4.17}$$

As a result, the hypothesis $\mathcal{H}_0$ can be asymptotically written as:

$$\mathcal{H}_0 : r_t \rightsquigarrow \mathcal{N}\left(0,\, \sigma_t^2\right). \tag{4.18}$$

On the opposite, by applying CLT to the hypothesis $\mathcal{H}_1$, we have:

$$\begin{aligned}
& \tfrac{d_t-(i_t-N_{a_t})(1-p_t)}{\sqrt{(i_t-N_{a_t})p_t(1-p_t)}} && \rightsquigarrow \mathcal{N}(0,1), \\
\Rightarrow\ & d_t-(i_t-N_{a_t})(1-p_t) && \rightsquigarrow \mathcal{N}(0,(i_t-N_{a_t})p_t(1-p_t)), \\
\Rightarrow\ & d_t && \rightsquigarrow \mathcal{N}((i_t-N_{a_t})(1-p_t),(i_t-N_{a_t})p_t(1-p_t)), \\
\Rightarrow\ & -\frac{d_t}{i_t} && \rightsquigarrow \mathcal{N}\left(-\frac{(i_t-N_{a_t})(1-p_t)}{i_t}, \frac{(i_t-N_{a_t})p_t(1-p_t)}{i_t^2}\right), \\
\Rightarrow\ & l_t = 1-\frac{d_t}{i_t} && \rightsquigarrow \mathcal{N}\left(1-\frac{(i_t-N_{a_t})(1-p_t)}{i_t}, \frac{(i_t-N_{a_t})p_t(1-p_t)}{i_t^2}\right), \\
\Rightarrow\ & r_t = l_t - p_t && \rightsquigarrow \mathcal{N}\left(1-p_t-\tfrac{(i_t-N_{a_t})(1-p_t)}{i_t}, \tfrac{p_t(1-p_t)}{i_t}-\tfrac{N_{a_t}p_t(1-p_t)}{i_t^2}\right), \\
\Rightarrow\ & r_t && \rightsquigarrow \mathcal{N}\left((1-p_t)\left[1-\frac{(i_t-N_{a_t})}{i_t}\right], \sigma_t^2-\frac{ap_t}{i_t}\right), \\
\Rightarrow\ & r_t && \rightsquigarrow \mathcal{N}\left(\frac{(1-p_t)N_{a_t}}{i_t}, \sigma_t^2-\frac{ap_t}{i_t}\right), \\
\Rightarrow\ & r_t && \rightsquigarrow \mathcal{N}\left(a,\, \sigma_t^2-\frac{ap_t}{i_t}\right).
\end{aligned} \tag{4.19}$$

Let σ_a^2 be the decrease in variance due to IFA:

$$0 < \sigma_a^2 = \frac{ap_t}{i_t} = \frac{(1-p_t)N_{a_t}p_t}{i_t^2} < \sigma_t^2. \tag{4.20}$$

Hence, the hypothesis $\mathcal{H}_1$ changes into:

$$\mathcal{H}_1 : r_t \rightsquigarrow \mathcal{N}\left(a,\, \sigma_t^2-\sigma_a^2\right). \tag{4.21}$$

One can note that σ_a^2 is caused by the increase of i_t from $i_t^\star$ in Eq. (4.18) to $i_t^\star + N_{a_t}$ in Eq. (4.21) during the attack, while the number of *Data* packet d_t does not change.

From (4.18) and (4.21), the testing problem (4.7) can reformulated as:

$$r_t \sim \begin{cases} \mathcal{N}\left(0,\sigma_t^2\right) & \text{under } \mathcal{H}_0, \\ \mathcal{N}\left(a,\sigma_t^2-\sigma_a^2\right) & \text{under } \mathcal{H}_1. \end{cases} \tag{4.22}$$

Equation (4.22) indicates that the parameter a entirely characterizes the impact of

IFA on the packet loss rate, hence in the followings, a will be used to feature the attack payload.

The likelihood ratio$\Lambda(x)$ of the problem (4.22) is given by:

$$\begin{aligned} \Lambda(x) &= \frac{f_1(x)}{f_0(x)} = \frac{\frac{1}{\sqrt{2\pi(\sigma_t^2-\sigma_a^2)}}\, exp\left(-\frac{(x-a)^2}{2(\sigma_t^2-\sigma_a^2)}\right)}{\frac{1}{\sqrt{2\pi\sigma_t^2}}\, exp\left(-\frac{x^2}{2\sigma_t^2}\right)}, \\ \Rightarrow \quad \Lambda(x) &= \frac{\sqrt{\sigma_t^2}}{\sqrt{\sigma_t^2-\sigma_a^2}} \exp\left(\frac{x^2}{2\sigma_t^2} - \frac{(x-a)^2}{2\sigma_t^2-2\sigma_a^2}\right). \end{aligned} \tag{4.23}$$

Proof for Likelihood Ratio's Monotonicity

To find the UMP test, we utilize the theorem [101, Theorem 3.4.1]. The theorem proposes a formulation of the UMP test between two hypotheses whose distribution depending on a scalar parameter, and their likelihood ratio is monotone. As such, we first need to prove that the likelihood ratio $\Lambda(x)$ is monotonic through its first derivative:

$$\begin{aligned} \frac{\partial\Lambda(x)}{\partial x} &= \frac{\sqrt{\sigma_t^2}}{\sqrt{\sigma_t^2-\sigma_a^2}}\left(\frac{x}{\sigma_t^2} - \frac{x-a}{\sigma_t^2-\sigma_a^2}\right)\exp\left(\frac{x^2}{2\sigma_t^2} - \frac{(x-a)^2}{2\sigma_t^2-2\sigma_a^2}\right), \\ \Rightarrow \quad \frac{\partial\Lambda(x)}{\partial x} &= \left(\frac{x}{\sigma_t^2} - \frac{x-a}{\sigma_t^2-\sigma_a^2}\right)\Lambda(x). \end{aligned} \tag{4.24}$$

Notice that $\forall x \in \mathbb{R}, \Lambda(x) > 0$. Therefore, the monotonicity of Eq. (4.24) depends on whether its first term is either positive or negative:

$$\begin{aligned} \frac{x}{\sigma_t^2} - \frac{x-a}{\sigma_t^2-\sigma_a^2} &\gtrless 0\,, \\ \Rightarrow \quad \frac{x(\sigma_t^2-\sigma_a^2) - x\sigma_t^2 + a\sigma_t^2}{\sigma_t^2(\sigma_t^2-\sigma_a^2)} &\gtrless 0\,, \\ \Rightarrow \quad \frac{-x\sigma_a^2 + a\sigma_t^2}{\sigma_t^2(\sigma_t^2-\sigma_a^2)} &\gtrless 0\,. \end{aligned} \tag{4.25}$$

According to Eq. (4.17), (4.20), $\sigma_t^2(\sigma_t^2-\sigma_a^2) > 0$. Therefore, Eq. (4.25) can be rewritten as follows:

$$-x\sigma_a^2 + a\sigma_t^2 \gtrless 0, \tag{4.26}$$

$$\Rightarrow \quad x\sigma_a^2 \gtrless a\sigma_t^2\,,$$

$$\Rightarrow \quad x \gtrless \frac{a\sigma_t^2}{\sigma_a^2}. \tag{4.27}$$

Replacing σ_t^2 and σ_a^2 by their definitions – see, respectively, Eq. (4.17) and (4.20) – Eq. (4.27) turns into:

$$\begin{aligned} x &\gtrless \frac{\frac{ap_t(1-p_t)}{i_t}}{\frac{ap_t}{i_t}}, \\ \Rightarrow \quad x &\gtrless (1-p_t). \end{aligned} \tag{4.28}$$

Because $r_t = \left(1 - \frac{d_t}{i_t}\right) - p_t$ – see Eq.(4.18) – Eq.(4.28) becomes:

$$1 - \frac{d_t}{i_t} - p_t \gtrless (1-p_t) \Rightarrow -\frac{d_t}{i_t} \gtrless 0. \tag{4.29}$$

Since i_t and d_t are, respectively, the number of *Interest* and *Data* packets, they are always non-negative numbers. Therfore, $\frac{d_t}{i_t} \geq 0$ or $-\frac{d_t}{i_t} \leq 0$ always holds true, while the other sign does not occurs. Note that in the case $i_t = 0$, the likelihood ratio $\Lambda(r_t)$ is degenerated because $\sigma_t = \infty$ and $\sigma_a = \infty$. In short, the monotone condition for likelihood ratio (4.24) is:

$$-\frac{d_t}{i_t} < 0, \tag{4.30}$$

and it is always true, implying that $\Lambda(x)$ is an increasing function for $x > 0$ which corresponds to the range of possible observations. According to [101, Theorem 3.4.1], the UMP test for the detection problem (4.22) is:

$$\delta^\star(r_t) = \begin{cases} \mathcal{H}_0 & \text{if } r_t \leq \tau^\star, \\ \mathcal{H}_1 & \text{if } r_t > \tau^\star, \end{cases} \tag{4.31}$$

in which $\tau^\star$ is a threshold such that the prescribed PFA is matched:

$$\begin{aligned} \mathbb{P}_0(r_t > \tau^\star) &= \alpha_0\,, \\ or \quad \mathbb{P}_0\left(r_t \leq \tau^\star\right) &= 1 - \alpha_0\,. \end{aligned} \tag{4.32}$$

To ease the threshold's calculation, we transform the null hypothesis $\mathcal{H}_0$ to the form of the standard normal distribution, i.e., with zero mean and unit variance:

$$\begin{aligned} \mathcal{H}_0: \quad r_t &\rightsquigarrow \mathcal{N}\left(0, \sigma_t^2\right), \\ \Rightarrow \mathcal{H}_0: \quad \frac{r_t}{\sigma_t} &\rightsquigarrow \mathcal{N}(0, 1). \end{aligned}$$

As such, the threshold $\tau^\star$ is found as follows:

$$\begin{aligned} (4.32) \Rightarrow \quad \mathbb{P}_0\left(\frac{r_t}{\sigma_t} \leq \frac{\tau^\star}{\sigma_t}\right) = \Phi\left(\frac{\tau^\star}{\sigma_t}\right) &= 1 - \alpha_0\,, \\ \Rightarrow \qquad\qquad \tau^\star &= \sigma_t \Phi^{-1}(1-\alpha_0)\,. \end{aligned} \tag{4.33}$$

where Φ and Φ^{-1} are, respectively, the cumulative density function and its inverse of the standard normal distribution.

The power of the UMP test $\beta_{\delta^\star(r_t)}$ is defined by:

$$\beta_{\delta^\star(r_t)} \quad = \quad \mathbb{P}_1(r_t > \tau^\star) = \; 1 - \mathbb{P}_1(r_t \leq \tau^\star). \tag{4.34}$$

Similarly to the transformation for the null hypothesis $\mathcal{H}_0$, we replicate the process for the alternative hypothesis $\mathcal{H}_1$:

$$\begin{aligned} \mathcal{H}_1 : \quad & r_t & & \rightsquigarrow \mathcal{N}\left(a,\, \sigma_t^2 - \sigma_a^2\right), \\ \Rightarrow \mathcal{H}_1 : \quad & r_t - a & & \rightsquigarrow \mathcal{N}\left(0,\, \sigma_t^2 - \sigma_a^2\right), \\ \Rightarrow \mathcal{H}_1 : \quad & \frac{r_t - a}{\sqrt{\sigma_t^2 - \sigma_a^2}} & & \rightsquigarrow \mathcal{N}\left(0,\, 1\right). \end{aligned}$$

Hence, the power of UMP test is computed as follows:

$$\begin{aligned} (4.34) \Rightarrow \beta_{\delta^\star(r_t)} \quad &= \quad 1 - \mathbb{P}_1\left(\frac{r_t - a}{\sqrt{\sigma_t^2 - \sigma_a^2}} \leq \frac{\tau^\star - a}{\sqrt{\sigma_t^2 - \sigma_a^2}}\right), \\ \Rightarrow \beta_{\delta^\star(r_t)} \quad &= \quad 1 - \Phi\left(\frac{\tau^\star - a}{\sqrt{\sigma_t^2 - \sigma_a^2}}\right) = 1 - \Phi\left(\frac{\sigma_t \Phi^{-1}(1-\alpha_0) - a}{\sqrt{\sigma_t^2 - \sigma_a^2}}\right) \end{aligned} \tag{4.35}$$

According to the definition of convergence in distribution [101, Definition 11.2.1], in the virtue of Portmanteau [101, Theorem 11.2.1] and to the continuous mapping theorem [101, Theorem 11.2.13], the power function of the test $\delta^\star(r_t)$ converges to the power function of the MP test $\delta^\star(d_t)$, defined in (4.9). Hence, the proposed test $\delta^\star(r_t)$ is *Asymptotically UMP* (AUMP) for testing problem (4.7) and its statistical properties are presented in the following Proposition (1):

Proposition 1. *Assuming that the number of Interests i_t tends to infinity, for any prescribed false-alarm probability α_0, the decision threshold, $\tau^\star$, given by:*

$$\tau^\star(\alpha_0) = \sigma_t \Phi^{-1}\left(1 - \alpha_0\right), \tag{4.36}$$

guarantees that the test $\delta^\star$ (4.31) is in $\mathcal{K}_{\alpha_0}$. Here Φ and Φ^{-1} are the standard normal cumulative distribution function and its inverse function, respectively. Using the decision threshold given in (4.36) the power function of the UMP test $\delta^\star$ (4.31) is given by:

$$\beta_{\delta^\star(r_t)}(a) = 1 - \Phi\left(\frac{\sigma_t \Phi^{-1}\left(1 - \alpha_0\right) - a}{\sqrt{\sigma_t^2 - \sigma_a^2}}\right). \tag{4.37}$$

The assessment of the statistical performance of AUMP test $\delta^\star(r_t)$ serves as an upper bound on the detection performance that is expected from any practical detection method for IFA. Another interesting aspect of the proposed asymptotic approach is that it is possible to set a threshold that satisfies a prescribed false-alarm probability. This threshold only depends on the prescribed PFA and the number of *Interests* i_t and packet loss rate p_t which are both known. This approach dramatically simplifies the problems of dealing with the binomial distribution whose cumulative distribution function is difficult to compute. However, a notable consequence of the underlying binomial distribution is that residual packet loss rate r_t has both its expectation and its variance impacted by IFA. Hence the power function of proposed AUMP test not only depends on the attack payload a but also, to a lesser extent, on the impact of IFA on r_t's variance through the denominator $\sqrt{\sigma_t^2 - \sigma_a^2}$.

4.4 Generalized Likelihood Ratio Test for Unknown Packet Loss Rate

In reality, as the amount of *Interests* and *Data* fluctuate over time, one cannot know precisely the packet loss rate a priori, and thus the AUMP test cannot be used in practice, and its performance is hardly achievable. As a result, we address in this section a more realistic case in which the expected packet loss rate p_t is unknown. In such a situation, a usual approach consists in designing a GLRT by substituting the unknown parameter p_t with its Maximum Likelihood Estimation from measurements of ℓ_t. We first introduce a model to estimate the unknown packet loss rate. Then, we design a GLRT based on the proposed model and establish its theoretical properties. The validation of the theoretical finding with data from real NDN deployment is addressed in Section 4.6.

4.4.1 Packet Loss Rate Model

Simple Regression Model for Packet Loss Rate

To model the packet loss rate, we consider a window with integer size $N > 0$ of most recent measurements of packet loss rate $\boldsymbol{\ell}_t = [\ell_{t-N+1}, \ldots, \ell_t]^T$. Under hypothesis $\mathcal{H}_0$, the fluctuation of the packet loss rate is limited and smooth [102, 103], hence can be modeled by a polynomial – a well-known model in approximation theory with many applications [104–106] – as follows:

$$\ell_{t-N+n} = x_0 + x_1 n + x_2 n^2 + \ldots + x_{q-1} n^{q-1} + \epsilon_{t-N+n},\ n \in \{1, \ldots, N\} \tag{4.38}$$

where q is the order of the polynomial, i.e. the number of coefficients x_j with $j \in \{1, \ldots, q-1\}$ to be fit, and ϵ_t is the error at an instant t. Moreover, to achieve the desired properties of the model, it is assumed that our model follows the Gauss-Markov contiditions [107], expressed by following equations:

- $\mathbb{E}(\epsilon_t) = 0$: implies that the error has a zero mean;
- $var(\epsilon_t) = \sigma^2 < \infty$: indicates that the error has the same and constant variance;
- $cov(\epsilon_i, \epsilon_j) = 0, \forall i \neq j$: means that distinct error terms are uncorrelated to each other.

Transforming the Hypotheses

In matrix notation, the Gauss-Markov conditions can be rewritten as:

$$\mathbb{E}(\boldsymbol{\epsilon}_t) = \mathbf{o}, \qquad cov(\boldsymbol{\epsilon}_t) = \sigma^2 \boldsymbol{I}_N \tag{4.39}$$

where $\boldsymbol{o}$ is a vector of zeros, $cov(\boldsymbol{\epsilon}_t)$ is the covariance matrix of $\boldsymbol{\epsilon}_t$ and $\boldsymbol{I}_N$ is the identity matrix of size N, i.e., a square matrix with all elements on the main diagonal equal 1. As such, Eq. (4.38) can be written in matrix notation as:

$$\boldsymbol{\ell}_t \quad = \quad \mathbf{H}\mathbf{x}_t + \boldsymbol{\epsilon}_t \tag{4.40}$$

where $\mathbf{x}_t = [x_0, \ldots, x_{q-1}]^T$ is the vector of the q coefficients of the polynomial; $\boldsymbol{\epsilon_t} = [\epsilon_{t-N+1}, \ldots, \epsilon_t]^T$ is the error vector of N measurements taken into account and $\mathbf{H}$ is a matrix of size $N \times q$ whose elements $h_{(n,j)} = n^j$, $n \in \{1, \ldots, N\}$, $j \in \{0, \ldots, q-1\}$:

$$\mathbf{H} = \begin{pmatrix} h_{(1,0)} & h_{(1,1)} & \ldots & h_{(1,q-1)} \\ h_{(2,0)} & h_{(2,1)} & \ldots & h_{(2,q-1)} \\ \vdots & \vdots & \ddots & \vdots \\ h_{(N,0)} & h_{(N,1)} & \ldots & h_{(N,q-1)} \end{pmatrix} = \begin{pmatrix} 1 & 1^1 & \ldots & 1^{q-1} \\ 1 & 2^1 & \ldots & 2^{q-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & N^1 & \ldots & N^{q-1} \end{pmatrix}$$

Such a model has been widely used in signal processing [108, 109], for applications in Internet traffic modeling [110, 111] and image processing [112–114]. Furthermore, under the Gauss-Markov conditions, the expectation of $\boldsymbol{\ell_t}$ becomes:

$$\mathbb{E}(\boldsymbol{\ell}_t) \quad = \quad \mathbb{E}(\mathbf{H}\mathbf{x}_t + \boldsymbol{\epsilon}_t) = \mathbb{E}(\mathbf{H}\mathbf{x}_t) + \mathbb{E}(\boldsymbol{\epsilon}_t) = \mathbf{H}\mathbf{x}_t \tag{4.41}$$

On the other hand, assuming that packet loss rate measurements ℓ_t are statistically independent, it follows from the asymptotic distribution in Eq. (4.15) that

under hypothesis $\mathcal{H}_0$, the measured packet loss rate $\boldsymbol{\ell}_t$ can be modeled as:

$$\mathcal{H}_0 : \quad \boldsymbol{\ell}_t \rightsquigarrow \mathcal{N}(\boldsymbol{p}_t = \mathbf{H}\mathbf{x}_t \,,\, \Sigma_0), \tag{4.42}$$

where $\mathbf{\Sigma}_0$ is a diagonal covariance matrix whose elements are given by:

$$\mathbf{\Sigma}_{0(n,n)} = \frac{p_n(1-p_n)}{i_n} \,,\, n \in \{t-N+1, \ldots, t\}.$$

When IFA starts at an instant t, the packet loss rate will increase at the last samples. As a result, under hypothesis $\mathcal{H}_1$, as i_t tends to infinity, $\boldsymbol{\ell}_t$ can be modeled as:

$$\mathcal{H}_1 : \boldsymbol{\ell}_t \rightsquigarrow \mathcal{N}\left(\mathbf{H}\mathbf{x}_t + a\mathbf{v}_a \,,\, \mathbf{\Sigma}_0 - \mathbf{\Sigma}_a\right), \tag{4.43}$$

where:

- a is the attack payload, as in Eq. (4.6);
- $\mathbf{v}_a$ is a vector of N elements, indicating instants where the packet loss rate increases because of the attack, e.g. $\mathbf{v}_a = [0, 0, \ldots, 0, 1]^T$ indicates that only the last sample is corrupted by IFA;
- $\mathbf{\Sigma}_a$ is a diagonal matrix representing the decrease of $\boldsymbol{\ell}_t$'s variance due to IFA. Its element are given by:

$$\mathbf{\Sigma}_{a(n,n)} = \begin{cases} 0 & \text{, if } \mathbf{v}_{a(n)} = 0, \\ \dfrac{ap_t}{i_t} & \text{, if } \mathbf{v}_{a(n)} = 1, \end{cases}$$

  implying that the variance decrease, as in Eq. (4.20), only applies to the corrupted samples.

Packet Loss Rate Estimator

For the regression model Eq. (4.40), the *Ordinary Least Square* (OLS) estimator of the coefficient vector $\mathbf{x}_t$ is given by:

$$\widetilde{\mathbf{x}}_t = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\boldsymbol{\ell}_t.$$

Under the Gauss-Markov conditions, it is well known that the OLS estimation is equivalent to the maximum likelihood estimation. Therefore, the maximum likelihood estimation of the packet loss rate $\mathbf{p}_t$ is the OLS estimation given by:

$$\widetilde{\mathbf{p}}_t = \mathbf{H}\widetilde{\mathbf{x}}_t = \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\boldsymbol{\ell}_t. \tag{4.44}$$

4.4.2 Proposed Test and its Properties

As a result, the estimated residual packet loss rate $\mathbf{r}_t$ (Eq. (4.14)) are defined as:

$$\begin{aligned} & \widetilde{\mathbf{r}}_t &&= \boldsymbol{\ell}_t - \widetilde{\mathbf{p}}_t \\ \Rightarrow\ & \widetilde{\mathbf{r}}_t &&= \boldsymbol{\ell}_t - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\boldsymbol{\ell}_t \\ \Rightarrow\ & \widetilde{\mathbf{r}}_t &&= [\mathbf{I}_N - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T]\boldsymbol{\ell}_t = \mathbf{H}^{\perp}\boldsymbol{\ell}_t. \end{aligned} \tag{4.45}$$

where $\mathbf{H}^{\perp} = [\mathbf{I}_N - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T]$ is a square matrix of size $N \times N$ that represents the projection onto the orthogonal complement of the subspace spanned by the columns of $\mathbf{H}$. Thus, the hypothesis $\mathcal{H}_0$ in Eq. (4.42) turns into:

$$\begin{aligned} \mathcal{H}_0 :\quad & \widetilde{\mathbf{r}}_t = \mathbf{H}^{\perp}\boldsymbol{\ell}_t && \rightsquigarrow \mathcal{N}(\mathbf{H}^{\perp}\mathbf{H}\mathbf{x}_t \quad , \quad \mathbf{H}^{\perp}\Sigma_0{\mathbf{H}^{\perp}}^T), \\ \Rightarrow \mathcal{H}_0 :\quad & \widetilde{\mathbf{r}}_t && \rightsquigarrow \mathcal{N}(\boldsymbol{o} \quad , \quad \mathbf{H}^{\perp}\Sigma_0{\mathbf{H}^{\perp}}^T) \end{aligned} \tag{4.46}$$

and the hypothesis $\mathcal{H}_1$ in Eq. (4.43) becomes:

$$\begin{aligned} \mathcal{H}_1 :\quad & \widetilde{\mathbf{r}}_t = \mathbf{H}^{\perp}\boldsymbol{\ell}_t && \rightsquigarrow \mathcal{N}\left[\mathbf{H}^{\perp}(\mathbf{H}\mathbf{x}_t + a\mathbf{v}_a) \quad , \quad \mathbf{H}^{\perp}(\boldsymbol{\Sigma}_0 - \boldsymbol{\Sigma}_a)\right], \\ \Rightarrow \mathcal{H}_1 :\quad & \widetilde{\mathbf{r}}_t && \rightsquigarrow \mathcal{N}\left(\mathbf{H}^{\perp}\mathbf{H}\mathbf{x}_t + a\mathbf{H}^{\perp}\mathbf{v}_a \quad , \quad \mathbf{H}^{\perp}\boldsymbol{\Sigma}_0{\mathbf{H}^{\perp}}^T - \mathbf{H}^{\perp}\boldsymbol{\Sigma}_a{\mathbf{H}^{\perp}}^T\right), \\ \Rightarrow \mathcal{H}_1 :\quad & \widetilde{\mathbf{r}}_t && \rightsquigarrow \mathcal{N}\left(a\widetilde{\mathbf{v}}_a \quad , \quad \mathbf{H}^{\perp}\boldsymbol{\Sigma}_0{\mathbf{H}^{\perp}}^T - \mathbf{H}^{\perp}\boldsymbol{\Sigma}_a{\mathbf{H}^{\perp}}^T\right), \end{aligned} \tag{4.47}$$

where $\widetilde{\mathbf{v}}_a = \mathbf{H}^{\perp}\mathbf{v}_a$ is a column vector of N elements that represents the IFA footprint obtained by estimating and removing the expected packet loss rate (4.45). It is noteworthy that both $\mathbf{H}^{\perp}$ and $\mathbf{v}_a$ are known once the configuration is already fixed. Therefore, $\widetilde{\mathbf{v}}_a$ is calculated only once. As a result, the IFA detection problem for unknown loss rate can now be formulated as:

$$\begin{cases} \mathcal{H}_0 : & \widetilde{\mathbf{r}}_t \rightsquigarrow \mathcal{N}\left(\boldsymbol{o} \quad , \quad \mathbf{H}^{\perp}\Sigma_0{\mathbf{H}^{\perp}}^T\right), \\ \mathcal{H}_1 : & \widetilde{\mathbf{r}}_t \rightsquigarrow \mathcal{N}\left(a\widetilde{\mathbf{v}}_a \quad , \quad \mathbf{H}^{\perp}\boldsymbol{\Sigma}_0{\mathbf{H}^{\perp}}^T - \mathbf{H}^{\perp}\boldsymbol{\Sigma}_a{\mathbf{H}^{\perp}}^T\right). \end{cases} \tag{4.48}$$

One should notice that the IFA impacts both the expectation and the covariance of the residuals $\widetilde{\mathbf{r}}_t$, as previously discussed in Section 4.3.

As previously mentioned, the GLRT can be designed by replacing the unknown parameter p_t in the LRT (4.31) with its Maximum Likelihood Estimator $\widetilde{p_t}$ (4.44). However, it is noteworthy that in Eq. (4.48), essentially consists in testing the presence of a known signal $a\widetilde{\mathbf{v}}_a$ in the presence Gaussian noise. It is well-known, see [115] and [111,116] for applications, that is such a case a sufficient statistics can be obtained through the projection of observations $\widetilde{\mathbf{r}}_t$ onto the signal $a\widetilde{\mathbf{v}}_a$ resulting in the scalar value $a\widetilde{\mathbf{v}}_a^{\perp}\widetilde{\mathbf{r}}_t$. $\widetilde{\mathbf{c}}_t = \widetilde{\mathbf{v}}_a^T\widetilde{\mathbf{r}}_t$ as the test statistic. From the distribution of the residuals

$\widetilde{\mathbf{r}}$ (4.48), it is straightforward that:

$$\widetilde{\mathbf{c}}_t = \widetilde{\mathbf{v}}_a^T \widetilde{\mathbf{r}}_t \rightsquigarrow \begin{cases} \mathcal{N}\left(\mathbf{0}, s_0^2\right) & \text{under } \mathcal{H}_0, \\ \mathcal{N}\left(a\|\widetilde{\mathbf{v}}_a\|_2^2, s_0^2 - s_a^2\right) & \text{under } \mathcal{H}_1. \end{cases} \tag{4.49}$$

where

- $\|\widetilde{\mathbf{v}}_a\|_2^2 = \widetilde{\mathbf{v}}_a^T \widetilde{\mathbf{v}}_a$ is the square of $\widetilde{\mathbf{v}}_a$'s Euclidian norm, given by:

$$\|\widetilde{\mathbf{v}}_a\|_2^2 = \widetilde{\mathbf{v}}_{a(1)}^2 + \ldots + \widetilde{\mathbf{v}}_{a(N)}^2;$$

- the GLR variance s_0^2 under $\mathcal{H}_0$ and the decrease of variance s_a^2 under $\mathcal{H}_1$ are given by:

$$s_0^2 = \widetilde{\mathbf{v}}_a^T \mathbf{H}^\perp \mathbf{\Sigma}_0 {\mathbf{H}^\perp}^T \widetilde{\mathbf{v}}_a \quad , \quad s_a^2 = \widetilde{\mathbf{v}}_a^T \mathbf{H}^\perp \mathbf{\Sigma}_a {\mathbf{H}^\perp}^T \widetilde{\mathbf{v}}_a. \tag{4.50}$$

The GLRT hence becomes:

$$\widetilde{\delta}(\widetilde{\mathbf{r}}_t) = \begin{cases} \mathcal{H}_0 & \text{if } \widetilde{\mathbf{v}}_a^T \widetilde{\mathbf{r}}_t \leq \widetilde{\tau}, \\ \mathcal{H}_1 & \text{if } \widetilde{\mathbf{v}}_a^T \widetilde{\mathbf{r}}_t > \widetilde{\tau}. \end{cases} \tag{4.51}$$

It is noteworthy that $\widetilde{\mathbf{v}}_a^T \widetilde{\mathbf{r}}_t = (\mathbf{H}^\perp \mathbf{v}_a)^T (\mathbf{H}^\perp \boldsymbol{\ell}_t) = \mathbf{v}_a^T {\mathbf{H}^\perp}^T \mathbf{H}^\perp \boldsymbol{\ell}_t$. While $\mathbf{H}^\perp$ is fixed, $\mathbf{v}_a$ is chosen a priori. As such, one has to compute the term $\mathbf{v}_a^T {\mathbf{H}^\perp}^T \mathbf{H}^\perp$ only once and multiplies it with $\boldsymbol{\ell}_t$ for each new sample.

Similarly to Proposition (1), based on the distribution of the GLR (4.49), we can establish the decision threshold and the power function of the proposed GLRT:

Proposition 2. *Assuming that the number of incoming Interests i_t tends to infinity, for any prescribed false-alarm probability α_0, the decision threshold $\widetilde{\tau}$ is given by:*

$$\widetilde{\tau} = s_0 \Phi^{-1}\left(1 - \alpha_0\right), \tag{4.52}$$

as in Eq. (4.36), guarantees that the test $\widetilde{\delta}$, see Eq. (4.51), is in $\mathcal{K}_{\alpha_0}$. Using the decision threshold given in Eq. (4.52), the power function of the GLRT test (4.51) is given by:

$$\beta_{\widetilde{\delta}(\widetilde{\mathbf{r}}_t)}(a) = 1 - \Phi\left(\frac{s_0 \Phi^{-1}(1-\alpha_0) - a\|\widetilde{\mathbf{v}}_a\|_2^2}{\sqrt{s_0^2 - s_a^2}}\right). \tag{4.53}$$

From the power function (4.53), it is noteworthy that the main cause for the optimality loss of the proposed GLRT is the factor $\|\widetilde{\mathbf{v}}_a\|_2^2$. It is because a non-negligible

proportion of the packet loss rate changed due to IFA will be modeled as part of the regular change of legitimate traffic.

4.5 From Snapshot Test to Sequential Detection

The optimal tests presented in Section 4.3 and 4.4 are devoted to the analysis of a single router face at a specific time. In practice, the packet loss rate may not always fluctuate smoothly. It can increase abruptly at a specific time, leading to a false alarm. However, if it is not due to the IFA, the increase is unlikely to be repeated over time. Such noise can be mostly avoided by gathering consecutive samples. Moreover, the IFA traffic can be distributed over multiple attacker-controlled hosts so that its footprint at a specific time and a specific router's face becomes small enough, impeding the instantaneous detection. Collecting evidence over time can make the detection more sensitive to such an elusive footprint. This section extends the previous "snapshot" detection method by taking into account previous observations within a sequential framework.

In the literature, the problem of change-point detection[1] has been extensively studied. In brief, the sequential analysis framework not only aims at detecting a specific event with the highest accuracy, regarding false-alarm and missed-detection probability but also introduces the delay as the third criterion of detection performance. Given a sequence of observations $\{x_1, \ldots, x_t\}$, a change-point detection scheme is defined by a stopping rule $S(x_1, \ldots, x_t) \mapsto \{0, 1\}$ such that when IFA is detected for the first time at an instant S_t, the stopping rule $S(x_1, \ldots, x_t) = 1$. Let ν be the IFA's starting time. As such, $S_t \geq \nu$ when the attack is correctly detected, and the detection delay is defined as $DD = S_t - \nu$. On the opposite, when $S_t < 0$, a false alarm is triggered after the run length S_t.

Several methods have been proposed in the literature for change-point detection. This section uses the well-known *CUmulative SUM* (CUSUM), initially proposed in [119]. The reason for this choice is twofold. First, we have empirically observed that the CUSUM provides best overall performance. Secondly, the CUSUM has been shown to be optimal in several cases according to the so-called Lorden's criterion [120] that consists in minimizing the average of worst-case's detection delay, defined as:

$$\sup_{\nu \in \mathbb{N}} \mathbb{E}\left[S_t - \nu | S_t \geq \nu\right] \tag{4.54}$$

[1] Note that in the literature, the term "change-point" detection, as in [117] and [118], usually refers to sequential problem in which samples distribution changes at a given time ν.

for a given worst-case's average *Run Length To False-Alarm* (RL2FA), defined as:

$$\inf_{\nu \in \mathbb{N}} \mathbb{E}\left[S_t | S_t < \nu\right] \tag{4.55}$$

For a given face at which observations $\ell_1, \ldots, \ell_t$, see Eq. (4.2), are collected, the CUSUM C_t is defined by the following recurrence relation:

$$C_t = \max\left(0\,;\, C_{t-1} + \widetilde{\mathbf{v}}_a^T \widetilde{\mathbf{r}}_t - \kappa\right), \text{with } C_0 = 0, \tag{4.56}$$

with $C_0 = 0$, where $\mathbf{v}_a^T \widetilde{\mathbf{r}}_t$, see Eq. (4.49), is the likelihood ratio between hypotheses $\mathcal{H}_0$ and $\mathcal{H}_1$ computed with observation ℓ_t and κ is a constant to be set. The main idea of CUSUM is to compute sequential LRs and reset it to zero whenever it goes below zero, given that observations are independent and the change-point has not occurred yet. The constant κ can be interpreted as the "sensitivity" of the CUSUM. A large κ will make the reset of C_t to 0 more frequently but may delay the detection. On the contrary, a small κ allows a faster detection at the price of a less frequently reset CUSUM, hence yielding a smaller average RL2FA.

4.6 Numerical Results

In this section, we assess our proposed detection with both simulated data and real data. The simulated data is necessary to validate the intrinsic statistical properties of our detection since, in the simulation environment, we can entirely control the attack power regardless of the NDN network conditions. In a second step, the proposed detection is assessed with data from the real deployment of NDN coupled with IP (see Sections 3.3.2 and 3.3.4). Each subsection begins with a description of the deployed topology, utilized tools, experiment setup, followed by the evaluations on the PFA guarantee and the detection power – the two objectives of hypothesis testing with the Neyman-Pearson approach.

4.6.1 Assessment of the Statistical Properties with Simulation

Simulation Tool and Topology

In this section, two sets of numerical results are presented. First, results obtained on data simulated in MATLAB are presented to verify the sharpness of the theoretical findings. Then, we use ndnSIM [121] – an open-source NDN simulation provided by the NDN project. Indeed, ndnSIM faithfully implements the components of an NDN network, allowing us to consider every aspect of the network. To compare

the performance of our approach to existing ones, we reuse a topology from [53], illustrated in Figure 4.1 - a binary tree with eight hosts, intermediate routers and one content provider for the evaluation. The topology represents one of the worst cases to defend against IFA since all *Interests* are aggregated to upper links, and eventually to the content provider.

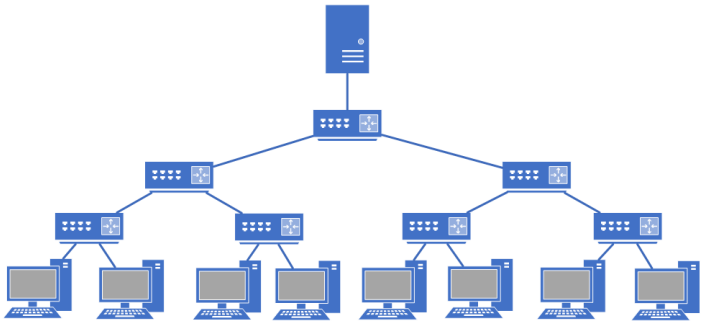


FIGURE 4.1: Topology for data simulation in ndnSIM

Simulation Setup

In our simulations, the number of *Interests* is generated from a Poisson distribution whose mean value is drawn from a uniform distribution. Also, the actual packet loss rate follows an auto-regressive model. Such a model has been widely used to model both users' requests evolution and packet loss rate in computer network [102, 122] and can be easily implemented in ndnSIM. More specifically, the packet loss rate is initialized at $p_0 = 0.05$, then its expectations are given by $p_t = p_{t-1} + u$ with u drawn from a uniform distribution with zero mean. To avoid the computational problem and to have a realistic behavior, the sign of u is flipped if $p_t < 0$ or if $p_t > 0.25$, which is a quite high value in practice. Several values for those parameters have been tested, and the obtained results show similar trends.

For the proposed GLRT, a set of $N = 50$ samples is used, and the degree of the polynomial is $q - 1 = 4$. Hence, the matrix $\mathbf{H}$ has the size 50×5. In all the figures, unless explicitly stated otherwise, the quickest detection is considered, i.e., the last sample is supposed to be corrupted. Therefore, the footprint of IFA on the packet loss rate is characterized by $\mathbf{v}_a = [0, \ldots, 1]^T$, yielding a footprint after packet loss rate estimation $\widetilde{\mathbf{v}}_a$ with $\|\widetilde{\mathbf{v}}_a\|_2^2 \approx 0.6$.

Simulation Results Analysis

Numerical Results on MATLAB Simulated Data Figure 4.2 shows a comparison between the theoretical and empirical performance of the proposed GLRT on MATLAB simulated data. The figure consists of the PFA, in dark and light blue, and detection power, in dark and light red, that are plotted as a function of decision threshold $\widetilde{\tau}$ as in Eq. (4.52). The figure exhibits that even for thresholds corresponding to probabilities as small as $\alpha_0 = 10^{-3}$, the empirical results are close to the theoretical ones despite the asymptotic approach proposed and the mean number of *Interests* is 12000. This also shows the sharpness of the theoretical findings and the relevance of the proposed model.

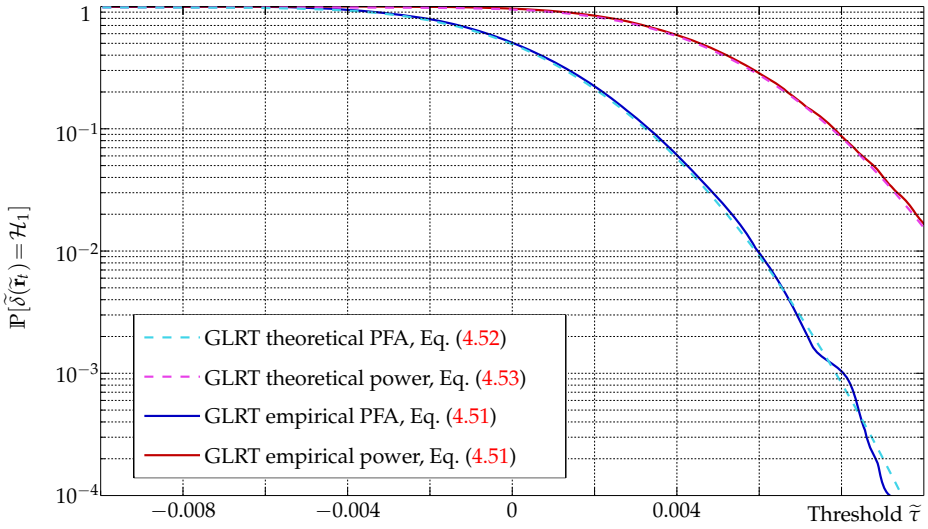


FIGURE 4.2: Comparison between the theoretical and empirical performance of the proposed GLRT with MATLAB simulated data.

Figure 4.3 then compares the theoretical and empirical power for both the optimal LRT and proposed GLRT. The mean number of *Interests* is 12000. The power function is plotted as a function of the anomaly $a \in [0, 0.02]$, corresponding to N_a values from 0 to 294 malicious *Interest* packets. The power is computed with two prescribed PFA $\alpha_0 = 0.01$ and $\alpha_0 = 0.1$. The figure reinforces the relevance of the theoretical findings since empirical power functions are close to the theoretical ones. However, for a low prescribed PFA such as $\alpha_0 = 0.01$, empirical results are slightly less accurate since a larger set of sample is required to achieve such a low PFA.

Numerical Results on ndnSIM Data Figure 4.4 presents the comparison between the proposed GLRT theoretical and empirical PFA as a function of detection threshold $\widetilde{\tau}$ using ndnSIM data. Unlike MATLAB simulated data, the actual value of packet loss rate is unknown in ndnSIM data because such feature was not available in ndnSIM when we run simulations for results in this section (in June 2015). Therefore, the optimal LRT cannot be included in this comparison. Compared to MATLAB simulated data, the number of samples collected in ndnSIM is much smaller since

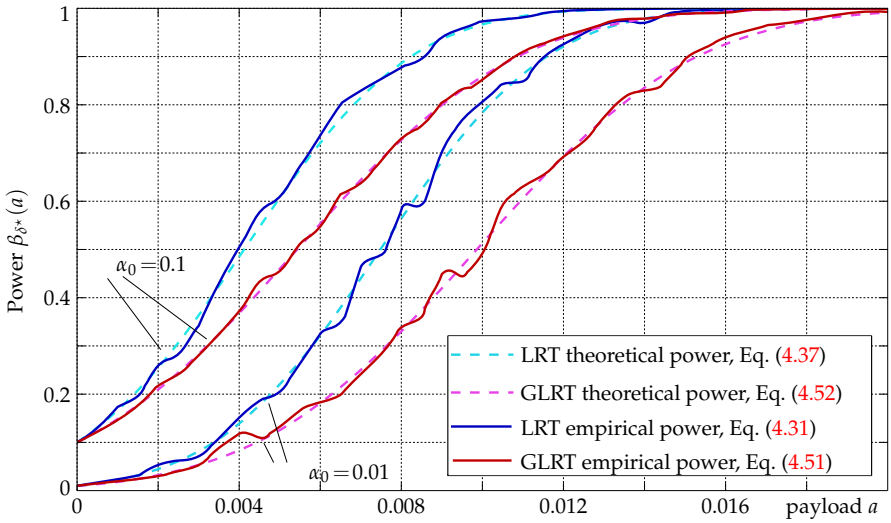


FIGURE 4.3: Comparison between empirical and theoretical detection power function for both the optimal LRT and the proposed GLRT on MATLAB simulated data.

running ndnSIM is time-consuming. One should also note that the mean number of *Interests* i_t in ndnSIM data is 3000 packets, which is relatively small as compared to infinity as our assumption in the asymptotic approach (see Proposition 2). Nevertheless, the empirical PFA still matches the theoretical one. This result is crucial as it demonstrates the relevance of the asymptotic approach with data from a reliable NDN simulator, even when the number of samples is limited, and the mean number of *Interests* is relatively small as compared to our assumption.

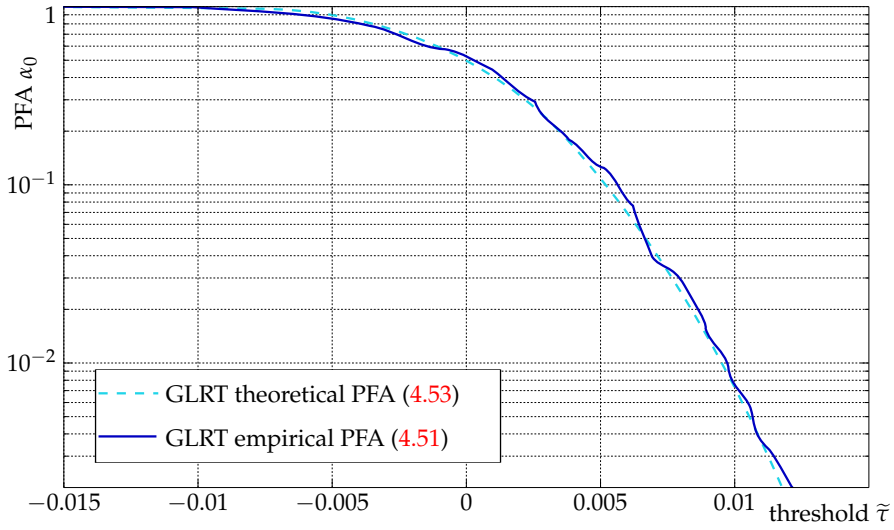


FIGURE 4.4: Comparison between empirical and theoretical PFA of the proposed GLRT on ndnSIM data.

Figure 4.5 depicts the *Receiver Operating Characteristic* (ROC) curves for various numbers of corrupted samples, denoted $M = \|\mathbf{v}_a\| \in [1, 3, 7]$, with corresponding

$\|\widetilde{\mathbf{v}}_a\|_2^2$ are 0.6, 0.73, and 1.07, to compare the theoretical and empirical performance of the proposed GLRT with ndnSIM data. Note that the closer the curve to the upper left corner of the plot, the better the detection. In addition, to be considered as acceptable, a curve must be higher than the diagonal $\beta = \alpha_0$ which represents the performance of a random guess. As expected, the power increases with the number of corrupted samples. This result emphasizes that the proposed method can be adapted to focus on the quickest detection by detecting whether the one most recent sample is corrupted at the cost of lower detection accuracy. On the contrary, it is possible to increase the detection delay by expecting more samples corrupted by the IFA, to ensure a higher detection accuracy. Moreover, Figure 4.5 also exhibits a comparison with the detector proposed in [53], that utilizes the instantaneous packet loss rate d_t/i_t with a fixed threshold for detection (see Section 2.2.1). It is noteworthy that such a statistic is widely used in the state of the art for IFA detection, and yet it cannot deal with the non-stationary users' behavior. Indeed, the figure shows that the proposed detector [53] performs slightly better than the random guess, while the proposed GLRT achieves much better performance, even with the quickest detection ($M = 1$).

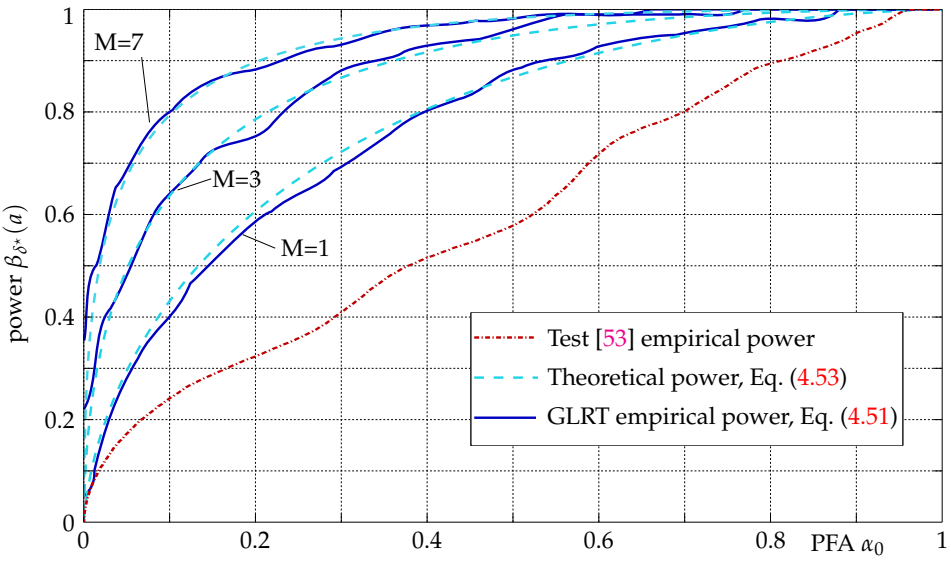


FIGURE 4.5: ROC curves for the proposed GLRT with different number of corrupted samples.

4.6.2 Performance Evaluation under Real Conditions

In this section, we use the data collected from the real NDN testbed and scenarios described in Section 3.3.4. First, we verify the behavior of the empirical packet loss rate when IFA is carried with the proposed attack scenario in reality. The ability to guarantee a prescribed PFA of the proposed GLRT is demonstrated along with the

selection of best parameters for empirical applications. Then, we assess the performance of the snapshot test and the CUSUM test against IFA in real NDN deployment. We also integrate the test [53] (see Section 2.2.1) to the CUSUM equation and assess its performance to compare our proposed test with the state of the art in IFA detection.

Empirical Packet Loss Rate in Reality

With data from the real NDN architecture, we observe a difference between the theoretical and the empirical packet loss rate, depicted in Figure 4.6. In theory, when the attack starts, the packet loss rate is expected to increase and then maintained at a specific value during the attack period. Note that for easy understanding, the theoretical loss rate in Figure 4.6 belongs to an apparent attack in which the loss packet rate is multiplied by a factor of 10 (from 2% to 20%) after the IFA starts. However, the empirical data shows that the packet loss rate repeatedly increases and drop during the attack. These changes occur quite abruptly and seem not to follow any particular pattern. This phenomenon can be explained as follows. At an instant t, there is a certain amount of *Interests* created by the attacker. *Data* packets for those *Interests* will not arrive in the same instant due to the additional delay caused by the malicious server, and hence increasing packet loss rate measured at the instant t. Delayed *Data* packets will eventually arrive later at an instant $t + x, x > 0$, compensating for the absence of *Data* packets caused in this $t + x$ instant, hence decreasing the measured packet loss rate. Thus, the loss packet rate during attack changes quite abruptly depending on the exact number of attack *Interests* sent over each sampling period and the exact delay of *Data* packets which are both stochastic processes. Such an observation demonstrates the difficulty of IFA detection in a real deployment with the snapshot test because the test statistic can drop down, even when the attack is still going on, thus dampening the detection performance.

Guarantee of False-Alarm Probability and Detector Configuration Selection

The most important property of the proposed statistical methodology lies in its ability to guarantee a prescribed PFA. To validate this aspect, we evaluate the empirical packet loss rate under all the data collected under $\mathcal{H}_0$, that is with more than 48000 samples. For being comprehensive, several parameters of the proposed method have been selected, namely the window length N, the model degree q and the number M of corrupted samples it is aimed at detecting. Figure 4.7 depicts the empirically measured PFA as a function of threshold $\widetilde{\tau}$, as well as a comparison with the

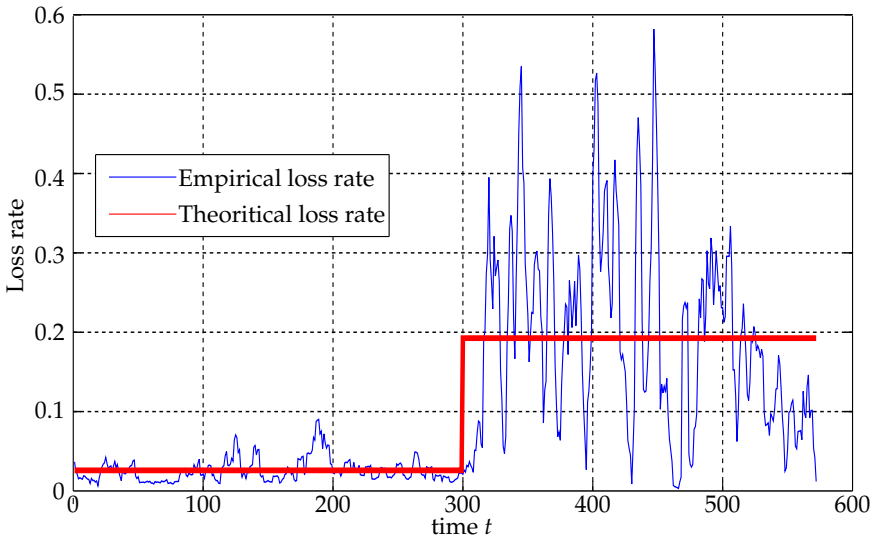


FIGURE 4.6: Theoretical and empirical loss rate

theoretical PFA given in (4.52). This figure shows that empirical results match theoretical ones in the range of $PFA > 5.10^{-3}$, demonstrating the accuracy of our model under $\mathcal{H}_0$. In addition, PFA for different detector's configurations is close, exhibiting the flexibility in guaranteeing a prescribed PFA under various configurations.

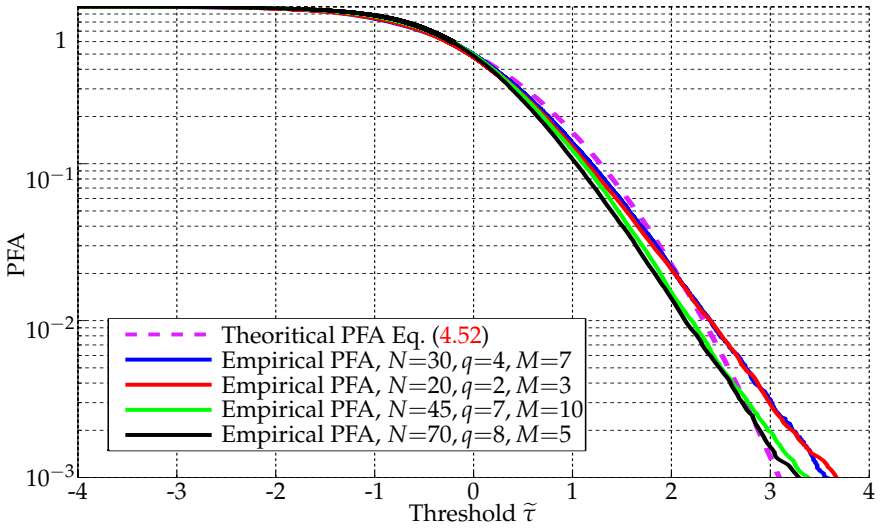


FIGURE 4.7: PFA as a function of threshold $\widetilde{\tau}$ for various detector's configurations, including window size N, polynomial degree q and number of corrupted samples $\|\mathbf{v}_a\|$.

Sequential Detector Performance

To emphasize the advantages of gathering consecutive samples, Figure 4.8 and 4.9 compare the performance of the snapshot test in Eq. (4.51) with the proposed sequential detection method based on the CUSUM in Eq. (4.56) on two different aspects. More specifically, Figure 4.8 illustrates the detection power under a maximum

detection delay constraint $\mathbb{P}\left[S_t - \nu \leq M_{\max}\right]$ as a function of the average RL2FA, see (4.55). The maximal detection delay is set to $M_{\max} = 10$ seconds. On the other hand, Figure 4.9 depicts the average detection delay, i.e., $\mathbb{E}[S_t - \nu | S_t \geq \nu]$, as a function of the average RL2FA.

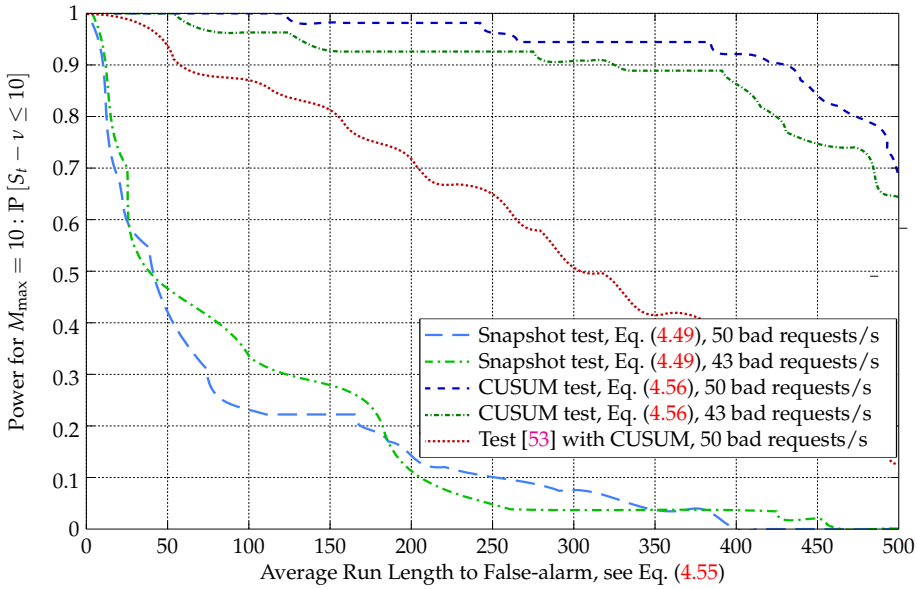


FIGURE 4.8: Power of sequential detection method (probability of detection with maximal constraint delay) as a function of average RL2FA.

The comparison is made on both figures for two different attack payloads, 43 and 50 bad requests/s. For the CUSUM test, the constant κ is set to 0.005 which corresponds roughly to the 1% largest values of detection statistics $\widetilde{\mathbf{v}}_a^T \widetilde{\mathbf{r}}$ under $\mathcal{H}_0$ (see Figure 4.2). For the snapshot test, the detection delay parameter M is set to $10s$. To be exhaustive, these figures also offer a comparison with the proposed detector in [53] by replacing the proposed GLRT statistics $\mathbf{v}_a^T \widetilde{\mathbf{r}}_t$ with the test [53] in the CUSUM equation (4.56).

Figure 4.8 and 4.9 clearly show that the significant gain obtained by gathering consecutive samples. More specifically, Figure 4.8 exhibits that, for an average RL2FA of 300 seconds (5 minutes), the probability of detecting an IFA after 10 seconds increased from roughly 5% for the snapshot test to more than 90% using the CUSUM procedure. Meanwhile, the test [53] with CUSUM only achieves 50% probability of detecting an IFA with the same constraint. Similarly, Figure 4.9 shows that, for the same average RL2FA of 300 seconds, the average detection delay is decreased by a factor of about 8, from about 40 seconds for the snapshot test to 5 seconds for the CUSUM test. Also, the average delay of the proposed CUSUM test is about six times lower than 28 seconds for the test [53] with CUSUM.

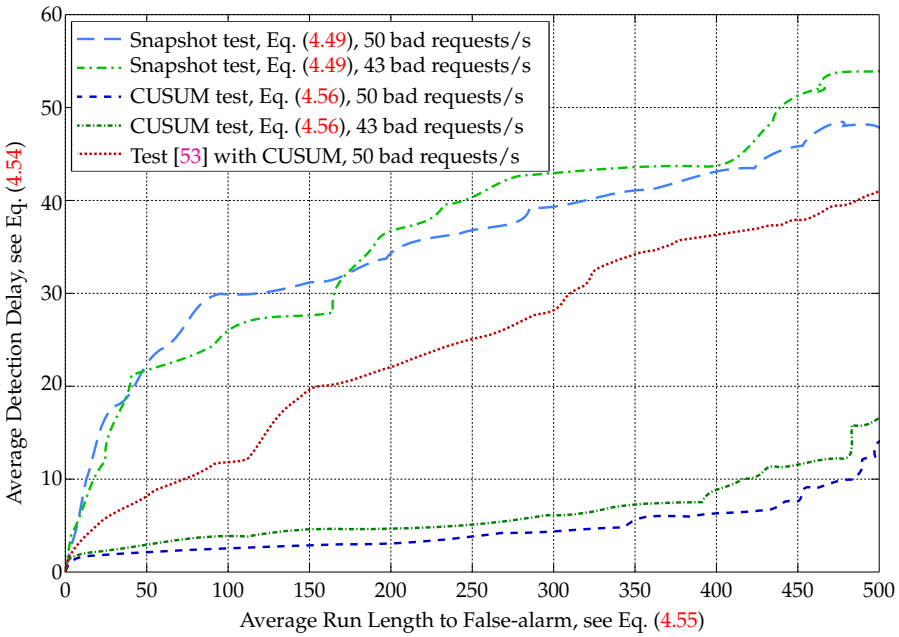


FIGURE 4.9: Average Detection Delay as a function of average RL2FA.

4.7 Conclusion

This chapter addressed the micro detector for the envisaged NDN security monitoring plane using hypothesis testing theory. The IFA (see Section 2.2.1) is leveraged as a use-case for evaluation. The metric utilized to detect IFA is the satisfaction ratio, inspired by existing works on the topic (see Section 2.2.1). The chapter first presented the basic framework of hypothesis testing theory. Then, we formally defined the detection problem and tackled it in the ideal case where the traffic parameter is known in advance. The result is an AUMP test serving as an upper bound for the performance that one can expect in other cases. Next, we addressed a more realistic scenario in which the traffic parameter is unknown, resulting in a GLRT. The detector was then extended to a sequential version using the well-known CUSUM to further improve the detection accuracy. To evaluate the detection scheme, we first assess the performance of the snapshot GLRT using simulation. Results exhibited the relevance of the approach and its advantage over the previous works, especially the state of the art IFA detection method in [53]. Finally, we evaluated the overall performance of our proposal with data collected from the real experimental testbed. Results showed the significant gain by the sequential detector, regarding the average detection delay and the probability of true alarm with the constraint of maximum detection delay.

It is noteworthy that introducing a mitigation once the detection has not been mature and reliable enough can waste the network resources and penalizing legitimate users, leading instability to the network. As such, in the next chapter, we

continue on developing a security monitoring plane for reliably capturing security anomalies, based on micro detectors design validated in this chapter. Therefore, obtained results are important because they demonstrate the relevance of our framework for designing micro detector.

Chapter 5

Toward A Security Monitoring Plane for Named Data Networking

Contents

In this chapter, we will tackle the third limitation, which is the lack of a generic security solution in the literature (see Section 2.6), by presenting our proposal for *Security Monitoring Plane* (SMP) of NDN. The proposed SMP must detect not only discovered NDN attacks in the literature but also ones that haven't been discovered so far. Due to CPA's elusive impact on multiple metrics (see Chapter 3), it is chosen as a use-case to evaluate our proposal. We first motivate for an SMP in NDN. Secondly, to facilitate the reading of this chapter, we will provide background knowledge, including the network management plane, NFD management protocol and

principles of *Baysian Networks* (BNs) – the methodology used to design our proposed SMP. Thirdly, we present our main contributions, including (1) an exhaustive list of metrics that potentially feature the status of an NDN node by inspecting router's components; (2) a generic micro detector that analyses each metric individually to notify its abnormal behaviors and (3) a *Bayesian Network Classifier* (BNC) that correlates the alarms of micro detectors. Finally, the proposed BNC's performance against CPA is demonstrated with data from our real NDN testbed. Numerical results show the relevance of our micro detectors, as well as the effectiveness of the classifier, regarding the number of samples for training, the detection delay, the accuracy and its trade-off with the detection window.

The work in this chapter is conducted in collaboration with our partners from Montimage[1], France, and especially Hoang-Long Mai whose PhD. is co-supervised by UTT, Montimage, and LORIA. We design the generic micro detector while our partners run experiments and develop the mechanism to extract metrics. Jointly, we come up with the metric list, the proposed BNC and analyze the obtained results. The contribution in this chapter has been presented as a full paper in an international conference, see [123].

5.1 Motivation for a Security Monitoring Plane in NDN

Although a dedicated detector can yield good performance for a particular attack, the designing process is time-consuming and requires effort. Moreover, such a detector is inapplicable to other threats because of differences in attacks' characteristics, i.e., some attacks' impact can be featured by a sole metric, such as IFA, while others are more elusive and can hardly be characterized with a single metric. Therefore, the network operator needs a generic SMP that can tackle not only identified attacks in the NDN's literature but also ones that possibly emerge in the future.

The Security Information and Event Management (SIEM) is an industrial approach for security management that is relevant to our envisaged SMP. This approach combines functions of *Security Information Management* (SIM) and *Security Event Management* (SEM) into one system. A SIEM slution provides the following principal functions [124]:

1. Collecting and aggregating logs from multiple sources of the system;
2. Continuously monitoring incidents;
3. Correlating logs and events to detect security threats;

[1] See http://www.montimage.com

4. Issuing alert notifications.

The proposed SMP mainly address the last two functions of SIEM, i.e., correlating logs to detect security threats and issuing alerts, since they bring novelty and add a more scientific contribution to the chapter. More specifically, we focus on proposing a metric list to be monitored in an NDN node, designing a generic micro detector and providing a correlation engine based on BN to locally detect potential security threats.

5.2 Background

In order to facilitate the reading, this section provides some background knowledge. We first provide an overview of the network management plane which is the content of this chapter. Secondly, we present an existing work on NFD management protocol. Thirdly, we introduce principles of BN, the mathematics tool that we used for our proposed SMP.

5.2.1 An Overview of Network Management Plane

An essential requirement in networking is that the network operator must be able to manage his system and guarantee its operation in order to delivery as good as possible experience to users. As such, a management plane is indispensable. The management plane is the element of the system that monitors data on performance and traffic characteristics, diagnoses possible problems and helps the network operator configure the network accordingly to environmental conditions.

Organization of Network Management Plane

From an organizational perspective, the management plane can be distributed across administrative levels [125], forming a hierarchical paradigm. For each level, there is a *manager* – a program responsible for managing the corresponding system [126]. The network operator will interact with the top-level manager to get the broad view of the whole system. At the lowest level is the *agent* – a program running in a managed element, for instance, a network device or a component of a distributed system [127]. The agent is in charge of monitoring and reporting required information of the managed element and carrying out specific tasks instructed by its manager.

Principal Functions in Network Management

From the functional point of view, the network management can be divided into five principal functions as follows [125]:

- *Fault management*: involves in detecting problems in the managed object, run diagnostic tests and correct the faults;
- *Configuration management*: permits network managers to control over the managed object's configuration to reduce congestion, isolate faults or match user needs;
- *Performance management*: consist in monitoring and evaluating the performance of the managed objects;
- *Accounting management*: allow the network manager to determine and allocate cost and charges for the use of his resources;
- *Security management*: relate to services that protects the network resources and entities from security threats.

Since our main interest is the security of NDN's data plane (see Section 2.1), the contribution of this chapter will mainly focus on the *security management* function.

5.2.2 Related Work - NFD Management Protocol

NFD management protocol provides the capability to monitor and control NFD. It defines several management modules; each is in charge of an NFD's component. There are four major management modules, described in the following [128]:

- *Forwarder Status*: provides information about NFD and basic statistics about the forwarder, such as NFD's version, startup time, *Interest*/*Data*/*NACK* packet counts;
- *Face Management*: provides commands to create, destroy faces and change attributes of an existing one. This module also maintains a dataset of faces and their counters, channels and offers a notification stream for face creating and destroying events;
- *FIB Management*: support commands to insert, update, and delete FIB entries and nexthop records. Besides, it maintains a dataset of FIB entries and nexthop records;

- *Strategy Choice Management*: offers commands to choose the forwarding strategy (see Section 1.3.2) for a namespace and maintains a dataset of strategy choices.

To convey a management action that modifies NFD state, it is obligatory to use a control command[2], a form of *signed Interests* – *Interests* that are authenticated by embedding a signature into the last component of its name. As a result, NFD can determine whether the issuer is authorized to perform the specified action. Upon receiving the control command, the management module replies with control responses to inform whether the command is successful or failed. Control responses have status codes and describe the action that was performed or any errors that occurred. Management actions that query the current state of NFD do not need to be authenticated. These actions are defined in NFD Management Protocol as status datasets and are currently implemented in NFD as a simple *Interest*/*Data* exchange.

We remark that the NFD management protocol mainly focuses on the control capability for NFD. Other existing works are available [129, 130], but insufficient for security monitoring, and especially for data plane security since its database for data-plane-related components (CS and PIT)is limited to a few metrics. As such, for the proposed SMP, we plan to propose an exhaustive metrics list that is more relevant for security monitoring. Also, for metrics that are unavailable in NFD management protocol, we intend to collect them from NFD log.

5.2.3 Bayesian Network

A Bayesian Network (BN) is a popular graphical model that encodes probabilistic relationships among a set of correlated random variables [131–133]. BN has been used in a wide range of applications that involve making a decision based on collected and processed data, such as diagnostics [134], predictive analytics [135] and classification. The reasons for our choice of BN are many. First is its ability to correlate most of the events with their impacts on a small set of metrics. By repeating the process for all metrics, BN eventually leverages all of those relations to classify the event under observations. Secondly, BN allows visualizing causal relationships between monitored metrics [136] with its structure. Our domain expertise in NDN protocol and implementation, hence, can be leveraged and easily integrated to the anomaly detection. Finally, BN allows designing the anomaly detection at multiple levels, e.g., local detectors for aggregating local metrics, as well as a global detector for combining local detectors' alarms.

[2]See: https://redmine.named-data.net/projects/nfd/wiki/ControlCommand.

Principles of Bayesian Networks

Let $\boldsymbol{B} = (\mathcal{S}, \boldsymbol{\theta})$ be a BN which structure $\mathcal{S}$ and parameters $\boldsymbol{\theta}$. The structure S [131, Definition 2.3] consists of a set of nodes $\boldsymbol{X}$ and a set of directed edges $\boldsymbol{E}$. Each node $X_i \in \boldsymbol{X}$ represents a random variable X_i whose values can be discrete or continuous. An edge from node X_i to node X_j represents a statistical conditional dependence between the corresponding variables. As such, X_i is called a parent of X_j and X_j is called a child of X_i. For a BN structure, the nodes in $\boldsymbol{V}$ together with the directed edges in $\boldsymbol{E}$ form a *Directed Acyclic Graph* (DAG) – a graph contains no directed path $X_i \rightarrow \ldots \rightarrow X_j$ that forms a circle, i.e., $X_i = X_j$. Moreover, each variable X_j is associated with a *Conditional Probability Distribution* (CPD) that characterizes its relationship with its parents, defined as:

$$\mathbb{P}(X_j | pa(X_j)), \tag{5.1}$$

where $pa(X_j)$ includes all parent nodes of X_j. When variables are discrete, CPD takes the form of a table, hence called *Conditional Probability Table* (CPT). If X_i has no parents, its CPD becomes the prior probability distribution $\mathbb{P}(X_i)$. The set of all CPT of a BN is referred as its parameters $\boldsymbol{\theta}$. An evidence $\mathbf{e}$ is a vector $(X_1 = x_1, \ldots, X_n = x_n)$ containing observed values of (but not necesarily all) variables. Some variables value in an evidence can be missing. An evidence $\mathbf{e}$ is considered to be "complete" if all variables' values are observed, otherwise it is considered as "incomplete".

Joint Probability Distribution in a Bayesian Network: a joint probability distribution over a set of random variables $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ is a function that specifies the probability $\mathbb{P}(X_1 = x_1, \ldots, X_n = x_n)$ for every combination of values of $\{X_1, \ldots, X_n\}$ [133, Definition 1.8]. Besides, a joint probability must satisfy the following conditions:

$$\begin{aligned} 0 \leq \quad & \mathbb{P}(X_1 = x_1, \ldots, X_n = x_n) \quad \leq 1, \\ \text{and} \quad & \textstyle\sum_{x_1,\ldots,x_n} \mathbb{P}(X_1 = x_1, \ldots, X_n = x_n) \quad = 1, \end{aligned}$$

where $\sum_{x_1,\ldots,x_n}$ means the sum as the variables $x_1, \ldots, x_n$ go through all possible values in their corresponding spaces.

One should note that, as the number of variables increases, the number of combinations will grow exponentially, hence calculating $\mathbb{P}(\boldsymbol{X})$ becomes more difficult. However, BN allows computing the joint probability distribution more compactly

with its chain rule [131, Theorem 2.1], defined as:

$$\mathbb{P}(\boldsymbol{X}) = \prod_{i=1}^{n} \mathbb{P}(X_i | pa(X_i)).$$

Bayesian Network Classifier: one of the applications of BN is for classification. A Bayesian Network Classifier (BNC) is a BN used for classification in which one of its nodes takes values in a finite set $\mathcal{C}$ of all possible classes or events that one needs to distinguish [137]. Given a BN structure $\mathcal{S}$ and an evidence $\mathbf{e}$, BNC will return the class $\widehat{c} \in \mathcal{C}$ that has the maximum posterior estimation $\widehat{c} = \max_{c \in \mathcal{C}} \mathbb{P}(c | x_1, \ldots, x_n)$. In general, the observed metrics in computer networking are not entirely predictable. Based on a probabilistic on stochastic framework, BN can naturally handle the underlying uncertainty of observed metrics. It is valuable to know how certain the classifier is about its classification results. Such insight allows better controlling the cost of making errors.

Building a Bayesian Network

Despite its broad application, constructing an appropriate BN remains a challenging task mainly due to the complexity of real-world problem domains [138]. In this subsection, we briefly discussed two major tasks when building a BN: (1) establishing the BN structure (i.e., the qualitative part) and (2) estimating the parameters (or the CPD, i.e., the quantitative part).

Building Structure: to build the BN structure $\mathcal{S}$, one first needs to identify variables. In our case, those variables are metrics that we will identify for the proposed SMP. Afterward, causal relationships are integrated to the graph in the form of directed edges between variables. When the domain experts are unavailable, the BN structure can be learned from data. Several BN structure learning algorithms have been proposed and can be categorized into three approaches:

- Constraint-based approach [139, 140]: this approach tests the conditional independence between variables, attempting to search for a structure that is consistent with the observed dependencies and independencies;
- Score-based approach [141, 142]: algorithms in this category define a score and a search method. The score is a measure of the goodness-of-fit between the probability distribution of the candidate network and the true joint distribution implied by the data. A searching method then looks for the structure that maximizes the score;

- Hybrid approaches [143, 144] combine the previous approaches.

The structure obtained from these algorithms, however, will depend on the characteristics of the dataset. When the domain experts are available, the BN structure can be constructed manually. This holds true for our case in which the expertise of NDN protocol and implementation can be leveraged to reason causal relationships between identified variables.

Parameter Estimation: after the structure has been decided, the BN parameters $\boldsymbol{\theta}$ can be assigned manually by the domain expert. However, in most of the cases, the parameters are estimated from data. Let $\mathcal{D}$ be a dataset of evidences. A dataset $\mathcal{D}$ is "complete" if all of its evidences are complete. Because all metrics' values can be monitored, we only considered parameter estimation approaches for complete data. For complete data, there are two major parameter estimation approach: maximum likelihood estimation and Bayesian estimation [131]. While the Bayesian estimation requires with a prior probability distribution, the maximum likelihood estimation does not require such knowledge. In our problem, since we do no prior probability distribution of any nodes, we decide to use the maximum likelihood estimation to learn the parameters $\boldsymbol{\theta}$. If we assume that all cases in $\mathcal{D}$ are independent given the model, then the likelihood of $\boldsymbol{B}$, given the dataset $\mathcal{D}$, is defined as:

$$\mathcal{L}(\boldsymbol{B}|\mathcal{D}) = \prod_{d \in \mathcal{D}} \mathbb{P}(d|\boldsymbol{B}). \tag{5.2}$$

The estimated parameters $\hat{\boldsymbol{\theta}}$ are ones that maximize the likelihood of $\boldsymbol{B}$, given dataset $\mathcal{D}$:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{B}_{\boldsymbol{\theta}}|\mathcal{D}). \tag{5.3}$$

Further details and other algorithms for parameter estimation with incomplete data can be found in [131, 133, 145].

Inference in a Bayesian Network

As a follow-up of BN's structure building and parameter estimation, inference in BN is the process of computing a posterior probability distribution of unobserved variables, given the values of observed ones as evidence [146]. Algorithms for inference in BN can be categorized as *exact* or *approximate*. The exact inference, as referred by its name, provides an accurate value (i.e., having a closed-form solution) for the posterior probability by combining repeated applications of Bayes's theorem with local computations. *Junction Tree* (JT) [147] is one of the best-known exact inference algorithms. Nevertheless, the exact inference may be time-consuming, or sometimes

a closed-form solution does not exist. Hence, its feasibility is limited, mostly when all nodes are discrete or have Gaussian distribution [148]. For BNs that are large and complex, the exact inference is impractical.

Alternatively, the approximate inference yields an inexact solution that has high probability being within a small distance to the correct answer [149]. This approach sacrifices the accuracy for a quick estimation of posteriors and broad applicability. A few approximate inference methods include stochastic simulation algorithms, model simplification methods, search-based methods and loopy belief propagation. The details of major steps of these inference algorithms are beyond the scope of this thesis. A detailed explanation on inference in BN and inference algorithms can be found in [131–133, 145, 146, 150, 151].

In our proposed SMP, we envisage to use BN to correlate alarms of micro detectors whose output value is discrete (i.e., *no alarm* or *alarm*). As such, all nodes in our BN are discrete, thus making the exact inference algorithm feasible. Moreover, an exact inference algorithm, especially JT algorithm, can yield an accurate result compensating for the loss in accuracy of the generic micro detector design.

5.3 A Bayesian Network Classifier for Anomaly Detection in NDN

This section presents the first elements towards an SMP for NDN. First, a comprehensive list of metrics to be monitored in an NDN node is presented. We design a micro detector to raise the alarm whenever a metric significantly shifts from its normal behavior. Finally, the results from all micro detectors are combined in a correlation engine based on BN whose structure is built on the thorough expertise of both NDN specification and implementation.

5.3.1 An Exhautive Metrics List for NDN Monitoring

This subsection introduces a list of metrics to be monitored in an NDN node. Such a list must be able to feature the node's behavior and to distinguish abnormal traffic and helps identify the underlying type of event. To build an exhaustive metrics list, all relevant components inside an NDN node are considered, including (1) *Faces*; (2) CS and (3) PIT. We deliberately decide not to cover the FIB in our metrics list because our main interest is the security of NDN data plane (see Section 2.1) while the FIB belongs to the control plane. Also, we argue that the FIB malfunctions can be indirectly captured by other metrics that we describe as follows.

Through *Faces* (see Section 1.3.2), an NDN router receives and forwards packets (i.e., *Interest*, *Data*, *NACK*). Apparent metrics for this component include *In Interest*, *In Data*, *In NACK*, *Out Interest*, *Out Data*, *Out NACK*, which are the numbers of incoming and outgoing packets in the sampling period. These metrics, which are similar to *Simple Network Management Protocol* (SNMP) [152] counters, allow determining various traffic's characteristics such as the volume, the frequency, the correlation between requests (*Interest*) and contents (*Data*). Moreover, since the router can drop packets according to its strategy, it is also proposed to monitor the number of dropped packets (i.e., *Drop Interest*, *Drop Data*, *Drop NACK*). Because it is not expected from nodes to send and receive dropped packets, such metrics can help reveal an anomaly in NDN operations.

The CS is NDN router's local cache. During its operation, cache misses and hits occur. Depending on the cache replacement policy, the router can decide to insert a new *Data* into the CS. Hence, for the CS, we monitor the number of occurrences of the miss (*CS Miss*), hit (*CS Hit*) and insert (*CS Insert*) events during a specified interval. Because a cache usually stores popular content to improve the delivery performance, changes in those metrics can reveal information related to the content popularity, e.g., when users prefer to watch a new trending video, or when a router is forced to cache unpopular content.

The PIT is a database where an NDN router tracks valid *Interest* it forwarded as well as reverse-path forwards *Data* packets. Apparent metrics for such a component include the number of PIT entries created (*PIT Create*), deleted (*PIT Delete*) during a time interval and the current number of entries (*PIT Number*). Moreover, since the NDN router aggregates *Interest* for the same content, created entries may be updated during the operation. Thus we also monitor *PIT Update*, the number of updates in the PIT per interval. Besides, an *Interest* has a lifetime period, indicating the time it can exist in the PIT, waiting for a *Data*. If there is no *Data* or *NACK*, the matching PIT entry expires, and the request becomes unsatisfied. Such a situation is unlikely and probably related to abnormality, hence the number of its occurences should be monitored (*PIT Unsatisfied*). Furthermore, the *Interest* lifetime can be tuned by NDN users, making it stay longer in the PIT. Deliberately increasing this lifetime could be an attempt to launch an attack (e.g., IFA [93]). Such an attack type on the forwarding plane can be featured by the entry's existing time (i.e., time elapsed since the entry is created until its removal). Beyond, such information can also be relevant to network latency issues. To feature the existing time of entries in the PIT, *PIT Exist Time* stands for the average of each value considered in the sampling period.

Table 5.1 summarizes the proposed metric list and their descriptions.

TABLE 5.1: List of metrics to be monitored in an NDN node

	Metric	Description
Faces	*In Interest*	Periodic number of incoming *Interest*
	In Data	Periodic number of incoming *Data*
	In NACK	Periodic number of incoming *NACK*
	Out Interest	Periodic number of outgoing *Interest*
	Out Data	Periodic number of outgoing *Data*
	Out NACK	Periodic number of outgoing *NACK*
	Drop Interest	Periodic number of dropped *Interest*
	Drop Data	Periodic number of dropped *Data*
	Drop NACK	Periodic number of dropped *NACK*
CS	*CS Insert*	Periodic number of cache insert
	CS Miss	Periodic number of cache miss
	CS Hit	Periodic number of cache hit
PIT	*PIT Create*	Periodic number of PIT entries created
	PIT Update	Periodic number of updates in PIT
	PIT Delete	Periodic number of PIT entries deleted
	PIT Unsatisfied	Periodic number of PIT entries unsatisfied
	PIT Number	Current number of PIT entries
	PIT Exist Time	Average of PIT entries' existing time

5.3.2 Generic Micro Detector Using Hypothesis Testing Theory

The metric list stands for a quantitative measure to estimate the status of an NDN node. Any aberration of these metrics could be a clue about an ongoing anomaly. Therefore, in this subsection, a generic micro detector is presented to detect any significant change of a metric from its normal behavior. The methodology used to build these micro detectors is the statistical hypothesis testing theory with the Neyman-Pearson two-criteria approach that has been presented in Chapter 4.

Let us denote x_i , $i = \{1, \ldots, t\}$ a metric value observed at a time instant i. Considering the necessity for simple micro detectors, we deliberately decided to model all the metrics using the normal (Gaussian) distribution which belongs to the family of exponential distributions. This distribution has already been presented in Chapter 4 on IFA detection in NDN and has shown its accuracy on real data of a single metric. It is also assumed that an anomaly is expected to change the average values of metrics much more than their variance. Using such a model for all metrics is a trade-off between accuracy and simplicity.

The problem considered at micro detector level is to identify whether a metric

value (1) lies within its normal behavior, (2) significantly decreases or (3) significantly increases. These three cases are defined, respectively, by the following statistical hypotheses:

$$x_t, \dots x_{t-n+1} \sim \begin{cases} \mathcal{N}(\mu_0; \sigma^2) & \text{under } \mathcal{H}_0, \\ \mathcal{N}(\mu_1; \sigma^2)\,,\ \mu_1 < \mu_0 & \text{under } \mathcal{H}_1, \\ \mathcal{N}(\mu_2; \sigma^2)\,,\ \mu_2 > \mu_0 & \text{under } \mathcal{H}_2, \end{cases} \tag{5.4}$$

where n is the window size considered for the detection; μ_0, μ_1, μ_2 are the means of normal distributions in the corresponding cases, and σ^2 is the variance of the metric. The problem presented in (5.4) can be addressed easily by using a straightforward extension of Neyman-Pearson approach for multiple hypotheses referred to as "minimax constrained test," see details in [116, 153], whose solution is presented merely here by the following test:

$$\delta\left(x_t, \dots x_{t-n+1}\right) \begin{cases} \mathcal{H}_0 & \text{if } \tau_1 \leq \sum_{t-n+1}^{t} x_t \leq \tau_2. \\ \mathcal{H}_1 & \text{if } \sum_{t-n+1}^{t} x_t < \tau_1, \\ \mathcal{H}_2 & \text{if } \sum_{t-n+1}^{t} x_t > \tau_2, \end{cases} \tag{5.5}$$

The thresholds τ_1 and τ_2 that guarantee the prescribed PFA are established as follows:

$$\tau_1 = \Phi^{-1}\left(\alpha_0/2\right)\sqrt{n}\sigma + n\mu_0 \tag{5.6}$$

$$\tau_2 = \Phi^{-1}(1-\alpha_0/2)\sqrt{n}\sigma + n\mu_0 \tag{5.7}$$

where Φ and Φ^{-1} respectively represent the standard normal cumulative distribution function and its inverse function; α_0 is the desired PFA of the micro detector. Eqs. (5.6) and (5.7) show that the thresholds τ_1 and τ_2 are functions of the prescribed PFA α_0, the window size considered for the detection n, the estimated mean and variance of metric's distribution under normal behavior, respectively, μ_0, σ^2. While α_0 and n are chosen based on the requirements for the micro detector, μ_0, σ^2 can be estimated from metric's normal behavior. In short, the threshold τ can be computed in advance and guarantees the desired PFA, regardless the metric's behavior under attack. Moreover, using the decision threshold given in (5.6)–(5.7), the detection power of the micro detectors is provided by:

$$\beta_1(\mu_1) = \Phi\left[\Phi^{-1}\left(\frac{\alpha_0}{2}\right)\sqrt{n}\sigma + \sqrt{n}\frac{\mu_0-\mu_1}{\sigma}\right] \tag{5.8}$$

$$\beta_2(\mu_2) = 1 - \Phi\left[\Phi^{-1}\left(1-\frac{\alpha_0}{2}\right)\sqrt{n}\sigma + \sqrt{n}\frac{\mu_0-\mu_2}{\sigma}\right] \tag{5.9}$$

where μ_1, μ_2 is the mean of metric's distribution under $\mathcal{H}_1$ and $\mathcal{H}_2$, respectively. Note that these values are always unknown as they depend on the attack payload, and so does the detection power. As explained above, the only parameters that can be set by the operator are α_0 and n. While α_0 represents the PFA, the number of samples used n can be tuned to find a trade-off between quick and accurate detection. Although increasing n can enhance the detection power (5.8)–(5.9), it moves the thresholds (5.6)–(5.7) apart from μ_0 and hence delays the detection. On the other hand, decreasing n reduces the detection delay at the cost of lower detection power or higher probability of missed detection.

5.3.3 Alarms Correlation Engine Using Bayesian Network Classifier

Various attack types lead to different effects on a metric. For instance, IFA mostly impacts the packet loss rate while CPA impacts cache metric, as featured in Chapter 3. As stated in the previous section, applying the normal distribution with a constant variance to all metrics is a trade-off between the accuracy and the simplicity of the micro detector. Thus, alarms from a single micro detector cannot timely detect and characterize an occurring anomaly in an NDN node entirely. In this section, we compensate for this loss of accuracy by correlating alarms from micro detectors.

We propose to sketch the BN structure based on NFD forwarding pipelines, instead of using BN structure learning algorithm. Analyzing NFD forwarding pipelines reveals the causal (cause and effect) relationships between metrics. Directed edges are then drawn from the causing metrics to the effected ones. We argue that such an approach yields a more reliable and robust BN structure since it is grounded on our NDN expertise and the knowledge about NFD [5]. Meanwhile, learning BN structure from data requires an adequate dataset covering all possible node behaviors in real life, which is not feasible in an experimental environment.

A forwarding pipeline (or pipeline for short) is a series of steps that operate on a packet or a PIT entry, triggered by a specific event. In NFD, there are thirteen pipelines in total. They can be divided into two categories: *externally initiated* and *internally initiated*. The former includes pipelines that are triggered by an external event, e.g., incoming packet, while the latter is triggered by a step in an *externally initiated* pipeline. We focus on describing the first category, including: (1) *Incoming Interest*; (2) *Interest unsatisfied*; (3) *Incoming Data* and (4) *Incoming NACK*. Other pipelines that are mentioned in the following descriptions belong to the *interally initiated* category. One can find more detail about them in [5]. It is worth noting that the actual pipelines in [5] cover many details in the NFD practical implementation. For the sake of conciseness, we deliberately simplify the pipelines to retain the most relevant information for the proposed metric list, as well as to keep the BN explanation

straightforward and understandable. It is noteworthy that despite their similarity, the pipelines descriptions in this section provide more technical details in NFD implementation as compared to Figure 1.1 whose purpose is to exhibit the primary operation of NDN router.

Incoming Interest pipelines

Figure 5.1 illustrates the *incoming Interest pipeline*. When an *Interest* arrives, NFD first checks and drops it if it violates the `/localhost` prefix which is reserved for internal communications between components. By using the recorded *Name* and *Nonce* in PIT entries, NFD can detect duplicated *Interests* (see Section 1.3.1). In such case, NFD drops the *Interest* and sends a *NACK* with a *Duplicate* error code to notify the downstream. Otherwise, NFD executes a PIT lookup and then either inserts a new PIT entry or updates the corresponding one by canceling its *unsatisfy timer*[3]. Hence, we deduce that *Out NACK*, *PIT Create*, *Drop Interest*, and *PIT Update* are affected by *In Interest*. Subsequently, NFD performs the CS lookup for a cached *Data* and enters the *CS Miss* or *CS Hit pipelines* accordingly, implying an influence of *In Interest* on *CS Miss* and *CS Hit*.

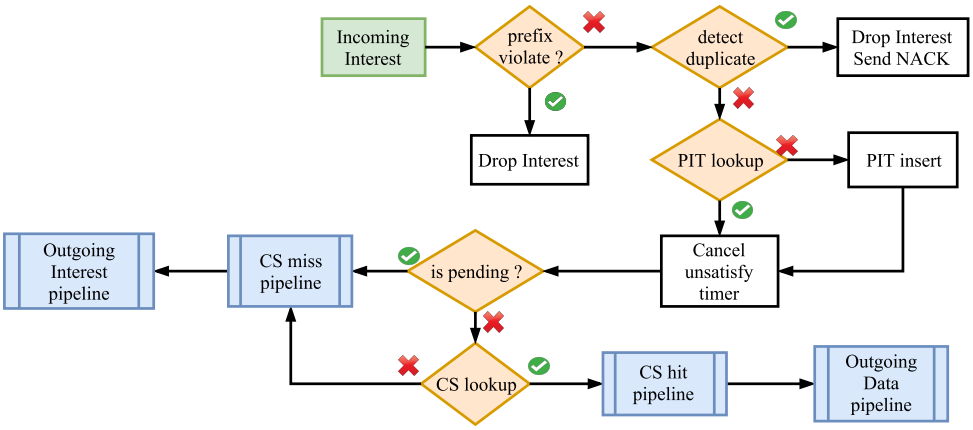


FIGURE 5.1: Simplified incoming *Interest* pipeline (modified from [5])

In case of a cache hit, the corresponding PIT entry will be removed after a while, and the matching *Data* is passed to the *outgoing Data pipeline* where NFD verifies and drop the cached *Data* packet if there is any prefix violation before sending it to downstream. As a result, *CS Hit* impacts *PIT Delete*, *Drop Data*, and *Out Data*.

In the *CS Miss pipeline*, the corresponding PIT entry is updated once again by adding the *Interest*'s incoming face and setting the *unsatisfy timer*. Hence, *CS Miss* also affects *PIT Update*. Besides, the *Outgoing Interest pipeline* is triggered. Once again, the prefix violation will be verified before being forwarded to other nodes.

[3] A timer indicates the lifetime of a PIT entry. Because a new valid *Interest* arrives for the PIT entry, so that the lifetime of the PIT entry needs to be extended [5]

Therefore, *CS Miss* influences *Out Interest* and *Drop Interest*. Besides, it is noteworthy that *PIT Exist Time* and *PIT Number* also change whenever a PIT entry is removed or created. Hence, both *PIT Create* and *PIT Delete* impact *PIT Exist Time* and *PIT Number*.

Interest unsatisfied pipeline

After forwarding an *Interest*, the NDN node waits for a *Data* or *NACK* packet from the upstream, but only for a while. Each entry in the PIT has an *unsatisfy timer*. When this timer expires, NFD considers that the PIT entry was already alive for a substantial duration since no upstream node can satisfy the *Interest*. Thus, NFD removes the entry from the PIT. Hence, *PIT Unsatisfied* causes *PIT Delete*.

Incoming Data pipeline

Figure 5.2 depicts the *incoming Data pipeline*. When a *Data* arrives, NFD first checks and drops *Data* packets if there is a prefix violation. NFD then verifies whether the *Data* matches any PIT entry. If no matching PIT entry is found, the *Data* is considered unsolicited. Depending on the policy of NFD, unsolicited *Data* can be dropped or inserted in the CS. Otherwise, the *Data* is inserted in the CS. Note that even if the pipeline decides to insert the *Data* to the CS, whether it is stored and how long it will stay in the CS is determined by CS admission and replacement policy [5]. Thus, *In Data* affects *Drop Data* and *CS Insert*.

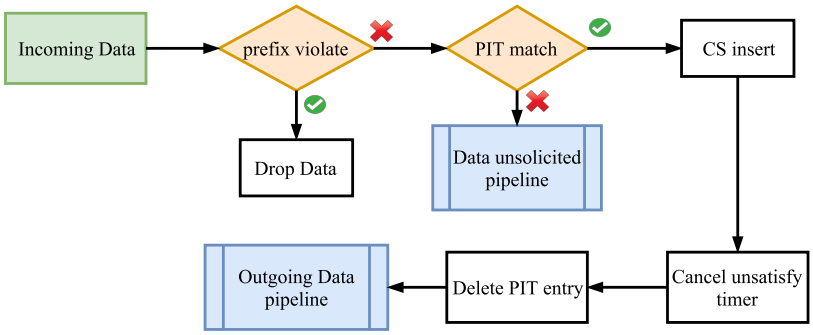


FIGURE 5.2: Incoming *Data* pipeline (modified from [5])

When a *Data* is inserted into the CS, NFD will cancel the *unsatisfy timer*[4] for each matching PIT entries, implying PIT updates. After a while, the corresponding PIT entry is deleted, and the *Data* packet is passed to the *Outgoing Data pipeline*, where NFD verifies and drops the *Data* if there is any prefix violation before it is forwarded to downstream. Thus, *CS Insert* impacts *PIT Delete*, *Drop Data*, *Out Data* as well as *PIT Update*.

[4]Because the pending *Interest* is now getting satisfied [5]

Incoming NACK pipeline

When a *NACK* comes, NFD will look for a matching PIT entry. The *NACK* will be dropped if there is no relevant PIT entry. Otherwise, the corresponding PIT entry is updated by erasing all records about incoming faces of *Interests*, indicating *PIT Update*, before sending an outgoing *NACK* to downstream nodes. Therefore, *In NACK* influences *Drop NACK*, *Out NACK*, and *PIT Update*.

Proposed Bayesian Network Structure

Considering all the dependencies described above between metrics, we integrate them in the BN structure and depict the result in Figure 5.3. Each node corresponds to a micro detector associated with a specific metric. The nodes are colored based on the router component to which they belong. The *Anomaly* node represents the anomalies that can occur in the NDN network, and its value will be the output of the classifier, e.g., "IFA", "CPA".

5.4 Numerical Result

In this section, we explain the mechanism used to extract metrics from NFD logs. Next, we present the topology and the four scenarios we have considered to evaluate the proposed BNC, followed by setups implemented for our experiments. The micro detector's model relevance is then evaluated. Afterward, we address the learning efficiency of BNC with cross-validation. Finally, the performance of BNC is demonstrated regarding the impacts of the attack rate, the attack scenario, and the BNC location, as well as how it can be improved by tuning the detector window.

5.4.1 Metric Extraction Mechanism

The NFD Management Protocol (see Section 5.2.2) enables collecting data related to the status of an NDN node (e.g., *In Interest*, *PIT Number*). However, given the metric list that we need to collect, these statistics are not sufficient. To the best of our knowledge, currently, there is no mechanism to collect metrics that are unavailable in NFD Management Protocol, such as *CS Hit*, *CS Miss*, *CS Insert*, *Drop Interest*, *Drop Data*, *Drop NACK*. Therefore, our partners from Montimage has developed an extension for *Montimage Monitoring Tool* (MMT) probe to collect those metrics. Since NFD is still under development, they avoid modifying its implementation. Instead, MMT probe runs independently from NFD and extracts the necessary information from the NFD log.

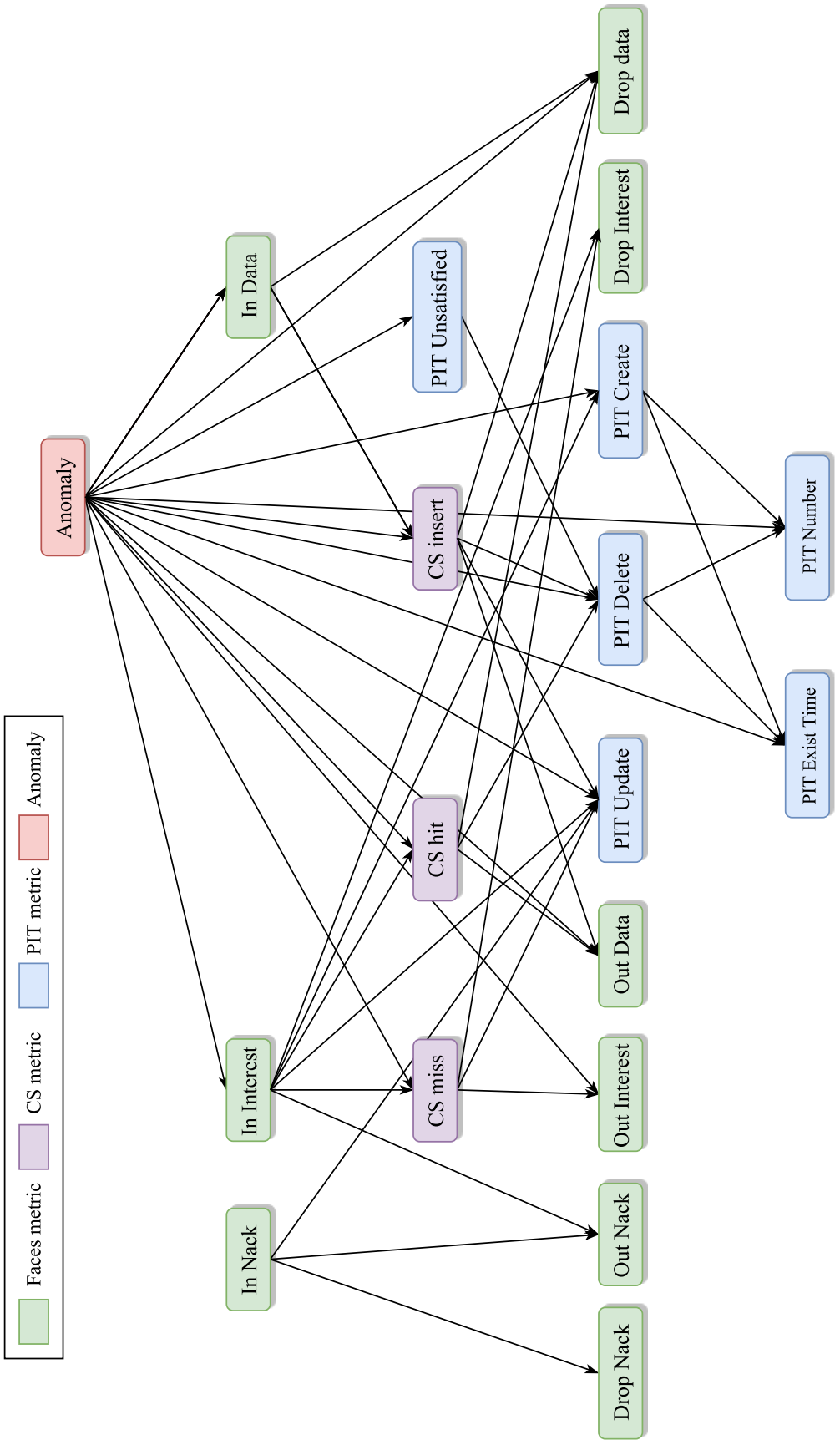


FIGURE 5.3: The proposed Bayesian Network Classifier's Structure

(1)	(2)	(3)	(4)	(5)	(6)
1503332255.605719	DEBUG	[Forwarder]	onIncomingInterest	face=264	interest=/com/good/content4
1503332255.605777	DEBUG	[Forwarder]	onContentStoreMiss		interest=/com/good/content4
1503332255.605933	DEBUG	[Forwarder]	onOutgoingInterest	face=265	interest=/com/good/content4

FIGURE 5.4: NFD log trace example

Figure 5.4 shows an example of NFD log trace. Each log line contains information about an event occurring in NFD, including: (1) the event's timestamp; (2) the logging level, indicating how much detailed the log is; (3) the NFD module that produces this log line; (4) the event name; (5) the face and (6) the corresponding *Interest*. Some metrics can be interpreted directly from these log entries, while others are deduced from the log. For instance, events such as *onIncomingInterest*, *OnContentStoreMiss*, *onOutgoingInterest* provide information to directly update, respectively, *In Interest*, *CS Miss*, and *Out Interest*, metrics. Although NFD does not record explicitly a PIT entry creation/update (*PIT Create/PIT Update*), it can be deduced if the incoming *Interest* packet (*onIncomingInterest*) is not found in the cache (*OnContentStoreMiss*) and is forwarded to the next hop (*onOutgoingInterest*).

5.4.2 Experiment Topology and Setup

Since we aim at dealing with sophisticated attacks like CPA, we reuse Section 3.4.3's topology (reproduced in Figure 5.5 for readability) and experiment configuration to evaluate the performance of the proposed BNC.

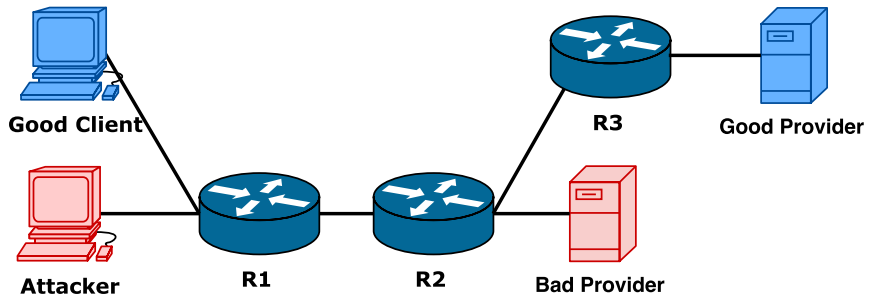


FIGURE 5.5: Use-case topology for Content Poisoning Attack

Experiment Scenarios

In order to evaluate the proposed BNC under various type of traffic, we reproduced four scenarios, namely *normal traffic*, *double traffic*, *CPA best route* and *CPA multicast*.

- *Normal traffic*: there is only legitimate traffic issued by a good client in this scenario. The number of *Interests* generated follows a Poison distribution, and the requested content is selected according to a Zipf's law. Those *Interests* are replied to by the *Data* from the good provider which connects through R3.

- *Double traffic*: in this scenario, two good clients are connected through R1, and a good provider is connected through R3. Each client in this scenario behaves like in *Normal traffic*. We consider this scenario not only to the explore metrics' behavior in case of abrupt changes in the network traffic but also to provide a learning dataset that challenges the BNC, by evaluating to what extent it can distinguish legitimate traffic changes against malicious ones.

The last two scenarios, *CPA multicast*, and *CPA best route* are already introduced in Section 3.4.2. For the CPA unsolicited scenario, we argue that it can be prevented with ease by a patch of NFD, and hence it is not reproduced in this evaluation.

Experiment Setup

An MMT probe is installed in each router to extract and collect data for our selected metrics. The dataset is then provided to micro detectors implemented in MATLAB. We especially utilized the MATLAB Bayesian Network Toolbox [148] for the implementation, parameter learning, and inference of the proposed BN. The BN structure has been built already in Section 5.3.3. Alarms from micro detectors are gathered to learn parameters of the proposed BN structure. Since the dataset is complete (i.e., all nodes can be observed) and we have no prior probability distribution of any nodes, we use the maximum likelihood estimation (Section 5.2.3) for parameter learning. Also, because micro detector's alarms are discrete and all nodes are observable, using exact inference is reasonable. To infer the value of *Anomaly* node from an observation of metrics, we utilized the junction tree engine [132].

For this evaluation, we need three datasets, namely (1) micro-detector-tuning, (2) BNC learning and (3) BNC testing dataset. The first dataset only has normal traffic and is collected during one week to tune the configuration for the micro detectors. Meanwhile, the BNC learning dataset is collected for scenarios with the following specific settings. The mean of the good client's *Interest* rate is the same and equals 10 *Interest*/s. The mean *Interest* rate of the second user (*Double traffic*) and the attacker (*CPA best route* and *CPA multicast*) also equal 10 *Interest*/s. The objective of this setting is to help BN differentiate between malicious and additional legitimate traffic even if the user and the attacker have the same rate. As such, the BNC will be trained to classify one of the three outputs, namely (1) normal, (2) additional traffic and (3) CPA.

For the BNC testing dataset, we gather the metrics from two scenarios: (1) *CPA bestroute* and (2) *CPA multicast*. For each scenario, we execute experiments with different attack rates in the range [1..100] Interests/s following a log scale. For each setting, five experiments were conducted. Each experiment lasts ten minutes and

has two periods. The first five minutes only has good client traffic, while the attack occurs during the second period. Constants in our experiments are summarized in Table 5.2. Since we now aim at evaluating the BNC and not featuring the CPA, one should note that these configurations are not entirely similar to what we used in Section 3.4.3.

TABLE 5.2: Bayesian Network classifier experiment constants

Constant	Value
Number of contents	10000 contents
Data's freshness period	4 seconds
Good provider link latency	100ms
Bad provider link latency	10ms
Clients' mean *Interest* rate by default	10 Interests/second
Zipf distribution factor	1.5
MMT probe's sampling period	5 seconds
Duration per experiment	10 minutes
Repetitions per attack rate	5
NFD version	0.5.1 (latest version by September 2017)

5.4.3 Micro Detector Evaluation

Relevance of the Micro Detector's Model

As mentioned in 5.3.2, due to the diversity of the behavior of the metrics when anomalies occur, we focus on correctly modeling metrics in normal traffic. Figure 5.6 depicts the kernel estimated density function for some illustrative metrics (*In Interest*, *CS Hit*, *PIT Number*) and their approximated normal distributions. The figure shows that for most of our metrics (e.g., *In Interest* and *CS Hit*), the empirical distribution is close to the normal distribution, indicating the relevance of the model. Nevertheless, the model does not fit well with some metrics (e.g., *PIT Number*), because their value range is close to zero and the variance is narrow. However, to retain the simplicity and the reusability of the micro detector, we deliberately accept this lack of accuracy in the modeling for this minor part of metrics and intend to compensate it by correlating other micro detectors' alarms.

Guarantee of False Alarm Rate for Micro Detectors

Figure 5.7 illustrates the theoretical and the empirical PFA of our micro detectors for different metrics. Each metric's threshold was normalized by the mean and standard deviation of its normal behavior so that the performance for various metrics can be demonstrated in the same figure. For most of the metrics (e.g., *CS Hit*, *In*

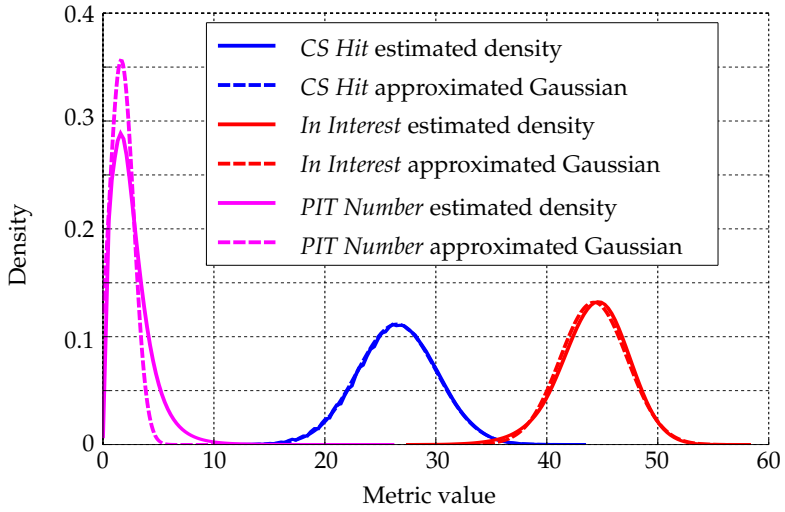


FIGURE 5.6: Illustrative metrics' distributions in normal traffic

Interest), the empirical and the theoretical PFA match closely, implying the ability to guarantee the prescribed PFA of the micro detector and the relevance of the model. Meanwhile, for a few metrics (e.g., *PIT Number*), our micro detector cannot ensure the performance for a small prescribed PFA. As stated in the previous subsection, this phenomenon results from the fact that these metrics are not well modeled by the normal distribution. However, as shown in the following section, the performance can be enhanced by combining micro detectors. Hence the modeling errors on their distribution are compensated by each other.

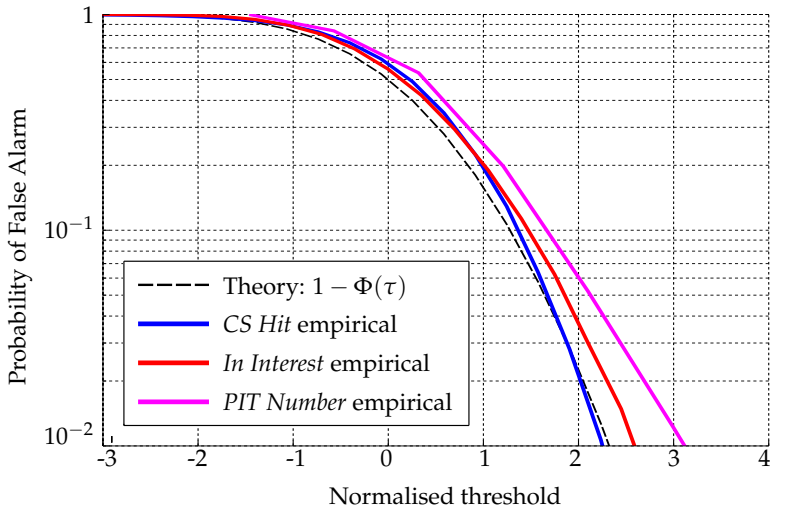


FIGURE 5.7: Guarantee of prescribed PFA for micro detectors of illustrative metrics

5.4.4 Learning Parameters of the Proposed Bayesian Network

To evaluate the learning efficiency of BNC when the size of training set varies, we use the usual k-folds cross-validation method with $k = 5$. Consequently, the BNC learning dataset, which includes data collected from all four scenarios mentioned in Section 5.4.2, is divided into five subsets. Each subset will, in turn, be used as testing data and the remaining four will be used as training data. The average misclassification rate (i.e., the total number of misclassified samples over the total number of samples) over training subsets is defined as *train error*, while the one obtained over testing subsets is called *cross-validation error*. Figure 5.8 shows the learning curves of the proposed BNC when the size of training dataset per scenario changes. When the size of training set per scenario increases, the misclassification error starts decreasing. An optimal value is achieved around 280 training samples per scenario. After that, the misclassification error keeps increasing due to the well-known phenomenon of over-fitting. Therefore, the optimal value of 280 samples for the training set per scenario has been chosen, corresponding to about 23 minutes of collecting samples.

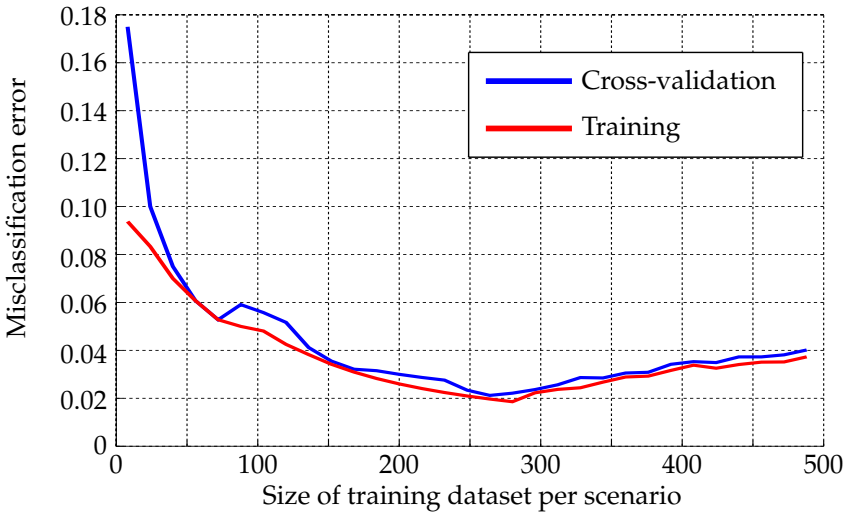


FIGURE 5.8: Learning curve of the proposed BNC

5.4.5 Bayesian Network Classifier Evaluation

Figures 5.9 and 5.10 respectively exhibit the accuracy (i.e., the total number of correctly classified samples over the total number of samples) and the detection delay of the proposed BNC when the attacker rate changes. We will discuss these two figures according to three factors, including the impact of the attack rate, the attack scenario, and the classifier's location. It is noteworthy that most of the previous works propose seamless mitigations against CPA. Moreover, CPA in previous works

is performed in simulation or with strong assumptions while in our work, we propose a detailed attack protocol to carry out CPA in a real deployment. As such, we cannot compare our results with others'.

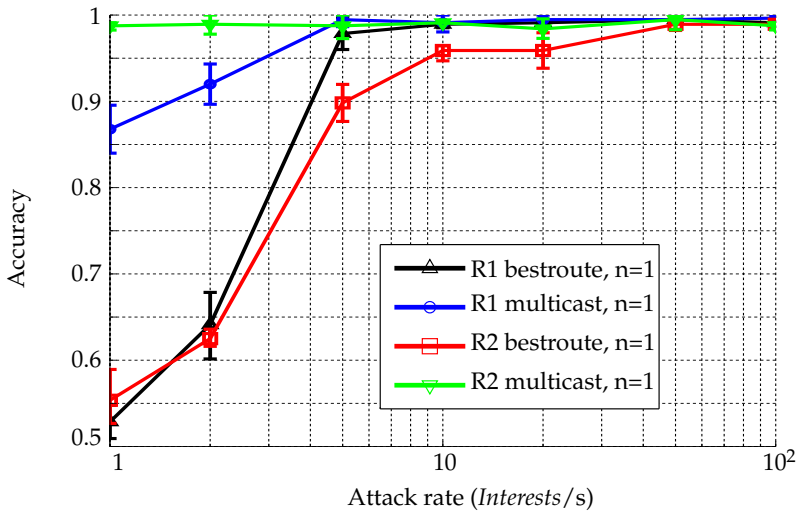


FIGURE 5.9: Accuracy of the proposed classifier

Regarding the attack rate's effect, the figures indicate that when the attack rate is lower than the *Interest* rate under normal traffic, BNC has a weak performance with low accuracy, a high delay and those results have a large variance. It is because when the attack rate is small as compared to the good client's rate, the attack becomes ineffective and its trace is hardly distinguished from the normal traffic. On the other hand, BNC achieves over 95% of accuracy with a delay of about one sample when the attack rate starts getting higher good client rate.

Considering the attack scenario, Figures 5.9 and 5.10 indicate that, at low attack rate (i.e., smaller than 10 Interests/s of the good client), the accuracy and detection delay of BNC against *CPA multicast* is much better than ones of *CPA bestroute*. The reason is that, in *CPA multicast*, the attacker's *Interests* are forwarded to the bad provider more easily thanks to the nature of multicast forwarding strategy. Therefore, even with a small attack rate, the attack is still successful, forcing the good client to re-issue *Interest* to retrieve good *Data*. Thus, it has a stronger footprint on the various metrics, yielding a better detection accuracy. On the other hand, for higher attack rates, the BNC performance is nearly the same for both scenarios. In *CPA bestroute*, the attacker needs to send *Interests* quickly enough to trick the router to use the second best route. As such, the *CPA bestroute* with a high attack rate is more effective, and its behavior becomes more evident.

Regarding the classifier location's impact, in *CPA bestroute*, BNC at R1 is more accurate than at R2 because R1 receives *Interest* packets from clients and attackers

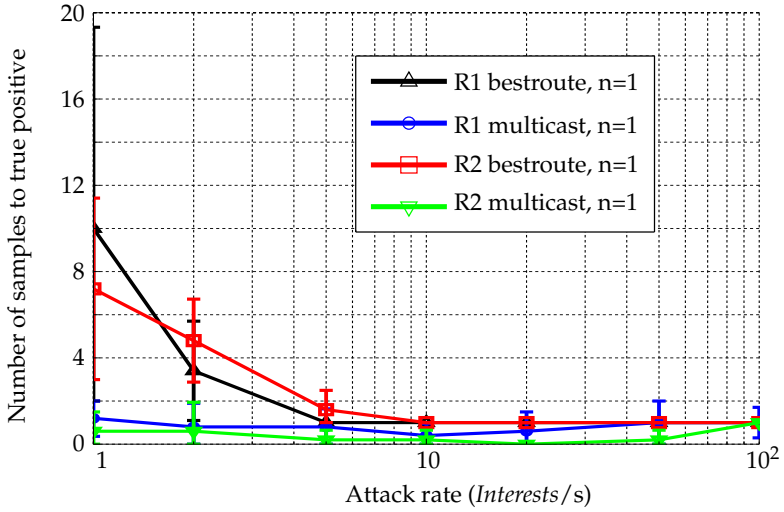


FIGURE 5.10: Delay to CPA true positive

first. As a result, its metrics will be more affected than those of R2. Meanwhile in *CPA multicast*, since R2 is more likely to be poisoned due to the multicast forwarding, its metrics are impacted more obviously than that of R1. Therefore, BNC in R2 achieves better accuracy than in R1.

Eventually, it is important to point out that the accuracy of BNC can be improved by increasing the detection window n of the micro detector. Figure 5.11 plots the BNC accuracy as attack rate changes, with different values of detection window n used by micro detectors. The figure shows that for the worst case of *CPA bestroute* with 1 Interest/s, the accuracy increased from 53% up to 93% by raising the n from 1 to 5. On the other hand, increasing the detection window is unnecessary for high attack rate since it does not enhance the accuracy. Besides, it is noteworthy that increasing the detection window will increase the detection delay of micro detectors (Eq. 5.8-5.9). Therefore, this trade-off should be considered carefully in the deployment.

5.5 Conclusion

This chapter presented the first element towards the design and implementation of a generic SMP for NDN. We first motivated for the necessity of the SMP for NDN. Afterward, we provided background knowledge for the chapter, including the network management plane, the NFD management protocol and fundamentals of BN. The chapter's main contribution consists in the SMP for NDN, which compose of an exhaustive metric list to be monitored, a generic micro detector and a correlation engine. The proposed list included 18 metrics that allows featuring behaviors

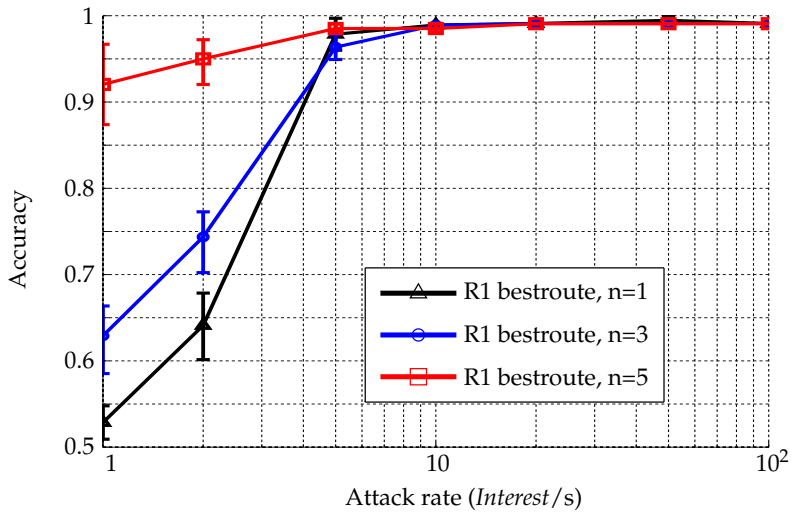


FIGURE 5.11: Effect of detection window

of an NDN node. Each metric is associated with a generic micro-detector that is designed based on the hypothesis testing theory to detect any abnormal variation from the metric's normal behavior. Casual relationships between micro detectors were revealed from an analyze of NFD forwarding pipelines. Those relationships were then integrated into a correlation engine based on BNC, allowing the detection of any abnormal security event occurring in an NDN node. CPA is selected as a use-case to validate our proposal. More specifically, two attack scenarios of the CPA have been considered in a real testbed, and they demonstrate the capability of our solution to accurately detect these attacks at different network locations and with various rates.

Conclusions and Perspectives

Conclusions

Ever since it was designed to connect distant computers decades ago, the Internet has evolved significantly. Therefore, the host-centric communication model no longer fits the Internet's primary usage of accessing contents and gradually reveals its limits. In such a context, a clean-slate approach for the future Internet by utilizing the content-centric communication model has emerged, resulting in ICN architectures. Among them, we focused on NDN because it has gained the reputation in the ICN community as the most promising proposal. NDN identifies content objects with content names instead of identifying hosts with IP address. Also, it uses the content-based routing paradigm, i.e., network elements will route a content object by using its name. Ubiquitous in-network caching is employed to improve the delivery performance for popular content. Besides, a digital signature is associated with each content object to ensure integrity and authenticity. By envisaging the security requirement in the design, NDN intrinsically mitigates attacks inherited from IP network. After more than a decade of development, NDN has reached a certain technological maturity to be considered for deployment. As such, deployment efforts and experiments in the live environment are lively and drawing the significant attention from the community. Despite its potential and maturity, NDN can hardly be adopted by ISP and deployed in their infrastructure without considering its security. Although NDN intrinsic security features can prevent some attacks inherent from IP, novel feature and router components also introduce new security threats to NDN. Given such a context, we propose in this thesis an SMP for the NDN's data plane in the real deployment.

We have investigated in discovered attacks and existing works in the security of NDN to be aware of its current status. We divided security threats of NDN's data plane into three categories, based on the mainly exploited NDN component: *PIT* attacks, *CS* attacks and *Name* attacks. For each category, we explained attacks' fundamental ideas and reviewed some remarkable existing works. We have evaluated that IFA and CPA are the most critical attacks and considered them as two use-cases to further study for our SMP proposal. Although the two attacks have drawn the

attention from the NDN board, their prior works share several disadvantages that must be addressed by our proposal. First, a majority of previous works is evaluated in a simulated environment, leaving the performance in real conditions unknown. Secondly, existing proposals do not provide reliable detection. More specifically, they cannot guarantee a prescribed PFA. Undesired false alarms can waste network resources or accidentally penalize legitimate clients, leading to weak performance. Thirdly, most of the existing works in NDN security focus on a particular attack. The literature lacks a generic security solution that can tackle not only existing threats but also potential attacks revealed in the future. As such, we envisage that our SMP for NDN security will be built on reliable micro detectors whose alarms are correlated to identify abnormal security events at an NDN node in the real deployment.

We tackled the first limitation by revisiting and fully characterizing IFA and CPA in a real deployment of NDN. Since IFA is well-studied in the literature, it is studied in an environment where IP and NDN coexist in isolated domains interconnected by dedicated gateways to deliver the HTTP service. Investing IFA in such an environment would bring more novelty. Despite the attacker's lack control over issued *Interests* and the *NACK* packet's existence, we propose an IFA scenario that succeeds in degrading user's experience when accessing websites. Meanwhile, CPA is an NDN-specific threat that lacks a detailed attack scenario. Thus, we studied CPA in a pure NDN environment in order to prove for its actual threat. We have identified three attack protocols that profit from weaknesses in NDN protocol and implementation. Results in our NDN native testbed exhibit CPA's impacts on essential network entities. It is also noteworthy that while IFA is entirely characterized by the satisfaction ratio as a majority of related works, CPA's effects are more elusive and thus, can hardly be tackled by a detector based on a solely dedicated metric.

To overcome the literature's drawback of unreliable detection, we have designed a micro detector using statistical hypothesis testing. Such a methodology allows us to set up a threshold guaranteeing the desired PFA and establish an expected theoretical performance. We leverage IFA as a use-case to evaluate our micro detector because it can be fully characterized by a single metric of satisfaction ratio. The obtained GLRT detector was first evaluated in a simulation environment to evaluate its intrinsic properties. Results exhibit the relevance of the proposed model with the close match of empirical and theoretical results, as well as the ability to guarantee a prescribed PFA and to establish the trade-off between detection power and delay. Moreover, to further improve the detection accuracy, we developed a sequential version based on the initial GLRT. Data collected from the attack scenarios in NDN-IP testbed demonstrate a significant gain obtained by the sequential detector, regarding the average detection delay and the probability of true alarm with the constraint

of maximum detection delay. As we envisage an SMP based on micro detectors to capture security anomalies, the reliability of the micro detector significantly impacts the performance of the overall proposal. Therefore, obtained results are remarkable because they demonstrate the relevance of our framework for designing micro detector.

Finally, we come up with the proposal of a generic SMP for NDN. Our proposal SMP consists an exhaustive metric list for monitoring, a design of generic micro detector and a correlation engine. We have inspected NFD forwarding pipelines and identified 18 metrics to characterize an NDN node's behavior. Associated with each metric is a generic micro detector based on the hypothesis testing theory. The micro detector is responsible for catching any abnormal change of the metric from its normal behavior and can guarantee a prescribed PFA. From the analyze of NFD forwarding pipelines, we have identified the causal relationship between metrics and integrated them to a BNC that correlates alarms from micro detectors to capture any unusual security event in an NDN node. To validate our proposal, two CPA scenarios were deployed in a real testbed. Although the micro detector's performance is weak for a minority of metrics, collected data demonstrate the accuracy of the overall SMP in detecting CPA at different network locations and with various rates.

Perspectives

The work achieved during this thesis opens several perspectives for NDN security management. We first discuss reconsideration of NDN attacks in an environment where NDN is coupled with IP. Next, we propose some possible extension of the proposed SMP regarding several axes, e.g., collaborative detection scheme between nodes for a trace-back countermeasure, evaluate the proposal's performance with other NDN attacks.

We are interested in considering attacks' scenarios in real testbeds of NDN coupled with IP. NDN is a disruptive paradigm that proposes to replace the conventional host-centric model with the novel content-centric one. For this reason, deploying NDN on a large scale is hardly feasible in the near feature. It is, however, possible to deploy NDN on isolated islands interconnected to the existing IP network with dedicated NDN/IP gateways. This scenario is more credible as it provides a smooth transition from the current IP network to NDN without disrupting current users. More complications are added, thus forcing attackers to adapt their attack scenarios. Therefore, the feasibility of NDN attacks (e.g., CPA, time analysis, cache pollution) need to be revised considering the actual deployment condition, just like what we did with IFA.

As security solutions are proposed, attack scenarios evolve as well, trying to negate existing security solutions. We believe that there is no exception for our proposal. In order to prepare for attack improvements in which the proposed detection scheme might be deceived with low-rate malicious traffic, enhancing our security management plane by investigating a collaborative detection scheme is an appealing perspective. In our solution, the Bayesian network is leveraged to correlate alarms from micro detectors of a single node. The engine can also be extended to correlate warnings from other nodes. Once a node detects an aberration, it sends the alarm to its neighbors or the network manager. Gathering and correlating alarms from multiple sources could significantly enhance the detection accuracy, and enable a countermeasure that can track the anomaly back to its sources. We also want to address other NDN attacks with our SMP to evaluate its comprehensiveness and extensibility. The path toward this aim would be similar to what has been done in this thesis, i.e., investigating attack scenarios and reproducing the attack in a real testbed, collecting data for parameter estimation of Bayesian network and evaluating it under various attack scenarios.

Appendix A

Résumé de la thèse en français

Contents

A.1 Introduction

A.1.1 Contexte

L'architecture actuelle d'Internet a été conçue il y a des décennies alors que l'intérêt principal était de connecter des systèmes distants. À ce moment-là, le modèle de communication se concentrait principalement sur *où* les systèmes sont. Au fil du temps, l'Internet et son utilisation ont considérablement évolué. Les ordinateurs et les appareils ont augmenté en nombre grâce à un prix toujours plus abordable. La vitesse de traitement et la capacité de mémoire ont également été grandement améliorées. Les utilisateurs peuvent maintenant rejoindre plus facilement l'Internet pour accéder à une énorme quantité de contenus et de services, et ils se soucient seulement du *quoi*, à savoir le contenu ou les services auxquels ils accèdent et non plus *où* il se trouvent. En tant que tel, le modèle de communication original ne correspond plus à l'usage principal de l'Internet. En outre, l'Internet a été conçu à l'origine sans considération d'éventuels problèmes émergents tels que la sécurité, la gestion, la qualité du service et la mobilité. Par conséquent, à mesure que de nouveaux besoins ou problèmes émergent, l'architecture devient plus lourde avec l'usage de "correctifs" pour résoudre ces problèmes.

Comme le *quoi* devient l'intérêt principal, le contenu devrait maintenant devenir la primitive essentielle des couches protocolaires de niveau réseau. Une approche pour l'Internet du futur aui fait table rase vis à vis de l'existnat a attiré l'attention de la communauté de la recherche, aboutissant à plusieurs architectures *Information-Centric Networking* (ICN). Parmi ces propositions, *Named Data Networking* (NDN) est la proposition d'architecture la plus populaire. NDN est basé sur le paradigme de routage orienté contenu, mis en oeubre par le biais d'un nommage des contenus directement au niveau du réseau. Il propose également une mise en cache des contenus sur le réseau pour améliorer les performances leur livraison et il associe une signature à chaque objet de contenu, apportant un premier niveau de sécurité.

A.1.2 Problèmatique

Après une décennie de recherche et de développement, la proposition NDN est maintenant assez mature avec une spécification d'architecture et de protocole et une implémentation fonctionnelle et maintenue *NDN Forwarding Daemon* (NFD). En conséquences, les efforts de recherche dans la communauté NDN se sont déplacés vers les considérations de gestion, de déploiement et de sécurité. Plus spécifiquement, en intégrant des mécanismes de sécurité par conception, NDN atténue intrinsèquement plusieurs attaques héritées du réseau IP. Néanmoins, en introduisant de nouvelles fonctionnalités et de nouveaux composants, NDN expose également le réseau à nouvelles failles de sécurité. Parmi celles-ci, l'attaque par inondation d'intérêts (*Interest Flooding Attack* – IFA) et l'attaque par empoisonnement de contenu (*Content Poisoning Attack* – CPA) sont les plus importantes. Ces deux attaques ont attiré l'attention de la communauté NDN. Cependant, les travaux précédents sur ces sujets partagent plusieurs inconvénients :

- La plupart des travaux précédents évaluent leurs propositions avec des résultats obtenus dans un environnement simulé, posant la question de leur performance effective dans un contexte de déploiement réel ;
- Dans les travaux existants avec détection d'attaque, les auteurs n'indiquent pas clairement comment ils choisissent leur seuil de détection ou s'ils requièrent une phase d'apprentissage non triviale. Un seuil mal défini peut conduire à une détection mal adaptée et non fiable, ce qui entraîne de fausses alarmes qui gaspillent les ressources du réseau et peuvent pénaliser le client légitime ;
- Les travaux existants se concentrent généralement sur une attaque NDN particulière. La littérature manque d'une solution de sécurité générique capable de répondre à la fois aux menaces de sécurité existantes et aux menaces potentielles qui n'ont pas encore été révélées à ce jour.

Un plan de gestion de la sécurité apparait donc indispensable si l'on veut que NDN soit adopté et déployé par les opérateurs et fournisseurs d'accès de l'Internet dans leurs infrastructures opérationnelles.

A.1.3 Contributions

L'objectif de cette thèse est de proposer un plan de surveillance de la sécurité utilisable dans le cadre d'un déploiement réel de NDN et qui puisse résoudre efficacement les failles de sécurité existantes et permettre d'envisager d'autres menaces non encore révélées à ce jour. Le plan proposé est constitué de micro détecteurs fiables basés sur des tests d'hypothèses statistiques.

Nous soutenons qu'une détection peu fiable entraînera de nombreuses fausses alarmes. Une fois associées à une contre-mesure, elles peuvent provoquer une instabilité, un gaspillage des ressources du réseau et une pénalisation des utilisateurs légitimes. En tant que tel, nous privilégions délibérément l'établissement d'une détection fiable afin de garantir un taux prescrit de fausses alarmes, plutôt que de nous précipiter prématurément vers une solution d'atténuation du SMP envisagé.

Comme cas d'utilisation pour cette proposition, nous caractérisons d'abord les attaques de sécurité NDN les plus importantes - IFA et CPA - dans le cadre d'un déploiement réel de NDN. Une telle investigation n'a jamais été faite dans des travaux antérieurs. Nous identifions les paramètres d'attaque et mettons en évidence leur impact sur les entités réseau.

Comme IFA peut être caractérisée par une seule métrique, nous abordons ensuite sa détection avec un micro-détecteur en utilisant la théorie des tests d'hypothèses statistiques. La méthodologie permet de garantir une probabilité de fausses alarmes (*Probability of False-Alarms* – PFA) donnée tout en maximisant la puissance de détection. Le micro-détecteur proposé est ensuite étendu à une version séquentielle pour améliorer sa précision. Nous évaluons finalement la performance globale du micro-détecteur en utilisant des données issues de simulations et d'expérimentations réelles.

Pour les attaques qui affectent plusieurs aspects de l'état des nœuds NDN, tels que l'attaque CPA, mais aussi pour les attaques potentielles non révélées à ce jour, nous proposons une solution multi-métriques avec corrélation d'alarmes fondée sur un ensemble exhaustif de mesures basées sur une analyse approfondie des flux de traitements de paquets NFD. Chaque mesure est associée à un micro-détecteur pour détecter tout changement anormal avec une PFA maitrisée. Par la suite, un moteur de corrélation basé sur un réseau bayésien est proposé pour combiner des alarmes de chaque micro-détecteur et ainsi identifier tout événement de sécurité anormal dans un nœud NDN. Pour valider notre proposition, deux scénarios d'attaque CPA ont été considérés dans un environnement d'expérimentation réel. Les données collectées démontrent la capacité de notre solution à détecter l'attaque CPA avec précision à différents taux. Au delà, le plan de surveillance de la sécurité proposé est générique. En tant que tel, une perspective possible de ce travail, consisterait à déployer d'autres attaques de sécurité NDN et vérifier les performances de notre proposition. De plus, le réseau bayésien proposé pourrait aussi être étendu afin de corréler les alarmes provenant de différents nœuds, aidant ainsi à identifier la source de l'anomalie.

A.1.4 Organisation

Le reste de ce résumé est organisé comme suit. Dans la Section A.2, nous présentons les concepts clés d'ICN et la manière dont NDN réalise ces concepts dans son architecture. Dans la Section A.3, nous fournissons un état de l'art sur la sécurité NDN en proposant une taxonomie pour classer les attaques NDN, en introduisant le principe des attaques et des remarques sur les travaux existants en lien avec ces problématiques. Cet état des lieux est ensuite analysé avec une évaluation du risque des attaques NDN selon un ensemble d'attributs donnés. Sur la base des résultats de cette évaluation, nous avons sélectionné IFA et CPA comme deux cas d'utilisation pour notre étude. Dans la Section A.4, nous démontrons la nécessité de considérer les menaces de sécurité NDN dans un déploiement réel, puis nous présentons les deux attaques sélectionnées, ainsi que leurs impacts dans un environnement expérimental NDN. Dans la Section A.5, nous présentons notre conception de micro-détecteur IFA, étape par étape, en utilisant la théorie des tests d'hypothèses statistiques. Le schéma de détection est étendu à une version séquentielle pour améliorer la précision du micro-détecteur. Nous en évaluons la performance globale en utilisant des données issues de simulations et d'expérimentations réelles. La Section A.6 présente le plan de surveillance de la sécurité que nous proposons pour NDN. Un ensemble exhaustif de métriques est proposé sur la base d'une analyse approfondie des flux de traitements des paquets NFD. Chaque métrique est associée à un micro-détecteur capable de capturer les déviations anormales vis à vis d'un comportement attendu de la métrique avec une PFA maitrisée. Les alarmes provenant de micro-détecteurs sont corrélées par un moteur basé sur un réseau bayésien afin d'identifier tout événement de sécurité anormal dans un nœud NDN. En s'appuyant sur les données de CPA mesurées dans notre environnement expérimental NDN, nous fournissons des résultats numériques qui démontrent la pertinence et la performance de notre approche. La Section A.7 tire les conclusions de ce travail en dresse les perspectives.

A.2 Information-Centric et Named Data Networking

Dans cette section, nous présentons les concepts clés du paradigme ICN et la manière dont NDN implémente ces concepts clés dans son architecture.

A.2.1 Caractéristiques clés d'ICN

Les architectures ICN sont les résultats d'une approche pour la définition de nouvelles architectures pour l'Internet du Futur. Les caractéristiques importantes des architectures ICN comprennent le nommage de l'information, l'acheminement de l'information, la sécurité et la mobilité [6].

Nommage de l'Information

ICN utilise un modèle de communication centré sur l'information (ou le contenu) : un contenu est nommé et appelé *objet de données nommé* (NDO), et se place au coeur des primitives du réseau. En nommant les objets de données indépendamment de leur emplacement, la demande de contenu et sa récupération se produisent par la seule utilisation de ce nom comme critère d'acheminement. Dans ce qui suit, le terme «utilisateur» désigne l'entité qui demande un NDO, tandis que le terme «fournisseur» désigne celui qui publie un NDO. Au lieu de spécifier l'adresse IP du fournisseur, les utilisateurs indiquement simplement le nom des données et le réseau détermine la source optimale pour récupérer le NDO. En tant que tel, toutes les copies d'un NDO sont considérées comme équivalentes et peuvent répondre à la demande. Dans le paradigme ICN, il y a deux types de schémas de nommage : plat et hiérarchique [7].

Acheminement de l'Information

L'Internet actuel manque de mécanismes naturels pour fournir une quantité massive de données de manière efficace. Le réseau se concentre uniquement sur la transmission des paquets aussi rapidement que possible sans trouver d'optimisation pour la livraison [2, 9]. Le paradigme ICN gère efficacement la fourniture d'informations avec trois fonctionnalités : la mise en cache, la résolution de noms et le routage de données. Dans une architecture ICN, les NDO populaires transitant par le réseau sont mis en cache, ce qui permet de répondre plus rapidement à d'autres demandes, ce qui réduit le trafic redondant et minimise la latence. La fonctionnalité de *résolution de nom* regroupe les requêtes et résoud un nom de données demandé en informations relatives aux les sources de données optimales, tandis que *routage des données* est responsable du transfert de l'objet NDO de la source à l'utilisateur. Les deux mécanismes peuvent être couplées ou découplées.

Sécurité de l'Information

Le paradgime ICN considère intrinsèquement les exigences de sécurité dans sa conception. La communication d'ICN est pilotée par les demandes des utilisateurs : un flux de données est acheminé seulement si quelqu'un a explicitement demandé ces données. En tant que tel, les noeuds de réseau peuvent refuser un flux non sollicité et réduire les données indésirables. En outre, les architectures ICN fournissent des moyens pour vérifier l'intégrité des données, telle qu'une signature numérique ou un nom auto-certifiant. De plus, les NDO sont mis en cache dans les éléments du

réseau pour résoudre d'autres requêtes, améliorant ainsi la disponibilité des informations et la résistance aux attaques par déni de service. De plus, comme une architecture ICN ne présente pas d'identificateur pour les hôtes, il déconnecte l'utilisateur et le fournisseur, protégeant ainsi leur vie privée.

Gestion de la Mobilité

L'Internet actuel n'a pas été conçu explicitement pour gérer des hôtes mobiles, bien que cet usage soit très populaire de nos jours. Au contraire, ICN gère les NDO individuellement et indépendamment de leur emplacement. En tant que tel, ICN ne nécessite pas de connexion de bout en bout pour l'échange de données, ce qui simplifie le problème de mobilité. Un utilisateur mobile a juste besoin d'envoyer de nouvelles demandes les NDO lorsqu'il se déplace vers un nouvel emplacement. Pendant ce temps, lorsqu'une source de données change de localisation, le réseau peut utiliser des copies mises en cache ou d'autres sources de données alternatives pour répondre aux demandes des utilisateurs tout en mettant à jour les informations de routage pour le nouvel emplacement de la source de données.

A.2.2 Named Data Networking

La plus populaire des architectures ICN - *Named Data Networking* (NDN) [11] est un projet de recherche du programme *Future Internet Architecture*, financé par la *National Science Fundation* (NSF) des États-Unis.

Schéma de Nommage et Paquets

NDN utilise un schéma de nommage hiérarchique. Un nom de contenu NDN (ou *préfixe*) est constitué de plusieurs composants de nom , comme par exemple, `/utt/videos/utt_intro.mpg`. Un tel schéma est interprétable par un utilisateur et reflète les relations des éléments de données. En outre, des demandes similaires pour le même préfixe peuvent également être agrégées, facilitant ainsi l'évolutivité de l'architecture [2]. Un schéma de nommage est nécessaire pour définir et attribuer des noms de premier niveau, garantissant l'unicité du nom de contenu entre les fournisseurs.

NDN utilise deux paquets principaux, à savoir *Interest* et *Data*. L'*Interest* porte la requête de l'utilisateur pour le contenu, tandis que le *Data* contient l'objet NDO lui-même. Un paquet *Interest* comprend les champs importants suivants : *Name* (requis), *Nonce* (requis), *InterestLifetime* et *Selector*. *Name* indique le préfixe du contenu demandé. *Nonce* est une valeur aléatoire générée par l'utilisateur pour détecter les boucles de *Interests*. *InterestLifetime* spécifie le temps restant en millisecondes (4000ms par défaut) avant que l'*Interest* expire. Puisque NDN utilise le plus long

préfixe-correspondant, un *Interest* peut avoir plusieurs *Data* correspondants. Le champ *Selector* fournit des informations supplémentaires pour sélectionner le seul *Data*.

Un *Data* paquet comprend les champs suivants : *Name* (requis), *MetaInfo*, *Content* (requis) et *Signature* (requis). *Name* contient le nom complet de l'objet NDO porté à l'intérieur. *MetaInfo*: fournit des informations supplémentaires sur le *Data*. *Content* contient le NDO réel. *Signature* inclut deux sous-champs *SignatureInfo* et *SignatureValue*. *SignatureInfo* fournit des informations pertinentes pour obtenir le certificat parent. *SignatureValue* représente les bits réels de la signature, calculés en prenant le hachage sur tous les autres champs (y compris *SignatureInfo*), puis en chiffrant avec la clé privée du fournisseur.

Routeur et routage basé sur le nom

Le routeur NDN a trois composants principaux. Le premier est la *base d'informations de transfert (FIB)* qui contient une liste de prochains sauts à qui envoyer les *Interests* reçus. Chaque entrée de la FIB comprend un préfixe et une liste de faces sortantes pour transmettre l'*Interest* au fournisseur de contenu. Le deuxième composant est la *table d'intérêt en attente (PIT)* qui conserve les traces de *Interests* transmises. Chaque entrée PIT se compose d'un préfixe, d'une liste de faces entrantes de *Interests* en attente pour ce préfixe et d'une liste de faces sortantes auxquelles les *Interests* agrégés sont transmis. Ces informations seront utilisées ultérieurement pour renvoyer *Data* aux utilisateurs. Le troisième composant est la *cache de contenu (CS)* qui fournit une mise en cache sur le chemin pour améliorer les performances de livraison dans NDN.

Lorsqu'un *Interest* arrive, le routeur consulte le CS. Si une correspondance *Data* existe, le routeur transmet immédiatement cette copie mise en cache à la face entrante de l'*Interest*. S'il y a plusieurs *Data* trouvés, le champ *Selector* (Section A.2.2) de l'*Interest* est utilisé pour décider quel *Data* est retourné. Si aucun *Data* mis en cache n'est trouvé, le routeur consulte sa PIT. Si une entrée correspondant au préfixe le plus long est trouvée et que la valeur de *Nonce* n'est pas encore enregistrée (ce qui signifie que cet *Interest* n'est pas une boucle), la face entrante de *Interest* est ajouté à l'entrée PIT correspondante. L'*Interest* sera rejeté parce qu'il y a déjà un *Interest* pour ce *Data* envoyé en amont. S'il n'y a pas d'entrée PIT correspondante, le routeur vérifie la FIB pour une correspondance de préfixe le plus long pour trouver une face sortante. En fonction de la stratégie de transfert, l'*Interest* peut être redirigé vers le meilleur itinéraire ou plusieurs faces sortantes. Lorsqu'aucune entrée correspondante dans la FIB n'est trouvée, le routeur supprime l'*Interest* ou envoie un paquet *Negative-ACKnowledge* (*NACK*).

Quand un paquet *Data* arrive, le routeur consulte d'abord sa PIT. S'il n'y a pas d'entrée PIT correspondante, le paquet *Data* est considéré comme non sollicité et rejeté. Sinon, le paquet *Data* est transmis à toutes les faces entrantes de l'entrée PIT correspondante. Le routeur décide également d'insérer ce paquet *Data* dans le CS. Après avoir renvoyé le paquet *Data*, l'entrée PIT correspondante est effacée, ce qui implique que «Un *Interest* récupère au plus un paquet *Data* ». Les entrées PIT qui ne reçoivent aucune correspondance *Data* après un certain temps sont considérées comme obsolètes et effacées.

Transport

NDN n'a pas de couche de transport dédiée. Des fonctions de livraison fiables et de contrôle de congestion sont déléguées aux applications, ou couches supérieures et au composant de stratégie dans le plan d'acheminement. La segmentation et le réassemblage des segments sont effectués en utilisant des composants de nom. Les informations requises pour le transport figurent dans le nom du contenu. NDN peut fonctionner sur un service de livraison de paquets non fiable. Pour assurer une livraison fiable, les entrées PIT insatisfaites expirent après une période et sont rejetées, ce qui oblige l'utilisateur à émettre un autre *Interest* s'il veut toujours obtenir le contenu. Les caches aident aussi à la retransmission de données pendant la congestion. Si un routeur met déjà en cache un paquet *Data* avant de le supprimé pour cause de lien congestionné, les *Interests* retransmis seront satisfaits par cette copie mise en cache, ce qui réduit la latence pendant la congestion. De plus, le contrôle de flux et le contrôle de congestion ne dépendent plus des hôtes finaux. La PIT permet de réduire significativement les *Interests* redondants et *Data* non sollicités. Le champ *Nonce* permet également de détecter et de supprimer les *Interest* en boucle. Enfin, chaque entrée FIB et chaque entrée PIT enregistrent plusieurs faces, prenant ainsi en charge les trajets multiples et fournissant une connectivité riche.

Fonctions de sécurité intrinsèques

Le champ *Signature* dans le paquet *Data* assure intrinsèquement l'authenticité et l'intégrité. *SignatureValue* est calculé par une fonction de hachage portant sur tous les autres champs, puis chiffré avec la clé privée du fournisseur de contenu. Tous les noeuds recevant un paquet *Data* peuvent récupérer la clé publique de ce fournisseur avec des informations dans *SignatureInfo* et l'utiliser pour déchiffrer le champ *SignatureValue*. Le résultat est ensuite comparé au hachage sur tous les autres champs de *Data*. Le processus de vérification échouera si l'objet NDO est modifié pendant la transmission ou si la clé publique récupérée est incorrecte. De plus, pour vérifier l'authenticité du fournisseur, l'utilisateur doit faire confiance au propriétaire de la clé publique utilisée pour signer le paquet *Data*. La structure de nom hiérarchique

facilite la création de relations de confiance.

A.3 Sécurité de NDN - État de l'art

Dans cette section, nous présentons un état de l'art de la sécurité de NDN. Pour commencer, nous proposons une taxonomie pour classifier les attaques NDN. Pour chaque attaque, nous présentons son idée fondamentale ainsi qu'un résumé des travaux relatifs existants. Ensuite, nous sélectionnons un ensemble d'attaques comme cas d'utilisation pour la suite de nos travaux.

A.3.1 Etude existantes sur la sécurité NDN et taxonomie proposée

Depuis que NDN a été proposé, plusieurs études sur la sécurité NDN ont été conduites [47–49], et chacune d'elles catégorise les attaques NDN selon différents critères. Après avoir examiné les avantages et les inconvénients de ces études sur la sécurité NDN, nous proposons notre propre taxonomie pour la sécurité NDN. Fondée sur les caractéristiques NDN qui sont exploitées par l'attaque, la taxonomie se compose de trois catégories : *routage*, *respect de la vie privée* et *mise en cache*. Pour chacune de ces attaques, nous expliquons leur principe et nous résumons les travaux antérieurs qui les concernent.

A.3.2 Attaques sur le routage

Les attaques dans cette catégorie sabotent les PIT et FIB (Section A.2.2) pour causer des dégâts. Une attaque cruciale de ce type est l'*Attaque par Inondation d'Intérêts* (IFA). D'autres attaques possibles dans cette catégorie sont la mise à jour excessive des données d'acheminement et l'annonce de fausses reoutes.

L'Attaque d'Inondation d'Intérêt

Dans une attaque IFA, l'attaquant inonde la PIT et les fournisseurs de contenu en envoyant un grand nombre de *Interests* sur une courte période, les empêchant ainsi de traiter les demandes légitimes. Cette attaque est identifiée par la communauté NDN comme l'une des attaques de sécurité les plus importantes. L'attaque IFA est la plus dangereux avec des noms inexistants, car elle affecte non seulement les fournisseurs de contenu, mais aussi l'infrastructure réseau. En outre, elle ne nécessite aucune connaissance des contenus existants car elle exploite des noms de contenu falsifiés, ce qui permet de la réaliser facilement. Les préfixes d'*Interests* malveillants peuvent être entièrement forgés par l'attaquant ou composés d'un préfixe valide et d'un suffixe aléatoire [52]. Les travaux précédents sur IFA peuvent être divisés en fonction de leur niveau de granularité. Les solutions grossières portent sur l'interface du routeur sous attaque (approche *par-interface*) [53]. Les solutions fines quant à elles

portent sur le préfixe malveillant utilisé pour lancer (*par-préfixe* approche) [64, 65].

Une majorité de solutions existantes sur IFA nécessite que les routeurs maintiennent des statistiques pour détecter les interfaces ou les préfixes attaqués. Cependant, les auteurs de ces solutions n'indiquent pas clairement comment ils choisissent le seuil de détection. Un seuil mal défini peut conduire à une détection rigide et non fiable, ce qui entraîne beaucoup de fausses alarmes qui gaspillent les ressources du réseau ou pénalisent accidentellement le client légitime. De plus, de nombreuses solutions proposées tirent parti de la collaboration des routeurs pour atténuer l'IFA, les rendant ainsi dépendants les uns des autres. Une fois qu'un routeur est compromis, il peut perturber d'autres routeurs en envoyant de fausses annonces. De plus, la plupart des travaux précédents évaluent leur proposition dans un environnement simulé, laissant ainsi leur performance dans une situation de déploiement réel, inconnue.

Mise à jour excessives et fausses annonces de routes

Dans le cadre d'une mise à jour excessive, un nœud malveillant envoie des mises à jour excessives du contenu et de la copie en cache sur le réseau. Il faudra du temps et des ressources aux routeurs NDN pour traiter et recalculer les routes optimisées pour le contenu. Lorsque le débit d'envoi de mises à jours est supérieur au temps de convergence, le réseau NDN sera surchargé, conduisant à une distribution de contenu erronée à grande échelle et empêchant la récupération de données sur le réseau [154]. Lors d'un piratage, un nœud contrôlé par un attaquant annonce de fausses informations de routage indiquant aux autres nœuds de rediriger les *Interests* vers l'attaquant. Ainsi, l'attaquant peut détourner les requêtes des utilisateurs proches et même filtrer ces requêtes pour provoquer un DoS [48]. L'attaquant peut eller plus loin, rendant son comportement plus sophistiqué, en redirigeant les *Interests* vers des emplacements appropriés après les avoir capturés. Tout semble être normal pour les utilisateurs, mais ils ne savent pas que leurs requêtes ont été enregistrées et qu'ils ont probablement récupéré un faux contenu. Les deux attaques sont principalement causées par la réception d'informations de routage provenant de nœuds malveillants. Par conséquent, la sécurisation du protocole de routage (e.g., *Named-data Linked State Routing* [3, 50]) est une solution efficace pour prévenir ces menaces.

A.3.3 Attaques sur le respect de la vie privée

Les attaques de confidentialité exploitent les failles des paquets *Datas* et du nom de contenu mis en cache pour acquérir des connaissances sur les demandes et le contenu des utilisateurs, violant ainsi le respect de leur vie privée.

Attaques temporelles

Le contenu mis en cache a un délai de récupération plus court que celui fourni par d'autres sources. L'attaque temporelle exploite ce fait pour distinguer les données issues d'un cache de celles fournies par un autre fournisseur, indiquant ainsi quels contenu sont sollicités par des utilisateurs sur le même lien que l'attaquant. L'attaquant peut estimer cette latence d'accès au cache en demandant un contenu, puis en le redemandant une deuxième fois et en mesurant la latence d'accès au cache d_c. Par la suite, l'attaquant envoie un *Interest* pour un contenu et mesure la latence d_i. Si $d_i \approx d_c$, l'attaquant peut déduire que le contenu a été récemment demandé par un autre client. Sinon, le contenu n'a pas demandé récemment ou a été supprimé du cache cible. La plupart des solutions existantes pour l'attaque temporelle proposent d'appliquer un retard artificiel au cache *Data* pour tromper l'attaquant, e.g., Asc et al. [81]. Cependant, cette approache élimine l'avantage de la mise en cache dans le délai de récupération du contenu.

Attaques de la confidentialité de nom

Il existe plusieurs façons d'exploiter la confidentialité des noms NDN. En surveillant le nom du contenu dans *Interests* (par exemple compromettre un routeur, ou inspecter un cache avec une attaque temporelle), on peut connaître les tendances dans les demandes des utilisateurs proches et le contenu populaire [84]. En outre, la mise en cache omniprésente NDN peut conserver des traces de communications. En utilisant le mécanisme de correspondance de préfixe le plus long dans NDN et le champ *Interest Exclude*, les attaquants peuvent envoyer itérativement *Interest* pour découvrir quels contenus sont conservés dans le cache [4, 30]. En tant que tels, les attaquants peuvent capturer des noms de contenu, acquérir des connaissances sur les communications, même quand il est déjà terminé. Dans le cas où l'attaque peut compromettre un routeur, il peut aussi *Interests* trou noir pour certains contenus spécifiques dans sa watchlist [85]. Les œuvres existantes ont de nombreuses approches pour protéger la confidentialité des noms, e.g., *application de réseau de données nommées anonymes* (ANDaNA) [87, 88], *mise en cache sélective* [30, 90].

A.3.4 Caching Attaques

Les attaques dans cette classe perturbent l'avantage du système de mise en cache NDN. *Attaque de l'empoisonnement du Contenu* (CPA) et l'attaque de pollution de cache sont deux attaques typiques dans cette catégorie.

Attaque de l'Empoisonnement du Contenu

Dans la CPA, un attaquant injectera un *Data* malveillant dans le cache du routeur pour résoudre les *Interests* légitimes. Une telle attaque exploite les caches en réseau dans NDN pour propager un *Data* malveillant à autant d'utilisateurs que possible. Pour augmenter l'échelle de l'effet, l'attaquant forgerait du faux contenu pour un nom de contenu populaire. Malicieux *Data* porte toujours un nom valide, mais son contenu a été modifié. Les solutions proposées pour détecter et atténuer le CPA peuvent être divisées en trois groupes, à savoir (1) *Amélioration des intérêts* [66], (2) *amélioration de la vérification* [69] et (3) *basé sur les commentaires* [67,71].

Cependant, les travaux précédents sur CPA partagent certains inconvénients communs. La première est l'incohérence dans l'évaluation de l'impact de l'ACP. Comme leurs auteurs associent généralement une telle évaluation à leurs solutions proposées, la compréhension de ce phénomène est partielle et biaisée en faveur des solutions proposées. Deuxièmement, les scénarios de simulation reposent sur une attaque ponctuelle dans laquelle les clients s'arrêtent souvent lorsqu'ils reçoivent un *Data* légitime, tandis que la CPA est plus susceptible d'opérer comme un flux, laissant ainsi des angles morts sur le phénomène. Troisièmement, la plupart des travaux précédents surestiment le CPA avec des comportements irréalistes, par exemple, des caches pré-pollués avec un pourcentage de *Data* malveillant. Bien qu'il y ait des explications sur le scénario d'attaque et sur la manière dont *Data* est inséré dans les caches, ils sont insuffisants pour expliquer pourquoi la CPA peut atteindre un pourcentage aussi élevé de *Data* malveillants dans les caches.

Attaques de pollution de cache

Dans la pollution du cache, l'attaquant dégrade les avantages de la mise en cache pour les utilisateurs légitimes en forçant les caches à stocker le contenu de manière irrégulière. Cette attaque a été largement étudiée dans IP, en particulier pour la mise en cache web [73]. Les travaux existants sur l'attaque de pollution de cache peuvent être divisés en deux groupes : *décision de mise en cache* (e.g., *CacheShield* [75]) ou *détection - atténuation* (e.g., Guo et al [76]).

Une majorité des travaux existants sur les attaques de pollution de cache est évaluée uniquement avec des résultats de simulation, remettant ainsi en question leur performance dans la réalité. En outre, il n'y a pas de consensus sur le scénario d'évaluation des attaques par cache NDN dans la communauté scientifique jusqu'à présent [79], ce qui rend les résultats incomparables. De plus, la plupart d'entre eux ne peuvent cibler qu'un seul type de pollution de cache, ce qui nécessite que le routeur exécute des solutions complémentaires pour éviter la pollution des caches.

A.3.5 Résumé des attaques NDN et sélection de cas d'utilisation

Dans cette section, nous évaluons les attaques ci-dessus et sélectionnons un ensemble d'attaques pour les travaux conduits dans les chapitres suivants. La Tableau A.1 récapitule notre évaluation des attaques NDN selon les critères suivants: impact sur la confidentialité, impact sur l'intégrité, impact sur la disponibilité, échelle des effets, nombre de travaux antérieurs et reproductibilité. Les valeurs d'évaluation comprennent: *H* pour high; *M* pour moyen; *L* pour faible et *Blank* si aucun impact n'est quantifiable. On peut noter que la plupart des attaques dans NDN n'affectent

TABLE A.1: Résumé des attaques NDN

	Impact sur la confidentialité	Impact sur l'intégrité	Impact sur la disponibilité	Échelle des effets	Nombre de travaux antérieurs	Reproductibilité
Inondation d'Intérêt			H	H	H	H
Mis à jour excessive			H	H	L	L
Hijack	H	L	L	L	L	L
Attaque temporelle	M			M	L	H
Confidentialité de nom	M			M	M	L
Empoisonnement du contenu		M	M	H	M	M
Pollution de cache			M	M	M	M

aucun utilisateur spécifique. Puisque NDN n'a pas d'identificateur d'hôte, les attaques doivent cibler les routeurs pour causer des dommages, affectant indirectement les utilisateurs proches. En ce qui concerne le nombre de travaux antérieurs, le tableau montre que la plupart des efforts de recherche se concentrent sur l'attaque IFA. Sa reproductibilité est par ailleurs élevée, ce qui facilite d'autres études dans un contexte de déploiement réel. En outre, le tableau montre que les atteintes à la vie privée, l'attaque CPA et à la pollution de cache attirent également l'attention de la communauté NDN, mais dans une moindre mesure. L'attaque de confidentialité de nom, cependant, manque de réproductabilité et n'est donc pas sélectionnée. En ce qui concerne l'empoisonnement du contenu et la pollution des caches, les deux attaques ont des valeurs de classement similaires. Cependant, le CPA montre un impact à plus grande échelle et sur davantage d'attributs (par exemeple, l'intégrité). En résumé, nous avons choisi délibérément IFA et CPA comme les deux attaques NDN à étudier dans les chapitres suivants.

A.4 Caractérisation de l'IFA et du CPA en Déploiement Réel

Dans cette section, nous discutons d'abord de la nécessité d'étudier la sécurité NDN dans des conditions réalistes. Ensuite, nous introduisons les conditions de tests que

nous avons utilisées pour nos différentes expériences. Par la suite, nous présentons les résultats obtenus par le reproduction des attaques sélectionnées, à savoir IFA et CPA.

A.4.1 Nécessité de caractériser les attaques en conditions réelles

Lors des phases de conception initiales de NDN, aucun routeur NDN dédié n'était disponible pour mettre en oeuvre diverses expérimentations, tandis que les chercheurs avaient besoin d'une plate-forme flexible pour éprouver leurs travaux sur NDN. Pour répondre à une telle demande, le simulateur NDN [121] a été introduit, prenant en charge de nombreux travaux antérieurs. Au fil du temps, le protocole NDN s'est stabilisé et il propose maintenant une implémentation (*NDN Forwarding Daemon – NFD*) fonctionnelle [5]. Les travaux récents se sont donc déplacés vers le déploiement, la gestion et la sécurité. Par ailleurs, la mise en oeuvre de solutions expérimentales, s'est vue facilitée par les techniques de virtualisation [155]. Ainsi, NDN et sa sécurité doivent être étudiés dans des conditions de déploiement au plus proche du réel pour comprendre pleinement la faisabilité des attaques et leurs impacts, chose qui ne peut pas être couverte par des moyens de simulation.

A.4.2 Conditions d'évaluation des attaques NDN

Les expériences pour chaque attaque sont menées sur une topologie qui comprend plusieurs clients, des routeurs intermédiaires avec NFD installé et des serveurs/-fournisseurs NDN. L'environnement de déploiement de cette topologie qui sert de cas d'utilisation à notre étude expérimentale de caractérisation des attaques est décrite dans Figure A.1.

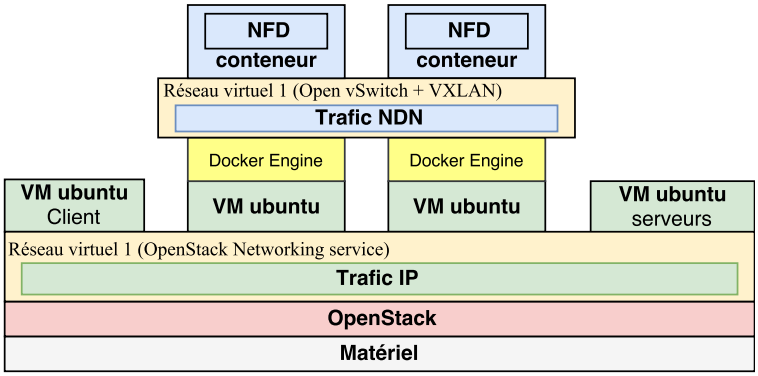


FIGURE A.1: Environnement de déploiement des expérimentations de caractérisation d'attaques NDN

On utilise l'environnement de Cloud OpenStack pour contrôler les ressources sur plusieurs équipements physiques. Pour chaque nœud, une machine virtuelle (VM) est créée avec un système d'exploitation Ubuntu. Les machines virtuelles sont connectées à un réseau virtuel via le service de mise en réseau OpenStack. Les VM

émulant des nœuds réseaux, elles hébergent des conteneurs Docker dans lesquels sont installés NFD. Pour permettre le partage de ces conteneurs sur différentes machines virtuelles, OpenVSwitch est installé dans chaque machine virtuelle en tant que structure de transport d'un conteneur à un autre. En tant que tel, le trafic NDN est encapsulé dans les canaux IP/UDP et transporté à travers un tunnel VxLAN.

A.4.3 Caractérisation de l'attaque IFA

Dans cette section, on étudie la faisabilité de l'attaque IFA (Section A.3.2) et nous soulignons son impact dans le déploiement réel de NDN.

Scénario étudié de NDN couplé à une application IP

Pour des raisons de continuité des services opérés, la transition du réseau IP actuel vers NDN ne peut pas être effectuée en une fois. Ainsi, on étudie un scénario dans lequel un fournisseur d'accès Internet (ISP) couple un réseau NDN au réseau IP existant pour fournir le service Web. HTTP est sélectionné comme exemple d'application IP, car il est le plus service le plus utilisé sur Internet. Dans ce scénario, un îlot NDN est déployé à l'intérieur du réseau central d'un FAI pour tirer parti des avantages de son système de mise en cache et de livraison de données à faible latence. Comme les utilisateurs actuels d'Internet n'implémentent pas NDN, leur trafic IP sera transmis à des passerelles dédiées [92] qui traduisent le trafic HTTP en trafic NDN et vice versa. Il existe deux types de passerelles, à savoir *ingress gateway* (iGW) et *egress gateway* (eGW). Bien que toutes les opérations des passerelles et l'existence de l'îlot NDN soient transparentes pour les utilisateurs du réseau, ils peuvent néanmoins bénéficier des avantages de NDN sans aucun effort d'adaptation de leur part.

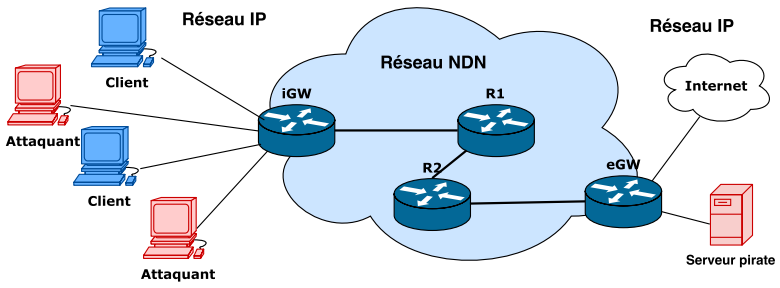


FIGURE A.2: Un exemple illustratif de NDN couplé à IP

Scénario IFA proposé dans un contexte de déploiement réel

Le mécanisme IFA dans la littérature n'est plus réalisable dans le scénario étudié en raison de l'existence de *NACK* et du manque de contrôle de l'attaquant sur le trafic NDN. Cependant, nous avons démontré que la menace de l'IFA persiste. Dans le scénario étudié, l'attaque IFA peut être lancée avec un serveur web malveillant et

des robots contrôlés par l'attaquant (Figure A.2). Pour ce faire, les robots vont naviguer sur le site hébergé sur le serveur malveillant, qui retarde intentionnellement sa réponse. Ainsi, les *Interests* malveillants occuperont les routeurs NDN et par conséquent, les utilisateurs légitimes subiront des délais plus longs lors de l'accès à un site Web. On note que le délai de retard de l'attaque ne peut toutefois pas être trop long et rester sous le seuil de déclenchement d'un paquet *NACK*. Le paquet *Interest* expirant automatiquement et libérant des ressources pour le routeur, il est difficile d'arrêter un routeur lorsque l'attaquant n'a pas le contrôle de *Interests* dans le NDN. Par conséquent, un tel scénario est susceptible de ralentir le réseau plutôt que de nier complètement le service.

Configuration expérimentale

Pour caractériser les impacts de l'attaque IFA, nous avons déployé la topologie décrite dans la Figure A.2, en utilisant les outils décrits dans la Section A.4.2. Le réseau NDN est constitué de quatre nœuds avec NFD (v0.5.1 avec *NACK* implémenté). La *iGW* est connectée aux utilisateurs Web, tandis que la *eGW* est connectée à l'Internet et au serveur malveillant exécutant un serveur HTTP Apache. Les données provenant des expériences sont recueillies en utilisant la sonde Montimage Monitoring Tool (MMT). Toutes les 4 secondes, la sonde génère un échantillon incluant le nombre de *Interests* entrants et sortants *Data* pour l'interface de chaque routeur. Pour générer du trafic Web, nous avons développé un émulateur d'utilisateur à l'aide de l'API Jaunt. Le site web consulté est sélectionné au hasard parmi une liste donnée de 90 sites populaires, sur la base d'une distribution Zipf [94].

On peut ainsi contrôler la quantité de trafic généré en modifiant : le nombre d'émulateurs, le nombre de processus légers associé à chaque émulateur et le temps moyen de navigation émulé. Pour faciliter la compréhension, la quantité de trafic est présentée en termes de requêtes HTTP par seconde. Chaque expérience comprend 15 minutes sans IFA suivies de 15 minutes avec le trafic IFA. Dans toutes les expériences, il y a 80 requêtes HTTP légitimes par seconde. Du côté de l'attaquant, les émulateurs malveillants ne demandent que le site hébergé sur le serveur de l'attaquant et le parcourent rapidement. On fait varier la puissance d'attaque, et chaque configuration d'attaque est exécutée 10 fois. Le retard du serveur malveillant suit une distribution uniforme $\mathcal{U}(4.9, 5.1)s$. Le site hébergé contient beaucoup d'objets[1] afin que les sessions vers le serveur malveillant durent plus longtemps.

[1] Pour être précis, le code HTML du Web malveillant est de 60 Ko et le site web complet, y compris les objets, de 1,6 Mo. Ainsi, une requête HTML aura pour résultat plus d'un paquet Data de 4 Ko dans NDN, ce qui augmentera le traitement du routeur. Avec les pages Web actuelles et la taille de leur contenu (plus volumineuses que notre site Web), l'efficacité de l'IFA proposé pourrait être considérablement supérieure.

Phénomène d'attaque

Dans le cadre de l'attaque IFA, le réseau NDN est occupé de manière intensive et les utilisateurs subissent un délai plus long lors du chargement d'un site Web. Pour fournir une vue globale de l'effet IFA sur la totalité de la liste de sites Web avec différents retards, on enregistre le temps de chargement pour chaque site Web, 20 fois de suite, sans navigation (i.e. passage d'une page à une autre sur un même site). Les sites web qui prennent trop de temps pour récupérer leur contenu (> 300) sont considérés comme inaccessibles. La Figure A.3 illustre le retard moyen sous attaque en fonction du délai moyen sans attaque. Chaque point représente la mesure d'un site Web individuel. La ligne pointillée noire montre l'équation $y = x$ pour la lisibilité. Les lignes pleines montrent les résultats d'une régression linéaire de ces mesures, en utilisant l'erreur quadratique moyenne. On peut noter que l'attaque est réussie même quand elle a une faible puissance (ligne verte). Cette tendance n'est cependant pas significative puisqu'elle est assez proche de la ligne pointillée. Lorsque la puissance d'attaque augmente, le retard augmente, tel qu'indiqué par l'augmentation de la pente.

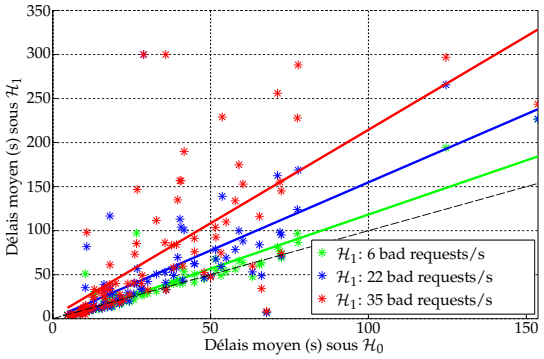


FIGURE A.3: Effet de l'attaque sur le délai d'un site web individuel

A.4.4 Caractérisation de l'attaque CPA

Dans cette section, nous décrivons trois scénarios pour effectuer une attaque CPA. Nous effectuons ensuite des expériences avec différentes configurations d'attaques et nous mettons en évidence les impacts sur les entités du réseau. Ensuite, nous présentons le résultat d'une *Analyse par Composantes Principales* (ACP) qui met en évidence les corrélations de métriques et de paramètres. Le travail dans cette section est le résultat de notre collaboration avec des collègues du LORIA, Université de Lorraine, Nancy.

Topologie et Comportement des Entités

Pour étudier l'impact de l'attaque CPA, nous implémentons la topologie représentée dans la Figure A.4, en utilisant les outils présentés dans la Section A.4.2. Nous croyons que notre topologie et les comportements de nos entités sont suffisants pour

démontrer la faisabilité de l'attaque ACP et ses impacts pour les raisons suivantes. Premièrement, elle présente le rôle général des différents nœuds impliqués et reflète un réseau typique avec un routeur d'accès R1, un routeur central R2 et un routeur d'accès R1 qui fournit le système d'accès et de mise en cache au serveur légitime. Deuxièmement, le comportement de l'utilisateur est plus réaliste, car il utilise des champs optionnels de *Interest* packet (Section A.2.2). Les clients légitimes n'acceptent que les bons paquets *Datas* et peuvent utiliser le champ *Exclude* pour exlure un paquet *Data* empoisonné, dont la signature est invalide. Au contraire, les attaquants excluent les bons paquets *Datas*, favorisant ainsi la dissémination des paquets *Datas* du serveur pirate. Troisièmement, la topologie représente l'un des cas les plus graves, car la disponibilité légitime du contenu est limitée à une route vers le bon serveur. En outre, le serveur légitime a un délai plus long dans la livraison du contenu que le serveur pirate. Cependant, ce délai est réduit pour les *Interests* une fois qu'un paquet *Data* est mis en cache dans R1. En outre, en raison de la validité de son identité, nous considérons que le chemin vers le serveur légitime a un coût inférieur à celui du serveur pirate. Le serveur légitime met automatiquement à jour ses paquets *Datas* quand leur *FreshnessPeriod* expire. La topologie considère aussi que le serveur pirate se connecte à R2. Pour accroitre son impact sur le serveur légitime, nous considérons que l'attaquant sélectionne toujours le contenu le plus populaire à empoisonner, étant donné une distribution de popularité. De plus, le serveur pirate mettra à jour ses paquets *Datas* à chaque fois qu'un mauvais paquet *Data* actuel commence à être exclu, maintenant la persistance de l'attaque.

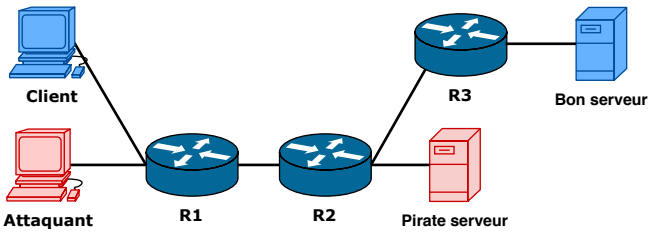


FIGURE A.4: Topologie étudiée pour CPA

Attaques Scénarios

Dans cette sous-section, nous décrivons trois scénarios CPA, à savoir *unsolicited*, *multicast* et *bestroute*. Le scénario *unsolicited* exploite une faille découverte dans NFD (v0.4.1). Dans cette version, NFD accepte un paquet *Data* même si sa face entrante ne correspond pas à la face sortante du précédent *Interest*. Par conséquent, un paquet *Data* non sollicité de faces sans route établie légitime peut être considéré comme valide, donc consommer des entrées de la PIT. Par conséquent, l'attaquant peut déployer un serveur non sollicité en prenant le contrôle d'un client se connectant à R2, puis envoie un mauvais paquet *Data* avec des noms de contenu populaires aléatoires

à R2 pour correspondre à des *Interests* en attente de R2 et être inséré dans le cache.

Dans le scénario *multicast*, R2 exécute la stratégie de transfert de ses *Interests* vers toutes ses faces. Ainsi, en recevant un *Interest*, il transmet l'*Interest* à toutes les faces admissibles dans l'entrée FIB de correspondance. Les clients incorrects envoient régulièrement des *Interests* pour le nom de contenu ciblé, mais excluent les copies actuelles de bon paquet *Data* pour contourner les caches. Cela oblige R1 à transmettre ces *Interests* à R2, qui les multidiffusent aux deux serveurs. Par conséquent, un paquet *Data* est retourné par les deux serveurs. En raison du délai plus court, le mauvais paquet *Data* arrive en premier à R2, résoud l'entrée PIT correspondante et est mis en cache dans R1 et R2.

Dans le scénario *bestroute*, R2 exécute la stratégie de bestroute - le paramètre par défaut de NFD [5]. Par conséquent, R2 transmet un *Interest* à la face avec le coût le plus bas dans l'entrée FIB correspondante. S'il y a deux faces avec le même coût le plus bas, le routeur utilise la premiere enregistrée. Après la transmission d'un *Interest*, le routeur NDN supprimera l'*Interest* similaire pendant un intervalle de suppression de retransmission. Après un tel intervalle, un *Interest* similaire est transmis à la prochaine face la moins chère qui n'a pas été utilisée auparavant, ouvrant ainsi la porte au serveur pirate. Grâce à la collaboration avec des clients malveillants, des *Interests* supplémentaires sont générés, forçant R2 à utiliser l'autre route vers le serveur pirate, poussant ainsi les mauvais paquets *Datas* vers R1 et R2[2].

Configuration expérimentale

La plupart de nos valeurs de configuration constantes sont motivées dans [79]. Il y a 10000 contenus dans la popularité. Chaque expérience dure 10 minutes, avec les 5 premières minutes sans CPA suivie de 5 minutes avec CPA. Chaque configuration d'attaque est répétée 5 fois, et tous les points de la courbe représentent la moyenne de toutes les mesures sur la période d'attaque et sur les 5 répétitions, avec un intervalle de confiance de 95%. Le comportement du client légitime et de l'attaquant est implémenté à l'aide de jNDN. Le nombre de *Interests* envoyés par un client suit une distribution de Poisson, et le nom de contenu demandé est sélectionné à partir d'une distribution Zipf. Un tel modèle a été utilisé par des travaux antérieurs dans les ICN [97, 98]. Le taux d'envoi d'un client légitime est de 10 *Interests*/s par défaut. Une fois que le client a reçu un paquet *Data* légitime, il mémorise les noms exclus à utiliser ultérieurement lorsqu'il demande à nouveau le même contenu. Un

[2] Il est à noter que lors de la rédaction de ce rapport de thèse, la spécification du format de paquet NDN a été mise à jour vers la version 0.3. Dans cette version, le champ d'*Interest* à exclure a été supprimé. Cependant, nous avons examiné la version mise à jour de la bibliothèque NDN. En spécifiant un nom de contenu complet pouvant uniquement être identifié par un fournisseur incorrect, les attaquants extraient délibérément des *Data* incorrectes et les routeurs les mettent en cache. La situation est encore pire pour un bon client, car sans champ Exclure, les bons clients ne peuvent désormais plus éviter les *Data* malveillantes précédemment reçues.

tel comportement est plus sûr et plus réaliste parce que les clients ne savent pas si le mauvais paquet *Data* existe encore dans le réseau.

Nous considérons le *taux d'attaque* comme le paramètre principal qui influe sur le succès de l'attaque. Pour le scénario *unsolicited*, le taux d'attaque est le nombre de mauvais paquets *Datas* envoyés par seconde par le serveur *unsolicited* dans l'intervalle $[10, 1000]$ suivant une échelle logarithmique. La valeur par défaut est 50 *Data*/s. Pour les scénarios *multicast* et *bestroute*, le taux d'attaque est le nombre moyen de mauvais *Interests* envoyés par seconde, qui varie dans l'intervalle $[1, 1000]$ suivant une échelle logarithmique et est défini sur 10s comme valeur par défaut. Le deuxième paramètre d'attaque est le *pourcentage de contenu empoisonnés*. Sa valeur varie dans l'intervalle $[0.01, 10]$ du pourcentage de la popularité du contenu, suivant une échelle logarithmique et elle est définie à 1% comme valeur par défaut.

Phénomène d'attaque

Dans cette sous-section, nous étudions l'impact de l'attaque sur le client, le serveur et les caches des routeurs. L'impact sur le client légitime est évalué par le pourcentage de mauvais paquets *Datas* qu'un client légitime reçoit. Nos résultats montrent que dans le scénario *bestroute*, le client légitime souffre le moins de l'attaque CPA. Les dégâts augmentent légèrement lorsque le taux d'attaque est supérieur à celui du trafic légitime et restent presque les mêmes pour les taux d'attaque plus élevés. Pendant ce temps, le scénario *unsolicited* empoisonne plus efficacement le client légitime, en particulier avec un taux d'attaque élevé. D'un autre côté, le scénario *multicast* ne présente pas de tendance évidente. Cependant, il est clair qu'avec un taux d'attaque élevé, son effet est plus sévère que *bestroute* et moins sévère que le scénario *unsolicited*. Les résultats montrent également que le nombre de contenus cibles n'a pas un impact clair sur le client légitime dans tous les scénarios. De plus, le nombre de contenus empoisonnés varie, le scénario *bestroute* maintient toujours sa protection contre CPA. Cela implique que si un attaquant veut améliorer les dégâts sur des clients légitimes, il devrait plutôt se concentrer sur quelques-uns très populaires.

Pour évaluer l'impact sur le serveur légitime, nous mesurons la différence entre le nombre moyen d'*Interests* que le serveur reçoit par seconde après et avant l'attaque, ce qui montre la charge supplémentaire induite sur le serveur. Le résultat montre que le scénario *unsolicited* n'a aucun impact sur le serveur légitime quelle que soit la valeur des paramètres d'attaque. Par contre, pour les scénarios *bestroute* et *multicast*, l'effet sur le serveur est lié au taux d'attaque et a une tendance similaire. Les deux scénarios montrent une croissance presque linéaire de la surcharge du serveur avec la gamme de contenus empoisonnés.

L'impact sur le cache des routeurs est évalué par la proportion moyenne de

"bons hits" (*cache hit* par bon paquet *Data*), "mauvais hits" (*cache hit* par mauvais paquet *Data*) et "miss" (aucun paquet *Data* n'est trouvé dans le cache) dans les caches de R1 et R2 en fonction du taux d'attaque pour chaque scénario. Nos résultats indiquent que le scénario *unsolicited* est plus efficace en polluant les caches des routeurs (fort pourcentage de "mauvais hits") dans le chemin vers le client, mais son efficacité diminue quand il traverse les routeurs. Nous pouvons également remarquer que le plus grand nombre d'échecs sur R1 par rapport aux autres scénarios a de bonnes chances de finir comme "mauvais hit" sur R2. En ce qui concerne les scénarios *bestroute* et *multicast*, ils ont un impact moins important sur le routeur principal R2 mais ont toujours un impact important sur la diminution de "bons hits" sur le routeur d'accès R1.

Empreinte de l'attaque avec une analyse par composantes principales

Afin de comprendre globalement nos données, à variables multiples, nous avons effectué une ACP sur l'ensemble global des données afin de révéler les corrélations de toutes les métriques et de tous les paramètres. Les deux premières composantes représentent 80,5% de la variance totale des données. La première composante, représentant 56,5% de la variance des données, est caractérisée par un fort impact sur le "mauvais hit", des ressources gaspillées pour ces "mauvais hits" des deux routeurs, ainsi qu'un nombre élevé de mauvaises réponses au bon client. En tant que tel, cette première composante représente l'impact attendu principal de l'attaque CPA avec l'injection de mauvais paquets *Datas* dans le cache des routeurs. Au contraire, la deuxième composante, représentant 24% de variance des données, montre un impact similaire sur le nombre de mauvaises réponses au client, mais un impact beaucoup plus important sur le taux de "miss" du routeur d'accès, dans une moindre mesure le routeur principal et le trafic supplémentaire vers le fournisseur. Cela montre l'effet secondaire de l'attaque CPA qui empêche les routeurs de mettre en cache les bons paquets *Datas*, créant ainsi un taux plus élevé d'échecs et de trafic vers le bon fournisseur.

La Figure A.5 présente la projection des mesures individuelles sur les axes des deux premières composantes. La flèche en trait plein représente la projection moyenne de chaque scénario et le '+' représente les projections d'expériences individuelles. Les couleurs bleue, verte et rouge représentent respectivement le scénario *unsolicited*, *multicast* et *bestroute*. Le cercle cyan indique les projections des expériences ayant le moins d'impact (taux d'attaque le plus faible). Les flèches pointillées en noir indiquent la direction des mesures lorsque le taux d'attaque augmente. La figure montre que le scénario *unsolicited* a une empreinte spécifique principalement capturée par la première composante principale, confirmant son fort impact d'injection

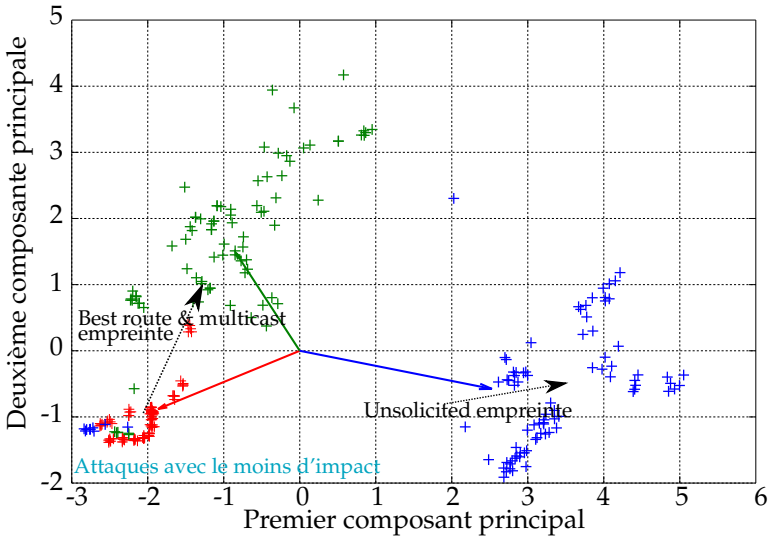


FIGURE A.5: Projections des mesures sur les deux premiers composants principaux

de mauvais paquets *Datas* dans le cache des routeurs (pourcentage éléevé de "mauvais hits"). En outre, les scénarios *bestroute* et *multicast* montrent un impact similaire lorsque le taux d'attaque augmente, principalement représenté par la seconde composante principale et, dans une moindre mesure, par la première composante principale. En effet, ces scénarios créent un taux plus élevé de "miss", et autant les bons que les mauvais clients tentent d'empêcher la mise en cache des mauvais et bons paquets *Datas*, respectivement. Cela explique aussi le plus grand nombre de *Interests* envoyés au fournisseur.

A.5 Un micro-détecteur fondé sur la théorie des tests d'hypothèses

Dans cette section, nous proposons un micro-détecteur fiable pour notre plan de surveillance. Puisque l'attaque IFA peut être caractérisée par une seule mesure (Section A.4), elle est sélectionnée comme premier cas d'utilisation pour la conception de notre micro-détecteur.

A.5.1 Formalisation du problème

Dans cette section, nous représentons le problème de détection de l'attaque IFA sous la forme d'un test d'hypothèses statistiques et nous en mettons en évidence les difficultés.

Définitions

Soit i_t et d_t, respectivement, le nombre d'*Interests* entrants et de paquets *Datas* sortants à un instant t mesuré pour une face d'un routeur. Le ratio (d_t/i_t) est le taux de satisfaction d'*Interest*, un indicateur largement utilisé dans la littérature pour la

détection de l'attaque IFA. Idéalement, pour chaque face, chaque *Interest* entrant devrait aboutir à un *Data* sortant. Cependant, une partie des paquets peut être perdue. Par conséquent, on définit $\ell_t = 1 - \frac{d_t}{i_t}$ le taux de perte mesuré à chaque instant. On suppose que tous les *Interests* ont la même probabilité d'être perdus [53, 54], noté $p_t \geq 0$. Dans la situation normale, une telle probabilité correspond à un taux de pertes mesuré attendu, c'est-à-dire $\mathbb{E}(\ell_t) = p_t$. Ainsi, d_t suit une distribution binomiale $\mathcal{B}(i_t, 1 - p_t)$. Lorsque l'attaque IFA se produit, une partie importante d'*Interests* est envoyée au serveur pirate qui retarde leur réponse, entraînant une augmentation brutale de ℓ_t.

Soit $i_t^\star$ et N_{a_t}, respectivement, sont le nombre de *Interests* légitimes et de *Interests* malveillants reçus à un instant t pendant l'attaque. Ainsi, l'attaque IFA peut être caractérisée par une augmentation du nombre de *Interests* $i_t = i_t^\star + N_{a_t}$. Il est intéressant de noter que la distinction entre les éléments légitimes et l'ensemble du flux de données est impossible dans la pratique. De plus, comme les *Interests* N_{a_t} malveillants sont assujétis à un délai, l'espérance du taux de perte instantané est augmentée comme suit :

$$a = \mathbb{E}(\ell_t) - p_t = \frac{(1 - p_t)N_{a_t}}{i_t} > 0. \tag{A.1}$$

Problème de détection

Le problème de détection de l'attaque IFA consiste à choisir entre deux hypothèses : $\mathcal{H}_0$ impliquant que "d_t *est cohérent avec ce qui est attendu de* i_t *et* p_t" et $\mathcal{H}_1$ impliquant que "d_t *est significativement plus bas que ce qui est attendu de* i_t *et* p_t". Ces deux expressions peuvent être écrites formellement comme les hypothèses statistiques suivantes :

$$\begin{cases} \mathcal{H}_0 \left(p_t, N_{a_t} = 0\right) : d_t \sim \mathcal{B}\left(i_t, 1 - p_t\right), \\ \mathcal{H}_1 \left(p_t, N_{a_t} > 0\right) : d_t \sim \mathcal{B}\left(i_t - N_{a_t}, 1 - p_t\right). \end{cases} \tag{A.2}$$

Nous nous concentrons sur l'approche bi-critères de Neyman-Pearson qui vise à la fois à garantir une probabilité de fausses alarmes (*Probability of False Alarm – PFA*) garantie tout en maximisant la puissance de détection. Les hypothèses formulées dans l'Eq. (A.2) mettent en évidence deux difficultés principales de ce problème de test. Premièrement, l'attaque IFA affecte à la fois l'espérance et la variance des distributions, ce qui rend difficile le contrôle de la PFA du test. Deuxièmement, la charge utile d'attaque N_{a_t} et le taux de perte attendu p_t sont inconnus en pratique. Concevoir un test pour de telles hypothèses et établir ses performances sont beaucoup plus difficiles que pour celles qui dépendent d'un seul paramètre.

A.5.2 Test du rapport de vraisemblance optimal pour le p_t connu

Dans cette section, nous abordons le problème de détection dans le cas idéal lorsque le taux de pertes $0 < p_t < 1$ est connu à l'avance. Les résultats d'un tel cas fourniront une limite supérieure aux performances que l'on peut attendre dans d'autres cas de ce même problème. Nous avons utilisé le *théorème de la limite centrale* (CLT) (voir [101, Théorème 11.2.5]) pour transformer asymptotiquement le problème de détection et trouver le test du rapport de vraisemblance (LRT) optimal. Soit $r_t = \ell_t - p_t$ le taux de pertes résiduel, i.e., la différence entre les taux de pertes observées et prévues. Après avoir appliqué le CLT, le problème du test, voir l'Eq. (A.2), peut être reformulé comme :

$$r_t \sim \begin{cases} \mathcal{N}\left(0, \sigma_t^2\right) & \text{under } \mathcal{H}_0, \\ \mathcal{N}\left(a, \sigma_t^2 - \sigma_a^2\right) & \text{under } \mathcal{H}_1. \end{cases} \tag{A.3}$$

où $\sigma_t^2 = \frac{p_t(1-p_t)}{i_t} > 0$ est la variance sous $\mathcal{H}_0$ et $\sigma_a^2 = \frac{ap_t}{i_t} < \sigma_t^2$ est la diminution de la variance induite par l'attaque IFA. On peut noter que σ_a^2 est causé par l'augmentation de i_t pendant l'attaque, alors que d_t ne change pas. Le paramètre a intègre l'impact de l'attaque IFA sur le taux de perte, et sera donc utilisé pour indiquer la charge de l'attaque.

Pour trouver le test *uniformément le plus puissant* (UMP), on utilise le théorème [101, Théorème 3.4.1] qui propose une formulation du test UMP entre deux hypothèses dont la distribution dépend d'un paramètre scalaire, et dont leur LRT est monotone. Nous avons prouvé que le LR $\Lambda(r_t)$ du problème dans l'Eq. (A.3) est monotone si $\frac{d_t}{i_t} > 0$, ce qui est toujours vrai. Ainsi, selon [101, Théorème 3.4.1], le test UMP pour le problème de détection dans l'Eq. (A.3) est :

$$\delta^\star(r_t) = \begin{cases} \mathcal{H}_0 & \text{if } r_t \leq \tau^\star, \\ \mathcal{H}_1 & \text{if } r_t > \tau^\star. \end{cases} \tag{A.4}$$

Le test proposé $\delta^\star(r_t)$ est *Asymptotiquement UMP* (AUMP) pour tester le problème dans l'Eq. (A.2) et ses propriétés statistiques sont présentées dans la proposition suivante:

Proposition 3. *En supposant que le nombre de Interests i_t tend vers l'infini, pour toute probabilité de fausse alarme prescrite α_0, le seuil de décision, $\tau^\star$, donné par :*

$$\tau^\star(\alpha_0) = \sigma_t \Phi^{-1}\left(1 - \alpha_0\right), \tag{A.5}$$

qui garantit que le test $\delta^\star$, voir l'Eq. (A.4), est dans $\mathcal{K}_{\alpha_0}$. Ici, Φ et Φ^{-1} sont la fonction de distribution cumulative normale standard et sa fonction inverse, respectivement. En utilisant le seuil de décision donné dans l'Eq. (A.5) la fonction de puissance du test UMP $\delta^\star$, voir l'Eq. (A.4), est donnée par :

$$\beta_{\delta^\star(r_t)}(a) = 1 - \Phi\left(\frac{\sigma_t \Phi^{-1}(1-\alpha_0) - a}{\sqrt{\sigma_t^2 - \sigma_a^2}}\right). \tag{A.6}$$

Notez que le seuil $\tau^\star$ dépend uniquement de α_0, i_t et p_t qui sont connus. Cette approche simplifie les problèmes avec la distribution binomiale. Cependant, une conséquence notable de la distribution binomiale sous-jacente est que l'attaque IFA influe à la fois sur l'espérance et la variance de r. Par conséquent, la fonction de puissance du test AUMP proposé dépend non seulement de la charge utile d'attaque a mais aussi de l'impact de l'IFA sur la variance de r_t.

A.5.3 Test du rapport de vraisemblance généralisé pour le p_t inconnu

En réalité, on connaît rarement le taux de pertes de paquets a priori. Ainsi, le test AUMP est difficilement réalisable. On aborde dans cette section un cas réaliste dans lequel le p_t est inconnu. Une approche habituelle consiste à concevoir un *test du rapport de vraisemblance généralisé* (GLRT) en substituant l'inconnue p_t à son *estimation de vraisemblance maximale* (MLE) à partir des mesures de ℓ_t.

Modélisation du taux de perte de paquets

Pour modéliser le taux de pertes, nous considérons une fenêtre de taille $N > 0$ des mesures les plus récentes du taux de pertes $\boldsymbol{\ell}_t = [\ell_{t-N+1}, \ldots, \ell_t]^T$. Sous $\mathcal{H}_0$, la fluctuation du taux de pertes l_t est limitée et continue [102, 103], donc l_t peut être modélisé par un polynôme. Ainsi, $\boldsymbol{\ell}_t$ peut être écrit en notation matricielle comme :

$$\boldsymbol{\ell}_t = \mathbf{H}\mathbf{x}_t + \boldsymbol{\epsilon}_t \tag{A.7}$$

où $\mathbf{x}_t = [x_0, \ldots, x_{q-1}]^T$ est le vecteur des coefficients q du polynôme ; $\boldsymbol{\epsilon_t} = [\epsilon_{t-N+1}, \ldots, \epsilon_t]^T$ est le vecteur d'erreurs des N mesures considérées ; et $\mathbf{H}$ est une matrice de taille $N \times q$ dont les éléments $h_{(n,j)} = n^j$, $n \in \{1, \ldots, N\}$, $j \in \{0, \ldots, q-1\}$. Un tel modèle a été largement utilisé dans le traitement du signal, voir [110, 112, 113] pour des applications dans la modélisation du trafic Internet et le traitement des images. Dans les conditions de Gauss-Markov, il est bien connu que l'estimation *le plus petit carré ordinaire* (OLS) est équivalente à la MLE. Ainsi, le MLE de $\mathbf{p}_t$ est donnée par :

$$\widetilde{\mathbf{p}}_t = \mathbf{H}\widetilde{\mathbf{x}}_t = \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\boldsymbol{\ell}_t. \tag{A.8}$$

Proposed Test and its Properties

Sous les conditions de Gauss-Markov [107], l'espérance de $\boldsymbol{\ell_t}$ devient $\mathbf{H}\mathbf{x}_t$. En conséquence, le taux de pertes résiduel estimé $\mathbf{r}_t$ est défini comme $\widetilde{\mathbf{r}}_t = \boldsymbol{\ell}_t - \widetilde{\mathbf{p}}_t = \mathbf{H}^\perp\boldsymbol{\ell}_t$

où $\mathbf{H}^{\perp} = \mathbf{I}_N - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T$ est une matrice carrée de taille N représentant la projection au complément orthogonal du sous-espace engendré par les colonnes de $\mathbf{H}$. Ainsi, le problème de détection pour le taux de perte inconnu est formulé comme:

$$\begin{cases} \mathcal{H}_0 : & \widetilde{\mathbf{r}}_t \rightsquigarrow \mathcal{N}\left(\boldsymbol{o} \quad , \quad \mathbf{H}^{\perp}\Sigma_0\mathbf{H}^{\perp\,T}\right), \\ \mathcal{H}_1 : & \widetilde{\mathbf{r}}_t \rightsquigarrow \mathcal{N}\left(a\widetilde{\mathbf{v}}_a \quad , \quad \mathbf{H}^{\perp}\boldsymbol{\Sigma}_0\mathbf{H}^{\perp\,T} - \mathbf{H}^{\perp}\boldsymbol{\Sigma}_a\mathbf{H}^{\perp\,T}\right). \end{cases} \tag{A.9}$$

où $\boldsymbol{\Sigma}_0$ est une matrice de covariance diagonale dont les éléments sont donnés par $\boldsymbol{\Sigma}_{0(n,n)} = \frac{p_n(1-p_n)}{i_n}$, $n \in \{t-N+1,\ldots,t\}$; a est la charge de l'attaque ; $\mathbf{v}_a$ est un vecteur binaire indiquant des instants où le taux de perte augmente à cause de l'IFA ; $\boldsymbol{\Sigma}_a$ est une matrice diagonale représentant la diminution de la variance de ℓ_t due à l'IFA ; $\widetilde{\mathbf{v}}_a = \mathbf{H}^{\perp}\mathbf{v}_a$ représentant l'empreinte IFA obtenue en estimant et en supprimant le taux de perte attendu. Il est à noter que $\mathbf{H}^{\perp}$ et $\mathbf{v}_a$ sont connus une fois la configuration corrigée.

Le GLRT est atteint en intégrant l'inconnue p_t dans le LRT (A.4) avec son MLE $\widetilde{p}_t$ (A.8). Cependant, il est à noter que dans Eq. (A.9), la variable du test $\widetilde{\mathbf{r}}_t$ et les paramètres d'hypothèses sont des vecteurs. Pour réaliser un test ordonné et compact, il est proposé d'utiliser la valeur scalaire $\widetilde{\mathbf{c}}_t = \widetilde{\mathbf{v}}_a^T\widetilde{\mathbf{r}}_t$ comme statistique de test :

$$\widetilde{\mathbf{c}}_t = \widetilde{\mathbf{v}}_a^T\widetilde{\mathbf{r}}_t \rightsquigarrow \begin{cases} \mathcal{N}\left(\mathbf{0}\,,\, s_0^2\right) & \text{under } \mathcal{H}_0, \\ \mathcal{N}\left(a\|\widetilde{\mathbf{v}}_a\|_2^2\,,\, s_0^2 - s_a^2\right) & \text{under } \mathcal{H}_1. \end{cases} \tag{A.10}$$

où $\|\widetilde{\mathbf{v}}_a\|_2^2 = \widetilde{\mathbf{v}}_a^T\widetilde{\mathbf{v}}_a$; $s_0^2 = \widetilde{\mathbf{v}}_a^T\mathbf{H}^{\perp}\boldsymbol{\Sigma}_0\mathbf{H}^{\perp\,T}\widetilde{\mathbf{v}}_a$ la variance GLR sous $\mathcal{H}_0$ and $s_a^2 = \widetilde{\mathbf{v}}_a^T\mathbf{H}^{\perp}\boldsymbol{\Sigma}_a\mathbf{H}^{\perp\,T}\widetilde{\mathbf{v}}_a$ la diminution de la variance sous $\mathcal{H}_1$. Le GLRT devient donc :

$$\widetilde{\delta}(\widetilde{\mathbf{r}}_t) = \begin{cases} \mathcal{H}_0 & \text{if } \widetilde{\mathbf{v}}_a^T\widetilde{\mathbf{r}}_t \leq \widetilde{\tau}, \\ \mathcal{H}_1 & \text{if } \widetilde{\mathbf{v}}_a^T\widetilde{\mathbf{r}}_t > \widetilde{\tau}. \end{cases} \tag{A.11}$$

Basée sur la distribution de la GLR (A.10), On peut établir le seuil de décision et la fonction de puissance du GLRT proposé :

Proposition 4. *En supposant que le nombre d'Interests entrants i_t tend vers l'infini, pour toute probabilité de fausse alarme prescrite α_0, le seuil de décision $\widetilde{\tau}$ est donné par :*

$$\widetilde{\tau} = \Phi^{-1}\left(1-\alpha_0\right)s_0, \tag{A.12}$$

garantit que le test $\widetilde{\delta}$, voir l'Eq. (A.11), est dans $\mathcal{K}_{\alpha_0}$. En utilisant le seuil de décision donné dans l'Eq. (A.12), la fonction de puissance du test UMP, voir (A.11), est donnée par :

$$\beta_{\widetilde{\delta}(\widetilde{\mathbf{r}}_t)}(a) = 1 - \Phi\left(\frac{s_0\Phi^{-1}(1-\alpha_0) - a\|\widetilde{\mathbf{v}}_a\|_2^2}{\sqrt{s_0^2 - s_a^2}}\right). \tag{A.13}$$

À partir de la fonction de puissance dans l'Eq. (A.13), il est à noter que la cause principale de la perte d'optimalité du GLRT proposé est le facteur $\|\widetilde{\mathbf{v}}\|_2^2$. Cela vient du fait qu'une partie non négligeable du taux de perte modifié en raison de l'IFA sera modélisée dans le cadre du changement régulier du trafic légitime.

A.5.4 Du test instantané à la détection séquentielle

En pratique, le taux de perte peut ne pas fluctuer régulièrement mais augmenter brusquement à un moment donné, conduisant à une fausse alarme. Par ailleurs, il est possible que l'empreinte d'attaque soit trop évasive pour empêcher sa détection instantanée. Cette section étend la méthode de détection «snapshot» précédente en prenant en compte les observations précédentes dans un cadre séquentiel pour gérer ces situations. Le cadre d'analyse séquentielle [117, 118] vise non seulement à détecter un événement spécifique avec la plus grande précision, en ce qui concerne le PFA et la puissance de détection, mais introduit également le retard comme troisième critère de performance. Le délai de détection est défini comme $DD = S_t - \nu$, où ν est l'heure de début de l'attaque IFA et $S_t \geq \nu$ est l'instant où l'attaque IFA est détectée pour la première fois.

Nous avons utilisé le *CUmulative SUM* (CUSUM) [119] bien connu qui a été montré comme optimal selon le critère de Lorden [120]. Le critère consiste à minimiser le retard de détection moyen du pire cas pour la moyenne *Run Length To False-Alarm* (RL2FA) du pire cas. L'idée principale de CUSUM est de calculer les LRs séquentiels et de les remettre à zéro chaque fois qu'il devient inférieur à zéro, étant donné que les observations sont indépendantes et que le point de changement n'est pas encore arrivé. Le CUSUM C_t est défini par l'équation récurrente suivante :

$$C_t = \max\left(0\,;\, C_{t-1} + \widetilde{\mathbf{v}}_a^T \widetilde{\mathbf{r}}_t - \kappa\right) \tag{A.14}$$

où $\mathbf{v}_a^T \widetilde{\mathbf{r}}_t$ est le GLR entre $\mathcal{H}_0$ et $\mathcal{H}_1$, voir Eq. (A.10), et κ est la sensibilité du CUSUM, à définir. Un gros κ nécessitera plus fréquemment la réinitialisation de C_t à 0 mais pourra aussi retarder la détection.

A.5.5 Résultats numériques

Dans cette section, nous évaluons la détection proposée avec des données simulées et des données réelles.

Évaluation des propriétés statistiques avec simulation

Nous avons évalué notre proposition avec deux ensembles de données simulées, l'une dans MATLAB et l'autre dans ndnSIM [121]. On réutilise une topologie de [53] qui utilise un arbre binaire avec huit hôtes, des routeurs intermédiaires et un

fournisseur de contenu. La topologie représente l'un des pires cas d'attaque IFA puisque tous les *Interests* sont agrégés vers les liens supérieurs et finalement vers le fournisseur de contenu. La valeur de i_t est générée à partir d'une distribution de Poisson dont la valeur moyenne est tirée d'une distribution uniforme. De plus, le taux de perte réel suit un modèle auto-régressif. Un tel modèle a été largement utilisé pour modéliser à la fois l'évolution des requêtes des utilisateurs et le taux de pertes dans un réseau informatique [102, 122]. Le taux de pertes est initialisé à $p_0 = 0,05$, puis son évolution est donnée par $p_t = p_{t-1} + u$ avec u tiré d'une distribution uniforme avec une moyenne nulle. Pour le GLRT proposé, nous fixons $N = 50$ et $q - 1 = 4$. Dans toutes les figures, sauf indication contraire, l'empreinte de l'attaque IFA est caractérisée par $\mathbf{v}_a = [0, \ldots, 1]^T$.

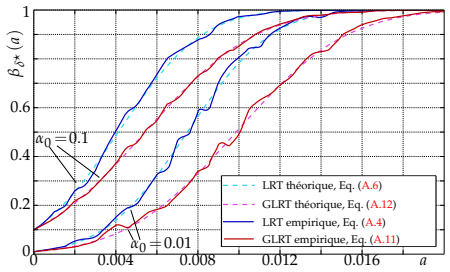
Pour les données simulées dans MATLAB, la Figure A.6a compare la puissance théorique et empirique du LRT optimal et du GLRT proposé avec deux PFA prescrites $\alpha_0 = 0,01$ et $0,1$. La fonction de puissance est tracée en fonction de l'anomalie $a \in [0, 0.02]$. La figure montre la pertinence des résultats théoriques puisque la puissance empirique est proche de la puissance théorique. Cependant, pour une PFA faible prescrite, telle que $\alpha_0 = 0,01$, les résultats empiriques sont légèrement moins précis car un plus grand nombre d'échantillons est nécessaire pour obtenir une PFA aussi faible.

Comparé aux données simulées par MATLAB, le nombre d'échantillons collectés dans ndnSIM est beaucoup plus petit car l'exécution de ndnSIM prend du temps. Cependant, les résultats obtenus montrent que la PFA empirique correspond toujours à la PFA théorique, indiquant la capacité à garantir la PFA prescrite. La Figure A.6b représente les courbes ROC pour différents nombres d'échantillons corrompus, notés $M =\in [1, 3, 5]$ pour comparer les performances théoriques et empiriques du GLRT. Comme prévu, la puissance augmente avec le nombre d'échantillons corrompus. Ainsi, la méthode proposée peut être adaptée pour assurer une plus grande précision de détection au détriment du délai de détection. La figure présente également une comparaison avec le détecteur proposé dans [53], qui repose sur le taux de satisfaction instantané bien connu d_t/i_t et un seuil fixe. Il convient de noter qu'une telle statistique ne peut pas traiter les comportements non stationnaires des utilisateurs. En effet, la figure montre que le GLRT proposé atteint une bien meilleure performance, même avec la détection la plus rapide ($M = 1$).

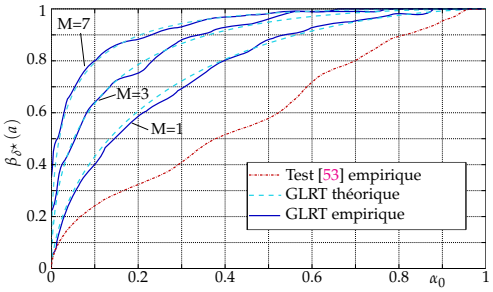
Évaluation du rendement en conditions réelles

Dans cette section, nous utilisons les données collectées à partir de notre environnement expérimental NDN et les scénarios décrits dans la Section A.4.3.

Taux de perte de paquets empirique réel : avec les données de l'architecture



(A) Puissance avec les données MATLAB

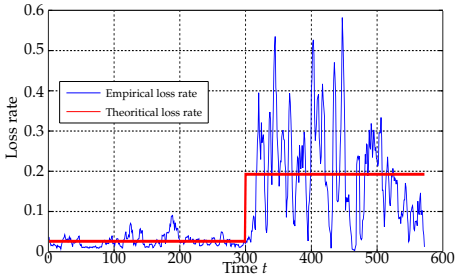


(B) Courbes ROC avec les données ndnSIM

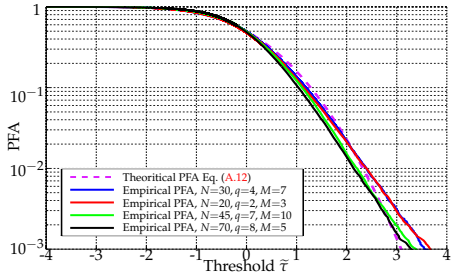
FIGURE A.6: Performances GLRT avec les données simulées

NDN réelle, nous observons une différence entre le taux de perte théorique et le taux de perte empirique, représenté dans la Figure A.7a. En théorie, lorsque l'attaque IFA commence, le taux de pertes devrait augmenter puis être maintenu à une valeur spécifique pendant la période d'attaque. Cependant, les données empiriques montrent que le taux de perte augmente et diminue de façon répétée pendant l'attaque. Ces changements se produisent assez brusquement et ne semblent suivre aucun schéma particulier. Ce phénomène est dû au fait que les paquets *Datas* retardés arriveront plus tard à un instant $t + x, x > 0$, compensant l'absence de paquets *Datas* provoqués dans cet instant $t + x$, changeant ainsi ℓ_t abruptement.

Garantie de la probabilité de fausse alarme et de la sélection de la configuration du détecteur : une propriété importante d'un test statistique est de garantir un PFA α_0 prescrit. Avec les données collectées sous $\mathcal{H}_0$, plusieurs configurations de détection ont été testées, à savoir la longueur de la fenêtre N, le degré du modèle q et le nombre M d'échantillons corrompus. Figure A.7b représente empiriquement la PFA en fonction du seuil $\widetilde{\tau}$, ainsi qu'une comparaison avec le PFA théorique donné dans l'Eq (A.12). Cette figure montre que les résultats empiriques correspondent aux résultats théoriques de l'ordre de $PFA > 5.10^{-3}$, ce qui démontre la précision de notre modèle sous $\mathcal{H}_0$. En outre, le PFA pour les différentes configurations de détecteur sont proches, montrant ainsi la flexibilité en garantissant α_0.



(A) Taux de pertes théorique et empirique

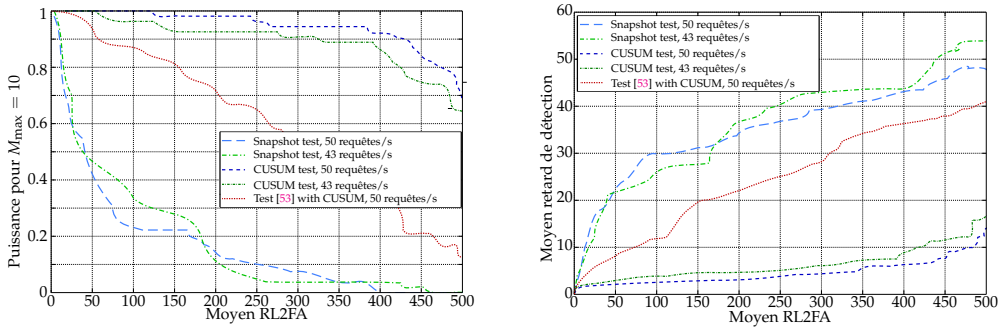


(B) PFA for différentes configurations du détecteur

FIGURE A.7: en fonction du RL2FA moyen

Performance du détecteur séquentiel : pour comparer les performances du test instantané dans l'Eq. (A.11) avec la détection CUSUM dans l'Eq. (A.14), la Figure A.8a et A.8b représentent, respectivement, la puissance de détection sous une contrainte de délai de détection maximum $\mathbb{P}\left[S_t - \nu \leq M_{\max}\right]$ et le délai de détection moyen (Section A.5.4) en fonction du RL2FA moyen. La comparaison est faite sur les deux figures pour deux charges d'attaque différentes, respectivement 43 et 50 mauvaise requêtes/s. Pour le test CUSUM, la constante κ est fixée à 0.005. Pour le test d'instantané, M est défini sur $10s$. Ces figures offrent également une comparaison avec le détecteur proposé dans [53] en remplaçant les statistiques GLRT proposées par le test [53] dans l'équation CUSUM (A.14).

La Figure A.8a et A.8b montrent clairement le gain significatif obtenu par la collecte d'échantillons consécutifs. Plus spécifiquement, la Figure A.8a montre que, pour un RL2FA moyen de 300 secondes (5 minutes), la probabilité de détecter un IFA après 10 secondes augmente d'environ 5 % pour le test instantané à plus de 90% en utilisant la procédure CUSUM. De même, la Figure A.8b montre que, pour le même RL2FA moyen, le délai moyen de détection est diminué d'environ 8, passant d'environ 40 secondes pour le test instantané à 5 secondes pour le test CUSUM.



(A) Puissance avec contrainte de $M_{\max}$ (B) Retard moyen de détection

FIGURE A.8: Performances du détecteur séquentiel

A.6 Conception d'un plan de surveillance de la sécurité pour NDN

Dans cette section, nous présentons notre conception d'un plan de surveillance de la sécurité pour NDN. Premièrement, nous montrons l'intéret d'un tel plan pour NDN, et passons brièvement en revue la littérature des réseaux Bayesiens, la méthodologie utilisée pour concevoir notre plan de surveillance. Deuxièmement, nous présentons nos principales contributions, qui comprennent une liste exhaustive des métriques à surveiller, un micro-détecteur polyvalent pour indiquer le comportement anormal de chaque métrique et un classifier fondé sur un réseau Bayesien (*Bayesian Network Classifier* – BNC) qui corrèle les alarmes des micro détecteurs. Enfin, la performance

du classificateur proposé par rapport au CPA est établie avec les données de notre environnement expérimental NDN. Les travaux de ce chapitre sont menés en collaboration avec nos partenaires de la société Montimage à Paris.

A.6.1 Besoin d'un plan de surveillance dans NDN

Bien qu'un détecteur dédié donne de bonnes performances pour une attaque particulière, le processus de conception est long et fastidieux. De plus, un tel détecteur est inapplicable à d'autres menaces en raison de différences dans les caractéristiques des attaques. Par conséquent, un opérateur de réseau a besoin d'une solution générique capable de s'attaquer non seulement aux attaques identifiées dans la littérature du NDN, mais aussi à celles qui pourraient apparaître à l'avenir. Un plan de surveillance de la sécurité peut offrir une solution de gestion de la sécurité avec les fonctions principales suivantes: (1) collecter et agréger les logs provenant de sources multiples; (2) surveiller en permanence les incidents; (3) corréler les logs et les événements pour détecter les menaces de sécurité et (4) émettre des notifications d'alerte [124]. L'accent est mis sur la proposition d'une liste de métriques à surveiller dans un nœud NDN et la fourniture d'un moteur de corrélation basé sur un réseau bayésien (BN) pour détecter localement les menaces de sécurité potentielles.

A.6.2 Introduction aux réseaux Bayésiens

Le moteur de corrélation proposé pour NDN est basé sur un réseau Bayésien (*Bayesian Network* – BN) [131–133] - une approche probabiliste orientée graphes qui formalise les relations causales tout en gérant l'incertitude en utilisant la théorie des probabilités. Un BN se compose de deux parties principales - la structure et les paramètres. Chaque nœud de sa structure représente un événement par une variable aléatoires. Une relation entre deux événements est visualisée par une arête entre deux nœuds correspondants, en commençant par le parent (l'événement cause) et en pointant vers un enfant (l'événement affecté). Les relations des nœuds sont quantifiées par leurs paramètres, c'est-à-dire la probabilité de la valeur d'un enfant étant donné les valeurs de ses parents. Ainsi, étant donné les valeurs d'un ensemble de nœuds, on peut déduire la probabilité de la valeur d'un nœud spécifique.

Les raisons de notre choix d'un BN sont nombreuses. Premièrement, dans un BN, la plupart des événements avec leur impact peuvent être corrélés avec un petit ensemble de métriques. La répétition du processus pour toutes les mesures et toutes les variables permet à un BN d'utiliser toutes ces relations pour classer une anomalie. Deuxièmement, comme un BN est un modèle graphique, l'ensemble du système de détection des anomalies peut s'appuyer sur une approche hiérarchique avec des métriques locales associées à des détecteurs locaux, et éventuellement un détecteur global jouant le rôle de composant racine pour la sécurité du réseau. Enfin, un BN

peut traiter efficacement la nature aléatoire sous-jacente des mesures observées en utilisant l'approche probabiliste bayésienne.

L'approche par BN a été utilisé dans un large éventail d'applications telles que le diagnostic [134] et la classification. Un BNC (Bayesian Network Classifier) est un BN utilisé pour la classification dans lequel l'un de ses nœuds prend des valeurs dans un ensemble fini $\mathcal{C}$ de toutes les classes possibles [137]. Un BNC renverra la classe $\widehat{c} \in \mathcal{C}$ qui a l'estimation postérieure maximale avec un BN et un ensemble de données observées.

A.6.3 Proposition de BNC pour la détection d'anomalies dans NDN

Cette section présente notre solution de surveillance de la sécurité pour NDN utilisant un BNC, comprenant une liste complète de métriques, une conception de micro-détecteur et un moteur de corrélation basé sur un BN.

Métriques surveillées

Cette section présente une liste de métriques à surveiller dans un nœud NDN. Une telle liste doit pouvoir mettre en évidence le comportement du nœud et distinguer le trafic normal du trafic anormal. Pour construire une liste exhaustive, tous les composants pertinents à l'intérieur d'un nœud NDN sont considérés, à savoir : (1) les *Faces*; (2) le CS; (3) la PIT et (4) la FIB. Cependant, on ne considère délibérément pas la FIB, car elle appartient au plan de contrôle dont les changements ne se produisent qu'en raison de configurations de routage statiques ou d'annonces du protocole de routage. Par conséquent, ses métriques sont moins susceptibles de changer, elles sont donc moins utiles pour caractériser rapidement le comportement d'un nœud. De plus, on affirmons que les dysfonctionnements de la FIB peuvent être capturés par d'autres métriques que nous considérons.

Un routeur NDN reçoit et transfère des paquets (c'est-à-dire, *Interest*, *Data*, *NACK*) par ses *Faces*. Les mesures qui cractérisent ces dernières incluent *In Interest*, *In Data*, *In NACK*, *Out Interest*, *Out Data*, *Out NACK*, qui sont les nombres de paquets entrants et sortants dans la période d'échantillonnage. Puisque le routeur peut supprimer des paquets, on surveille également le nombre de paquets abandonnés (c'est-à-dire, *Drop Interest*, *Drop Data*, *Drop NACK*). En ce qui concerne le CS, pendant son fonctionnement, des cache "miss" et des "hits" se produisent. Selon la politique de remplacement de son contenu, le routeur peut décider d'insérer un nouveau *Data* dans le CS. Par conséquent, pour le CS, on surveille le nombre d'occurrences des événements "miss" (*CS Miss*), "hit" (*CS Hit*) et "insert" (*CS Insert*) pendant un intervalle spécifié. Pour la PIT, les métriques de caractérisation comprennent le nombre d'entrées PIT créées (*PIT Create*), supprimées (*PIT Delete*) pendant un intervalle de

temps et le nombre actuel d'entrées (*PIT Number*). De plus, les entrées créées seront aussi mises à jour pendant le fonctionneement du réseau. Ainsi, on surveille également *PIT Update*, le nombre de mises à jour dans la PIT par intervalle de temps. En outre, un *Interest* a une durée de vie et l'entrée PIT peut expirer après un certain temps. On mesure donc également l'occurrence de l'expiration (*PIT Unsatisfied*) et le temps moyen d'existante des entrées dans la PIT (*PIT Exist Time*).

Micro Détecteur

Dans cette sous-section, un micro-détecteur est présenté pour détecter tout changement significatif d'une métrique de son comportement normal en utilisant la théorie des tests d'hypothèses statistiques avec l'approche à deux critères de Neyman-Pearson présentée dans la Section A.6.3 . Notons x_i une valeur métrique observée à un instant i. Considérant le besoin de construire des micro-détecteurs simples, on a délibérément décidé de modéliser toutes les métriques en utilisant la distribution normale (gaussienne). Il est également supposé qu'une anomalie modifie les valeurs moyennes des métriques beaucoup plus que leur variance. Par conséquent, le problème considéré au niveau du micro-détecteur est finalement défini par les hypothèses suivantes :

$$x_t, \dots x_{t-n+1} \sim \begin{cases} \mathcal{N}(\mu_0; \sigma^2) & \text{under } \mathcal{H}_0, \\ \mathcal{N}(\mu_1; \sigma^2)\,,\ \mu_1 < \mu_0 & \text{under } \mathcal{H}_1, \\ \mathcal{N}(\mu_2; \sigma^2)\,,\ \mu_2 > \mu_0 & \text{under } \mathcal{H}_2, \end{cases} \tag{A.15}$$

où n est la taille de la fenêtre pour la détection. Le problème présenté dans l'Eq. (A.15) peut être résolu facilement en utilisant une simple extension de l'approche de Neyman-Pearson pour plusieurs hypothèses appelées *test contraint minimax*, voir les détails dans [116, 153]. La solution est présentée simplement par le test suivant :

$$\delta\left(x_t, \dots x_{t-n+1}\right) \begin{cases} \mathcal{H}_0 & \text{if } \tau_1 \leq \sum_{t-n+1}^{t} x_t \leq \tau_2. \\ \mathcal{H}_1 & \text{if } \sum_{t-n+1}^{t} x_t < \tau_1, \\ \mathcal{H}_2 & \text{if } \sum_{t-n+1}^{t} x_t > \tau_2, \end{cases} \tag{A.16}$$

Les seuils τ_1 et τ_2 garantissant la PFA prescrite sont établis comme suit :

$$\tau_1 = \Phi^{-1}\left(\alpha_0/2\right)\sqrt{n}\sigma + n\mu_0 \tag{A.17}$$

$$\tau_2 = \Phi^{-1}(1-\alpha_0/2)\sqrt{n}\sigma + n\mu_0 \tag{A.18}$$

où Φ et Φ^{-1} représentent respectivement la fonction de distribution cumulative normale standard et sa fonction inverse, α_0 est le PFA désiré. Les équations montrent que les seuils τ_1 et τ_2 sont des fonctions de $\alpha_0, n, \mu_0, \sigma^2$. Alors que α_0 et n sont choisis en fonction des besoins de l'administrateur du réseau, μ_0 et σ^2 peuvent être estimés à partir du comportement normal de la métrique. Ainsi, le seuil τ peut être calculé

à l'avance et garantit le PFA désiré, quel que soit le comportement de la métrique attaquée. De plus, en utilisant le seuil de décision donné dans les equations (A.17) et (A.18), la puissance de détection des micro-détecteurs est donnée par :

$$\beta_1 = \Phi\left[\Phi^{-1}\left(\frac{\alpha_0}{2}\right)\sqrt{n}\sigma + \sqrt{n}\frac{\mu_0 - \mu_1}{\sigma}\right], \tag{A.19}$$

$$\beta_2 = 1 - \Phi\left[\Phi^{-1}\left(1 - \frac{\alpha_0}{2}\right)\sqrt{n}\sigma + \sqrt{n}\frac{\mu_0 - \mu_2}{\sigma}\right]. \tag{A.20}$$

Les seuls paramètres pouvant être définis par l'administrateur du réseau sont α_0 et n. Alors que α_0 représente la PFA prescrite, le nombre d'échantillons n peut être réglé pour trouver un compromis entre une détection rapide et précise.

Proposition d'un classifieur fondé sur un réseau Bayésien

Différents types d'attaque conduisent à des effets différents sur une métrique. Par conséquent, les alarmes provenant d'un seul micro-détecteur ne peuvent pas détecter et caractériser entièrement une anomalie apparaissant dans un nœud NDN. La combinaison d'alarmes provenant de micro détecteurs est donc essentielle. Pour démontrer les relations causales entre les micro détecteurs, une structure de BN est proposée et représentée dans la Figure A.9, dont le nœud correspond au micro-détecteur d'une métrique. Le noeud *Anomalie* représente les anomalies possibles dans un routeur NDN, et sa valeur est la sortie du classificateur. La structure du BN est construite sur la base des flux de traitement de paquets NFD (c'est-à-dire une série d'étapes effectuées par NFD sur un paquet ou une entrée PIT, déclenchée par un événement spécifique), au lieu d'utiliser un algorithme d'apprentissage de structure de BN. On soutient qu'une telle approche fournit une structure de BN plus fiable et plus robuste puisqu'elle est basée sur notre expertise NDN et sur nos connaissances sur NFD [5]. A l'inverse, l'apprentissage de la structure d'un BN à partir des données nécessite un ensemble de données adéquat couvrant tous les comportements de nœuds possibles dans la vie réelle, ce qui n'est pas réalisable dans un environnement expérimental.

A.6.4 Résultats numériques

Dans cette section, on présente la configuration de nos expériences et les résultats de l'évaluation effectuée pour le BNC proposé.

Topologie expérimentale et configuration

Comme CPA est le cas d'utilisation sélectionné, on réutilise la topologie de la Figure A.4 et expérimentons la configuration pour l'évaluation de BNC. On a reproduit quatre scénarios, à savoir *trafic normal*, *trafic double*, *CPA bestroute* et *CPA multicast*.

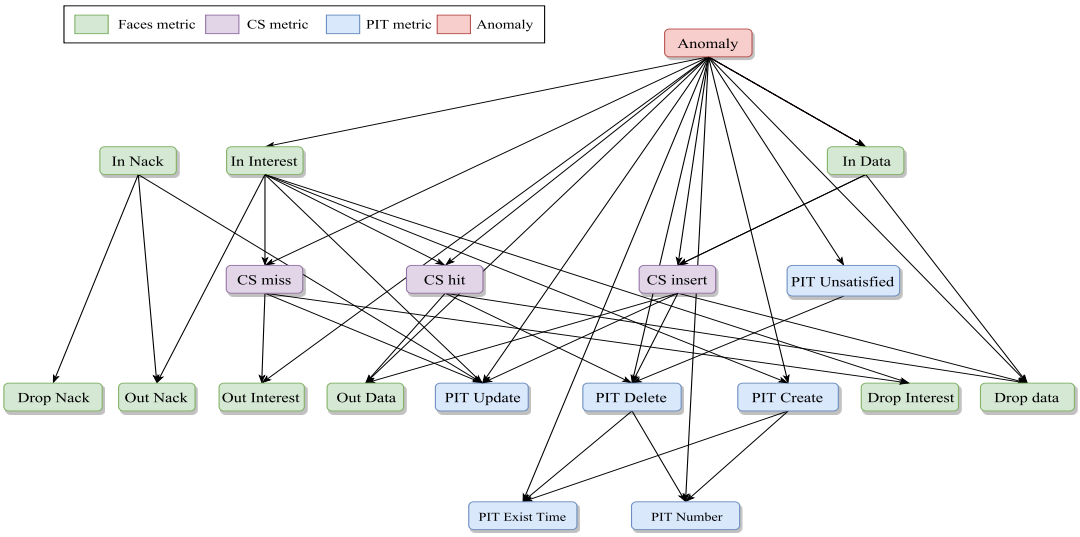


FIGURE A.9: Le réseau Bayésien proposé

Dans le premier, il n'y a qu'un trafic légitime émis par un bon client. Le nombre de *Interests* générés suit une distribution de Poisson, et le contenu demandé est sélectionné en fonction d'une loi de Zipf. Dans le second scénario, deux bons clients sont connectés via R1. Chaque client se comporte comme dans le scénario de *trafic normal*. Nous considérons ce scénario non seulement pour le comportement des métriques d'exploration en cas de changements brusques du trafic réseau mais également pour fournir un jeu de données d'apprentissage qui défie le BNC, en évaluant dans quelle mesure il peut distinguer les changements légitimes du trafic du traffic réellement malveillant. Les deux derniers scénarios, *multicast*, et *bestroute* de CPA sont déjà présentés dans la Section A.4.4.

Pour collecter toutes les métriques proposées, nous avons construit un outil en utilisant *Montimage Monitoring Tool* (MMT)[3] pour les extraire des logs NFD. Bien qu'un protocole de gestion NFD soit disponible, il est insuffisant de collecter toutes les métriques que l'on considère ici. Une sonde MMT est installée dans chaque routeur pour extraire et collecter des données pour nos métriques sélectionnées. L'ensemble de données est ensuite fourni aux micro détecteurs mis en œuvre dans MATLAB. On utilise la boîte à outils MATLAB Bayesian Network [148] pour la mise en œuvre, l'apprentissage des paramètres et l'inférence du BN proposé. La structure du BN est déjà présentée dans la Section A.6.3. Etant donné que l'ensemble de données est complet (c'est-à-dire que tous les nœuds peuvent être observés) et que on n'a aucune distribution de probabilité antérieure des nœuds, nous utilisons l'estimation du maximum de vraisemblance pour l'apprentissage des paramètres. De plus, comme les alarmes du micro-détecteur sont discrètes et que tous les nœuds sont observables, il est raisonnable d'utiliser une inférence exacte. Pour inférer la

[3]Voir http://www.montimage.com/products.html

valeur du noeud *Anomaly*, on a utilisé le moteur de l'arbre de jonction [132].

Pour l'évaluation, nous avons besoin de trois jeux de données, à savoir le réglage du micro-détecteur, l'apprentissage BNC et l'ensemble de données de test. Le premier est collecté pendant une semaine avec seulement un trafic normal pour régler la configuration des micro détecteurs. Le deuxième ensemble de données pour l'apprentissage de BNC est composé de données de chaque scénario avec les paramètres suivants. Le taux moyen pour chaque bon client et l'attaquant (*CPA bestroute* et *CPA multicast*) est le même et est égal à 10 *Interests*/s. L'objectif est d'aider le BN à différencier le trafic légitime malveillant et supplémentaire même si l'utilisateur et l'attaquant ont le même débit. En tant que tel, le BNC est entrainé pour classer l'un des trois résultats, à savoir (1) normal, (2) trafic supplémentaire et (3) CPA. Le troisième est l'ensemble de données de test. On rassemble les métriques des scénarios *CPA bestroute* et *CPA multicast*. Pour chaque scénario, nous exécutons des expériences avec des taux d'attaque de l'ordre de [1..100] *Interests*/s suivant une échelle logarithmique. Chaque réglage est répété cinq fois. Chaque expérience dure dix minutes et a deux périodes. Les cinq premières minutes présentent seulement un trafic légitime, tandis que l'attaque se produit pendant la deuxième période.

Évaluation des micro-détecteurs

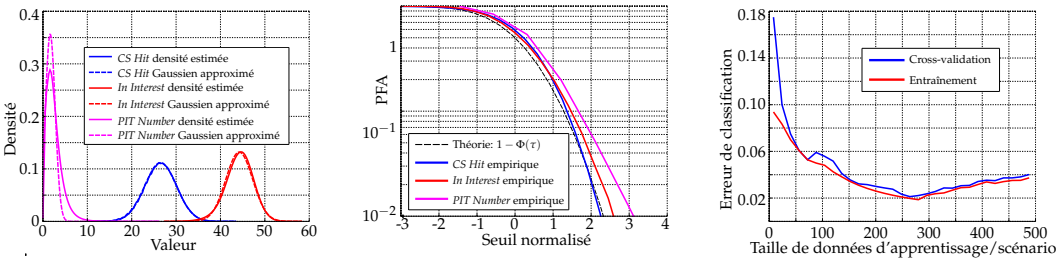
Pertinence du modèle de micro-détecteur : en raison de la diversité des comportements anormaux des métriques, on se concentre sur la modélisation correcte des métriques dans le trafic normal. La Figure A.10a représente la fonction de densité estimée du noyau pour les métriques illustratives et leurs distributions normales approchées. La figure montre que pour la plupart des métriques (par exemple, *In Interest* et *CS Hit*), la distribution empirique est proche de la distribution normale, indiquant la pertinence du modèle. Néanmoins, le modèle ne correspond pas bien à certaines métriques (par exemple, *PIT Number*), car leur plage de valeurs est proche de zéro et la variance est étroite. Cependant, pour conserver la simplicité et la réutilisabilité de notre approche par micro-détecteur, nous acceptons délibérément ce manque de précision pour une partie mineure des métriques et avons l'intention de le compenser en corrélant les alarmes d'autres micro-détecteurs.

Garantie du PFA pour les micro-détecteurs : La Figure A.10b illustre les PFA théoriques et empiriques de nos micro détecteurs pour différentes métriques en fonction du seuil normalisé. Pour la plupart des métriques (par exemple, *CS Hit*, *In Interest*), les PFA empirique et théorique correspondent étroitement, ce qui démontre la capacité de garantir un PFA prescrit et la pertinence du modèle suivi. Par contre, pour quelques métriques (par exemple, *PIT Number*), notre micro-détecteur ne peut pas garantir la performance d'un petit PFA prescrit parce que ces mesures

ne sont pas bien modélisées par la distribution normale. Cependant, comme nous le montrons dans la section suivante, la performance peut être améliorée en combinant les micro-détecteurs.

Paramètres d'apprentissage pour le réseau Bayésien proposé

Pour évaluer l'efficacité de l'apprentissage du BNC, on utilise la méthode de validation croisée habituelle des *k-folds* avec $k = 5$. Le taux moyen de classification erronée sur les sous-ensembles d'apprentissage est défini par l'*erreur d'apprentissage*, tandis que celui obtenu sur des sous-ensembles de test est appelé *erreur de validation croisée*. Figure A.10c montre les courbes d'apprentissage de la BNC proposée lorsque la taille de l'ensemble de données d'apprentissage par scénario change. La figure montre que lorsque la taille de l'ensemble de formation par scénario augmente, l'erreur de classification commence à diminuer. Une valeur optimale est d'environ 280 échantillons d'entraînement par scénario (environ 23 minutes de collecte d'échantillons). Après cela, l'erreur augmente en raison du sur-ajustement.



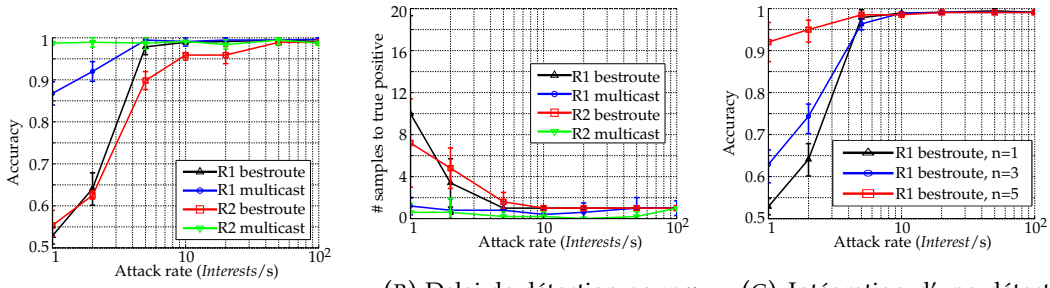
(A) Distributions des métriques dans un scénario de trafic normal (B) Garantie de PFA pour les micro-détecteurs (C) Courbe d'apprentissage du BNC

FIGURE A.10: Pertinence du classificateur bayésien proposé

Évaluation bayésienne des classificateurs de réseaux

Les figures A.11a et A.11b montrent respectivement la précision (i.e. le nombre total d'échantillons correctement classés sur le nombre total d'échantillons) et le délai de détection du BNC proposé quand le taux d'attaque change. Nous discutons ici ces deux indicateurs en fonction de l'impact du taux d'attaque, du scénario d'attaque et de l'emplacement du classificateur. En ce qui concerne le taux d'attaque, les chiffres indiquent que lorsque le taux d'attaque est inférieur à celui du bon client, le BNC a une faible précision, un retard élevé et ces résultats ont une variance élevée. C'est parce que l'attaque est inefficace et que sa trace se distingue à peine du trafic normal. A l'inverse, le BNC atteint plus de 95% de précision avec un retard moyen d'un échantillon lorsque le taux d'attaque est supérieur au taux du bon client.

Pour le scénario d'attaque, il est montré que, avec un taux d'attaque inférieur à 10 *Interests*/s du bon client, la précision et le délai de détection du BNC contre



(A) Exactitude, $n = 1$

(B) Delai de détection pour $n = 1$

(C) Intégration d'une détection à fenêtre

FIGURE A.11: Performance du BNC proposé

CPA multicast est bien meilleur que celui de *CPA bestroute*. C'est parce que, dans *CPA multicast*, les *Interests* malveillants sont transmis plus facilement au mauvais fournisseur. Par conséquent, même avec un faible taux d'attaque, le bon client est obligé de relancer *Interest* pour récupérer le bon *Data*. Ainsi, l'attaque laisse une empreinte plus forte. D'un autre côté, pour des taux d'attaque plus élevés, la performance de la BNC est presque la même pour les deux scénarios.

En ce qui concerne l'emplacement, dans *CPA bestroute*, le BNC au sein de R1 est plus précis qu'au sein de R2 parce que R1 est plus proche des clients et des attaquants. En conséquence, ses métriques sont plus affectées que celles de R2. Pendant ce temps, dans *CPA multicast*, étant donné que R2 est plus susceptible d'être empoisonné en raison du transfert de multidiffusion, ses métriques sont plus affectées que celles de R1. Par conséquent, le BNC dans R2 atteint une meilleure précision que dans R1.

La précision du BNC peut être améliorée en augmentant la fenêtre de détection n du micro-détecteur. Figure A.11c trace la précision à mesure que le taux d'attaque change, avec des valeurs différentes de n. Pour le pire cas de *CPA bestroute* avec 1 Interest / s, la précision passe de 53% à 93% en augmentant le n de 1 à 5. Par contre, l'augmentation de la fenêtre de détection est inutile pour un taux d'attaque élevé car elle n'améliore pas la précision. De plus, augmenter la fenêtre de détection augmentera le délai de détection des micro-détecteurs (Eq. (A.19) et Eq. (A.20)). Par conséquent, ce compromis doit être examiné attentivement dans une situation de déploiement.

A.7 Conclusions et perspectives

Depuis qu'il a été conçu pour connecter des ordinateurs distants il y a plusieurs décennies, l'Internet a considérablement évolué. Par conséquent, le modèle de communication centré sur les hôtes ne correspond plus à l'utilisation principale de l'Internet actuel, qui consiste à accéder à du contenu, et révèle progressivement ses limites. Dans un tel contexte, NDN est apparu comme la proposition la plus prometteuse

parmi les architectures ICN, une approche propre à l'Internet du Futur, en proposatn un modèle de communication centré sur le contenu. En intégrant des mécanismes de sécurité dès la conception, NDN atténue intrinsèquement les attaques héritées du réseau IP. Cependant, les nouvelles fonctionnalités et composants de NDN exposent également le réseau à de nouvelles failles. Dans cette thèse, nous avons proposé un plan de surveillance de la sécurité, capable de détecter des événements anormaux dans un nœud NDN.

A.7.1 Conclusions

Nous avons mis en évidence que les différences entre les ICN le réseau IP actuel concernent quatre caractéristiques clés, à savoir : le schéma de nommage, la résolution de noms, le routage des données, la mise en cache et la sécurité de l'information. Nous avons présenté le protocole NDN en détails, ainsi que d'autres propositions pour mettre en évidence différentes conceptions du concept ICN. Par la suite, nous avons comparé les architectures ICN présentées selon ces quatre caractéristiques clés. Pour chaque fonctionnalité, nous avons relevé les problématiques actuelles qui nécessitent davantage d'investigation, et en particulier les problèmatiques de sécurité.

Comme étude plus approfondie de la sécurité ICN, nous proposons une taxonomie pour classer et organiser les attaques sur la base des travaux existants. L'idée fondamentale de chaque attaque est expliquée, suivie d'une brève introduction de quelques mises œuvres existantes. En évaluant les attaques en fonction de critères sélectionnés, nous avons choisi IFA et CPA comme cas d'études pour l'application de notre proposition de plan de surveillance de la sécurité pour NDN. Bien que les deux attaques aient attiré l'attention de la communauté NDN, les travaux antérieurs portant sur leur détection présentent plusieurs inconvénients. Premièrement, une majorité des travaux antérieurs sont évalués avec des résultats issus d'un environnement simulé, laissant la performance dans des conditions réelles inconnues. Deuxièmement, les propositions de détection existantes n'indiquent pas comment elles choisissent le seuil de détection, ce qui conduit à une détection rigide et non fiable qui gaspille les ressources du réseau ou pénalise le client légitime. Troisièmement, les travaux existants se concentrent généralement sur une attaque NDN particulière. La littérature manque d'une solution de sécurité générale capable de répondre à la fois aux menaces de sécurité révélées et aux menaces potentielles qui n'ont pas encore été révélées à ce jour.

Ainsi, nos contributions sont triples et chacune répond aux inconvénients identifiés dans les travaux antérieurs. Considérant la maturité du paradigme NDN, nous avons démontré que l'étude des attaques NDN dans un contexte de déploiement

réel est indispensable. Comme première contribution, nous reconsidérons ainsi la faisabilité des attaques IFA et CPA et caractérisons leur impact dans un environnement expérimental NDN. L'attaque IFA est étudiée dans un environnement où IP et NDN coexistent dans des domaines isolés interconnectés par des passerelles dédiées pour fournir un service HTTP. Malgré le manque de contrôle de l'attaquant sur l'existence du paquet *Interests* et du paquet *NACK*, on propose un scénario IFA qui réussit à dégrader l'expérience des utilisateurs lorsqu'ils accèdent à des sites Web. Pour l'attaque CPA, nous avons identifié trois scénarios d'attaque qui profitent des faiblesses du protocole NDN et de sa mise en œuvre. Les résultats campagne de tests NDN présentent des effets différents sur les entités réseau essentielles. Alors que l'attaque IFA est entièrement caractérisée par le taux de satisfaction, on constate que les effets de l'attaque CPA sont plus insaisissables et, par conséquent, doivent être caractérisés par différentes métriques sur différentes composants d'un noeud NDN.

Pour traiter de la deuxième limite identifiée dans l'état de l'art, nous avons conçu un micro-détecteur en reposant sur des tests d'hypothèses statistiques. Une telle méthodologie permet d'établir un seuil garantissant une PFA souhaitée et d'établir une performance théorique attendue. Nous utilisons l'attaque IFA comme cas d'utilisation pour évaluer notre micro-détecteur car elle peut être entièrement caractérisée par une seule mesure de taux de satisfaction des *Interests* émis. Pour faire face à l'attaque, nous avons conçu un détecteur GLRT et évalué sa performance intrinsèque dans un environnement de simulation. Les résultats démontrent la pertinence du modèle proposé avec la correspondance étroite entre les résultats empiriques et théoriques, ainsi que la capacité de garantir une PFA prescrite et établir un compromis entre la puissance de détection et le retard. De plus, pour améliorer encore la précision de détection, nous avons développé une version séquentielle basée sur le GLRT initial. Les données collectées à partir des scénarios d'attaque dans notre environnement expérimental NDN-IP démontrent un gain significatif obtenu par le détecteur séquentiel, en ce qui concerne le délai de détection moyen et la probabilité de vrais positifs sous une contrainte de retard de détection maximum.

Dernier point mais non le moindre, nous avons abordé le troisième inconvénient avec une proposition de plan de surveillance pour la sécurité NDN, un élément essentiel pour le déploiement sécurisé de tout protocole réseau. À cette fin, nous avons inspecté les flux de traitements de paquets dans NFD et identifié 18 mesures pour caractériser le comportement d'un nœud NDN. Associé à chaque métrique est un micro-détecteur à usage général conçu sur la base de la théorie des tests d'hypothèses. Le micro-détecteur est responsable de détecter tout changement anormal de la métrique par rapport à son comportement normal. Le micro-détecteur a

été évalué à la fois théoriquement et empiriquement. À partir de l'analyse des les flux de traitements de paquets dans NFD, nous avons identifié la relation causale entre les métriques et les avons intégrées à un réseau Bayésien qui corrèle les alarmes des micro-détecteurs pour capturer tout événement de sécurité inhabituel dans un nœud NDN. Pour valider notre proposition, deux scénarios d'attaques CPA ont été déployés dans notre environnement expérimental, et les données collectées démontrent la capacité de notre solution à détecter avec précision ces attaques à différents emplacements du réseau et à différents débits.

A.7.2 Perspectives

Le travail réalisé au cours de cette thèse ouvre plusieurs perspectives pour la gestion de la sécurité NDN. Nous sommes intéressés par les scénarios d'attaques dans des environnements réels de déploiement de NDN couplé à IP. NDN est un paradigme perturbateur qui propose de remplacer le modèle conventionnel centré sur l'hôte par un nouveau modèle centré sur le contenu. Pour cette raison, déployer NDN à une grande échelle est difficilement réalisable dans un futur proche. Il est cependant possible de déployer des NDN sur des îlots isolés interconnectés au réseau IP existant avec des passerelles NDN/IP dédiées. Ce scénario est plus crédible car il fournit une transition en douceur du réseau IP actuel au NDN sans perturber les utilisateurs actuels. Plus de complications sont ajoutées, forçant ainsi les attaquants à adapter leurs scénarios d'attaque. Par conséquent, la faisabilité des attaques NDN doit être révisée en tenant compte des conditions réelles de déploiement, tout comme ce que nous avons fait avec IFA, mais pour d'autres types d'attaques.

À mesure que des solutions de sécurité sont proposées, les scénarios d'attaque évoluent également, en essayant de nier les solutions de sécurité existantes. Nous pensons qu'il n'y a pas d'exception pour notre proposition. Afin de préparer les améliorations d'attaque dans lesquelles le schéma de détection proposée pourrait être trompée avec un trafic malveillant à faible débit, l'amélioration de notre solution de corrélation d'alarmes par un réseau Bayésien en considérant un système de détection collaboratif est une perspective intéressante. Dans notre solution, le réseau Bayésien est mis à profit pour corréler les alarmes provenant de micro détecteurs d'un seul nœud. Le moteur peut également être étendu aux alertes corrélés provenant d'autres nœuds. Une fois qu'un nœud détecte une aberration, il pourrait partager ses alarmes avec ses voisins ou les envoyer à l'adminsitrateur du réseau. La collecte et la corrélation d'alarmes provenant de plusieurs sources pourraient améliorer considérablement la précision de la détection et permettre une contremesure permettant d'identifier alors la source de l'attaque. Nous proposons enfin d'aborder d'autres attaques NDN avec notre proposition de plan de surveillance

pour évaluer sa performance générale. La méthodologie dans ce cadre serait similaire à ce qui a été fait dans cette thèse, à savoir, étudier des scénarios d'attaque et réproduire l'attaque dans le vrai banc d'essai, collecter des données pour l'estimation des paramètres du BN et l'évaluer avec différentes puissances d'attaque.

Publications

International Journals

- **Tan Nguyen**, Hoang-Long Mai, Guillaume Doyen, Rémi Cogranne, Wissam Mallouli, Edgardo Montes-de-Oca, Olivier Festor. "*A Security Monitoring Plane for Named Data Networking Deployment.*" in IEEE Communications Magazine, Feature topic "Information-Centric Networking Security." (accepted with minor changes)

- **Tan Nguyen**, Hoang-Long Mai, Luong Nguyen, Rémi Cogranne, Guillaume Doyen, Moustapha El Aoun, Wissam Mallouli, Edgardo Montes de Oca and Olivier Festor. "*Reliable Detection of Interest Flooding Attack in Real Deployment of Named Data Networking*." in IEEE Transactions on Information Forensics and Security. (paper submitted)

International Conferences

[99] **Tan Nguyen**, Rémi Cogranne, and Guillaume Doyen. "*An optimal statistical test for robust detection against interest flooding attacks in CCN*." Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on. IEEE, 2015. [doi]

[100] **Tan N. Nguyen**, Rémi Cogranne, Guillaume Doyen, and Florent Retraint. "*Detection of interest flooding attacks in named data networking using hypothesis testing*." In Information Forensics and Security (WIFS), 2015 IEEE International Workshop on, pp. 1-6. IEEE, 2015. [doi]

[93] Hoang Long Mai, **Ngoc Tan Nguyen**, Guillaume Doyen, Alain Ploix, and Rémi Cogranne. "*On the readiness of NDN for a secure deployment: the case of pending interest table*." In IFIP International Conference on Autonomous Infrastructure, Management and Security, pp. 98-110. Springer, Cham, 2016. [doi]

[96] **Tan Nguyen**, Xavier Marchal, Guillaume Doyen, Thibault Cholez, and Rémi Cogranne. "*Content Poisoning in Named Data Networking: Comprehensive Characterization of real Deployment*." In Integrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on, pp. 72-80. IEEE, 2017. [doi]

[123] Hoang Long Mai, **Tan Nguyen**, Guillaume Doyen, Rémi Cogranne, Wissam Mallouli, Edgardo Montes de Oca, and Olivier Festor. "*Towards a Security Monitoring Plane for Named Data Networking and its Application against Content Poisoning Attack.*" In IEEE/IFIP Network Operations and Management Symposium (NOMS), 2018.

Presentations in National Conferences

- Ph.D Student session in Rendez-Vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information (RESSI), 2016 in Toulouse, France

Technical Reports

This thesis is within DOCTOR project, <ANR-14- CE28-0001>

- Deliverable D1.1 "Virtualization Techniques: Analysis and Selection." [report]
- Deliverable D2.1 "Security analysis of the virtualized NDN architecture." [report]
- Deliverable D2.2 "Security monitoring of NDN through virtualized components." [report]

Bibliography

[1] J. Tang, Z. Zhang, Y. Liu, and H. Zhang, "Identifying Interest Flooding in Named Data Networking," in *IEEE International Conference on Green Computing and Communications (GreenCom) and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*. IEEE, 2013, pp. 306–310.

[2] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 2, pp. 1024–1049, 2014.

[3] A. Hoque, S. O. Amin, A. Alyyan, B. Zhang, L. Zhang, and L. Wang, "Nlsr: Named-data link state routing protocol," in *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-centric Networking*. ACM, 2013, pp. 15–20.

[4] T. Lauinger, "Security & scalability of content-centric networking," Ph.D. dissertation, TU Darmstadt, 2010.

[5] A. Afanasyev, J. Shi, B. Zhang, L. Zhang, I. Moiseenko, Y. Yu, W. Shang, Y. Huang, J. P. Abraham, S. DiBenedetto, and others, "NFD developer's guide - revision 7," Technical Report NDN-0021, 2016. [Online]. Available: http://named-data.net/wp-content/uploads/2016/10/ndn-0021-7-nfd-developer-guide.pdf

[6] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox, "Information-centric Networking: Seeing the Forest for the Trees," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, ser. HotNets-X. New York, NY, USA: ACM, 2011, pp. 1:1–1:6. [Online]. Available: http://doi.acm.org/10.1145/2070562.2070563

[7] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker, "Naming in Content-oriented Architectures," in *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ser. ICN '11. New York, NY, USA: ACM, 2011, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/2018584.2018586

[8] L. Masinter, T. Berners-Lee, and R. T. Fielding, "Uniform Resource Identifier (URI): Generic Syntax," 2005. [Online]. Available: https://tools.ietf.org/html/rfc3986?ref=binfind.com/web

[9] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *Communications Magazine, IEEE*, vol. 50, no. 7, pp. 26–36, 2012.

[10] I. Abdullahi, S. Arif, and S. Hassan, "Survey on caching approaches in Information Centric Networking," *Journal of Network and Computer Applications*, vol. 56, pp. 48–59, Oct. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804515001381

[11] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang, and others, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[12] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 2009, pp. 1–12.

[13] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *ACM SIGCOMM Computer Communication Review*, vol. 37. ACM, 2007, pp. 181–192.

[14] S. Tarkoma, M. Ain, and K. Visala, "The Publish/Subscribe Internet Routing Paradigm (PSIRP): Designing the Future Internet Architecture." in *Future Internet Assembly*, 2009, pp. 102–111.

[15] N. Fotiou, P. Nikander, D. Trossen, G. C. Polyzos, and others, "Developing Information Networking Further: From PSIRP to PURSUIT." in *Broadnets*. Springer, 2010, pp. 1–13.

[16] R. Moskowitz, P. Nikander, E. P. Jokela, and T. Henderson, "Host Identity Protocol," 2008. [Online]. Available: http://www.rfc-editor.org/info/rfc5201

[17] D. Farinacci, C. Liu, S. Deering, D. Estrin, M. Handley, V. Jacobson, L. Wei, P. Sharma, D. Thaler, and A. Helmy, "Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification," 1998. [Online]. Available: http://buildbot.tools.ietf.org/html/rfc2362

[18] Z. Gu, Y. Wang, Q.-S. Hua, and F. C. M. Lau, *Rendezvous in Distributed Systems: Theory, Algorithms and Applications*. Springer, Aug. 2017.

[19] W. Shang, Z. Wang, A. Afanasyev, J. Burke, and L. Zhang, "Breaking Out of the Cloud: Local Trust Management and Rendezvous in Named Data Networking of Things," in *2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI)*, Apr. 2017, pp. 3–14.

[20] B. H. Bloom, "Space Time Trade-offs in Hash Coding with Allowable Errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970. [Online]. Available: http://doi.acm.org/10.1145/362686.362692

[21] D. Lagutin, "Redesigning internet-the packet level authentication architecture," *Licentiate's Thesis-Helsinki University of Technology*, 2008.

[22] A. Broder, M. Mitzenmacher, and A. B. I. M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," in *Internet Mathematics*, 2002, pp. 636–646.

[23] C. Dannewitz, D. Kutscher, B. Ohlman, S. eell, B. Ahlgren, and H. Karl, "Network of Information (NetInf)–An information-centric networking architecture," *Computer Communications*, vol. 36, no. 7, pp. 721–735, 2013.

[24] L. M. Correia, H. Abramowicz, M. Johnsson, and K. Wünstel, *Architecture and Design for the Future Internet: 4WARD Project*. Springer Science & Business Media, Jan. 2011.

[25] S. Farrell, C. Dannewitz, P. Hallam-Baker, D. Kutscher, and B. Ohlman, "Naming things with hashes," 2013.

[26] M. D'Ambrosio, C. Dannewitz, H. Karl, and V. Vercellone, "MDHT A Hierarchical Name Resolution Service for Information-centric Networks," in *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ser. ICN '11. New York, NY, USA: ACM, 2011, pp. 7–12. [Online]. Available: http://doi.acm.org/10.1145/2018584.2018587

[27] C. Dannewitz, M. D'Ambrosio, and V. Vercellone, "Hierarchical DHT-based name resolution for information-centric networks," *Computer Communications*, vol. 36, no. 7, pp. 736–749, Apr. 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366413000418

[28] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less pain, most of the gain: Incrementally deployable ICN," in *ACM SIGCOMM Computer Communication Review*, vol. 43. ACM, 2013, pp. 147–158.

[29] A. Araldo, D. Rossi, and F. Martignon, "Cost-Aware Caching: Caching More (Costly Items) for Less (ISPs Operational Expenditures)," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 5, pp. 1316–1330, May 2016.

[30] T. Lauinger, N. Laoutaris, P. Rodriguez, T. Strufe, E. Biersack, and E. Kirda, "Privacy risks in named data networking: what is the cost of performance?" *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 5, pp. 54–57, 2012.

[31] V. Lehman, A. Gawande, B. Zhang, L. Zhang, R. Aldecoa, D. Krioukov, and L. Wang, "An experimental investigation of hyperbolic routing with a smart forwarding plane in NDN," in *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, Jun. 2016, pp. 1–10.

[32] H. Yuan, P. Crowley, and T. Song, "Enhancing Scalable Name-Based Forwarding," in *2017 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, May 2017, pp. 60–69.

[33] S. Shannigrahi, C. Fan, and C. Papadopoulos, "Request Aggregation, Caching, and Forwarding Strategies for Improving Large Climate Data Distribution with NDN: A Case Study," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ser. ICN '17. New York, NY, USA: ACM, 2017, pp. 54–65. [Online]. Available: http://doi.acm.org/10.1145/3125719.3125722

[34] C. Fang, F. R. Yu, T. Huang, J. Liu, and Y. Liu, "A Survey of Green Information-Centric Networking: Research Issues and Challenges," *IEEE Communications Surveys Tutorials*, vol. 17, no. 3, pp. 1455–1472, 2015.

[35] A. Afanasyev, X. Jiang, Y. Yu, J. Tan, Y. Xia, A. Mankin, and L. Zhang, "NDNS A DNS-Like Name Service for NDN," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, Jul. 2017, pp. 1–9.

[36] W. Shang, Q. Ding, A. Marianantoni, J. Burke, and L. Zhang, "Securing building management systems using named data networking," *IEEE Network*, vol. 28, no. 3, pp. 50–56, May 2014.

[37] W. Shang, A. Bannis, T. Liang, Z. Wang, Y. Yu, A. Afanasyev, J. Thompson, J. Burke, B. Zhang, and L. Zhang, "Named Data Networking of Things (Invited Paper)," in *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*, Apr. 2016, pp. 117–128.

[38] M. Amadeo, C. Campolo, J. Quevedo, D. Corujo, A. Molinaro, A. Iera, R. L. Aguiar, and A. V. Vasilakos, "Information-centric networking for the internet of things: challenges and opportunities," *IEEE Network*, vol. 30, no. 2, pp. 92–100, Mar. 2016.

[39] A. Rahman, D. Trossen, D. Kutscher, and R. Ravindran, "Deployment Considerations for Information-Centric Networking (ICN)," Internet Engineering Task Force, Internet-Draft draft-irtf-icnrg-deployment-guidelines-00, Feb. 2018, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-irtf-icnrg-deployment-guidelines-00

[40] N. Alliance, "5g white paper," *Next generation mobile networks, white paper*, 2015.

[41] P. Gusev and J. Burke, "Ndn-rtc: Real-time videoconferencing over named data networking," in *Proceedings of the 2nd ACM Conference on Information-Centric Networking*. ACM, 2015, pp. 117–126.

[42] H. Asaeda, R. Li, and N. Choi, "Container-based unified testbed for information-centric networking," *IEEE Network*, vol. 28, no. 6, pp. 60–66, 2014.

[43] D. Trossen and G. Parisis, "Designing and realizing an information-centric internet," *IEEE Communications Magazine*, vol. 50, no. 7, 2012.

[44] G. White and G. Rutz, "Content delivery with content-centric networking," 2016.

[45] E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, and M. Wählisch, "Information centric networking in the iot: experiments with ndn in the wild," in *Proceedings of the 1st ACM Conference on Information-Centric Networking*. ACM, 2014, pp. 77–86.

[46] M. Sardara, L. Muscariello, J. Augé, M. Enguehard, A. Compagno, and G. Carofiglio, "Virtualized ICN (vICN): Towards a Unified Network Virtualization Framework for ICN Experimentation," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ser. ICN '17. New York, NY, USA: ACM, 2017, pp. 109–115. [Online]. Available: http://doi.acm.org/10.1145/3125719.3125726

[47] M. Aamir and S. M. A. Zaidi, "Denial-of-service in content centric (named data) networking: a tutorial and state-of-the-art survey," *Security and Communication Networks*, vol. 8, no. 11, pp. 2037–2059, Jul. 2015. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/sec.1149/abstract

[48] E. G. AbdAllah, H. S. Hassanein, and M. Zulkernine, "A survey of security attacks in information-centric networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1441–1454, 2015.

[49] R. Tourani, S. Misra, T. Mick, and G. Panwar, "Security, privacy, and access control in information-centric networking: A survey," *IEEE Communications Surveys & Tutorials*, 2017.

[50] V. Lehman, A. M. Hoque, Y. Yu, L. Wang, B. Zhang, and L. Zhang, "A secure link state routing protocol for NDN," *Technical Report NDN-0037*, 2016.

[51] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, "DoS and DDoS in Named Data Networking," in *Computer Communications and Networks (ICCCN), 2013 22nd International Conference on*. IEEE, 2013, pp. 1–7.

[52] S. Choi, K. Kim, S. Kim, and B.-h. Roh, "Threat of DoS by interest flooding attack in content-centric networking," in *Information Networking (ICOIN), 2013 International Conference on*. IEEE, 2013, pp. 315–319.

[53] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang, "Interest flooding attack and countermeasures in Named Data Networking," in *IFIP Networking Conference, 2013*. IEEE, 2013, pp. 1–9.

[54] A. Compagno, M. Conti, P. Gasti, and G. Tsudik, "Poseidon: Mitigating interest flooding DDoS attacks in named data networking," in *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*. IEEE, 2013, pp. 630–638.

[55] Y. Xin, Y. Li, W. Wang, W. Li, and X. Chen, "Detection of collusive interest flooding attacks in named data networking using wavelet analysis," in *Military Communications Conference (MILCOM), MILCOM 2017-2017 IEEE*. IEEE, 2017, pp. 557–562.

[56] H. Dai, Y. Wang, J. Fan, and B. Liu, "Mitigate ddos attacks in ndn by interest traceback," in *Proc. of IEEE INFOCOM NOMEN Workshop, IEEE Press, Piscataway, NJ, USA*, 2013.

[57] Z. Li and J. Bi, "Interest Cash An Application-based Countermeasure Against Interest Flooding for Dynamic Content in Named Data Networking," in *Proceedings of The Ninth International Conference on Future Internet Technologies*, ser. CFI '14. New York, NY, USA: ACM, 2014, pp. 2:1–2:6. [Online]. Available: http://doi.acm.org/10.1145/2619287.2619298

[58] R. You, R. Luo, and X. Lai, "Detecting and mitigating Interest Flooding Attack in Content Centric Networking," *Advances in Computer Science and Technology*, vol. 65, p. 251, 2014.

[59] K. Ding, Y. Liu, H.-H. Cho, H.-C. Chao, and T. K. Shih, "Cooperative detection and protection for interest flooding attacks in named data networking," *International Journal of Communication Systems*, vol. 29, no. 13, pp. 1968–1980, 2016.

[60] K. Wang, H. Zhou, Y. Qin, and H. Zhang, "Cooperative-Filter: countering Interest flooding attacks in named data networking," *Soft Computing*, vol. 18, no. 9, pp. 1803–1813, 2014.

[61] R. Shinohara, T. Kamimoto, K. Sato, and H. Shigeno, "Cache control method mitigating packet concentration of router caused by interest flooding attack," in *Trustcom/BigDataSE/ISPA, 2016 IEEE*. IEEE, 2016, pp. 324–331.

[62] A. Juels and J. G. Brainard, "Client Puzzles A Cryptographic Countermeasure Against Connection Depletion Attacks." in *NDSS*, vol. 99, 1999, pp. 151–165.

[63] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 2001.

[64] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," *ACM SIGCOMM computer communication review*, vol. 42, no. 3, pp. 62–67, 2012.

[65] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, "A case for stateful forwarding plane," *Computer Communications*, vol. 36, no. 7, pp. 779–791, 2013.

[66] C. Ghali, G. Tsudik, and E. Uzun, "Network-Layer Trust in Named-Data Networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 12–19, 2014.

[67] ——, "Needle in a Haystack: Mitigating Content Poisoning in Named-Data Networking," in *Proceedings of NDSS Workshop on Security of Emerging Networking Technologies (SENT)*, 2014.

[68] G. Bianchi, A. Detti, A. Caponi, and N. Blefari Melazzi, "Check before storing: What is the performance price of content integrity verification in LRU caching?" *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 3, pp. 59–67, 2013.

[69] D. Kim, S. Nam, J. Bi, and I. Yeom, "Efficient Content Verification in Named Data Networking," in *Proceedings of the 2nd International Conference on Information-Centric Networking*. ACM, 2015, pp. 109–116.

[70] I. Ribeiro, A. Rocha, C. Albuquerque, and F. Guimaraes, "On the possibility of mitigating content pollution in Content-Centric Networking," in *Local Computer Networks (LCN), 2014 IEEE 39th Conference on*. IEEE, 2014, pp. 498–501.

[71] S. DiBenedetto and C. Papadopoulos, "Mitigating Poisoned Content with Forwarding Strategy," in *The third Workshop on Name-Oriented Mobility (NOM)*. San Francisco, USA: IEEE, 2016.

[72] Z. Rezaeifar, J. Wang, and H. Oh, "A trust based method for mitigating cache poisoning in Name Data Networking," *Journal of Network and Computer Applications*, vol. 104, pp. 117–132, Feb. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804517304046

[73] L. Deng, Y. Gao, Y. Chen, and A. Kuzmanovic, "Pollution attacks and defenses for Internet caching systems," *Computer Networks*, vol. 52, no. 5, pp. 935–956, 2008.

[74] A. Karami and M. Guerrero-Zapata, "An anfis-based cache replacement method for mitigating cache pollution attacks in named data networking," *Computer Networks*, vol. 80, pp. 51–65, 2015.

[75] M. Xie, I. Widjaja, and H. Wang, "Enhancing cache robustness for content-centric networking," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 2426–2434.

[76] H. Guo, X. Wang, K. Chang, and Y. Tian, "Exploiting Path Diversity for Thwarting Pollution Attacks in Named Data Networking," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 9, pp. 2077–2090, Sep. 2016.

[77] M. Conti, P. Gasti, and M. Teoli, "A lightweight mechanism for detection of cache pollution attacks in Named Data Networking," *Computer Networks*, vol. 57, no. 16, pp. 3178–3191, 2013.

[78] D. E. Knuth, "The art of computer programming, 2: seminumerical algorithms, Addision Wesley," *Reading, MA*, 1998.

[79] D. Rossi and G. Rossini, "Caching performance of content centric networks under multi-path routing (and more)," *Relatório técnico, Telecom ParisTech*, 2011.

[80] A. Compagno, M. Conti, P. Gasti, L. V. Mancini, and G. Tsudik, "Violating Consumer Anonymity: Geo-Locating Nodes in Named Data Networking," in *Applied Cryptography and Network Security*, ser. Lecture Notes in Computer Science. Springer, Cham, Jun. 2015, pp. 243–262. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-28166-7_12

[81] G. Acs, M. Conti, P. Gasti, C. Ghali, and G. Tsudik, "Cache Privacy in Named-Data Networking," in *2013 IEEE 33rd International Conference on Distributed Computing Systems*, Jul. 2013, pp. 41–51.

[82] A. Mohaisen, X. Zhang, M. Schuchard, H. Xie, and Y. Kim, "Protecting access privacy of cached contents in information centric networks," in *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*. ACM, 2013, pp. 173–178.

[83] A. Mohaisen, H. Mekky, X. Zhang, H. Xie, and Y. Kim, "Timing Attacks on Access Privacy in Information Centric Networks and Countermeasures," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 6, pp. 675–687, Nov. 2015.

[84] C. Ghali, G. Tsudik, and C. A. Wood, "When Encryption is Not Enough: Privacy Attacks in Content-centric Networking," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ser. ICN '17. New York, NY, USA: ACM, 2017, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/3125719.3125723

[85] A. Chaabane, E. De Cristofaro, M. A. Kaafar, and E. Uzun, "Privacy in content-oriented networking: Threats and countermeasures," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 3, pp. 25–33, 2013.

[86] S. Arianfar, T. Koponen, B. Raghavan, and S. Shenker, "On Preserving Privacy in Content-oriented Networks," in *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ser. ICN '11. New York, NY, USA: ACM, 2011, pp. 19–24. [Online]. Available: http://doi.acm.org/10.1145/2018584.2018589

[87] S. DiBenedetto, P. Gasti, G. Tsudik, and E. Uzun, "ANDaNA: Anonymous named data networking application," *arXiv preprint arXiv:1112.2205*, 2011.

[88] N. Fotiou and G. C. Polyzos, "ICN privacy and name based security," in *Proceedings of the 1st international conference on Information-centric networking*. ACM, 2014, pp. 5–6.

[89] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," DTIC Document, Tech. Rep., 2004.

[90] T. Lauinger, N. Laoutaris, P. Rodriguez, T. Strufe, E. Biersack, and E. Kirda, "Privacy implications of ubiquitous caching in named data networking architectures," *Technical Report TR-iSecLab-0812-001, ISecLab, Tech. Rep.*, 2012.

[91] M. S. Merkow and J. Breithaupt, *Information security: Principles and practices*. Pearson Education, 2014.

[92] X. Marchal, M. El Aoun, B. Mathieu, W. Mallouli, T. Cholez, G. Doyen, P. Truong, A. Ploix, and E. M. De Oca, "A virtualized and monitored NDN infrastructure featuring a NDN/HTTP gateway," in *3rd ACM Conference on Information-Centric Networking (ACM-ICN'16)*. ACM, 2016, pp. 225–226.

[93] H. L. Mai, N. T. Nguyen, G. Doyen, A. Ploix, and R. Cogranne, "On the Readiness of NDN for a Secure Deployment: The Case of Pending Interest Table,"

in *IFIP International Conference on Autonomous Infrastructure, Management and Security*. Springer, 2016, pp. 98–110.

[94] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1. IEEE, 1999, pp. 126–134.

[95] V. S. Frost and B. Melamed, "Traffic modeling for telecommunications networks," *IEEE Communications Magazine*, vol. 32, no. 3, pp. 70–81, 1994.

[96] T. Nguyen, X. Marchal, G. Doyen, T. Cholez, and R. Cogranne, "Content Poisoning in Named Data Networking: Comprehensive characterization of real deployment," in *Integrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on*. IEEE, 2017, pp. 72–80.

[97] L. Muscariello, G. Carofiglio, and M. Gallo, "Bandwidth and storage sharing performance in information centric networking," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*. ACM, 2011, pp. 26–31.

[98] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino, "Modeling data transfer in content-centric networking," in *Proceedings of the 23rd international teletraffic congress*. International Teletraffic Congress, 2011, pp. 111–118.

[99] T. Nguyen, R. Cogranne, and G. Doyen, "An optimal statistical test for robust detection against interest flooding attacks in ccn," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*. IEEE, 2015, pp. 252–260.

[100] T. N. Nguyen, R. Cogranne, G. Doyen, and F. Retraint, "Detection of Interest flooding attacks in named data networking using hypothesis testing," in *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*. IEEE, 2015, pp. 1–6.

[101] E. L. Lehmann and J. P. Romano, *Testing statistical hypotheses*. Springer Science & Business Media, 2006.

[102] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of TCP/IP with stationary random losses," *IEEE/ACM Transactions on Networking (TON)*, vol. 13, no. 2, pp. 356–369, 2005.

[103] J.-C. Bolot, "Characterizing end-to-end packet delay and loss in the internet," *Journal of High Speed Networks*, vol. 2, no. 3, pp. 305–323, 1993.

[104] F. Hildebrand and P. Crout, "A least square procedure for solving integral equations by polynomial approximation," *Studies in Applied Mathematics*, vol. 20, no. 1-4, pp. 310–335, 1941.

[105] J. L. Walsh, "Approximation by polynomials in the complex domain," 1935.

[106] P. D. Crout, "An application of polynomial approximation to the solution of integral equations arising in physical problems," *Studies in Applied Mathematics*, vol. 19, no. 1-4, pp. 34–92, 1940.

[107] A. Sen and M. Srivastava, *Regression analysis: theory, methods, and applications*. Springer Science & Business Media, 2012.

[108] C. Rao, *Linear Statistical Inference and its Applications*, ser. Wiley Series in Probability and Statistics. Wiley, 2009.

[109] K. Tout, F. Retraint, and R. Cogranne, "Non-stationary process monitoring for change-point detection with known accuracy: Application to wheels coating inspection," *IEEE Access*, vol. 6, pp. 6709–6721, 2018.

[110] H. Yin, C. Lin, B. Sebastien, B. Li, and G. Min, "Network traffic prediction based on a new time series model," *International Journal of Communication Systems*, vol. 18, no. 8, pp. 711–729, 2005.

[111] R. Cogranne, G. Doyen, N. Ghadban, and B. Hammi, "Detecting botclouds at large scale: A decentralized and robust detection method for multi-tenant virtualized environments," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 68–82, March 2018.

[112] R. Cogranne and F. Retraint, "An asymptotically uniformly most powerful test for LSB matching detection," *IEEE transactions on information forensics and security*, vol. 8, no. 3, pp. 464–476, 2013.

[113] ——, "Statistical detection of defects in radiographic images using an adaptive parametric model," *Signal Processing*, vol. 96, pp. 173–189, 2014.

[114] V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 221–234, Feb 2016.

[115] L. L. Scharf and B. Friedlander, "Matched subspace detectors," *IEEE Transactions on signal processing*, vol. 42, no. 8, pp. 2146–2157, 1994.

[116] R. Cogranne and J. Fridrich, "Modeling and Extending the Ensemble Classifier for Steganalysis of Digital Images Using Hypothesis Testing Theory," *Information Forensics and Security, IEEE Transactions on*, vol. 10, no. 12, pp. 2627–2642, Dec. 2015.

[117] H. V. Poor and O. Hadjiliadis, *Quickest detection*. Cambridge University Press Cambridge, 2009, vol. 40.

[118] A. Tartakovsky, I. Nikiforov, and M. Basseville, *Sequential analysis: Hypothesis testing and changepoint detection*. CRC Press, 2014.

[119] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[120] G. Lorden, "Procedures for reacting to a change in distribution," *The Annals of Mathematical Statistics*, pp. 1897–1908, 1971.

[121] A. Afanasyev, I. Moiseenko, L. Zhang, and others, "ndnSIM: NDN simulator for NS-3," *University of California, Los Angeles, Tech. Rep*, 2012.

[122] S. Basu, A. Mukherjee, and S. Klivansky, "Time series models for internet traffic," in *INFOCOM'96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE*, vol. 2. IEEE, 1996, pp. 611–620.

[123] H. L. Mai, N. Tan, D. Guillaume, R. Cogranne, M. Wissam, M.-d.-O. Edgardo, and F. Olivier, "Towards a security monitoring plane for named data networking and its application against content poisoning attack," in *to appear In IEEE/IFIP Network Operations and Management Symposium (NOMS), 2018*. IEEE, 2018.

[124] I. Aguirre and S. Alonso, "Improving the Automation of Security Information Management: A Collaborative Approach," *IEEE Security Privacy*, vol. 10, no. 1, pp. 55–59, Jan. 2012.

[125] S. M. Klerer, "The OSI management architecture: an overview," *IEEE Network*, vol. 2, no. 2, pp. 20–29, Mar. 1988.

[126] J.-P. Martin-Flatin, S. Znaty, and J.-P. Hubaux, "A Survey of Distributed Enterprise Network and Systems Management Paradigms," *Journal of Network and Systems Management*, vol. 7, no. 1, pp. 9–26, Mar. 1999. [Online]. Available: https://link.springer.com/article/10.1023/A:1018761615354

[127] C. Sluman, "A tutorial on OSI management," *Computer Networks and ISDN Systems*, vol. 17, no. 4, pp. 270–278, Oct. 1989. [Online]. Available: http://www.sciencedirect.com/science/article/pii/016975289900038X

[128] A. Afanasyev, J. Shi, B. Zhang, L. Zhang, I. Moiseenko, Y. Yu, W. Shang, Y. Huang, J. P. Abraham, S. DiBenedetto, and others, "NFD developer's guide - revision 8," Technical Report NDN-0021, 2016. [Online]. Available: https://named-data.net/wp-content/uploads/2018/02/ndn-0021-8-nfd-developer-guide.pdf

[129] D. Pesavento, O. I. E. Mimouni, E. Newberry, L. Benmohamed, and A. Battou, "A Network Measurement Framework for Named Data Networks," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ser. ICN '17. New York, NY, USA: ACM, 2017, pp. 200–201. [Online]. Available: http://doi.acm.org/10.1145/3125719.3132113

[130] D. Mansour and C. Tschudin, "Towards a Monitoring Protocol Over Information-Centric Networks," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, ser. ACM-ICN '16. New York, NY, USA: ACM, 2016, pp. 60–64.

[131] T. D. Nielsen and F. V. Jensen, *Bayesian networks and decision graphs*. Springer Science & Business Media, 2009.

[132] D. Barber, *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[133] R. E. Neapolitan and others, *Learning bayesian networks*. Pearson Prentice Hall Upper Saddle River, NJ, 2004, vol. 38.

[134] P. J. Lucas, L. C. Van der Gaag, and A. Abu-Hanna, "Bayesian networks in biomedicine and health-care," *Artificial intelligence in medicine*, vol. 30, no. 3, pp. 201–214, 2004.

[135] M. E. Borsuk, C. A. Stow, and K. H. Reckhow, "A Bayesian network of eutrophication models for synthesis, prediction, and uncertainty analysis," *Ecological Modelling*, vol. 173, no. 2, pp. 219–239, 2004.

[136] D. Margaritis, "Learning Bayesian network model structure from data," Carnegie-Mellon University Pittsburgh PA School of Computer Science, Tech. Rep., 2003.

[137] F. Rubio, J. Martínez-Gómez, M. Julia Flores, and J. M. Puerta, "Comparison between Bayesian network classifiers and SVMs for semantic localization," *Expert Systems with Applications: An International Journal*, vol. 64, no. C, pp. 434–443, 2016.

[138] M. H. A. Hasanat, D. Ramachandram, and R. Mandava, "Bayesian belief network learning algorithms for modeling contextual relationships in natural imagery: a comparative study," *Artificial Intelligence Review*, vol. 34, no. 4, pp. 291–308, 2010.

[139] S. Yaramakala and D. Margaritis, "Speculative Markov blanket discovery for optimal feature selection," in *Data mining, fifth IEEE international conference on*. IEEE, 2005, pp. 4–pp.

[140] I. Tsamardinos, C. F. Aliferis, and A. Statnikov, "Time and sample efficient discovery of Markov blankets and direct causal relations," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 673–678.

[141] M. L. Wong and Y. Y. Guo, "Learning Bayesian networks from incomplete databases using a novel evolutionary algorithm," *Decision Support Systems*, vol. 45, no. 2, pp. 368–383, 2008.

[142] A. Niculescu-Mizil and R. Caruana, "Inductive transfer for Bayesian network structure learning," in *Artificial Intelligence and Statistics*, 2007, pp. 339–346.

[143] J. Jun-Zhong, H.-X. ZHANG, H. Ren-Bing, and L. Chun-Nian, "A Bayesian network learning algorithm based on independence test and ant colony optimization," *Acta Automatica Sinica*, vol. 35, no. 3, pp. 281–288, 2009.

[144] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," *Machine learning*, vol. 65, no. 1, pp. 31–78, 2006.

[145] H. Guo and W. Hsu, "A survey of algorithms for real-time Bayesian network inference," in *AAAI/KDD/UAI02 Joint Workshop on Real-Time Decision Support and Diagnosis Systems*. Edmonton, Canada, 2002.

[146] R. Nagarajan, M. Scutari, and S. Lèbre, "Bayesian networks in R," *Springer*, vol. 122, pp. 125–127, 2013.

[147] U. Kjærulff, "Triangulation of graphs–algorithms giving small total state space," 1990.

[148] K. Murphy and others, "The bayes net toolbox for matlab," *Computing science and statistics*, vol. 33, no. 2, pp. 1024–1034, 2001.

[149] E. Charniak, "Bayesian networks without tears." *AI magazine*, vol. 12, no. 4, p. 50, 1991.

[150] K. B. Korb and A. E. Nicholson, *Bayesian artificial intelligence*. CRC press, 2010.

[151] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[152] D. M. T. Rose and K. McCloghrie, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II," RFC 1213, Mar. 1991. [Online]. Available: https://rfc-editor.org/rfc/rfc1213.txt

[153] B. Baygün and I. Hero, A.O., "Optimal simultaneous detection and estimation under a false alarm constraint," *Information Theory, IEEE Transactions on*, vol. 41, no. 3, pp. 688 –703, May 1995. [Online]. Available: http://dx.doi.org/10.1109/18.382015

[154] M. Wählisch, T. C. Schmidt, and M. Vahlenkamp, "Backscatter from the Data Plane—Threats to Stability and Security in Information-Centric Networking," *arXiv preprint arXiv:1205.4778*, 2012.

[155] N. W. Paper, "Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges & Call for Action. Issue 1," Oct. 2012.