POLITECNICO DI MILANO

SCUOLA INTERPOLITECNICA DI DOTTORATO

Doctoral Program in Telecommunication Engineering

Final Dissertation

Analysis of Security Issues in Information Centric Networking

Giulia Mauri

Tutor
prof. Giacomo Verticale

Co-ordinator of the Research Doctorate Course
prof. Carlo Ettore Fiorini

04 April 2016

The family is
link to our past,
bridge to our future.
- A. Haley -

Abstract

NAMED Data Networking is funded by the National Science Foundation for the Future Internet Architecture project. Projects such as Named Data Networking (NDN) and Content Centric Networking (CCN) belong to the same program for defining a network where the focus is on "what" users care about and not on "where" they are. In this novel architecture, generally called Information Centric Networking (ICN), contents are addressed by their name and not by their location. Thus, the attention is shifted from users to content, resulting in a caching network that is more efficient and flexible than an IP network for content distribution and management, with beneficial effects on timely delivery. This simple change allows ICN networks to exploit the Internet's infrastructure, and, at the same time, to address some of the Internet's most important problems in security, scalability, and sustainability. Moreover, widespread use of caching provides advantages for users and providers, such as reduced network latency, higher content availability, bandwidth reduction, and server load balancing. Indeed, such architecture heavily relies on sound security mechanisms to deal with issues related to cache robustness against cache poisoning attacks.

However, this architecture faces totally new challenges. This thesis analyses the problems related to security, privacy, caching, and mobility. First, all data packets must be signed by the producer and verified by the consumer, bringing up the need for defining a public key distribution and management scheme. That is one of the main focus of this work. Moreover, the widespread use of in-network caching and data replication rises problems

relative to content access control and staleness detection.

Then, the use of named data together with caching opens new problems for user privacy and paves the way for cache pollution attacks. On the one hand, it is possible to trace users requests on name basis to decide which content to prefetch in caches to guarantee good performance in content retrieval. On the other hand, this tracing requires to infer users preferences and behavior invading user's privacy. Furthermore, exploiting caching could be an easy way for an attacker to create a botnet under its control for spreading unwanted contents. The trade-off between caching performance and user's privacy, and the access control and content management are other topics that are evaluated and solved in this thesis.

Finally, the widespread use of caching together with the new communication model could provide advantages for mobile communications. We think that optimally distributing contents in the network caches helps in performing a seamless handover. Thus, it is necessary to study the user behavior and the optimal content allocation in the caches.

Sommario

Il progetto Named Data Networking è finanziato dalla fondazione National Science Foundation per definire la futura architettura di Internet. Sia il progetto Named Data Networking (NDN) che Content Centric Networking (CCN) appartengono allo stesso programma per la definizione di una rete che sia basata su cosa interessa agli utenti e non dove si trova ciò che interessa agli utenti. In questa nuova architettura, chiamata più generalmente Information Centric Networking (ICN), i contenuti sono indirizzati grazie al loro nome e non alla loro posizione. Quindi l'attenzione si sposta dagli utenti ai contenuti, risultando così nella creazione di una rete di cache che è più efficiente e flessibile della tradizionale rete IP per la distribuzione dei contenuti e la gestione delle risorse ottenendo maggiori benefici. Questo semplice cambiamento permette alle reti ICN di sfruttare l'infrastruttura della Internet tradizionale e, allo stesso tempo, di risolvere alcuni dei più grossi problemi della stessa per quanto riguarda sicurezza, scalabilità e sostenibilità. Infatti, questa nuova architettura si appoggia su alcuni sistemi di sicurezza che permettono di affrontare e risolvere gran parte dei problemi legati alla resistenza contro gli attacchi di cache poisoning.

D'altra parte, questa nuova architettura deve risolvere problemi completamente nuovi. Questa tesi analizza i problemi legati a sicurezza, privacy, caching e mobilità. Dapprima consideriamo i problemi legati alla sicurezza. In particolare, dato che tutti i pacchetti devono essere firmati dal loro creatore e verificati dal consumatore, è necessario definire uno schema per la distribuzione e la gestione delle chiavi pubbliche e private, così come avviene nella PKI (Public Key Infrastructure) dello standard Internet. Questo

è uno dei principali argomenti della tesi. Inoltre, lo sfruttamento delle cache in rete e la replica di contenuti fanno sorgere problemi legati al controllo dell'accesso ai contenuti da parte degli utenti e il bisogno di rimuovere dalle cache i contenuti che sono scaduti o non validi.

Inoltre, l'utilizzo di contenuti con nomi e il loro caching pervasivo porta ad altri problemi come quello della privacy degli utenti e spiana la strada per attacchi di cache pollution. Da una parte, è possibile tracciare le richieste degli utenti sulla base dei nomi dei contenuti in modo da definire una politica di caching ottima per garantire buone prestazioni nel recupero dei contenuti. D'altra parte, per fare ciò è necessario ricavare le preferenze degli utenti invadendo in questo modo la loro privacy. In aggiunta, un attaccante potrebbe sfruttare le cache a suo vantaggio per creare una botnet che distribuisca in rete i suoi contenuti. Il trade-off tra prestazioni garantite dalla rete e la privacy degli utenti, e il controllo dell'accesso ai contenuti e la loro gestione sono altri argomenti affrontati in questa tesi.

Infine, sempre l'uso pervasivo delle cache insieme al nuovo modello di comunicazione comporta dei vantaggi per le comunicazioni mobili. Infatti pensiamo che sia possibile distribuire in maniera ottima i contenuti nella rete per ottenere migliori prestazioni in caso di handover. Quindi è necessario studiare il comportamento degli utenti per poter pre-allocare in maniera ottima i contenuti a cui sono interessati.

Contents

CHAPTER 1

Introduction

THE Internet is rapidly changing. New problems are arising as a consequence of its architecture. When the Internet was designed, the main issue to solve was to connect two end points which were located far away in a fixed position. However, the Internet is growing in terms of its size and number of applications that run on it. Thus, this brings to the need of defining a new clean-state architectural approach for the new Internet. The research community is defining the properties and requirements of the Future Internet.

The current Internet is facing the increased traffic volume by using distribution technologies, such as P2P (Peer to Peer) and CDN (Content Delivery Network), that are based on a communication model of accessing data by name, regardless of the source location, and on employing caching and content replication. However, different content providers and P2P applications rely on their technologies without having an unified solution for content distribution over the Internet. Thus, it is not easy to optimize network efficiency and performance.

The Future Internet architecture will be probably defined by the Information Centric Networking (ICN) paradigm. The information are named

at the network layer, thus making easier the content delivery to the users. Moreover, ICN solves some other issues of the Internet architecture such as mobility and security. Then, the in-network caching, that is one of the fundamental principles behind ICN, helps in improving the network efficiency and capabilities for information distribution. ICN is expected to evolve the Internet architecture by providing a network model more suitable for the current and future needs.

Firstly, the ICN paradigm uncouples information and location. Information can be located anywhere in the network but each information element is uniquely named. The receiver should know the name or the name's prefix of the information element it wants to retrieve. The network locates information in in-network caches. Then, the network is responsible for forwarding requests and responses on optimal paths. The key concept of ICN is naming data objects: names are important for making forwarding decisions and for matching requests to responses. It becomes noteworthy the validation of the name-content binding: the content should carry the information requested with the corresponding name.

Secondly, the ubiquitous network storage allows every node to answer to the requests for the cached object without the need to verify the node authenticity. The in-network caching provides a lot of advantages: enabling sharing, making communication more robust, supporting retransmission, and fast reacting to disruption. Moreover, the in-network caching together with the communication model allows new options for transport services, new interconnection and business models.

In summary, the new ICN framework could provide a lot of advantages to today's Internet. Moreover, it seems the natural evolution of the latter.

The design of an efficient ICN architecture poses several technical challenges. A recent Internet draft [33] presents an overview of the main open problems. Figure 1.1 represents the open challenges in the Information Centric Networking scenario that will be presented in the following of this section and will be evaluated and solved along the chapters of this thesis.

Firstly, data object authentication is a fundamental ICN feature. Since data objects are replicated in network caches, they can be modified by malicious entities. Thus, ICN should provide a security mechanism to verify origin and integrity of contents. Then, it is also necessary to define a *trust management* infrastructure to distribute the publisher's public key to the consumers. Moreover, data replication leads to a loss of control on content access and content dissemination. The content provider needs to know who accesses its contents, where and when its contents are used, and to revoke the access rights. The content provider also needs to update and synchro-

Figure 1.1: *The Information Centric Networking Open Challenges*

nize its content.

Beyond data network security in terms of data confidentiality and integrity, the ICN domain introduces new *privacy* issues related to the protection of what data could reveal. The most relevant goal is the protection of data that could reveal information about an individual along with his or her physical, cultural, economic, social characteristics, or personal behaviors. Both the user requests and the cached contents have a unique name and can reveal a lot of information about the users. Meanwhile, these data are useful to improve the network performance, for example to define a caching policy based on user. Thus, it is necessary to find a trade-off between network performance and users' privacy.

The in-network *caching* brings along improved efficiency, better scalability, and increased network performance but also attracts new kinds of attacks such as cache pollution attacks by which a malicious content producer could control a massive amount of storage for spreading malware, junk, and other attacker controlled content at a low price.

Then, the communication model and data replication in the network

caches should facilitate a seamless handover in a mobile scenario. A seamless transition in ICN ensures that the content retrieval does not suffer from intermittent connectivity. New challenges arise from the ICN mobile scenario. Especially, the *mobility* management should be coordinated between the network nodes and the users for optimizing caching policies and sizing.

Highly related to mobility, there is the problem of *routing* based on names. ICN routing comprises name resolution, content discovery, and data delivery. There is not a common consensus on how to manage these steps and different solutions are provided in literature. However, specific challenges of ICN routing are still open. The routing issues are strictly related to the *naming* convention. Indeed, two possible solutions have been proposed: hierarchical and flat namespaces. Each solution has its own advantages and drawbacks but also in this case there is not a definitive accepted proposal.

1.1 Contribution of this work

In this work, we cover some of the previous presented challenges in the Information Centric Networking scenario. All our proposals are original and, as far as we know, we are the first presenting such solutions. Firstly, we consider the problem of trust management, deeply inspecting the issue of distributing and retrieving up-to-date keys. Then, we evaluate the content access control and staleness management. Secondly, we describe the caching issues, in relation with users' privacy and cache pollution attacks. Finally, we also provide an overview of some optimization models for a vehicular ICN scenario.

The contents of this thesis, which has been developed between 2012 and 2015 during the PhD Program in Information Engineering in the Department of Electronics, Information and Bioengineering of Politecnico di Milano, are based on the following scientific publications:

1. Giulia Mauri and Giacomo Verticale "Distributing Key Revocation Status in Named Data Networking" *19th Eunice Workshop*, August 2013.

2. Giulia Mauri and Giacomo Verticale "On the Tradeoff between Performance and User Privacy in Information Centric Networking" *6th International Conference on New Technologies, Mobility & Security*, April 2014.

3. Federico Bruno, Matteo Cesana, Mario Gerla, Giulia Mauri, and Giacomo Verticale "Optimal Content Placement in ICN Vehicular Net-

works" *5th International Conference on Network of the Future*, September 2014.

4. Giulia Mauri, Riccardo Raspadori, Mario Gerla, and Giacomo Verticale "Exploiting Information Centric Networking to Build an Attacker-Controlled Content Delivery Network" *The 14th IFIP Annual Mediterranean Ad Hoc Networking Workshop*, June 2015.

5. Li Zhe, Jean-Charles Point, Selami Ciftci, Onur Eker, Giulia Mauri, Marco Savi, and Giacomo Verticale "ICN Based Shared Caching in Future Converged Fixed and Mobile Network" *The 16th International Conference on High Performance Switching and Routing*, July 2015.

6. Giulia Mauri and Giacomo Verticale "Up-to-data Key Retrieval for Information Centric Networking" *Transactions on Emerging Telecommunications Technologies*, 2015. SUBMITTED

7. Giulia Mauri, Federico Bruno, Matteo Cesana, Mario Gerla, and Giacomo Verticale "Content Placement in ICN Vehicular Networks" *Transaction on Vehicular Networks*, 2015. SUBMITTED

The remainder of the thesis is organized as follows.

Chapter 2 provides an overall view of the Information Centric Networking scenario, with a focus on the challenges and benefits introduced by this new architecture. The chapter also compares the most relevant initiatives aimed at defining the reference ICN framework. It concludes listing some of the open issues that will be covered in this thesis.

Chapter 3 briefly summarizes the fundamental definitions about the Named Data Networking architecture. This is the reference architecture for all the works described in this thesis: thus, when we write ICN, we have as reference architecture the NDN project.

Our proposed framework is compared to the recent scientific literature addressing the issues of security, privacy and mobility in ICN in **Chapter 4**.

Chapter 5 exploits the problem of how to retrieve up-to-date signing key in the ICN scenario. Since contents are stored and replicated into node caches, the content validity must be assured end-to-end. However, the use of digital signatures requires a key management infrastructure to manage the key life cycle. To perform a proper signature verification, a node needs

to know whether the signing key is valid or has been revoked. In this Chapter, we present a naive solution that implements the same functionalities of CRL, two reactive methods that are an adaptation of the OCSP to the ICN framework, and a distributed method that is an alternative to the traditional PKIX. Furthermore, in-network caching and data replication raise issues on content access management and control.

Such framework will be thoroughly discussed in Chapter 5, providing the following novel contributions:

- We provide, as far as we know, the first proposal for a definition of a PKI-like infrastructure to adapt the OCSP and CRL schemes to the ICN scenario.
- We propose a new solution based on a distributed method to better adapt the up-to-date key retrieval in a NDN-friendly way.
- We evaluate and compare the various proposals in terms of number of exchanged messages, latency and its standard deviation, and throughput.
- We show that our solution overcomes the main drawbacks of the standard schemes guaranteeing good network performance.

Then, **Chapter 6** describes how an attacker using compromised hosts can easily gather a massive amount of low-cost, and low-latency storage for malware, junk, and other attacker-controlled content. This ability provides a clear economic incentive for the adversary, and thus makes this kind of attack especially attractive. Thus, in Chapter 6:

- We describe a new kind of attack where attackers can do more than a DoS and build a large storage network that can be used to store junk content, malware, and in general any kind of attacker-controlled content.
- We evaluate and provide effectiveness of the attack by means of simulations.
- We consider a possible countermeasure, a blacklist fed by a honeypot, which we show to be effective.

Moreover, also **Chapter 7** considers a possible attack on ICN caches. In particular, a malicious node could exploit user-related information gathered from cached content to compromise the privacy of users. We propose a mathematical definition of user privacy suitable for the scenario, and we

evaluate the tradeoff between privacy and performance, in terms of latency. Then, in Chapter 7:

- We take the first step for defining the tradeoff between caching performance and user privacy guarantee.
- We provide a way to implement prefetching and we define some bounds for the users' privacy in this context.
- We present our attacker model, then we define a possible countermeasure to guarantee user privacy.

Finally, **Chapter 8** provides an ILP formulation of the problem of optimally distributing content in the network nodes to maximize the probability that a user retrieves the desired content in a Vehicle-to-Infrastructure scenario. The content distribution is subject to the storage capacity of each cache and the link capacity. We study how different system assumptions impact the success probability in different ways.
Therefore, in Chapter 8:

- We give an ILP formulation of the problem of optimally distributing content in the network nodes, while accounting for the available storage capacity and the available link capacity.
- We leverage the optimization framework to evaluate the impact on content retrievability of topology- and network-related parameters as the number and mobility models of moving users, the size of the content catalog and the location of the available caches.
- We show how the proposed model can be modified to find the minimum storage occupancy to achieve a given content retrievability level.
- We validate the results obtained from the optimization model against a Name Data Networking architecture through simulations in ndnSIM.

Final conclusions are drawn in **Chapter 9**.

CHAPTER 2

Information Centric Networking: architecture, requirements and benefits of the future network

THIS Chapter introduces the concept of Information Centric Networking and proposes a brief overall view of the structure of the future network, where the attention is shifted from users to contents. Potential benefits and challenges which arise in the new ICN scenario will also be discussed, as well as the most important projects around ICN. Moreover, the Chapter describes some open issues that will be covered in the next chapters.

2.1 What is "Information Centric Networking"?

The term "Information Centric Networking" (ICN) appeared around 2006, inspired by Van Jacobson's Google Tech Talk "A New Way to look at Networking". This talk focuses on a new direction for the future Internet toward a content centric architecture. Several different proposals have emerged in the last few years around this common ICN principle. The ob-

jective is to evolve from the Internet in order to fit the world that we have today. Nowadays, there is a huge content creation explosion and consequent dissemination of data over the network storages. Thus, the content itself is the key player of the future Internet. The content is wherever there is interest in it, it goes where it is requested, not anywhere else. The users ask for the content in which they are interested and they do not care from where it comes.

2.2 ICN Features

The ICN architecture differs from the standard TCP/IP protocol in the following ways:

- Request/Response communication model: it is the receiver that sends an interest message for the content it desires. After the request is sent, the network is responsible for providing the corresponding response from the best source. No data packet can be received if it is not explicitly requested with an interest packet.
- Hierarchical content naming scheme: each content has an arbitrary, user-defined name organized in a hierarchy. The object's name is persistent and unique. The routing is performed by doing longest-prefix match on names. Moreover, it is possible the aggregation of the request for the same information, i.e. request with the same prefix name.
- Cache-based architecture: all the network nodes store content and use them for satisfying future requests.
- Content security: all the content are signed by the producer, that certifies the binding between the content and its name. Thus, ICN provides name-data integrity, object authenticity, and origin verification.

2.3 Expected Benefits

Such benefits include:

- **Information-centric**: The content assumes a central role in the future Internet. Each content should be named with location-independent and application-independent names.
- **Efficient support for mobility**: The number of mobile devices continuously grows and needs efficient support for mobility. Mobile nodes can reissue interest message for content after handoff and the network is responsible to redirect the data to the nearest caches to the user.

- **Efficient support for multi-homing**: The future Internet should allow a host to be simultaneously connected to multiple networks.
- **Encouraging innovations**: Different network technologies should coexist and be contemporaneously deployed.
- **Enhanced security**: The user are allowed to refuse unwanted traffic and to control incoming and outgoing packets.
- **Enhanced scalability**: The routing should be more scalable in terms of routing tables.
- **Deployability**: The deployment of the new protocol should not incur in significant costs.

2.4 Relevant Initiatives

There are numerous approaches aimed at defining the reference ICN framework, including architectures, applicative scenarios, and standardizations. Here we briefly review the most significant ones.

- Content Centric Networking (CCN) [31], a US funded project;
- Named Data Networking (NDN) [57], a US funded project;
- Data-Oriented Architecture (DONA) [32], a project at Berkeley;
- Publish-Subscribe Internet Routing Paradigm (PSIRP) [24], a EU funded project, now in Publish-Subscribe Internet Technology, PURSUIT [23];
- Network of Information (NetInf) [17], currently in the Scalable & Adaptive Internet soLutions (SAIL) [5], a EU funded project;
- COntent Mediator architecture for content-aware nETworks (COMET) [25], a EU funded project;
- MobilityFirst [48], a US funded project;

Figure 2.1 show the ICN oriented projects together with the timeline.

All these ICN projects are currently under development and although with different approaches, they have some similar key functionalities. Here we report the main features, as resumed in [55]:

- **Naming**: Each piece of content in the network has a *name*. Naming can be flat, the content identifier is a cryptographic hash of a public key, or hierarchical, the content identifier is like a web URL. Usually, hierarchical names are human-readable, while flat names are not.

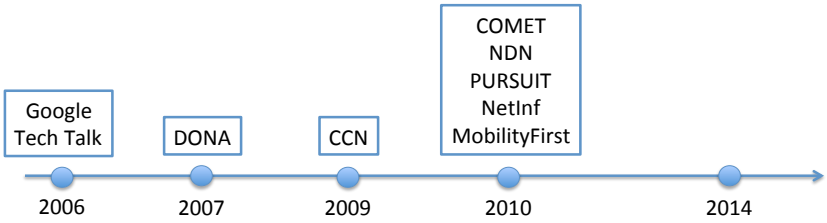

Figure 2.1: *Timeline of ICN milestones.*

- **Name resolution and data routing**: These two functions can be coupled or decoupled. In the first approach, the content request is routed to the provider and the data response follows the same path. While in the decoupled approach, the path followed by the data is not restricted to be the same of the request. Moreover, the name-based routing can be unstructured, mainly performed based on flooding, or structured exploiting a tree and a distributed hash table structure.
- **Caching**: There are two options: on-path and off-path caching. The on-path caching stores content along the path of the request, while off-path caching exploits content stored outside the path.
- **Mobility**: User mobility is easy to support, since new requests can be sent after a handoff. While, provider mobility is harder to manage, since name resolution and data routing should be updated.
- **Security**: It is highly related to the naming structure. The human-readable names need a trusted agent to authenticate the relation between a content and its name. While, self-certified names require a trusted agent to map the name to a human-readable one.

2.4.1 NDN and CCN

The Named Data Networking and the Content Centric Networking projects were born in California at the University of California, Los Angeles (UCLA) and, at the Palo Alto Research Center (PARC), respectively. The basic principles of both projects are very similar and sometimes also the same. In this Section, we provide a brief introduction relative to the CCN project. Then, in Chapter 3, we give a detailed description of the NDN proposal, that is the reference model for this thesis.

Data availability, security, and location-dependence are three problems that affect users in the Internet of today. CCN is build upon named data in order to replace named host and to shift from where to what. Thus,

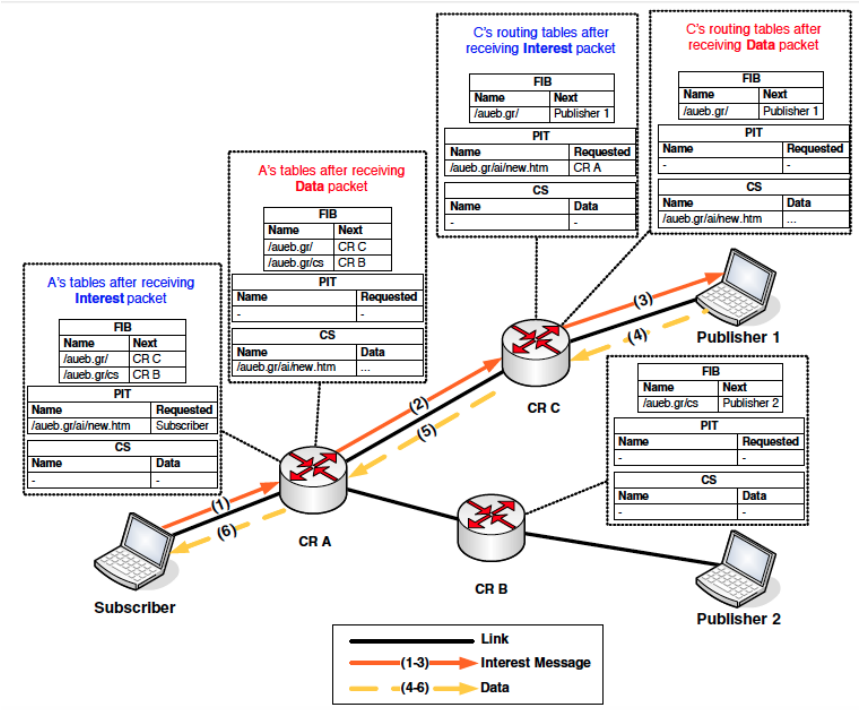

Figure 2.2: *CCN overview [55].*

each content object in the network has a hierarchical name that is similar to a URL. When an information is requested using its name, any piece of information whose name has the same requested name as prefix is the matching answer. The request is called Interest, while the response is the Data packet.

As depicted in Figure 2.2, the subscriber sends an Interest and the publisher responds with a Data. The intermediate nodes, the Content Routers (CRs), have three data structures which are used to forward Interest and Data packets. These databases are the Forwarding Information Base (FIB), the Pending Interest Table (PIT), and the Content Store (CS). The FIB and PIT are used for routing, while the CS is for storage. Additional details can be found in Chapter 3.

2.4.2 DONA

The Data Oriented Network Architecture (DONA) [32] was born from one particular idea: the shift from host-centric to data-centric applications in the today's Internet. Moreover, the authors identify three relevant issues

to be overcame: name persistence, data availability, and content authenticity. Thus, they propose to replace DNS names with flat and self-certifying names, and to replace DNS name resolution with a name-based primitive. In this way, name persistence and data authenticity can be guaranteed with names, while availability with name resolution service. The name resolution process happens by means of the route-by-name paradigm.

In DONA, principals own data and are responsible for them. Each principal has a public-private key pair. The name is as P:L, where P is the cryptographic hash of the principal's public key, and L is a label chosen by the principal. Each name is globally unique and self-certifying: each user can verify the authenticity and consequently the persistence of each data. Indeed, the client receive a triplet <data, public key, signature>, when it asks for a data using the name P:L. The route-by-name paradigm is based on resolution handlers (RH). Usually, a user sends a FIND(P:L) packet to retrieve an object named P:L. Then, RH is responsible to route the packet to the nearest copy. There is another kind of packet, REGISTER, that allows RHs to route the request in a proper way. Indeed, each RH has a registration table that maps a specific name to the next-hop and the distance to the copy.

As depicted in Figure 2.3, a FIND(P:L) packet is sent by a requester. When the RH receives the request, if there is an entry in the registration table, it routes the FIND(P:L) to the next-hop RH; otherwise, it forwards the packet to its parent. If there is a table miss, the packet is forwarded up in the tree hierarchy till an entry is found. Then, a data packet is sent back in response, through the reverse path or over a direct route.

Resolution handlers can implement caching. In order to assure that a data packet is routed through a particular RH, the RH itself should modify the incoming FIND by putting its address as source address. When, the RH receives the data, it can put it in its cache and make it available for the next requests. Usually, each data has a TTL to determine when it becomes stale.

Mobility is easily handled in DONA. The mobile clients can reissue FIND packets from the new location, then RHs are responsible for providing the closest copy of the data.

2.4.3 PSIRP and PURSUIT

The Publish-Subscribe Internet Routing Paradigm (PSIRP) [24] would like to address various challenges of the Internet. For example, one challenge is to distribute the contents to only those users which are really interested in and, at the same time, to guarantee the user's privacy and the digital rights.

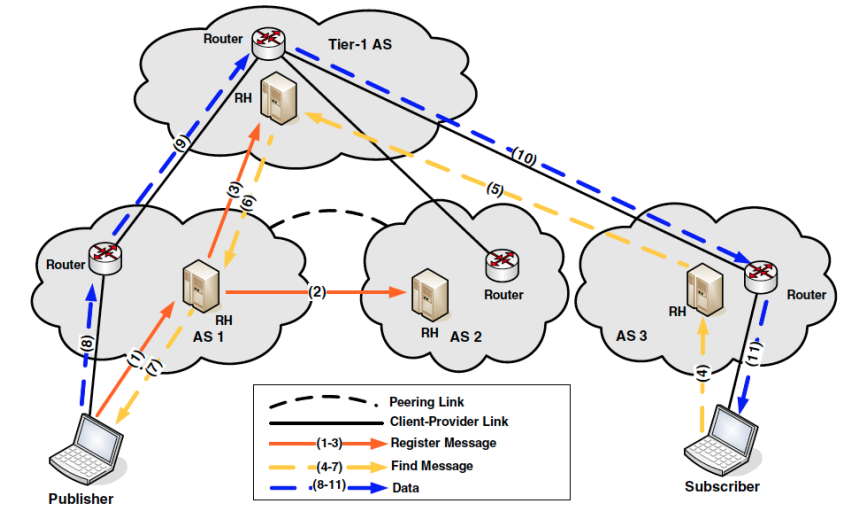

Figure 2.3: *DONA overview [55].*

A publish/subscribe paradigm helps in overcoming this challenge. Indeed, the end-users communicate their interest in a specific content, which is sent to them by the network when it is available.

As it can be expected, the information is the focus in the PSIRP architecture. Each information is composed of small pieces of data, generally called chunks, organized in a hierarchical way. Each chunk is uniquely identified by a label, that is used by the users to express their interest. The label is called *rendezvous identifier* (RId). A subclass of it is the *scope identifier* (SId). Moreover, the labels are flat and endpoint independent in order to separate location from identity.

Three are the roles in the PSIRP architecture: (i) publisher, the information provider; (ii) subscriber, the information consumer, and (iii) rendezvous point (RP). The latter is the node where the consumer's interest is matched with the publisher's content by means of a Rendezvous function. The RP is responsible for managing the information's scope. The publisher locates the eligible RPs, called *rendezvous nodes* (RN), and publishes the publication's metadata. The subscriber must be aware of the publication's RId and SId. When a user wants an information, he/she sends a subscription message to the publication's RP, identified by the SId. Then, the RP matches the subscription with the publication and forwards the publication from the publisher to the subscriber. The *forwarding identifiers* (FId) are path identifiers assigned to the publication and used by intermediary nodes

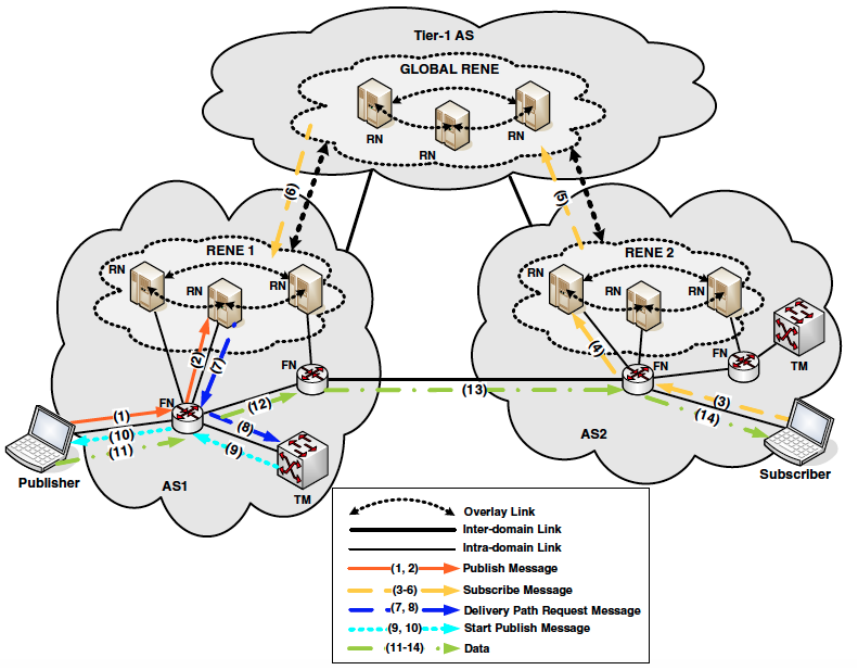

Figure 2.4: *PURSUIT overview [55].*

to forward the information to the requesting consumer. It is possible to cache publications along the path, but also off-path caching is enabled. Finally, the Topology function keeps the network topology updated and is responsible for creating the path for the information delivery.

Mobility can be easily managed in two ways: in a static or dynamic way. Moreover, multicast and caching facilitate mobility support and enhance performance.

The following Figure 2.4 depicts the PURSUIT architecture [23], where the Rendezvous Nodes (RN), the REndezvous NEtwork (RENE), the Forwarding Nodes (FW) and the Topology Manager (TM) are represented.

2.4.4 NetInf and SAIL

The Network of Information (NetInf) architecture [17] is a ICN approach that aims to define a new global-scale communication paradigm. NetInf would fulfill some important requirements for a content centric network. First of all, content distribution should be scalable and efficient; moreover, data access should be guaranteed in every network condition. Then, the content should have a persistent and location independent name. Finally,

user mobility and multihoming should be easily managed.

NetInf focuses on Named Data Objects (NDOs), which are Application Data Units (ADUs) and consist of their name, independent of network location, and the object itself. NDO names are flat, meaning that there is not a hierarchy in names. The NetInf protocol is message based: a NDO request (`GET`) is sent to the copies of the object and then the object is sent back. The routing and forwarding operations are name-based or can exploit name resolution services (NRS). In the latter case, the service map a NetInf name to network or host identifiers in different namespaces, which are called routing hint. A routing hint indicates where to find an object. NDOs are made available in the network by the publisher by means of a `PUBLISH` message. NDOs are spread in the network caches by means of on-path and off-path caching.

Mobility is simply supported: parts of NDO can be sent from different sources, if the requester moves in the network, it does not need to be involved or be aware of the changing sources.

The Figure 2.5 represents the NetInf architecture, now converged in the SAIL project [5], where CR represents the Content Routers.

2.4.5 COMET

The COntent Mediator architecture for content-aware nETworks (COMET) [25] project defines a new content oriented architecture for content distribution in a content aware fashion. The main problem of today's Internet is the lack of a global naming scheme and an infrastructure for content access. COMET starts from these two issues and solves them by designing a global naming scheme with a unified approach for content access and distribution, whatever its nature and location. COMET is based on the concept of mediation: an intermediate plane between the world of content and the world of data transmission is provided by the ISP (Internet Service Provider). This plane provides content, server, routing and network awareness, as detailed in the following.

The contents are identified by a Content-ID, that is machine readable, or a Content Name, that is human readable; both of them are globally unique for a particular content. The content's copies are placed in Content Servers and located by the Content registration process, that creates a relation between the Content-ID and the network locations. Then, the content can be used in Content Consumption that is composed of three operations: (i) Awareness, (ii) Content Resolution, and (iii) Content Delivery. The Awareness consists of the process of providing routing awareness with informa-

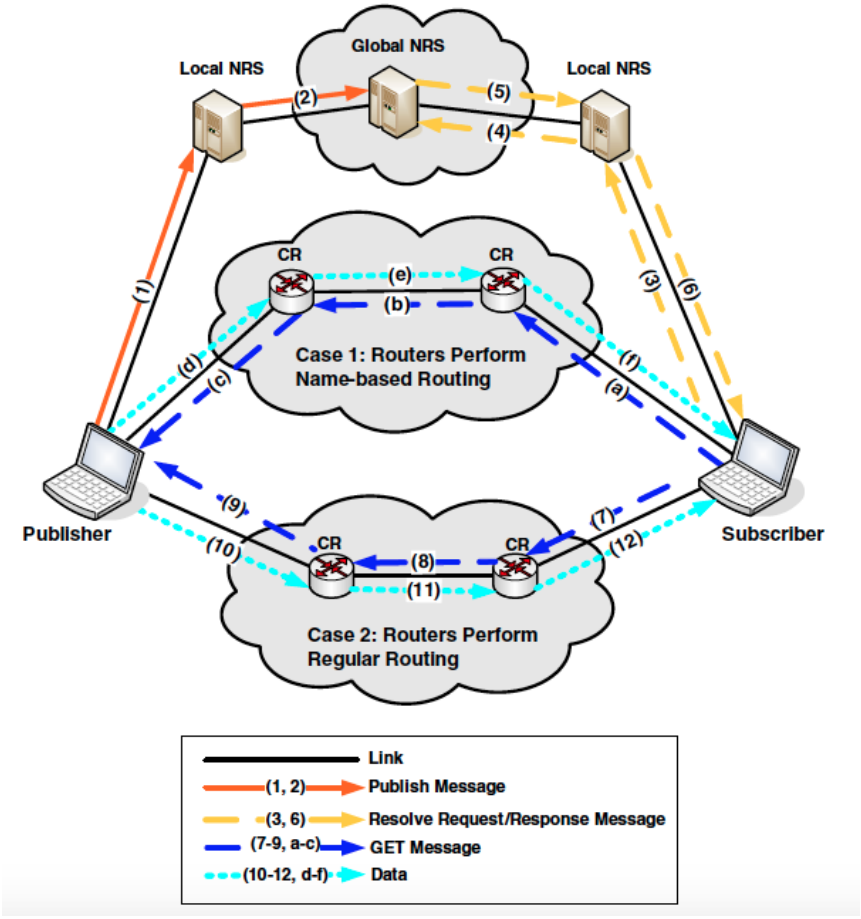

Figure 2.5: *SAIL overview [55].*

tion about the network topology, server awareness with information about the servers' conditions and network awareness with information about the link and path's condition in the network. Then, the Content Resolution starts with the client request of a content and ends with the system decision about which server and which path should be used for the Content Consumption. First, the name resolution process locates the object based on its name; then, the path discovery obtains the path from the Content Servers to the Client. Second, the decision process selects the best server and path; finally, the path configuration enforces this decision. At the end, the Content Delivery consists of sending the content to the Client according to the decision made during the Content Resolution process. All these operations are achieved thanks to two planes: the Content Mediation Plane (CMP) for

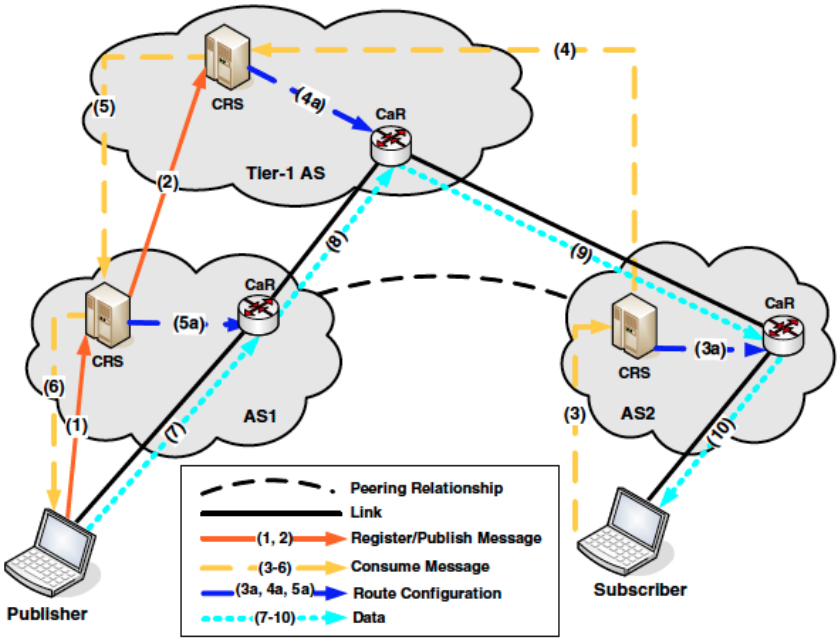

Figure 2.6: *COMET overview [55].*

name and content resolution; and the Content Forwarding Plane (CFP) for content delivery.

Special Content-aware Routers (CaRs) are responsible for mobility management; they track the users path and predict their future locations allowing the transfer of information in advance.

Figure 2.6 represents the decoupled COMET architecture.

2.4.6 MobilityFirst

The MobilityFirst network architecture [48] makes mobility to come first because the number of mobile devices will outnumber fixed hosts, and their traffic will surpass all other Internet traffic. Mobility and trustworthiness are the main goals of this architecture. However, there are other objectives to be fulfilled, such as a seamless support for host and network mobility.

The architecture is centered on a name-based service layer that uses flat names, that are globally unique identifiers (GUIDs), for network objects. GUIDs source and destination, together with a service identifier (SID), define a message to invoke the network services. Then, routing is performed by means of a fast global name resolution service (GNRS) that links the destination GUID to a set of network addresses (NAs). When a client wants

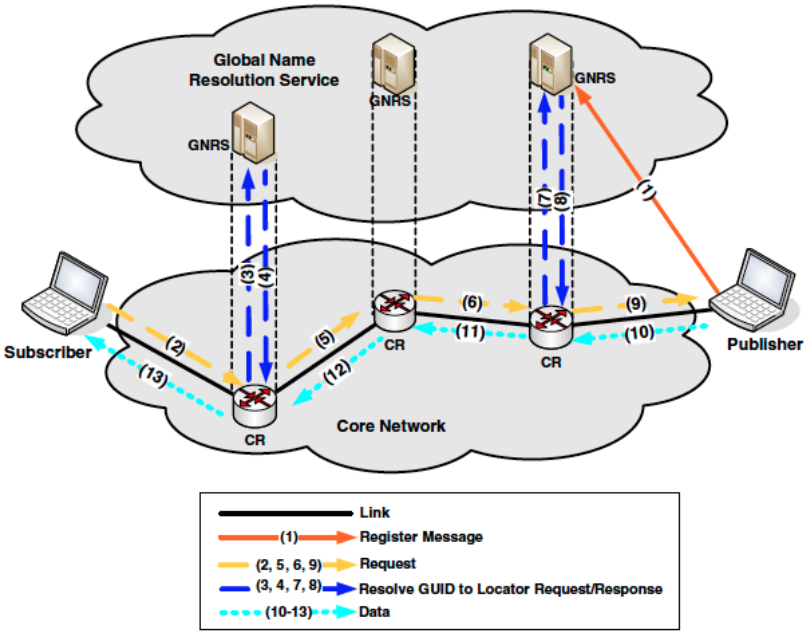

Figure 2.7: *MobilityFirst overview [55].*

a content, it sends a Get message with the corresponding GUID to the Content Router (CR). The latter ask the GNRS the correspondence between the GUID and the NAs. Then, the Get message is forwarded using the routing table to the publisher. On path caching is supported by caching messages at intermediate nodes. If a message is cached off path, the GNRS is informed to update its entries.

Mobility is managed by the GNRS, that updates its entries as an object changes its location point.

The following Figure 2.7 shows the MobilityFirst architecture.

2.4.7 Comparison

The main characteristic and differences of the ICN approaches are highlighted by the following Table 2.1.

Table 2.1: *Summary of characteristic of the ICN approaches [55].*

	CCN/NDN	DONA	PURSUIT	SAIL	COMET	MobilityFirst
Naming	Hierarchical	Flat	Flat	Flat	Unspecified	Flat
Name Resolution and Data Routing	Coupled	Coupled and Decoupled	Decoupled	Coupled, Decoupled, and Hybrid	Coupled	Decoupled
Caching	On-path caching at content routers. Off-path caching with additional routing information.	On-path caching at resolution handlers. Off-path caching with additional registrations.	On-path caching difficult. Off-path caching with additional registrations.	On-path caching at content routers. Off-path caching with additional routing information or registrations.	Probabilistic on-path caching at content routers. Off-path caching with additional registrations.	On-path caching at content routers. Off-path caching with additional registrations.
Mobility	Subscriber mobility via new requests. Interest flooding protocol for publisher mobility.	Subscriber mobility via new requests. Publisher mobility requires additional registrations.	Subscriber mobility via new requests. Publisher mobility requires updating the topology manager.	Subscriber mobility via new requests. Support for publisher mobility via routing hints in hybrid operation.	Specialized mobility-aware content routers at access network that exchange mobile context state.	Subscriber and publisher mobility via late binding to first reach the mobile area and then find the mobile.

2.5 Open Issues

Since the concept of ICN is quite new, there are a lot of issues and problems that are still open. Some of them have been inspected but not yet solved, while others are completely untouched.

- Naming: The main problem is whether to use hierarchical or flat names. It seems that hierarchical names are easier to manage and can be human-readable, but they do not scale well. While, flat names are manageable and can be self-certifying. Thus, there is no consensus on which type of name should be used.
- Name Resolution: The scalability in name resolution is a big challenge. The proposals to overcome this issue are DHT-based designs, a mesh-like inter-domain graph and the use of hashing to map names to IP addresses. However, a final solution has not been found.
- Data Routing: There are some proposals for the intra-domain level but very few for the inter-domain level. However, how this solutions scale to Internet size is a big open problem.
- Caching: Smart caching policies for storage management can improve the network performance and provide benefits for both users and providers. Thus, there is an urgent need for defining and deploying optimal caching policies and replication mechanisms.
- Mobility: Some efforts have been done on supporting user mobility. However, provider mobility is yet unsolved due to the non-scalability of name resolution systems.
- Security, Privacy and Trust: The need for a key management system is of paramount importance. Then, the new architecture raises new privacy threats and new kinds of attack that still need an efficient countermeasure. Also, a trust model should be defined.

CHAPTER 3

The Named Data Networking project

In this Chapter, some basic notions about the Named Data Networking (NDN) architecture are presented.

3.1 What is " NDN" ?

Named Data Networking starts from an earlier project, Content Centric Networking (CCN), which was born in 2006 thanks to Van Jacobson. NDN is funded by the U.S. National Science Foundation under its Future Architecture Program. This proposal moves from a host centric network architecture (IP) to a content centric network architecture (NDN). This evolution seems to be natural in today's Internet since the focus is ever more on contents than on users. Social networks, e-commerce, web surfing, video streaming are all focused on content object, on *what* user needs and, not on *where* it is. Thus, NDN changes the network from "delivering the packet to a given destination address to fetching data identified by a given name". Figure 3.1 represents the evolution from the IP architecture with its fixed point-to-point communication to a flexible distribution network with named content chunks.

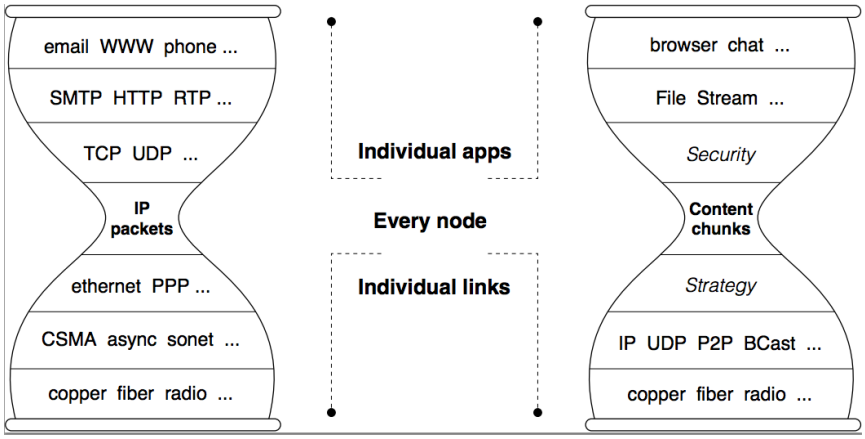

Figure 3.1: *The main building blocks of the NDN architecture are named content chunks, in contrast to IP architecture's fundamental unit of communication, which is an end-to-end channel between two endpoints identified by IP addresses. [56]*

3.2 NDN Architecture

3.2.1 Packets in the NDN Architecture

The communication follows the request/response paradigm. The request is the *Interest* packet, while the response is the *Data* packet. Both the packets are identified by a **name**. In order to be a valid response, the Data packet must carry the same prefix name of the corresponding request, the Interest packet. The name is used by the router nodes to forward the Interest to the data producer(s) and to forward the Data to the requesting consumer(s).

Figure 3.2 depicts the two types of packets in the NDN architecture.

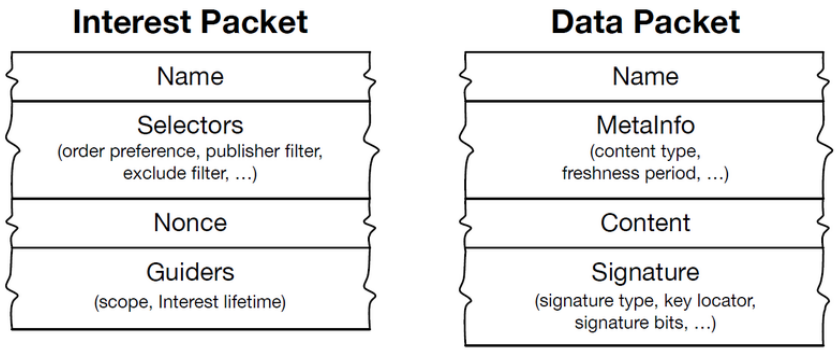

Figure 3.2: *Packets in the NDN Architecture [56].*

Interest Packet

The Interest packet is composed of the following fields:

- **Name**: is a hierarchical name for NDN content, which contains a sequence of name components. More details about the name format will be given in the following.
- **Selectors**:
  - `MinSuffixComponents/MaxSuffixComponents` refer to the name of name components. They allow a consumer to specify whether the name is the full name including the digest, or the full name excluding the digest, or the content name is known to be in a known range of legitimate components.
  - `PublisherPublicKeyLocator` specifies the name of the key used to sign the corresponding Data packet. More details will be given in the following.
  - `Exclude` allows a consumer to choose whether to exclude list and/or ranges of name components from the responding Data packet.
  - `ChildSelector` expresses a preference for which of the matching Data within a given content store should be returned.
  - `MustBeFresh` means that the router should not answer with a Data packet from its content store whose *FreshnessPeriod* has expired. This value is set by the producer for each Data packet.
- **Nonce**: is a random number that uniquely identifies the Interest packet.
- **Guiders**:
  - `Scope` limits how far the Interest may propagate.
  - `InterestLifeTime` is the time remaining before the Interest expires.

Data Packet

The Data packet is composed of the following fields:

- **Name**: is a hierarchical name for NDN content, which contains a sequence of name components. It must be the same of the corresponding Interest packet. More details about the name format will be given in the following.

- **MetaInfo**:
  - `ContentType` could be *default* that is the actual data bits identified by the data name, *LINK* is a name that identifies the actual data content and, *KEY* is a public key.
  - `FreshnessPeriod` indicates how long a node should wait after the arrival of this data before marking it as stale.
  - `FinalBlockId` is equal to the last name component of the final block and indicates the final block in a sequence of fragments.
- **Content** is the data itself.
- **Signature** is composed of `SignatureInfo` and `SignatureValue`. The first is included in the signature computation and describes the signature, signature algorithm, and other information such as the *KeyLocator*. The second is excluded from signature computation and is the actual bits of the signature and other supporting information. The signature is characterized by the `SignatureType` that could be *DigestSha256*, *SignatureSha256WithRsa*, or *SignatureSha256WithEcdsa*, and by the `keyLocator`. The latter is essential for retrieving the public key used to sign.

3.2.2 Forwarding at an NDN node

The packet forwarding process follows the scheme presented in Figure 3.3.

Each NDN router has three data sets: the Pending Interest Table (PIT), the Forwarding Information Base (FIB), and the Content Store (CS). The PIT stores the name prefixes that correspond to the Interests that the node could not satisfy and it sends to some other nodes. Moreover, the node must keep track of the requesting face(s), i.e. the node(s) asking the content, to send downstream the returned data. The FIB registers the prefix and the corresponding faces list, namely all the places data might be, to forward Interest packets toward possible nodes with the corresponding Data. The Content Store contains as long as possible the names and the data seen before as a buffer memory. This represents a temporary cache of Data packets the node has received.

When a router receives an Interest, it does the longest match lookup on Interest content name and checks if there is a correspondence into its tables, i.e. CS, PIT and FIB. If the Content Store caches the Data packet, the node sends out the content on the same face and throws out the satisfied Interest. Else, if the match is in the PIT, the corresponding entry is updated adding

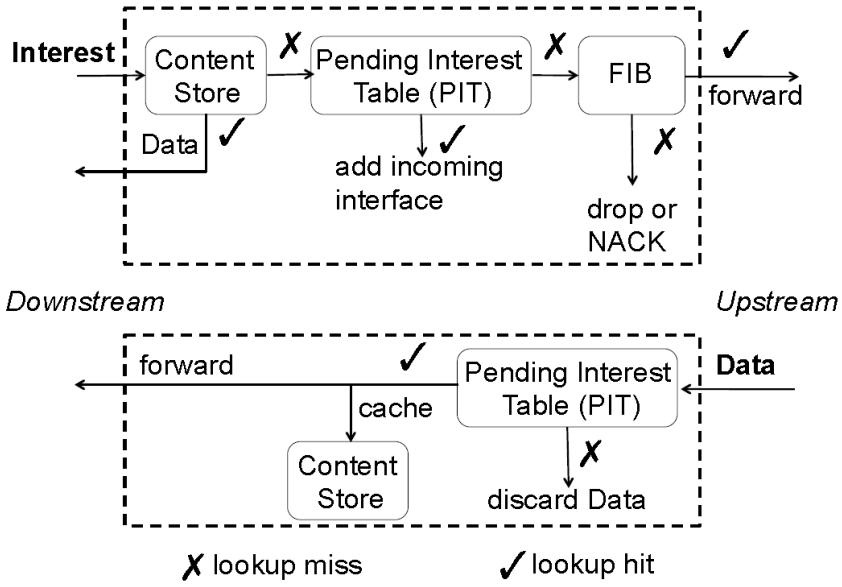

Figure 3.3: *Forwarding Process at an NDN node [56].*

the requesting face and the Interest is discarded. While, if the match is in the FIB, the Interest is sent out to the next hop faces and it is created a new entry in the PIT. Finally, if there is no match, the Interest is discarded because the node does not know how to find any matching data. The Data packet processing is quite similar, the router does a longest match lookup of Data packet name, if there is a match in the Content Store, the node throws it away because it is a copy. Otherwise, the node looks in the PIT and if there is a match, it sends the content to the requesting face(s) and adds it to the Content Store. A FIB match means an unrequested Data, so the node gets rid of the packet.

Names

Names have a central role in NDN. They are assumed to be hierarchically structured, as depicted in Figure 3.4. The name is composed of a number of components of arbitrary octets separated by "/" character. Names are opaque to the network: they have no meaning. This allows each application to choose the naming scheme that is more suitable for its needs.

Having a hierarchical structure helps applications to represent the context and relationships of data elements. As depicted in Figure 3.4, it is possible to define the temporal evolution of the content with the version

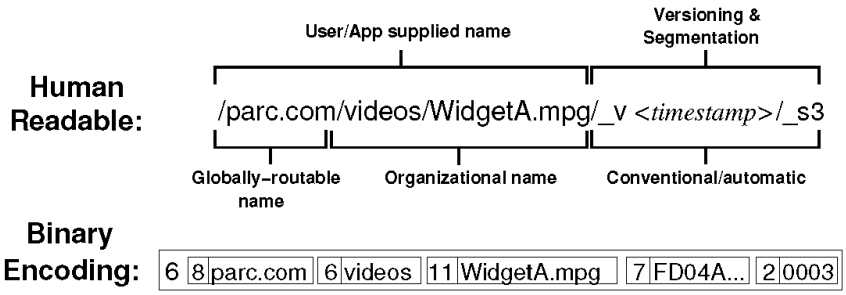

Figure 3.4: *Example of Data packet name [31].*

marker, `_v` and its segmentation with the segment marker `_s`. This structure provides several benefits: an explicit mapping between an application's data and its use in the network; a various number of possible names available to the developers; and a reduction of secondary notation. Moreover, in most cases the full name of a Data is not known so the requesting node specifies it relative (e.g. next or previous) to something whose name is known. This is possible using ordered tree of names.

3.2.3 Data-Centric Security

The data itself is secured thanks to a digital signature on every Data packet. A digital signature guarantees integrity, provenance and authenticity of the content. The content producer, P, is responsible for the digital signature over the content, C, and the corresponding name, N. Particularly, a content is made available in the network as $M_{N,C,P} = (N, C, Sign_P(N, C))$, where $Sign_P(N, C)$ is the producer's signature over the name and the content. The signature generation can follow one of the two forms: single blocks are individually signed using a standard public key algorithm, e.g. RSA with SHA256, or multiple blocks are signed together with an aggregated signature scheme, e.g. Merkle Hash Trees, [10]. A content consumer retrieves the content, C, using its name, N and it should be able to find the public key to use to verify $Sign_P(N, C)$.

As depicted in Figure 3.5, the NDN testbed root key signs the site's keys, which in turn sign the user's keys. Then, each user is responsible to sign and to maintain in a local repository the device and application keys. This model allows users to follow the trust chain from the leaf nodes (i.e. application and device) to the root key for verifying the validity of a key. In each Data packet, there is a "`KeyLocator/KeyName`" field, that one can use to fetch the key. Moreover, since the key itself is a Data packet, the key's

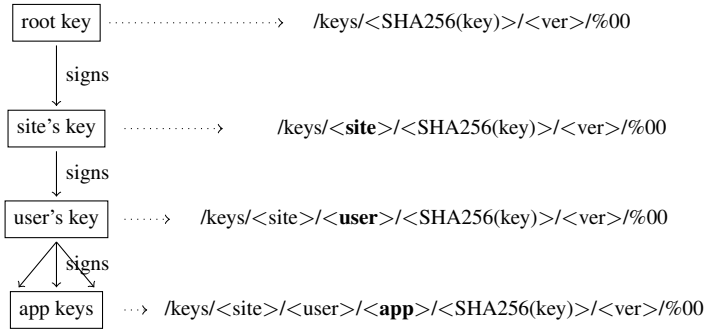

Figure 3.5: *Key model and naming in NDN.*

"`KeyLocator/KeyName`" field is used to reach a trusted anchor. The root key, that is "public knowledge" and self-signed, is used to verify the key needed to verify the Data packet. Furthermore, the Figure 3.5 shows the key name structure. Usually there is a common prefix "`/keys`", used to easily distinguish keys from contents, as the first part of the name; the middle part represents the path in the keys' subtree, namely the keys hierarchy in the network; the final part is the hash value of the corresponding public key. There could be another part of the name that carries the content version and segment, but it isn't mandatory.

3.2.4 Simplified Model of the NDN Protocol

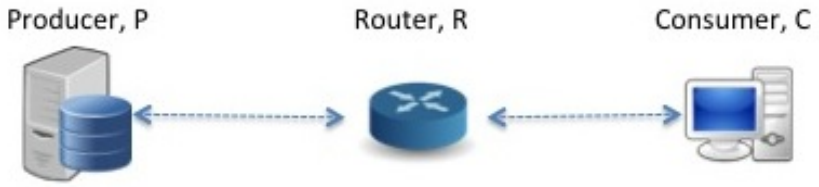

Figure 3.6: *The reference scenario.*

The NDN architecture comprises three different nodes, as shown in Figure 6.1:

- **Data Producer,** P: upon reception of an Interest packet, it answers with the corresponding Data packet. It signs a content by using its key.
- **Data Router,** R: upon reception of an Interest packet, it answers with

the corresponding Data packet, if it is present in its content store. Otherwise it forwards the request towards the correct Data Producer. Upon reception of a Data packet, it forwards it to the downstream Consumer. Moreover, it caches packet in its Content Store.

- **Data Consumer,** C: obtains data sending Interests with the desired data name.

Moreover, the architecture comprises a Trusted Authority (TA) that periodically updates the public/private key pairs. In the remainder of the thesis, we assume that the communication network is reliable and timely, i.e., no message can be lost due to communication delays or node malfunctioning.

The basic information centric protocol follows the request/response paradigm. A response is not given back if it is not received a request. The request message is called Interest, while the response Data Packet. Each Interest is uniquely identified by a Name and a Nonce. Then, the corresponding Data Packet must carry the same Name and the same Nonce. The Data Packet is always signed by its Producer following the RSA signature algorithm. From now on, an Interest packet is represented by `I(name)`, while a Data packet by $\texttt{D}_{signer}\texttt{(name)}$.

The simplified communication protocol consists of 7 phases:

1. **Setup**: the initial phase is performed only once to define the set of public parameters and to distribute them to the users.
2. **Key Gen**: this phase is performed time to time to generate the key pairs and to distribute them to the users. The Producer runs the key generation algorithm and gets (pk, sk). Then, it pushes to the TA the public key to be certified and keeps the private key secret. Finally, the Producer publishes off-line the public key pk into the nodes' repositories by means of the ccnx synchronization protocol.
3. **Create Data**: the Producer, P, produces and stores contents. Each content is represented by $m = \texttt{D}_P\texttt{(name\_m)}$ and the corresponding signature is $\sigma \leftarrow Sign_{sk_P}(H^s(m))$.
4. **Send Interest**: the Consumer, C, sends an Interest `I(name_m)` with the name of the data it wants to the next hops.

5a) **Forward Interest**: if the next hop, a Router R, does not have the content, it forwards the Interest `I(name_m)` to the next hop.

5b) **Send Data**: if the next hop, a Router R or the Producer P, has the content, it answers with the corresponding data $\texttt{D}_P\texttt{(name\_m)}$.

6) **Receive Data**: the Consumer, C, receives the data $\mathtt{D}_P\,(\mathtt{name\_m})$.

7) **Verify Data**: the Consumer, C, verifies the content: $Ver_{pk_P}(H^s(m), \sigma') \stackrel{?}{=} 1$.

The following Figure 3.7 shows the message exchange.

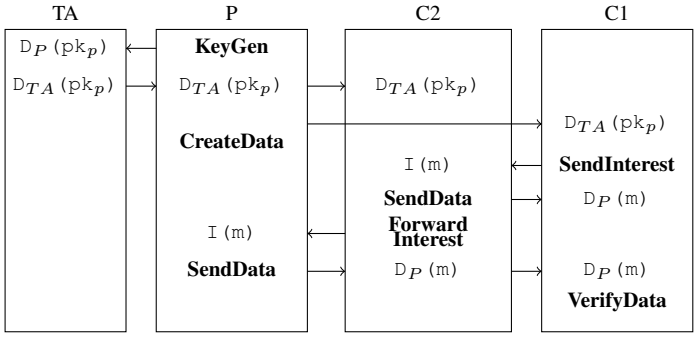

Figure 3.7: *The Communication Protocol.*

We suggest to verify all the packets at the end nodes to prevent cache pollution attacks or more sophisticated attacks. The standard verification procedure follows the protocol. However, the signature validation is a crucial problem. Each node needs to check the public key status, as described in the following.

The verification process should follow the next steps. After receiving a Data Packet, each end node in the network:

1. Checks the `KeyLocator` containing the signing key name or the key itself.
2. Checks if the key name is known and if the key is stored in the cache. If a match is found, the node checks the key status. If the key status was checked in the security window, then the Data packet can be verified.
3. Otherwise, the node expresses an Interest for the key, and waits till the reception and the validation of the key before verifying and accepting the content into its cache.

3.2.5 In-Network Storage

Each router in the network can cache the data packets in its Content Store for satisfying future requests. The Content Store is like a buffer memory in IP routers. However, NDN routers can reuse contents as long as they remain

Algorithm 3.1 Verification of Content Validity

```
Check the KeyLocator.
Check the key in the Content Store.
if a match is found then
   Check the key status.
   if Key status was checked in the window then
      Verify the Data packet.
      return
   end if
end if
Send an Interest for the key.
Wait for the key then check key and content.
```

in the Content Store. The most used caching policies are the Least Recency Used (LRU) and the Least Frequently Used (LFU), that evict the least recent or the least frequent requested content, respectively. Moreover, NDN supposes the presence of a more persistent and larger-volume in-network storage, the Repository, that is usually exploited for specific application needs. Caching named data raises new problems in terms of privacy. Indeed, naming and caching data reflect what users have requested, but it is harder to identify who has requested that contents.

3.2.6 CCNx Synchronization Protocol

Every CCN node has a repository where content objects are persistently stored in addition to the content store. The CCNx synchronization protocol allows repositories to be automatically up to date [39]. A set of contents whose prefix name is common, is called Collection. The contents in the collection are organized within a sync tree, that is built by the local Sync Agent. The agent computes an additive hash over that tree, the topmost hash is called root hash. Periodically, the agent sends Root Advise Interests to the neighboring nodes which synchronize their repositories comparing the root hash of the sync tree sent in the Interest with their own root hash. If they match, the collections are in synchronization. While if they do not, the collections are updated using the standard interest/data protocol and sending the different root hash between the nodes

CHAPTER 4

Related Work

In this Chapter we compare our proposed framework to the solutions already investigated in the context of the Information Centric Networking and we point out the main differences and innovative aspects with respect to them.

4.1 Trust Management and Cache Pollution

The content authenticity is a main security issue in a Information Centric Network. Since the first work on CCN, Smetters and Jacobson [51] rise the problem of content authentication. Their proposal is to authenticate the link between name and content using a digital signature scheme implemented by the content producer. The signature in each data packet is over the name, the content and other signed info. Each CCN data packet is publicly authenticable, anyone in the network can verify that the name-content binding was signed by a particular key. Then, it is the consumer that determines whether to trust the received content. The aim of this proposal is to provide evidence of content validity and origin. However, the authors do not inspect details about the key management and, also, the problem of key

revocation.

A certificate-based signature scheme that allows a signer to choose a certificate authority (CA) is presented by the paper [50]. In this work, the certificate acts as binding between the public key and its holder and as certificate authority's guarantee against partiality. The scheme is shown to be secure under three types of adversary, and also it is more practical and trustworthy then the standard schemes. The proposal to choose the CA is similar to our distributed protocol, where some trusted nodes are chosen by the end node to provide the key.

A recent work [28] suggests a trust management scheme for NDN. The paper provides the definition of the Interest-Key Binding (IKB) rule, that is also exploited along our work. The IKB rule binds the producer's public key with the consumer's interest. Our last two assumptions presented in Section 5.2.1 are complementary to the IKB rule. However, our work provides a solution for the management of key revocation that is left open in [28].

Furthermore, the paper [44] describes an architecture for video distribution based on ICN. It exploits Attribute-Based Encryption (ABE) to secure the content distribution. Finally, the authors focus on the user revocation problem. They propose to use a revocation authority that provides key-update for non-revoked users during a time window. This proposal is a good starting point for the definition of a revocation scheme but still needs to solve some challenges, such as content authentication.

The paper [36] suggests a Key Resolution Service (KRS) for CCN, that is, as far as we know, the only proposal relative to key management in ICN. This service allows to map a content name with the corresponding security information. The KRS is queried by the consumer node before sending the interest for a content. Thus, the consumer can obtain the public key certificate of a publisher or the content digest. This solution is a first practical attempt to mitigate content poisoning attack. The main drawback of the proposal is the presence of a local KRS server that could become a bottleneck. Moreover, the same paper presents some performance results relative to the average latency per request sent to the KRS server. The latency is measured as a function of the cache size and the number of KRS servers. Our work differs from [36] because we do not need a new network entity such as the KRS, we allow the consumer to choose its security window and also the keys are frequently refreshed. While, the authors of [41] present a platform used to obtain performance results about CRL and OCSP. The authors show the temporal behavior of CRL and OCSP in terms of the processing time. The results relative to CRL are shown to be around 1ms and

those relative to OCSP are between 25 and 30ms. These results are obtained over a standard IP network and end up in a small delay over the performance gathered with no certificate management. Our work extends those protocols for ICN showing that the resulting latency is comparable to the ICN setting with no key revocation management.

Furthermore, the paper [58] proposes a name-based trust and security mechanism. The idea is to use the identity of a user as public key by exploiting the identity-based cryptography (IBC). In this way, each user can verify authenticity and integrity by using the content name without following the certificate chain till the originator. The solution exploits the advantages of both PKI and IBC providing content integrity and trust in ICN networks. The paper gives an interesting overview on how to exploit an IBE scheme for key management but it does not consider the problem of key revocation.

If the problem of content integrity and trust is not efficiently solved, then the spreading of bad contents into the network is easier to implement. A lot of papers inspect and analyze the problem, also known as Denial of Service. Gasti et al. [26] present a first attempt to identify and mitigate DoS and DDoS in ICN. Particularly, the paper describes two types of attacks: interest flooding and content/cache poisoning. The first threat consists on sending a large number of interests requesting contents from the same set of producers; while the second aims to cause node to cache corrupted or false content objects, obstructing the retrieval of legitimate contents. The paper also discusses tentative countermeasures against the attacks but it does not evaluate their efficacy with a simulation. The paper represents a first step into the definition of content/cache poisoning in NDN scenario, but it does not consider the problem of key validity.

An interest flooding-based DDoS over NDN is analyzed also by papers [14] and [2], by considering that interests can be sent for non-existent contents and obviously fill up the victim's PIT. The authors in [14] propose proactive and reactive countermeasures, but then, they focus on reactive methods for detection of interest flooding via junk interests. Moreover, the authors suggest a mitigation technique, called Poseidon, which identifies traffic anomalies and keeps several statistics on expired interests. While, the authors in [2] define three mitigation methods with varying degree of implementation complexity against the attack. Particularly, they propose a token bucket approach and two satisfaction-based methods. Finally, both papers evaluate the benefits of the countermeasures through small-scale and large-scale simulations over real network topologies. However, the proposals cannot be exploited for the problem of content pollution because they are based on interest packets and, also, they do not consider the problem of

retrieval of key status.

Finally, a content poisoning attack in NDN is studied in [27]. In a content poisoning attack, the adversary injects junk content into the router caches. The authors suggest a content ranking algorithm for cached content to allow routers to distinguish between a fake or a valid content. When the consumer verifies the signature and detects a fake content, it sends an interest that excludes the received content. Thus, the router can assign a rank to each cached object that is updated when an interest for that content is received. The content with an highest rank is selected as response to an interest packet. The proposal is shown to be effective against content poisoning. However, the problem of key management still remains open and also the problem of spreading attacker-controlled contents in the network caches.

Thus, there is an urgent need for the definition of a scheme that opens the possibility to retrieve updated keys in the ICN scenario exploiting and adapting the well known solution for IP-based networks. This thesis inspects the problem of content poisoning and suggests three PKI-like solutions for the key management and, then, presents a distributed method that reflects the NDN distributed nature in Chapter 5. The thesis not only considers the problem of key management and consequently of content poisoning, but also analyzes the problem from another point of view. Indeed, we consider the attractiveness and the effectiveness of a cache pollution attack that aims to build an attacker-controlled Content Delivery Network. Moreover, our work highlights the incentive for the adversary in improving the caching ability of the access network and reducing the bandwidth requirement of the core network in Chapter 6.

4.2 User Privacy and Performance Tradeoff

There is a wide literature centered on the improvement of caching and prefetching and on the privacy challenges in the content centric scenario. However, only a few considers the trade-off between performance and privacy. The authors of [34] consider that there is a trade-off between privacy and performance. Indeed, using a countermeasure to preserve privacy obviously brings to reduced performance. The challenge is designing a countermeasure that provides users' privacy together with maximum performance. The paper analyzes the caching potential to improve network performance and takes into account the need for preserving users privacy-sensitive information. Moreover, it proposes a naïve countermeasure that detects and prevents a cache-based privacy attack. Then, the authors con-

sider two countermeasures: using selective caching or selective tunneling. However, the paper [34] considers caching policies based only on past communication and does not point out the improvement in performance. In this work, we compare three proactive algorithms and two reactive algorithms and we give results for the latency achieved by each one.

The proposal for an high performance caching scheme that also prevents cache pollution attacks is given in [54]. The authors have three principles for defining their scheme: (i) the scheme should be generic, suitable for CCN and today's IP Internet; (ii) the scheme should be effective with various cache replacement policies; (iii) the scheme should be simple and easy to be implemented. The proposal is CacheShield, an add-on proactive mechanism for the Content Store that can be used with any cache replacement algorithm and is compatible with existing attack detection techniques. CacheShield exploits a shielding function that distinguishes popular contents from unpopular ones. When there is a cache hit, CacheShield operates identically to a normal caching algorithm. While, when there is a cache miss, the shielding function decides whether to cache the new content or not based on a precomputed caching probability. If the content is not stored, the algorithm stores the content name or updates the number of requests relative to that name. The authors provide evidence of the robustness and the effectiveness of CacheShield using different replacement algorithms under different attack scenarios in several network topologies. However, the effectiveness is limited to the scenarios taken into consideration. Instead, the paper [15] illustrates the inefficacy of CacheShield against realistic adversaries. The authors also present a new lightweight detection method for cache pollution attacks. The mechanism is tested with simulations and provides accurate results in different topologies. Nonetheless, the paper does not suggest countermeasures against this kind of attacks.

Another solution for efficient caching is given in [12]. The authors propose WAVE, a chunk-based caching scheme that considers the content popularity and the inter-chunk relation. Moreover, WAVE is simple, can operate with any content routing scheme, and does not need a central server. Each router makes decisions individually, then an upstream router suggests the number of chunks to be stored to the downstream nodes by marking a chunk to be cached. The number of chunks to be cached is determined by the chunk marking window and exponentially increases according to the content popularity. WAVE distributes the chunks along the path from which the requests come in a hop-by-hop manner. Results, that take into account various caching policies, show that the hit ratio is improved and less frequent replacements are needed than the compared caching schemes. Both

previous papers consider the same Zipf-like popularity distribution for all the contents and they do not take into account problems relative to users privacy.

The paper [11] gives an overview of privacy challenges in Content-Oriented Networking. The authors distinguish different kinds of privacy: (i) caching privacy, as nodes cache content they can infer information about users interest; (ii) content privacy, any intermediate node can inspect content since is not encrypted; (iii) name privacy, if an attacker monitors the user's requests could infer sensitive information; (iv) signature privacy, the identity of the content signer is exposed by its signature on content. Then, the paper provides a detailed description about possible attacks on the privacy challenges described above. Relative to cache privacy, the paper describes timing attacks and protocol attacks. By measuring the response time, an attacker can decide whether a content has been retrieved from a particular router cache or not in a timing attack. While, a protocol attack exploits features and option of ICN interest packets to get sensitive information. Moreover, some potential countermeasures to defeat privacy threats are also suggested. However, they have drawbacks either in terms of caching performance or in preventing the possibility of having a user-based caching.

Cache privacy is also considered in [1]. The scenario presented is as follows. An adversary would like to determine if an user recently requested a content. If the user and the adversary share the first-hop router, it is possible to measure the RTT between the adversary and the router. Then, the adversary sends an interest for the content and measures again the RTT. Finally, it can compare the two RTTs and decide whether the content has been retrieved from the router's cache or not. The authors analyze timing and probing attacks and then they propose some countermeasures. First, they propose to mark content as private; however this method lowers the cache performance. Secondly, they describe some methods that guarantee a trade-off between performance and privacy based on random caching. Our work considers the trade-off between privacy and latency. However, it proposes also prefetching policies, a different adversary model and a different countermeasure.

4.3 ICN Vehicular Communication

Mobile ICNs have recently attracted much attention within the research community. The works [7], [52] and [60] provide comprehensive surveys on ongoing research in mobile ICN. Both papers highlight the benefits that

different ICN designs (e.g. DONA [32], CCN [31], NetInf [17], NDN [57]) provide to content and producer mobility. In details, the first work mainly focuses on open challenges that should be inspected, whereas the second explains how to exploit the named-data paradigm for making the mobility management easier. In particular, [7] scrutinizes the applicability of ICN paradigm in vehicular environments, by reviewing its core functionalities, such as named content retrieval, innate multicast support and in-network data caching. Then, [52] highlights the issues of producer mobility and dynamic routing. However, it also highlights the benefits in the management of user multihoming and handover. Moreover, [60] underlines the performance improvement due to in-network caching and the adaptability to various scenarios. Particularly, we think that the ICN paradigm naturally fits the requirements of a mobile network regardless the kind of network, i.e. VANET [29], MANET, Military Network [21] [20].

Furthermore, the paper [6] raises first the question whether CCN could be the solution for vehicular networks. The authors provide evidence that CCVN (Content Centric Vehicular Networking) performs better than the legacy TCP/IP-based architecture. Also this paper considers the advantages of using ICN for mobility support. The paper [35] proposes a mobility management scheme for CCN that confirms the previous suppositions. It evaluates the routing update latency and the delivery latency depending on the number of nodes. The results show that the mobile CCN scheme achieves better performance than the basic CCN protocol. Nonetheless, the work does not evaluate how to optimize content distribution in order to lower the perceived latency and to guarantee content retrievability.

Several papers present solutions to optimize the performance of ICN networks. The paper [9] analyzes the request flow in a Content Centric Networking (CCN) network by means of a Markovian process (MMRP). The authors compute the mean delivery time as a function of content popularity and size, network topology, content store size and interest rate. They also provide the failure probability in two different topologies. The paper highlights that the available bandwidth and the cache size are the most critical resources. In our work, we take into account both of them. Then, the authors of paper [13] suggests an optimization model that tries to minimize the energy consumption with the optimal content allocation. In our work, we consider the success probability in content retrieval but we do not care of energy efficiency. However, the authors consider a popularity distribution that follows the Zipf's law, as happens in most of the literature and also in our work. In addition, a proactive caching algorithm for NDN is proposed in [47]. The main idea is to proactively ask and cache contents

before the user moves from one access point to another. The simulations show that the proposed approach has better performance: lower handover cost, higher delivery ratio and shorter handover latency. The solution is implemented modifying the `INTEREST` packet and the basic communication protocol. Our solution does not require to add new functionalities or to modify the packets.

The problem of reducing the latency in content retrieval is well investigated by the research community in various contexts and a lot of papers opt for using pre-fetching. For example, the paper [43] tries to solve the problem of reducing latency, by predicting and prefetching those files that are most likely to be requested soon. The prefetching is done based on server advices to the client, who can choose to prefetch or not the suggested files. The results show improvements in the network latency perceived by users. However, the model makes probabilistic predictions for prefetching based on received requests. Our model is an optimization model for content predistribution based on a priori knowledge of the request probabilities. Moreover, a predictive prefetching method is also presented in [19]. The paper develops new handoffs and data transfer strategies in order to reduce the connection setup latency and the download latency. The results show that the performance of the vehicular WiFi network could be improved using the proposed strategies. However, the authors leave some open challenges, for example how to implement the estimation mechanism for prefetching. Our work does not consider the handoff strategy because the ICN context does not need to establish end-to-end connections. It considers a new optimization strategy for content prefetching.

Our work starts from the proposal of [30]. The authors analyze the optimal caching policy for file servers co-located with the APs in a WiFi-based content distribution community infrastructure. They formulate the content management problem as mixed integer programming with the aim of maximizing the file retrieval probability within a time interval. The model considers 100 content files whose popularity follows the Zipf's law with different skew parameters. Moreover, it uses 50 access points with a coverage of 250 meters. Our work stems from this work for the definition of the objective function and the storage capacity constraint. However, we consider additional constraints and we extend it for a multi-hop Named Data Networking (NDN) architecture.

CHAPTER 5

Up-to-date Key Retrieval for Information Centric Networking

In this Chapter we introduce how to retrieve up-to-date signing keys in the ICN scenario. In the usual public key infrastructure, the Certificate Revocation Lists (CRL) or the Online Certificate Status Protocol (OCSP) enable applications to obtain the revocation status of a certificate. However, the push-based distribution of Certificate Revocation Lists and the request/response paradigm of Online Certificate Status Protocol should be fit in the mechanism of named-data. We consider three possible approaches to distribute up-to-date keys in a similar way to CRL and OCSP. Then, we suggest a distributed method to retrieve the key that exploits the idea behind PERSPECTIVES, and that naturally fits the ICN scenario. Finally, we evaluate the number and size of exchanged messages of each solution, and then we compare the methods considering the perceived latency by the end nodes and the throughput on the network links.

[1]Part of the contents of this Chapter have appeared in: (i) Giulia Mauri, and Giacomo Verticale "Distributing Key Revocation Status in Named Data Networking", *EUNICE*, Aug 2013, (ii) Giulia Mauri, and Giacomo Verticale "Up-to-date Key Retrieval for Information Centric Networking", To appear in *Transactions on Emerging Telecommunications Technologies*

5.1 Introduction

Information Centric Networking (ICN) leverages in-network caching to provide efficient data distribution and better performance by replicating contents in multiple nodes to bring content nearer the users. Since contents are stored and replicated into node caches, the content validity must be assured end-to-end. Each content object carries a digital signature to provide a proof of its integrity, authenticity, and provenance. However, the use of digital signatures requires a key management infrastructure to manage the key life cycle. To perform a proper signature verification, a node needs to know whether the signing key is valid or it has been revoked.

In NDN, the content objects are divided into chunks, each digitally signed by its producer. Otherwise, the content chunks are organized together with their digest into a Manifest, which is signed by the producer. While each node could verify the signature before caching objects, most papers assume that verification is made only by the content consumer. Indeed, in order to perform the signature verification, a node needs the signer key, which can be easily retrieved by issuing a standard interest message. However, information about the key validity status is also necessary. In fact, a content signed with a compromised key may remain in cache for an indeterminate amount of time, and possibly be served to the end users. Even if caches implement a freshness mechanism that deletes a content that has been in the cache longer than a given threshold, a compromised node could resend data making extremely difficult to remove from the network the objects signed with a compromised key, resulting in a denial of service and paving the way for more sophisticated attacks.

The data object authentication is one of the research challenges presented in the IETF draft [33]. Indeed, there is an urgent need to define and support a mechanism to distribute updated publisher's public keys to the consumers of data objects. In the standard PKIX (Public Key Infrastructure Certificate X.509), the issue of delivering key revocation status to the end nodes is solved by using the CRL (Certificate Revocation Lists) [16] or the OCSP (Online Certificate Status Protocol) protocol [42]. We present a naive solution based on the ccnx-repository synchronization protocol that implements the same functionalities of CRL. Then, we suggest two reactive methods that are adaptation of the OCSP to the Information Centric Networking framework. In particular, the nonce-based scheme always retrieves the original key from the producer, and the timestamp-based method exploits timestamps over the keys to guarantee freshness.

Finally, we consider a distributed method like PERSPECTIVES [53] that

is an alternative to the traditional PKIX for authenticating the public keys. Indeed, we propose how to get up-to-date keys retrieving them from the nearest nodes in an NDN-friendly way.

The remainder of the chapter is structured as follows: Section 5.2 presents the network scenario and the attacker model together with the security definition. Section 5.3 proposes the traditional methods to distribute valid key and shows our novel scheme. Finally, Section 5.4 describes the evaluation scenario, and gives the performance results before the conclusions that are left for the last Section 5.5.

5.2 The Up-to-Date Key Retrieval Security Problem

5.2.1 Assumptions

1. Each Data packet contains a `KeyLocator` field containing the name of a public key, which can be fetched with the standard ICN mechanism.

2. A public key is valid if it is included in a certificate issued by a Trusted Authority and if there is a proof that it has not been revoked. A vulnerability period of duration W is acceptable.

3. The owner of a valid key is honest and only signs the contents that it is authorized to sign. This chapter does not discuss how to scope signatures or enforce name-key binding rules.

4. All the contents signed with an invalid key must be dropped by the Consumer, even if they were signed when the key was valid. This chapter does not discuss how to remove stale content from the router caches.

5.2.2 Attack Scenario and Security Definition

Our attacker model assumes an active, dishonest node, which can inject any content into the node caches. In particular, our attacker:

1. can obtain any valid content produced and signed in the network.

2. can insert any content of its choice in any Router content store.

3. can obtain any Producers' private key. In this case, the Producer immediately revokes the stolen key.

4. cannot obtain more than a single key in an interval W.

5. cannot break any cryptographic algorithm.

According to the attacker previously described, we provide our security definition:

Definition The key retrieval scheme is secure if no uncompromised Consumer accepts as valid a content signed with an invalid key, except in the case that less than a time W has passed since the key was revoked.

5.3 Key Retrieval Schemes

This Section reviews and provides additional details about the three key retrieval schemes that have been firstly presented in [37]. Then, we describe the proposal of this work that is a distributed protocol that overcomes the drawbacks of the three previous schemes.

The first protocol supposes to create a list with valid keys that are updated and signed by the Trusted Authority (TA), and then distributed to the network nodes using the ccnx synchronization protocol.

Moreover, we recall the two reactive protocols: the nonce-based and the timestamp-based schemes. Indeed, a user sends an Interest for a key to the Producer, and the latter answers with a Data packet containing the key and its signature. The main difference is the validity window that it is very small for the nonce based protocol and a chosen value from the users in the timestamp based protocol.

While, the fourth protocol exploits some trusted nodes that are responsible for providing a specific key, when they receive an Interest addressed to themselves. This solution easily fits the distributed nature of the Information Centric Networking paradigm.

The original ICN communication protocol does not need pervasive modification, the communication follows the standard Interest/Data packet exchange. While the Interest and Data packets need some changes. In particular, the Consumer sends an Interest for a key specifying the requirements for that key, as detailed in the following. Then, Producer or Routers answer with the corresponding key following the required criteria.

5.3.1 Protocol 1: Proactive method

Protocol P1 periodically distributes up-to-date keys to consumer nodes. Such predistribution can leverage either the proposed CCNx synchronization protocol [39] or, alternatively, the ChronoSync protocol [59].

The Trusted Authority and each Consumer keep a repository holding the public keys. Upon every key update, the TA pushes modifications to all the Consumers.

In order to enable key retrieval, the Trusted Authority defines a *key collection* where the Producers' public keys are listed.

The keys in the collection are then organized within a tree and the TA computes a *root hash* over that tree.

Whenever some change happens, a ROOTADVISE message with the root hash is sent to all the consumer nodes. The Consumer compares the root hash of its repository with the received root hash. If the hashes are equal, the repositories are up-to-date; otherwise the Consumer expresses an Interest to request the keys that have been modified.

Figure 5.1 shows the synchronization process. Notice that after the Consumer has compared the hashes, it sends Interests for the keys that are not up-to-date.

The main advantage of this solution is that the key repositories are kept synchronized and, therefore, no key retrieval is necessary when data arrives. On the other hand, the main disadvantage is that Consumers must keep a large number of keys for Producers even if they are not interested in their content. Additionally, as the number of keys grows, the update messages are more and more frequent resulting in a significant overhead. Additionally, the synchronization procedure must be repeated for each Consumer, potentially violating the security window for some other Consumers.

This key retrieval scheme is secure for whichever security window W because the key repositories are synchronized every time there is a key modification or revocation. Except of cases where the synchronization procedure incurs in significant delays.

5.3.2 Protocol 2: Nonce-based

Protocol P2 guarantees the up-to-date status of the key ensuring that the key is sent directly by the Trusted Authority.

Whenever a new Data packet arrives, the Consumer checks whether the corresponding key is available and executes Algorithm 3.1. If no valid key is found, an Interest for the key is sent by the Consumer node. The Interest must contain the "do not answer from content store" option and a nonce. Thus, the TA answers with a Data packet containing the root key and the same nonce.

Particularly, the Consumer node sends an Interest specifying the following fields: `name=key/pk_`P, `selector` equal to `answer origin`

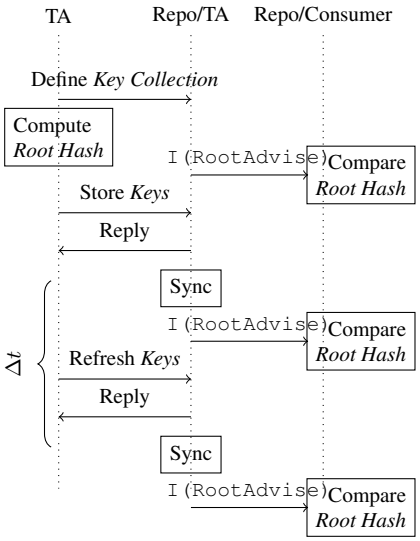

Figure 5.1: *Repository Synchronization. Note that the* `RootAdvise` *message is a special Interest message.*

`kind`, meaning "do not answer from content store", and a unique `nonce`, necessary for guaranteeing the uniqueness of the response. Note that `key/pk_`P is the name of the Producer's public key.

As soon as the Consumer receives the Data packet containing the key, it verifies that the message is signed by the TA and that the message includes the unique nonce, which is also part of the authenticated data. Then, the Consumer verifies the original message and stores the key in the Content Store. Algorithm 5.1 describes the operations performed by the Consumer.

Algorithm 5.1 Consumer in Protocol P2

Send `I(key/pk_`P`)`. This message includes a random nonce, n.
Wait until `D_`P`(key/pk_`P`)` is received.
Check that `D_`P`(key/pk_`P`)` includes n.
Check that `D_`P`(key/pk_`P`)` has been signed by a TA and the signature is valid.
Store the key in the CS.
Use the key to check the signature of pending messages.

It is worth noting that, by virtue of the `answer origin kind` option, the Router nodes only forward the Interest packets to the following nodes toward the TA.

The latter answers with a Data packet containing the `name=key/pk_`P, the `signed info` that are the same nonce sent in the Interest packet and the Publisher Public Key Digest that identifies the Producer, and the con-

tent that is the public key needed to verify the original packet. These data are signed by the TA.

Then, the Data packet is forwarded from the Trusted Authority to the Consumer.

The following Algorithm 5.2 describes the operations performed by the TA.

Algorithm 5.2 Trusted Authority in Protocol P2

Receive `I(key/pk_`P`)` with nonce n.
Put n in `D_`TA`(key/pk_`P`)`.
Forward `D_`TA`(key/pk_`P`)` to the Consumer.

This key retrieval scheme is secure if the security window respects the following condition: $W > RTT$, i.e. it should be bigger than the round trip time, RTT, between the Consumer and the Trusted Authority. Note that the tightest security window can be achieved only with the exact knowledge of the RTT. However, W can be chosen to be larger than the maximum network RTT. On the one hand, this protocol can guarantee security with very small windows, W, since the key is guaranteed to be fresh after each execution of the protocol. On the other hand, the TA can become a bottleneck.

5.3.3 Protocol 3: Timestamp-based

In Protocol P3, each incoming key is signed by the TA along with a timestamp. When the Consumer needs a key, it checks in the Content Store. If a matching key with timestamp T_0 is found, then the Consumer checks whether $T_0 + W$ falls later than the current time, in which case, the key is considered valid. Otherwise, the Consumer sends an Interest for the key to all its neighbors specifying in the name a timestamp, TS, that indicates a threshold validity.

The Consumer sends an Interest packet with `name=key/pk_`P/TS. A Router, having in its Content Store the key `key/pk_`P/T_r with $T_r > TS$, can answer with the corresponding Data packet.

Otherwise, the node forwards the Interest to the following node up to, possibly, the TA. The TA is assumed to generate key messages on-the-fly with the current time. Notice that system-wide clock synchronization is necessary for the correctness of the protocol.

This key retrieval scheme is secure if the security window is bigger than the chosen timestamp $W > TS$. The Consumer can choose any value for TS which is smaller than the current time and larger than the current time minus W. A small value will result in a quicker response from the network,

but also in more frequent key expirations. A larger value will result in a higher latency in obtaining the key, but in longer key durations.

5.3.4 Protocol 4: Distributed method

Protocol P4 overcomes the scalability limitations of the previous protocols by leveraging on specially trusted nodes, called Notaries, to provide keys on behalf of the TA. These Notaries are chosen by each Consumer and located in diverse network locations near the end users.

When the Consumer needs a valid key, it sends an Interest packet to a set of Notaries of size N_S asking for the content with `name=notary/`N_i`/key/pk_`P. Where N_i is the notary's name. This Interest can only be satisfied by the Notary as in Protocol P2. This protocol requires to add a FIB entry for each Notary in all the network nodes, so that they can route packets to *named notaries*. The other part of the name specifies that the content will be a key, `/key`, and in particular, the public key of Producer P, `pk_`P.

In order to increase the trust on the key validity, the Consumer can request the key to more than one notary, e.g. N_S, and wait for the answer of N_T notaries, where $N_T \leq N_S$. Moreover, we assume that the notaries are neighbor nodes and that could be part of different network areas in order to guarantee partition tolerance. If fewer than N_T of notaries answer to the Interest within a time interval Δt, the Consumer requests the key to the TA using the nonce-based method.

It is worth noting that the notaries sign the keys with their keys, which, therefore must be available at the Consumer or can be retrieved with one of the other protocols.

Algorithm 5.3 describes the operations performed by the Consumer.

Algorithm 5.3 Consumer in Protocol P4

Send `I(notary/`N_i`/key/pk_`P`)` to the chosen notaries.
Wait Δt.
if At least N_T `D_`N_i`(notary/`N_i`/key/pk_`P`)` packets are received **then**
  Check that the key signatures are valid.
  Store the key in the CS.
else {Δt expires}
  Retrieve `/key/pk_`P using the nonce-based method (Protocol 2).
end if

Notary nodes store a *key entry* for each Producer, P. The entry is a special content whose name is `/key/pk_`P, and it is composed of the key itself and the corresponding lifetime or the label *revoked*. The lifetime is generated by the key owner when the key is first used, and timely updated.

When the key's lifetime expires or the key becomes revoked, the key is marked as stale and is automatically dropped out of the cache.

The Notary can answer to the requesting Consumer with two messages: (i) the *key entry*, meaning that the Notary knows the key and it is not expired; (ii) the *key entry* with the label *revoked*, meaning that it is no longer valid to sign and must be dropped. The Data packet has `name=notary/`N_i`/key/pk_`P. The signed info is the Publisher Public Key Digest that identifies the Notary that has signed the content and the content is the *key entry* or the *key entry* with the label *revoked*. The packet is signed with the Notary key.

The following Algorithm 5.4 describes the operations performed by the Notary.

Algorithm 5.4 Notary N_i in Protocol P4

```
Receive I(notary/N_i/key/pk_P).
Remove name prefix and check Content Store.
if D(/key/pk_P) is found then
  if key lifetime != 0 && key is not revoked then
    send D_N_I(notary/N/key/pk_P) to the Consumer.
  else
    if key is revoked then
      send D_N_i(notary/N/key/pk_P) with label revoked
    end if
  end if
end if
```

This solution allows the Consumer to choose where to anchor its trust. Moreover, we do not create a bottleneck in the Trusted Authority, since each Notary can sign and forward the key entry. We only involve the TA when there is a new key request or when the key expires. Also, the key lifetime checking does not require a coordination between the nodes clock.

This key retrieval scheme is secure if the security window is $W > \Delta t + RTT$, i.e. it should be bigger than the waiting time window plus the round trip time between the Consumer and the TA if the Notaries do not answer within Δt. Usually, the security window, W, coincides with the key lifetime that is assigned to the key and signed by the TA, when the key is created.

5.4 Performance Evaluation

In this section we evaluate the number of exchanged messages as a function of the system parameters $|P|$, $|R|$, $|C|$, and $|N_T|$ of the protocol presented

in Section 3.2.4.

First, it is useful to discuss the possible size of the packets. As stated in [3], the minimum size of the Interest packet is 14 byte, let us call it mS_I. While, the maximum size is 196619 byte, let us represent it as MS_I. Thus, the minimum size of the Data packet is 6 byte, mS_D, while the maximum size is 196611 byte, MS_D. Then, we consider the frequency of requests λ, measured as the number of sent interests per second. Moreover, we define the miss probability at router node r, $p^r_{miss} = \sum_{r \in Path_{C \to r}} (1 - p^r_{hit})$, where p^r_{hit} is the hit probability at router node r, while $r \in Path_{C \to r}$ comprises all the router nodes on the path between the Consumer C and the Router r. Then, we define the stale probability, p^r_{stale}, that is the probability that the key timestamp stored in the Router r is expired.

5.4.1 Number and Size of Messages

During the **Key Gen** phase, the p-th Producer generates his key pair and pushes the public key to the Trusted Authority, that receives $|P|$ data packets. Then, the TA certifies the public keys and sends them back to the corresponding p-th Producer. Finally, the Trusted Authority pushes all the Producers' public keys to all the $|C|$ Consumers.

The **Send Interest** phase involves only the Consumers. The c-th Consumer sends λ Interest packets per seconds to the neighbor Routers.

During the **Forward Interest** phase, each r-th Router receives $\frac{\lambda|C|}{|R|} p^{r-1}_{miss}$ Interests and forwards to the next hops $\frac{\lambda|C|}{|R|} p^{r}_{miss}$ Interest packets.

The **Send Data** phase can include a Producer or a Router. The Router node receives $\frac{\lambda|C|}{|R|} p^{r-1}_{miss}$ Interest packets and answers with $\frac{\lambda|C|}{|R|} p^{r-1}_{miss}$ Data packets. While, the Producer node receives $\lambda|C|p^r_{miss}$ and answers with the same number of Data packets.

The c-th Consumer receives λ Data packets from the neighbor nodes during the **Receive Data** phase.

Finally, each Consumer verifies the Data packets received and checks the key validity in the **Verify Data** phase. The number of messages is different relative to the freshness protocol chosen. Starting from Protocol 1, for each key update, the TA pushes into nodes repositories the Producers' public key. Thus, it sends $|C||P|$ Data packets and each Consumer receives $|P|$ Data packets containing the Producer's public key.

Using the nonce-based method, P2, the c-th Consumer sends an Interest for the public key for each Producer. The r-th Router receives $|C||P|/|R|$ Interests and forwards them to the next hop until they reach the TA. The TA

answers with $|C||P|$ Data packets that are forwarded by the Routers till the c-th Consumer.

By choosing the timestamp-based Protocol 3, the c-th Consumer sends an Interest for each Producer's key specifying the time threshold that the key should not exceed. Each r Router receives $\frac{|C||P|}{|R|}p_{stale}^{r-1}$ Interests and checks their timestamp. If the keys are not stale, the Router answers with the corresponding $\frac{|C||P|}{|R|}p_{stale}^{r-1}$ Data packets. Otherwise, the Router forwards $|C||P|/|R|p_{stale}^{r}$ Interests to the next hop. The Interests can reach the TA, that sends on the reverse path the corresponding keys. Finally, the $|P|$ keys are received by the c-th Consumer.

Concluding with Protocol 4, the c-th Consumer sends $|N_T|$ Interests for each Producer's key to the chosen Notaries. Each of the chosen n-th Notary responds with the corresponding Data packet. If the Notary does not have the key or the key is stale, the Consumer sends an Interest using the Protocol 2.

We now evaluate the size of messages: Interest and Data packets. We follow the specifications on [3] for our analysis. Our Interest packets can have two names: `/data/Id_P/seq_no` or `/key/Id_P/seq_no`. Thus, the Interest name has different sizes depending on the packet type: $S_I($`/data/Id_P/seq_no`$) = mS_I + L(PPKD) + L(name) = 286$ byte or $S_I($`/key/Id_P/seq_no`$) = mS_I + L(PPKD) + L(name) = 280$ byte, where $L(PPKD) = 256$ byte, and $L(name)$ depends on the name length. Moreover, when the nonce-based method is used, the *Selector* field should be used for specifying the `AnswerOriginKind` that requires 3 byte more. While, when the timestamp-based method is used, the name comprises also the timestamp that requires 22 byte. If we use the Protocol 4, we need to add `/notary/Id_N` to the name and it is 11 byte more.

We can state the same about the Data packet names, i.e. the length depends on the protocol chosen. Usually, the content size changes depending on the contents, however, we assume that all the contents (i.e. data and keys) have the same size of 128 byte. Thus, the Data packet size is $S_D($`/data/Id_P/seq_no`$) = mS_D + L(sign) + L(content) + L(name) = 676$ byte or $S_D($`/key/Id_P/seq_no`$) = mS_D + L(sign) + L(content) + L(name) = 670$ byte, where $L(sign) = 260$ byte, $L(content) = 396$ byte, and $L(name)$ depends on the name length.

Table 5.1 compares the number of messages received and sent by each entity and reports the corresponding message sizes.

5.4.2 Assessment Scenario

To evaluate the impact of key retrieval on NDN nodes, we conduct simulations using the open-source *ndnSIM* package [4], which implements NDN protocol stack for NS-3 network simulator. We run simulations for two network topologies: i) a smaller tree topology, and ii) a larger mesh topology. The nodes' Content Stores use the Least Recently Used (LRU) cache replacement policy, the link between each pair of nodes is bidirectional and has a capacity of 1 Gbit/s and a latency of 5 ms.

The tree topology, in Figure 8.2, comprises 8 leaf nodes, the Consumers, 6 Router nodes that are the transit nodes organized into two levels and one root node, the Producer.

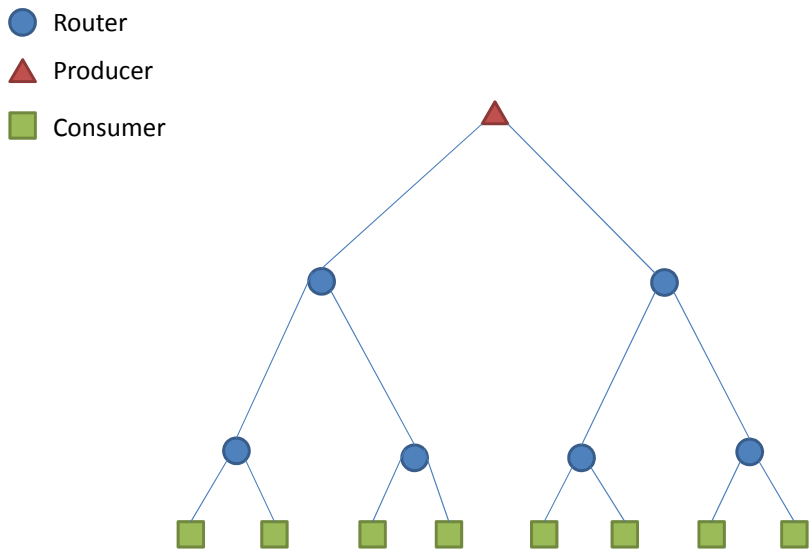

Figure 5.2: *Tree topology*

The mesh topology, in Figure 5.3, comprises 13 leaf nodes, the Consumers, 13 Router nodes and 13 root nodes, the Producers.

The Consumer nodes alternates On and Off periods following an exponential distribution. During On period, the Consumer requests content with a frequency f_c interests/second, with an exponential distribution with mean $1/f_c$ for inter-interests gap.

The contents of each Producer are organized into $C = 400$ popularity classes, each one with 34500 objects. Each content, c, has a probability of being requested, p_c, that follows the Zipf-Mandelbrot law: the more the content is popular, the higher the probability of being requested, hence

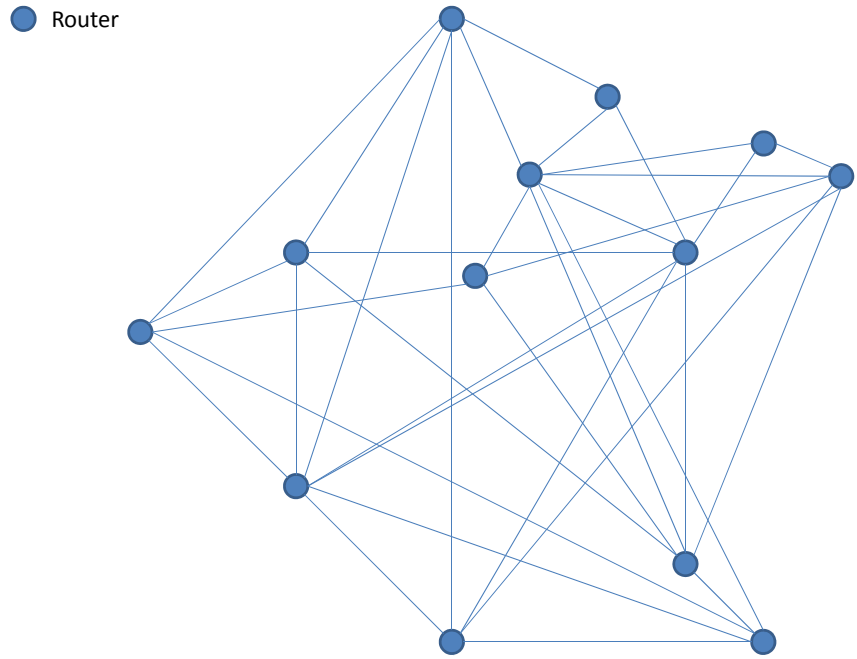

Figure 5.3: *Mesh topology. Each Router has a link with a Consumer and a Producer, not shown in the picture.*

$p_c = K(c+q)^{-\alpha}$, where $K = 1/\sum_{i=1}^{C}(c+q)^{-\alpha}$ and $\alpha = 0.8$ is the slope of the distribution. Since we put $q = 0$, the distribution becomes the Zipf law.

We assume that the number of Producers is equal to the number of Consumers. Thus, eight Producer applications are co-located in the root node in the tree topology. While, the Producer are distributed in the network as depicted in Figure 5.3 for the mesh topology. We also assume that the TA responsible for signing a Producer key is co-located with the Producer.

All the Routers and Consumers have a Content Store that allows up to 207000 entries to be cached and respects the content freshness. The simulations last 200 s. All the results are averaged over 10 simulations achieving a confidence interval of 95% or higher that yields a precision better than 1% .

We fixed unlimited freshness for "`/data`" packets and limited freshness for "`/key`" packets. Further, to distinguish between the nonce-based, the timestamp-based and the proactive methods, we assume the following:

1. in the proactive mode (P1), the Consumers always have the necessary keys;

2. in the nonce-based protocol (P2), the routers do not store keys;
3. in the timestamp-based protocol (P3), the validity threshold is chosen 10 seconds before the current time;
4. in the distributed method (P4), the key validity is 10 seconds, and routers do not store keys;
5. in P2, P3, and P4, the TA signing Producer P keys is co-located with the Producer itself;
6. in P4, each Consumer chooses the three nearest Consumers as notaries and waits for the first answer.

5.4.3 Numerical Results

In this Section, we compare the performance of the protocols discussed in this chapter. In particular, we report data concerning the latency depending on volume of requests, and the throughput on network links by distinguishing between tree and mesh topology.

Figures 5.4 and 5.5 show the average throughput on the input link of the Consumer nodes for the different key retrieval methods for the two topologies. As can be noted in Figure 5.4, in the tree topology the throughput on the link between the first level of Routers and the Consumer nodes grows with the frequency of requests and reaches saturation at about 250 Mbit/s, which is much less than the channel capacity. In this topology, the link at the Producer node is the bottleneck. The impact of the key retrieval protocol is negligible on the total traffic, and thus the different protocols show similar throughput figures.

A similar trend is depicted in Figure 5.5, which is relative to the mesh topology. Differently from the tree topology, the is no bottleneck and the throughput apporaches the channel capacity. As in the tree topology, the impact of the key retrieval protocol is negligible.

Figures 5.6 and 5.7 show the mean latency for content and key retrieval averaged over all the popularity classes as a function of the frequency of requests, i.e. $f_c = 2123, 9200, 18400, 27600$ interests/second, and depending on the network topology. In the tree topology, Figure 5.6, the lowest latency is obtained with P1, which grows slowly and stays under 0.3 s even with $f_c = 27600$. Protocol P3 and P4 show similar latency as P1 up to $f_c = 18400$, when their delay starts growing, reaching about 1 s for $f_c = 27600$. Finally, protocol P2 latency is comparable to the other methods only up to $f_c = 9200$, then the latency starts growing quickly, showing that P2 performance suffers from the congestion at the root node.

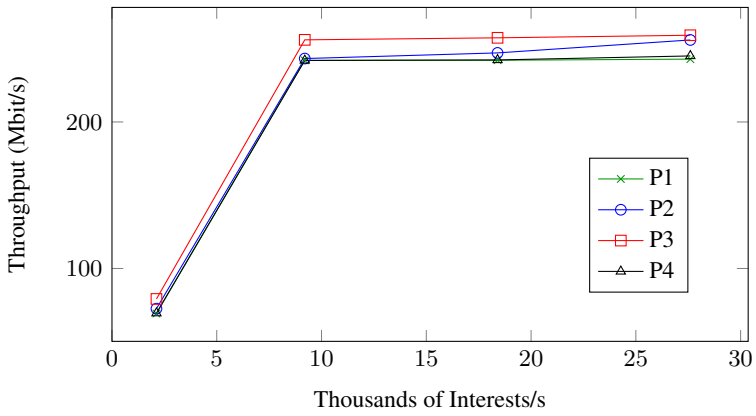

Figure 5.4: *Mean volume of Data packet received by the Consumer nodes in the tree topology.*

We have not taken into account a congestion control mechanism. Thus, in Figure 5.6, the values of latency relative to the biggest volumes of requests grow exponentially due to the congestion of the links. Before the channel saturation, in the left-hand side of the Figure, the additional latency of reactive, P2 and P3, and distributed protocols, P4, is negligible with respect to the proactive protocol, P1. The proactive protocol can be seen as the lower bound for the considered topologies. Indeed, since the keys are updated out-band, the depicted values represent the latency only in content retrieval.

The results relative to the mesh topology are presented in Figure 5.7. The trend is similar to the tree topology, but with some interesting differences. First, the average latency is lower for the same f_c, because the traffic can flow over multiple routes and the producer link stops being a bottleneck. Second, the performance of the distributed protocol (P4) is worse than P3 and halfway between P2 and P3, whereas in the tree topology the performance of P3 and P4 are similar. In fact, in the tree topology, P4 messages flow through the peripheral links, which are not congested. Instead, in the mesh topology, congestion may occur at any link and, thus, involve P4 messages. Additionally, P4 sends two or more interests for each requested key, thus increasing the traffic when compared to P3.

The same observations drawn for the tree topology can be written here. The results relative to P2, P3 and P4 add a negligible latency to the lower bound, P1, before the channel saturation. In this case, the saturation is asymptotically reached with a bigger volume of requests than the tree topology. Thus, we can say that, in both scenarios, our proposed solutions guar-

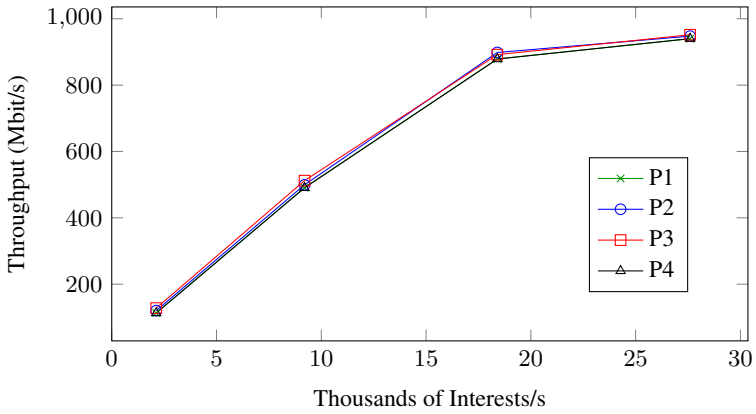

Figure 5.5: *Mean volume of Data packet received by the Consumer nodes in the mesh topology.*

antee the benefits of NDN in terms of small latencies in content delivery.

Figure 5.8 depicts the standard deviation relative to the mean latency for content and key retrieval averaged over all the popularity classes as a function of the frequency of requests, i.e. $f_c = 2123, 9200, 18400, 27600$ interests/second, and relative to the mesh topology. The figure shows that the standard deviation results are clustered closely around the values of the mean latency. Thus, the curves in Figure 5.8 have the same trend of those in Figure 5.7.

All our proposed solutions prevent nodes accepting a corrupted packet in their cache, except in the case that less than a time W has passed since the key was revoked. However, observing the results, there are some differences between the presented protocols. We notice that the timestamp-based method is the most flexible because it allows the user to choose the time threshold of validity. Moreover, this solution achieves the smallest latency and a good compromise between latency and throughput. However, we have to pay the price of clock synchronization of all the network nodes. While, the nonce-based method always guarantees keys' authenticity at the price of higher latency and of possible bottlenecks on the Trusted Authority node(s). Further, we believe that the proactive mode can be used when the channel capacity is limited and the network is overloaded, since it allows to manage the key retrieval off-line. Finally, the distributed method is the best solution not only for a good trade off between the evaluated performance parameters but also because it leaves the users the possibility to choose where to put their trust. Moreover, this method is more NDN-friendly and is perfect for distributed scenarios.

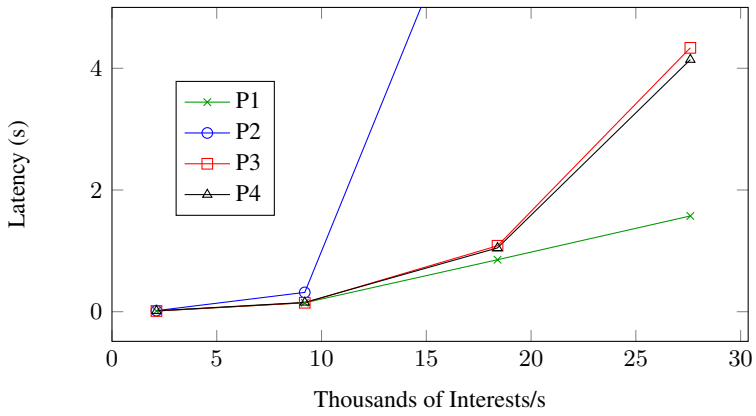

Figure 5.6: *Mean latency of all the popularity classes depending on volume of requests in the tree topology considering the alternative protocols. Precision better than 1% with confidence 95%.*

5.5 Conclusion

To our knowledge, this Chapter is the first one that deals with the problem of content freshness and revocation in ICN scenario. In particular, it compares different centralized and distributed approaches to distribute up-to-date keys in a NDN-friendly way. We provide the ICN framework with a trust management infrastructure. In particular, we present a proactive method that periodically distributes updated keys to the nodes, two reactive protocols that allow the TA or an intermediate node to send the up-to-date status of the key upon request, and a distributed method where some trusted nodes provide keys on behalf of the TA. Our results show that, even if the communication model undergoes a change, it is possible to maintain the benefits of an ICN network in terms of latency. Moreover, the results provide evidence that a distributed scheme is suitable for a network where the access network has spare capacities. While, if the access network is overloaded is better to choose a centralized protocol. However, in the latter case, the Trusted Authority can become a bottleneck. To overcome this issue, a key delegation scheme could be defined to allow a TA to entrust some other network entities to certify keys. This is a possible direction for an evolution of our work.

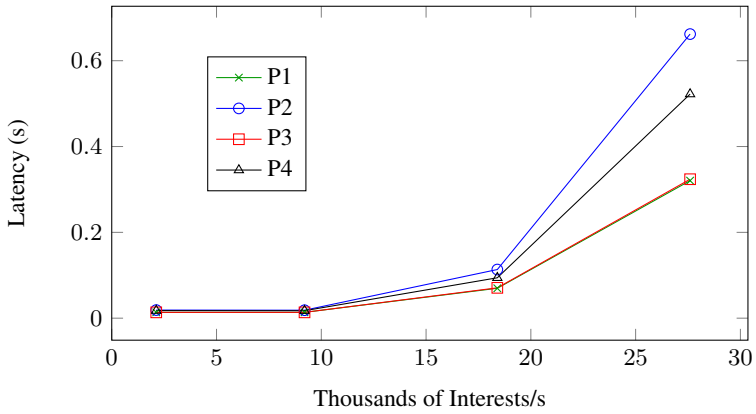

Figure 5.7: *Mean latency of all the popularity classes depending on the volume of requests in the mesh topology considering the alternative protocols. Confidence 95%.*

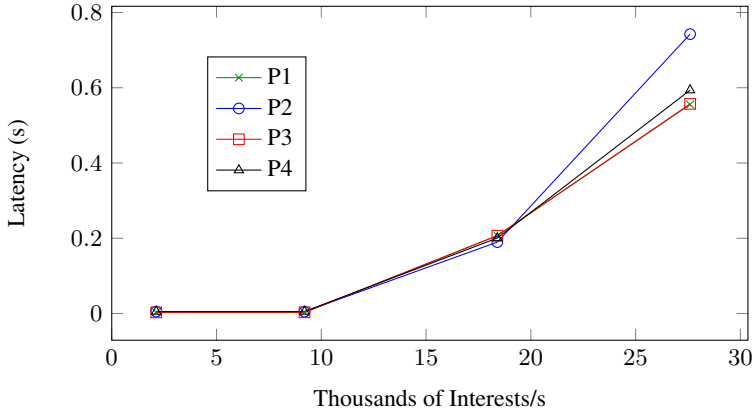

Figure 5.8: *Standard deviation relative to the mean latency of all the popularity classes depending on the volume of requests in the mesh topology considering the alternative protocols.*

Table 5.1: *Message Exchange*

Phase	Input Messages	Output Messages	Size In (byte)	Size Out (byte)
Trusted Authority				
Key Gen	$\|P\|$	$\|P\| + \|C\|\|P\|$	670	670
Verify P1	-	$\|C\|\|P\|$	-	670
Verify P2	$\|C\|\|P\|$	$\|C\|\|P\|$	283	670
Verify P3	$\|C\|\|P\|p_{stale}^{r-1}$	$\|C\|\|P\|p_{stale}^{r-1}$	303	692
Producer				
Key Gen	1	1	670	670
Send Data	$\lambda\|C\|p_{miss}^{r}$	$\lambda\|C\|p_{miss}^{r}$	286	676
Router				
Forward Int	$\frac{\lambda\|C\|}{\|R\|}p_{miss}^{r-1}$	$\frac{\lambda\|C\|}{\|R\|}p_{miss}^{r}$	286	286
Send Data	$\frac{\lambda\|C\|}{\|R\|}p_{miss}^{r-1}$	$\frac{\lambda\|C\|}{\|R\|}p_{miss}^{r-1}$	286	676
Verify P2	$\frac{2\|C\|\|P\|}{\|R\|}$	$\frac{2\|C\|\|P\|}{\|R\|}$	953	953
Verify P3	$\frac{2\|C\|\|P\|}{\|R\|}p_{stale}^{r-1}$	$\frac{2\|C\|\|P\|}{\|R\|}p_{stale}^{r-1}$	995	995
Consumer				
Key Gen	$\|P\|$	-	670	-
Send Int	-	λ	-	286
Receive	λ	-	676	-
Verify P1	$\|P\|$	-	670	-
Verify P2	$\|P\|$	$\|P\|$	670	283
Verify P3	$\|P\|$	$\|P\|$	692	303
Verify P4	$\|N_T\|\|P\|$	$\|N_T\|\|P\|$	684	294
Notary				
Verify P4	$\|C\|\|P\|$	$\|C\|\|P\|$	294	684

CHAPTER 6

Exploiting Information Centric Networking to Build an Attacker-Controlled Content Delivery Network

THIS Chapter discusses how an attacker using compromised hosts can easily gather a massive amount of low-cost, low-latency storage for malware, junk, and other attacker-controlled content. We conclude by considering a possible countermeasure, a blacklist fed by a honeypot, which we show to be effective.

6.1 Introduction

The Information Centric Networking (ICN) framework is emerging as a way to improve the performance of content delivery by leveraging on the pervasive use of caching. ICN nodes choose what content to keep in their Content Stores according to some reactive policy. This is an important

[1]Part of the contents of this Chapter have appeared in: Giulia Mauri, Riccardo Raspadori, Mario Gerla, and Giacomo Verticale "Exploiting Information Centric Networking to Build an Attacker-Controlled Content Delivery Network" *MedHocNet 2015*, Vilamoura, Portugal, June 2015

difference with respect to the pre-provision algorithms of standard Content Delivery Networks. Thus, the information centric paradigm provides a more neutral service to the user, but leaves room for a malicious user to implement entirely new attacks to the cache management process. The literature identifies two broad classes of attacks: cache pollution attacks, in which the attacker forces an ICN node to keep unpopular contents in its cache in order to exhaust it [2,26] and cache snooping, in which the cached content is used to get information about the downstream users [1,38].

The cache pollution attacks are not new in the networking scenario. Since proxy caching servers are widespread in the IP-based networks, a DoS attack is easy to deploy. The paper [18] presents false-locality and locality-disruption attacks and efficient methods to detect them. A false-locality attack happens when the adversary continuously requests the same set of files, creating a false file locality at the caches. Instead, in a locality-disruption attack, the attacker generates requests for otherwise unpopular content. The authors suggest to use a metric, the byte damage ratio, to measure the effects of the previous attacks. Then, they describe how to detect and mitigate such pollution attacks. Although the paper is not focused on ICN, it is a starting point for our work.

To our knowledge, most of the research on cache pollution attacks has focused on Denial-of-Service (DoS) attacks. This chapter shows that attackers can do more and build a large storage network that can be used to store junk content, malware, and in general any kind of attacker-controlled content. This storage is naturally very near to the end user and can be exploited to serve such content with low latency. This ability provides a clear economic incentive for the adversary, and thus makes this kind of attack especially attractive.

In our scenario, the attacker creates a *false locality* into the user caches by using compromised nodes to send a high number of requests for specific content objects. This way, the attacker can quickly spread its contents towards peripheral nodes. The attacker has an incentive in this attack because it can rent this storage to producers who, for any reason, cannot invest in infrastructure.

The attack is particularly effective when the network exploits the ability of CCN nodes of looking for content across several interfaces using for example a Nearest Replica Routing (NRR) as shown in [49]. A typical example would be an access network in which end-users are equipped with CCN set-top boxes. In such scenario, the access nodes could retrieve content objects requested by a user from the set-top box of another user in a peer-to-peer fashion, thus greatly enhancing the caching ability of the ac-

cess network and reducing the bandwidth requirement of the core network. By exploiting the low cache churn rate of low-activity users, the attacker can easily obtain control of a large fraction of the storage available at these users' premises.

The attacker model with its goals and capabilities is presented in Section 6.2. While, the Section 6.3 shows the evaluation scenario together with the parameters used for the attack analysis presented in the following Section 6.4. In Section 6.5, we suggest a countermeasure to mitigate the attack. We conclude in Section 6.6.

6.2 Attack Description

Our attacker is an entity whose goal is distributing content objects into end user storage, therefore the network can provide that contents with low latency to users asking for them. The attacker focuses its effort on the content objects whose name is in a list $\mathcal{A}$.

The attacker entity has the capability to compromise a set of consumer nodes, creating a botnet under its control. Particularly, the attacker can command:

- the frequency of requests sent by the compromised nodes and the distribution of the requests;
- the names of contents to be requested.

However, the attacker cannot change the cache size and the parameters related to the non compromised traffic: a compromised node sends interests according to the node owner's requests in addition to the attacker-controlled requests. Also, it is not able to compromise the router nodes, in particular it cannot modify their routing tables and Content Stores.

In order to prevent detection, the attacker must avoid starving the legitimate requests, therefore it should find a trade-off between its gain and the service degradation caused to the compromised nodes. We do not discuss how to find this trade-off, but evaluate the attack success depending on the number of compromised nodes.

6.3 Evaluation Scenario

To evaluate the impact of the attack, we consider the network scenario shown in Figure 6.1, in which the users have the following roles:

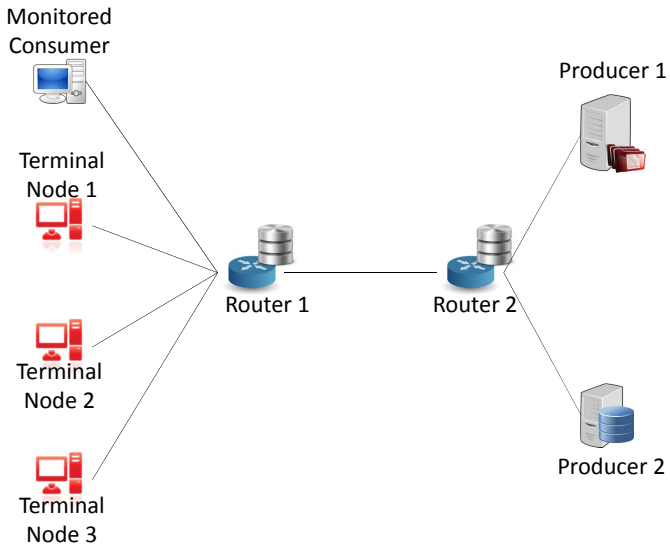

Figure 6.1: *The network topology*

Producer 1. Its Content Store contains all and only the content objects whose name prefix is `/prefix1`. We assume that the names in list $\mathcal{A}$ are a subset of the names starting with this prefix.

Producer 2. Its Content Store contains all and only the content objects whose name prefix is `/prefix2`.

Monitored Consumer. It sends interests for objects whose name starts with `prefix1` with a probability according to the Zipf's law. Consequently, this Consumer may ask for contents in $\mathcal{A}$ or not.

Terminal Nodes. They request contents whose name starts with `prefix2` following the Zipf's distribution. Additionally, one or more of these nodes can be compromised. A Compromised Node (CN) also requests contents from $\mathcal{A}$.

Routers. They forward interests and data packets using a simplified version of the NRR policy. When a router receives a data packet with name n from the downstream interface i, it adds the interface i to its FIB as the next hop for n. Then, when an interests arrives for n, and n is not in the Content Store, the router forwards the interest towards i. If, after a short timeout, the corresponding data packet does not arrive, the interest is forwarded upstream towards the Producer and the association (n, i) is removed from the FIB.

6.4 Attack Analysis

In this Section, we answer the following questions:

1. Can the producer take advantage from the attack and how can we measure this advantage?
2. What are the network conditions that mostly influence the benefits for the attacker?
3. How much the performance of the compromised node is decreased?

We compare different attacker's strategies for choosing the most convenient set of nodes to compromise to reach the goal presented in Section 6.2.

We conduct simulations using the open-source ndnSIM package [4], which implements the NDN protocol stack for the ns-3 network simulator. The simulations last 7200 s, after reaching the steady state. The shown results have a confidence of 95% or better.

We evaluate four scenarios that differ for the lack or presence of caches in the router nodes and for the RTT (Round Trip Time) between Users and Producers:

A. a short range network with RTT = 30 ms,
B. a long range network with RTT = 130 ms,

Both the scenarios can have the following cache configurations:

1. router content stores have negligible size.
2. both users and routers have content stores.

The link between each pair of nodes is 1 Gbit/s to avoid bottlenecks. Moreover, we suppose that the links do not introduce errors during transmission. The content catalog of each Producer comprises 10000 content objects. Each content has a size of 1000 bytes. The Content Store of the Monitored Consumer and of the Routers can store 100 contents, that is the 0.5% of the total. The Terminal Nodes have Content Stores of size 600 contents. All the Content Stores use a LRU policy, as mostly assumed by the literature [9, 40, 46].

The Monitored Consumer sends interests for content with name prefix `prefix1` following the Zipf's distribution with $\alpha = 0.9$ and request rate $\lambda_C = 100$ interests per second. The names in $\mathcal{A}$ correspond to the 1000 most popular objects with this prefix.

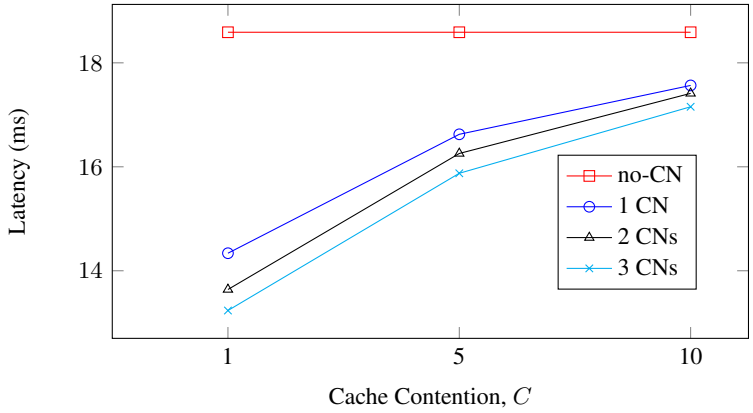

Figure 6.2: *Latency perceived by the Monitored Consumer. Scenario A1: RTT=30ms, negligible Router caches.*

The Terminal Nodes request content with name prefix `prefix2` with rate λ_{CN} and with names chosen according to the Zipf's law. If a node is compromised, then it also requests contents from $\mathcal{A}$ with rate $\mu = 10$ interests per second. The names are chosen uniformly from $\mathcal{A}$. We define a *cache contention* parameter, C, as the ratio between the legitimate requests and the attacker-controlled requests: $C = \lambda_{CN}/\mu$. In order to avoid detection, the attacker must keep its request rate low, therefore we will only consider scenarios in which $C \geq 1$. In the following, we depict results relative to $C = 1, 5, 10$, meaning that $\lambda_{CN} = 10, 50, 100$.

6.4.1 Effectiveness of the Attack

Figure 6.2 shows the latency perceived by the Monitored Consumer in retrieving the 1000 most popular contents versus the cache contention, C, which measures the ratio between the legitimate traffic and the attacker-controlled traffic at the compromised nodes. Results are shown for different numbers of Terminal Nodes that become Compromised Nodes (CN).

With no compromised nodes, the latency is about 18 ms, the delay remains constant and it is higher than the other scenarios. Whereas, in presence of the attack, the delay increases with an higher level of contention, i.e. when the attack is less obvious. However, even with a high contention, the latency is always lower than the values of the scenario without the attack. Thus, this figure shows that the attacker gets an advantage in pushing the content it controls towards the consumers.

Figure 6.3 shows the cache hit ratio measured at the Monitored Con-

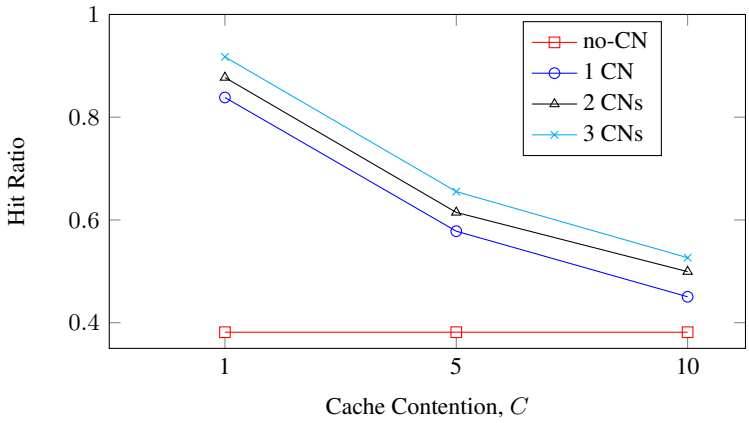

Figure 6.3: *Hit Ratio at the Monitored Consumer.*
Scenario A1: RTT=30ms, negligible Router caches.

sumer versus the cache contention, C. With no attack, the hit ratio is about 0.4.

In case of attack, the requests for attacker-controlled contents from the compromised nodes are routed also to the Monitored Consumer, increasing their perceived popularity and making them stay longer in the Content Store. When the attacker-controlled traffic is high ($C = 1$), the hit ratio grows significantly, to about 0.8 with a single compromised node up to almost 1 with three compromised nodes, meaning that attacker-controlled traffic is unlikely to be pushed out of the Content Store. Clearly, this behavior becomes weaker as the attack intensity decreases. With $C = 10$ and two or three compromised nodes, the effect of the attack on the Content Store of the Monitored Consumer can still be observed, while with a single compromised node, the effect of the attack is negligible.

Figure 6.4 presents the latency perceived by the Terminal Nodes in retrieving content from Producer 2, versus the cache contention, C. Results are shown for different numbers of Compromised Nodes (CN).

During the attack, the Terminal Nodes are forced to request content from Producer 1 and the perceived latency relative to content retrieved from Producer 2 grows. If the Terminal Nodes are not compromised, the latency is about 9 ms. In case of attack, the latency is always larger, showing that the attack bestows worse performance on the affected nodes. Clearly, the latency decreases with a weaker attack, specifically with $C = 10$ the service degradation is small and the attacker might be more difficult to detect.

Figure 6.5 shows the cache hit ratio measured at the Terminal Nodes

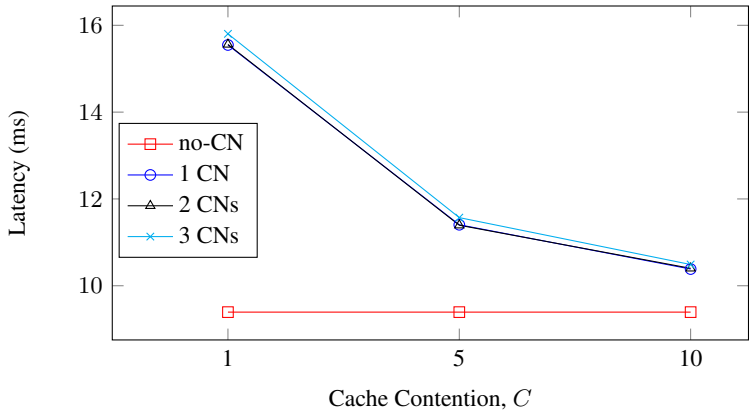

Figure 6.4: *Latency perceived by the Terminal Nodes.*
Scenario A1: RTT=30ms, negligible Router caches.

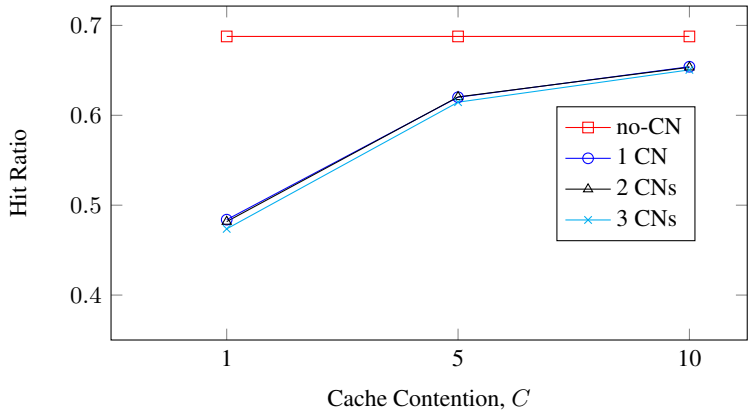

Figure 6.5: *Hit Ratio at the Terminal Nodes.*
Scenario A1: RTT=30ms, negligible Router caches.

depending on cache contention, C. Results are shown for different numbers of Compromised Nodes (CN).

With no attack, the hit ratio is about 0.7. In case of attack, the hit ratio decreases due to the additional requests for attacker-controlled content. With an higher percentage of attacker-controlled traffic, the hit ratio lowers significantly up to about 0.5. While, with $C = 10$ the effect of the attack on the Compromised Nodes hit ratio is negligible. As it can be noticed, both the perceived latency by the Terminal Nodes and the hit ratio are inversely proportional to the perceived latency by the Monitored Consumer and its hit ratio, meaning that the better the service perception for the Monitored Consumer, the higher the service degradation for the Terminal Nodes.

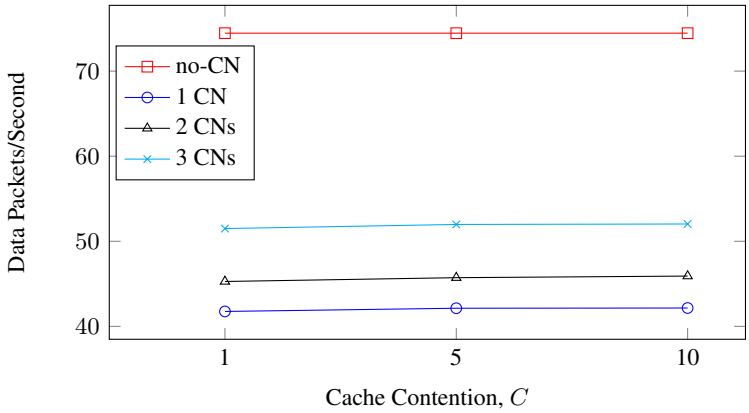

Figure 6.6: *Number of contents sent by the Producer 1. Scenario A1: RTT=30ms, negligible Router caches.*

Figure 6.6 describes the number of content packets sent per second by the Producer 1 versus the cache contention, C and for different numbers of Compromised Nodes (CN).

With no attack, the number of packets sent per second is around 75 packets/s. This number decreases when the Terminal Nodes start sending interest for attacker-controlled content. In particular, the smaller the number of Compromised Nodes, the smaller the number of content packets sent by the Producer 1. For example, the value with one Compromised Node is about 45 packets/s. As it can be expected, the results do not depend on the cache contention because the rate for the attacker-controlled contents, μ, is constant. Thus, this figure clearly depicts that the attacker, i.e. the malicious Producer, has benefits in terms of bandwidth savings. Indeed, the Compromised Nodes become responsible for pushing content to the requesting Consumer instead of the malicious Producer.

The previous results prove that it is possible for a malicious Producer to launch an attack in order to improve the user experience and to reduce the bandwidth and storage requirements of the core network. In the previous setups, it is possible to get both the advantages with an appropriate choice of the network parameters. In particular, the Producer can choose the number of Terminal Nodes to compromise, and the percentage of attacker-controller traffic. At a glance, if $C = 1$, the scenario seems the most advantageous. Indeed, the number of deliveries by the Producer is smaller, and the perceived latency by the Monitored Consumer is decreased more than the scenario with no attack. However, the performance relative to the Terminal Nodes are worse, and consequently the attack is easily noticeable. Thus, the Producer should find a trade-off between the advantages and the possibility to be detected.

6.4.2 Results for Scenario B1

This scenario assumes RTT = 130 ms, and negligible caches in the router nodes.

The results are similar to the scenario A1 presented in subsection 6.4.1. Figure 6.7 shows the metric relative to the latency perceived by the Monitored Consumer. The latency trend is constant with no attack and is about 80 ms. Also, the more the attacker-controlled traffic and the bigger the number of Compromised Nodes, the smaller the latency perceived by the Monitored Consumer. Thus, we can conclude that also if the distance, in terms of RTT, between the end nodes and the Producer is bigger, the attacker can launch an attack, that results to be effective, by controlling the behavior of some nodes.

We do not show the results for the scenarios A2 and B2 where router nodes have bigger caches, which provide similar figures and allow drawing similar conclusions.

Overall, the Producer can always take advantage from the attacker-controller traffic requested by the Terminal Nodes. However, it has to consider different aspects. First, it can choose to compromise only one Terminal Node with a small cache contention. In that case, the number of delivered contents is reduced, the latency perceived by the Monitored Consumer is lowered and consequently the hit ratio is increased. Nevertheless, the Compromised Nodes performance are worse than the other scenarios. Thus, the Producer could consider to compromise one or two Terminal Node(s) with medium cache contention, in order to take advantage and also to not be easily detected.

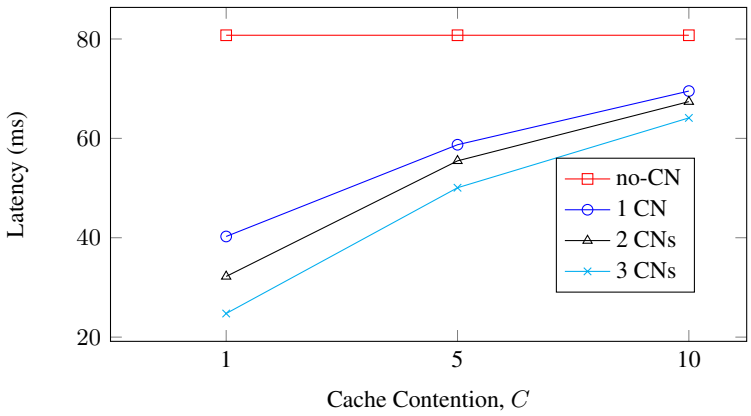

Figure 6.7: *Latency perceived by the Monitored Consumer. Scenario B1: RTT=130ms, negligible Router caches.*

As a conclusion, we can assert that the malicious producer, exploiting the presented attack, can significantly increase its performance in terms of throughput. Thus, performing a attack can make a Producer save a significant bandwidth in the network. The number of Compromised Nodes and their cache contention, C, influence the results. In particular, the more beneficial is the effect on the Consumer, the more negative is the effect on the Terminal Nodes performance. Finally, the cache size and their location also provide advantages.

6.5 Attack Mitigation

The previous Section shows how the attacker can exploit the ICN principles to pile up an attacker-controlled storage. In this Section, we suggest a simple countermeasure based on the idea that the list of names in $\mathcal{A}$ can be identified by means of a honeypot, and the nearest replica routing can be turned off for the identified names.

We install an honeypot over the Terminal Node 1, which we assume to have been compromised by deducing it from the degradation of its performance. This node also hides a monitoring station that collects the attacker-controlled requests. The list of these requests is then sent to Router 1 and stored in a blacklist. Thus, for the contents in this list, the standard NDN routing is used, instead of the modified one that exploits the nearest replica routing, thus reducing or eliminating the attacker's gain.

We evaluate the proposed countermeasure by means of simulations with ndnSIM. We consider the scenario A1 with RTT = 30ms, negligible caches

in the routers, and $C = 1$. Differently from the evaluation scenario in the previous Section, we assume that the Terminal Node 1 does not make any legitimate request and, as a consequence, all the requests coming from that node are malicious, i.e. for content from $\mathcal{A}$. In order to simulate the blacklist collection and the choice of the forwarding algorithm on the basis of the blacklist, we compute the a-priori probability for a malicious content to be in the blacklist, $Pr(Bl)$. Whenever an interest packet arrives for an attacker-controlled content, the Router 1 chooses to use the standard NDN routing with a probability $Pr(Bl)$, otherwise it uses the nearest replica routing. First, we find the average number of contents in the blacklist, N_{Bl}, as follows:

$$N_{Bl} = E[Pois(\mu T)] = \frac{e^{-\mu T}}{\mu T!}.$$

The number of contents in the blacklist depends on a Poisson's distribution $Pois(\mu T)$, where μ is the request rate and T is the monitoring window, meaning that every T seconds the blacklist is updated and sent to the Router node.

Then, we compute the probability, $Pr(Bl)$, as:

$$Pr(Bl) = \frac{N_{Bl}}{|\mathcal{A}|} = \frac{\mu T}{|\mathcal{A}|} = 1 - (1 - \frac{1}{|\mathcal{A}|})^{\mu T}.$$

Assuming $\mu = 10$ interest/s, $T = 120s, |\mathcal{A}| = 1000$, we get $Pr(Bl) \simeq 0,7$. Thus, for contents in the blacklist there is a probability of 70% to be routed using the standard NDN routing instead of the nearest replica routing. Then, considering a monitoring window of $T = 230s$, we get $Pr(Bl) \simeq 0,9$, which corresponds to a probability of 90%.

We measure the latency perceived by the Monitored Consumer in retrieving the attacker-controlled contents versus the number of Compromised Nodes (CN). The results are presented in Figure 6.8 for the different scenarios: with no attack, with attack, and with attack using the proposed mitigation technique.

The red square represents the latency perceived by the Monitored Consumer with no attack, and it is about 18 ms, as already shown in Figure 6.2. While, the latency perceived during the attack is represented by the blue dots and decreases with the number of Compromised Nodes. The Figure highlights an improvement of about 20% relative to the scenario without the attack. Finally, we represent the perceived latency after exploiting the countermeasure to mitigate the attack, the results are depicted with the black triangles. In that case, the results do not depend on the number of Compromised Nodes. We notice that the latency goes near the scenario

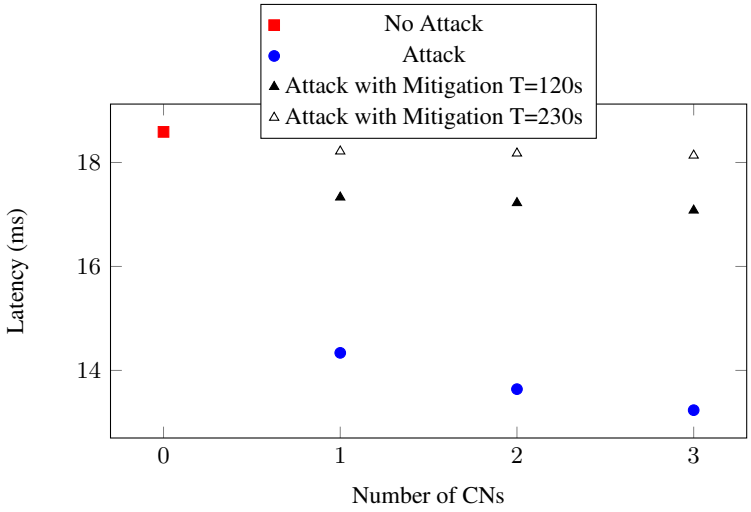

Figure 6.8: *Latency perceived by the Monitored Consumer in retrieving the attacker-controlled contents before the attack, during the attack and with the attack using the mitigation technique depending on the number of Compromised Nodes.*

without the attack, i.e. the latency is about 17 ms for $T = 120s$ and about 18ms for $T = 230s$. Thus, the improvement becomes of only 5% or 2%, respectively. We recall that we choose a monitoring window of $T = 120s$ or of $T = 230s$, that leads to a probability of 70% and of 90% for a content to be in the blacklist. Overall, we can conclude saying that the countermeasure is effective in reducing the attacker's incentive by decreasing its gain. If we enlarged the monitoring window, we would get a bigger probability and so we could bring back the perceived latency to the case without the attack. This means that the attacker completely looses its gain, as expected. Thus, it has no incentives in taking the control of the Terminal Node(s) for requesting its contents becoming a simple DoS attack. This, however, comes at the expense of higher complexity and less robustness to changing conditions.

6.6 Conclusion

This Chapter shows that, under realistic assumptions on the attacker ability, the ICN paradigm can be exploited by malicious users to take control of a large amount of storage near the end-user. This storage can be used to propagate junk, malware, or other content with low latency without requiring any investment in infrastructure. By means of simulations, we show that the attack is particularly effective, providing the attacker with band-

width savings and excellent performance in exchange for a limited effort.

Fortunately, there are some countermeasures that can be used to reduce the effectiveness and thus, the attractiveness of the attack. In particular, we show with simulations that a monitoring station, i.e. a honeypot, can significantly alleviate the issue by sending a blacklist of contents likely to be attacker-controlled to the routers nodes.

CHAPTER 7

The Tradeoff between Performance and User Privacy

THIS Chapter analyses and compares different caching policies for the improvement of network performance. However, this comes at the price of increased tracing of users communication and users behavior to define an optimal caching policy. Thus, the Chapter takes the first step for defining the tradeoff between caching performance and user privacy guarantee. Indeed, a malicious node could exploit user-related information to compromise the privacy of users. In particular, we provide a way to implement prefetching and we define some bounds for the users' privacy in this context.

7.1 Overview and Problem Formulation

The number of contents in the Internet quickly grows. Users continuously request content objects and want them as soon as possible. The need for

[1]Part of the contents of this Chapter have appeared in: (i) Giulia Mauri and Giacomo Verticale "On the Tradeoff between Performance and User Privacy in Information Centric Networking", *New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on*, Apr 2014

a network of caches is obvious. The core paradigm is that users send to the network an interest message indicating the name of a content and the network delivers it from the nearest node cache.

Nodes can either implement a reactive caching policy or use prefetching, which if the user behavior can be well predicted, provides better performance. The information for determining which objects to prefetch can be either generated using a server-hint method or a local method. In the former, a server provides hints, which are based on previous requested objects, to routers closer to the client. The routers prefetch the contents according to the hints received. In the latter, the local prefetcher uses only local information to determine what to prefetch. In this chapter, we consider prefetching policies based on information deduced from user's ranking.

Users have different preferences over the objects. Consequently, the probability of requesting a content differs from one user to another. This chapter captures this aspect by defining a dissimilarity metric between users. Our work stems from the naïve consideration that a prefetching policy that exploits optimal per-user knowledge has optimal performance but is not privacy-friendly. This chapter confirms both intuitions. We compare prefetching and reactive algorithms in scenarios with growing dissimilarity. Then, we try to enhance the privacy of the prefetching per-user policy and evaluate its cost in terms of performance degradation. In order to achieve this goal, we propose a mathematical definition of user privacy suitable for Information Centric Networks. Finally, we evaluate the tradeoff between privacy and latency.

The remainder of the chapter is structured as follows: Section 7.2 presents the model: we define the user dissimilarity, i.e. how user differs from each other, and we consider different caching policies. Section 7.3 proposes a possible way to implement prefetching in ICN scenario. The adversary model is given in Section 7.4, together with the privacy definitions and a possible countermeasure against privacy leakage. Section 7.5 shows the results for the latency perceived by user depending on popularity classes, the mean delay for the most popular content classes, using data perturbation or not, and the tradeoff between privacy and latency. Conclusion and future research directions are left for the final Section 7.6.

7.2 Definitions

7.2.1 User Dissimilarity

We assume a set of I different contents uniformly distributed into C classes of popularity. Each item of class c has a probability of being requested

equal to p_c, with $c = 1, ..., C$. The probability distribution follows the Zipf law, hence $p_c = K/c^\alpha$, where $K = 1/\sum_{i=1}^{C} \frac{1}{c^\alpha}$ and α is the slope of the distribution. In a more general scenario, the probability associated to the same content could be different for distinct users; therefore, we introduce a variable $p_{c,u}$ that defines the probability for a content of class c of being requested by the user u. This probability depends on the ranks that each user gives for each content class $r_{c,u}$, hence $p_{c,u} = K/r_{c,u}^\alpha$. Notice that the smaller the rank $r_{c,u}$, the more important is the content class c for the user u.

To capture how users differ from one another, we define the *dissimilarity* between two users as the number of swaps of two adjacent elements in order to obtain a permutation that maps the first user's ranks into the second one. In addition, we define dissimilarity of a set the minimum dissimilarity between all the users in the set and the natural ranking that gives the highest probability to class 1, the second highest to class 2, and so on.

7.2.2 Caching policy

There are various methods for choosing whether a content should be cached or not. Here, we consider five caching policies that can be distinguished into two classes: coordinated and uncoordinated. In the uncoordinated case, each router decides whether to cache a content using a reactive replacement strategy, e.g. LRU (Least Recently Used) and LFU (Least Frequently Used). While in the coordinate case, the routers take decisions based on a proactive computation on users ranking and on what other routers do.

We present the following prefetching and caching policies:

- **Prefetch by Popularity (PxP)**: the contents are stored into caches based on a reference popularity distribution. Assuming that the contents are organized into C classes of popularity, the ordered vector of content classes c = 1,..,C is used as reference for prefetching. The contents with a smaller class number are stored first into the nearest caches to the users. The caches that have the same distance from the users store the same contents.

- **Prefetch by Ranking (PxR)**: the contents are cached according to the popularity ranking resulting from the actual set of user in the network. The first content class to be stored is the class with the highest request probability obtained by averaging the request probability of each content class over all the users, i.e. $\sum_{u=1}^{N} r_{c,u}/N \quad \forall c \in C$, where N is

number of users in the network. Also in this case, the caches that have the same distance store the same contents.

- **Prefetch by User (PxU)**: the contents are put into caches according to the popularity ranking of the downstream users. The ranking of each user is used as reference, i.e. $r_{c,u}$. The most popular contents for a user are stored first in the cache nearest to this user. Each cache stores different content classes because users are different.
- **Least Recently Used (LRU)**: the least recently used item is discarded first from the cache. This algorithm requires keeping track of all contents when they are used.
- **Least Frequently Used (LFU)**: the contents that are used least often are discarded first. This algorithm requires counting how many requests for each content are received.

As can be noted from the previous descriptions, we need different amounts of information to choose what contents should be stored into the caches. A cache that uses a reactive policy should keep track of past requests and then, it decides which content should be discarded or stored for each interest received. Whereas, the family of prefetching methods is based on users' a priori preferences. In particular, these caching policies exploit the values of $r_{c,u}$ to store a content before it is requested. Unfortunately, it is not easy to know the users' rankings because they are privacy sensitive information. Moreover, obtaining more details on ranks produces an improvement in caching performance. In the next Section, we propose how to implement prefetching by user in a content centric scenario.

7.3 Prefetching by User in ICN

This section discusses a way to implement prefetching by user for the ICN nodes, according to the definition in section 7.2. An estimator installed over the router nodes computes the likelihood that a content will be requested in the next few times and gives this information to the neighbor nodes. Then, the node can choose whether or not to prefetch the content.

The estimator uses a prediction algorithm that could be based on that described in [43]. It is out of the scope of this work to define the detailed algorithm. However, we provide a brief description of it. However, the node can choose to send a *Prefetching Interest* for the content suggests by the algorithm. To distinguish between Interest and Prefetching Interest, we propose to modify the Interest packet adding the option `Prefetching`

into the *Selector* field. In particular, the `Prefetching` option can assume the following values: (i) **0**: it means that the Interest is an ordinary interest packet; (ii) **1**: it indicates that the Interest is sent for prefetching contents. Clearly, we also need to add an option to the PIT (Pending Interest Table), where it would be possible to keep track of the `Prefetching` bit.

7.4 Adversary Model and Countermeasure

Indeed, the ICN scenario solves the problem of privacy related to personally identifiable information, e.g. name, address, etc., but poses new challenges into the privacy of users behavior. In this Section, we present our attacker model, then we define a possible countermeasure to guarantee user privacy.

7.4.1 Adversary Model

First, we consider an attacker $\mathcal{A}$ that interacts with a challenger CH and we denote the interaction as $\mathcal{A}^{\text{CH}}$, where both the attacker and the challenger are Turing Machines (TM) computationally bounded. Particularly, we consider probabilistic polynomial time (PPT) TMs. The adversary goal is to identify a user within a set of subjects observing the contents stored in a chosen cache. Thus, the interaction continues until $\mathcal{A}$ returns an output.

The interaction between the challenger CH and the adversary $\mathcal{A}$ is described in the following Algorithm 7.1:

Algorithm 7.1 The $\mathcal{A}^{\text{CH}}$ game

1. CH chooses a user $u \xleftarrow{R} \mathcal{U}$ within the user space $\mathcal{U}$;
2. CH chooses a cache $k \xleftarrow{R} \mathcal{K}$ within the cache space $\mathcal{K}$;
3. CH gives l, the level of the cache k in a tree topology, and $k_{c_{(i)}}$, the i contents' classes stored in k to $\mathcal{A}$.
4. $\mathcal{A}$ outputs T, True, or F, False.

The Adversary wins the $\mathcal{A}^{\text{CH}}$ game if its output is:

$$\mathcal{A}_{win} = (T \wedge u \in \mathcal{U}_k) \vee (F \wedge u \notin \mathcal{U}_k)$$

where $u \in \mathcal{U}_k$ means that the user u is part of the set of downstream users of the cache k.

Thus, we give our privacy definition founding it on the notion of sender anonymity introduced in [45]:

Definition Anonymity of a subject from an attacker's perspective means that the attacker cannot sufficiently identify the subject within a set of subjects, the anonymity set.

From this description, we formalize our notion of δ-sender anonymity. The parameter δ, $0 \leq \delta \leq 1$, quantifies the probability that a distinguishing event happens, e.g. when a cache is compromised by the adversary, following the definition similar to CDP (Computational Differential Privacy) given in [8].

Any user u in the user space $\mathcal{U}$ of size N is equally probable to be a user served by the cache k for any PPT-adversary $\mathcal{A}$ with the capabilities previously described. This means that there exists δ such that:

$$\Pr[\mathcal{A}_{win}|u \xleftarrow{R} \mathcal{U}] \leq \frac{1}{N} + \delta \quad \forall k \xleftarrow{R} \mathcal{K} : u \in \mathcal{U}_k \tag{7.1}$$

$$\Pr[\mathcal{A}_{win}|u \xleftarrow{R} \mathcal{U}] \leq (1 - \frac{1}{N}) + \delta \quad \forall k \xleftarrow{R} \mathcal{K} : u \notin \mathcal{U}_k \tag{7.2}$$

where $x \leftarrow X(x \xleftarrow{R} X)$ means that x is drawn (uniformly at random) from the set X. It is easy to note that the smaller is δ, the more the privacy is preserved. The following Section presents a possible way to guarantee user anonymity.

7.4.2 Proposed Countermeasure: Data Perturbation

We propose to use a *data perturbation* technique in order to lower the adversary probability of winning, $Pr[\mathcal{A}_{win}]$. A data perturbation method attempts to preserve privacy by modifying values of the sensitive attributes using a randomized process.

In the ICN scenario, the cached contents are privacy sensitive information. A router node determines which contents are to be cached based on interests received from users. Obviously, the user sends interests for contents which is more interested in based on its ranking vector. Thus, the router node can profile its downstream users observing the contents it caches. In order to hide his/her preferences, a user can send interests for contents that are not really valuable for him/her.

In particular, these interests are sent based on a new ranking vector $r^p_{c,u}$ that is obtained making p permutations on the original ranking vector of the user $r_{c,u}$. The following algorithm 7.2 shows how to compute the perturbed ranking vector.

Algorithm 7.2 The data perturbation

for $1 \leq i \leq p$ **do**
1. Extract a random number w in $[0, C]$;
2. Swap $r_{c,u}[w]$ with $r_{c,u}[w+1]$

end for
return the perturbed ranking vector $r^p_{c,u}$

Thus, the interests are sent based on the new rankings $r^p_{c,u}$, which does not really represent the user u but a user with different preferences over the contents. As we show later, introducing a perturbation guarantees user privacy but lowers the performance.

7.5 Results

This Section presents the results obtained from both a simulative scenario and an analytic method. The simulation scenario takes advantage of ndnSIM [4] to evaluate the perceived latency by users using the LRU and LFU policies. While, a Montecarlo simulator, implemented with Matlab, is used to evaluate proactive policies and the adversary's advantage.

From now on, we consider a tree topology, as depicted in Figure 7.1, where there are four leaf nodes that send interests for the contents and one root node that generates the data packets. Moreover, the network is organized into three routers' levels that can cache the contents and answer to interests.

In this scenario, the probability of winning the game defined in the section 7.4 by algorithm 7.1 given by equations 7.1 and 7.2 is computed as follows:

$$\Pr[T \leftarrow \mathcal{A}^{\text{CH}}] = \Pr[T|u \in \mathcal{U}_k] \leq \frac{1}{2^{L-1}} + \delta_T.$$

$$Pr[F \leftarrow \mathcal{A}^{\text{CH}}] = Pr[F|u \notin \mathcal{U}_k] \leq (1 - \frac{1}{2^{L-1}}) + \delta_F.$$

where L is the maximum number of the tree's levels and $T \leftarrow \mathcal{A}^{\text{CH}}$ means that the adversary output is True, conversely $F \leftarrow \mathcal{A}^{\text{CH}}$, means False. Moreover, the adversary advantage δ is given by $\max(\delta_T, \delta_F)$.

In our simulations, having as reference the setup parameters used in [9], we consider a set of $I = 13.8 \cdot 10^6$ items divided into $C = 400$ classes of popularity. Each user has a rank vector $r_{c,u}$ from which the probability $p_{c,u}$ is computed according to the Zipf law. The slope of the probability distribution α is equal to 2. We consider four cases of dissimilarity with $d = 100, 1000, 10000, 100000$, where bigger d means more different users. All contents are assumed to be of the same size, i.e. $10kB$, and to be

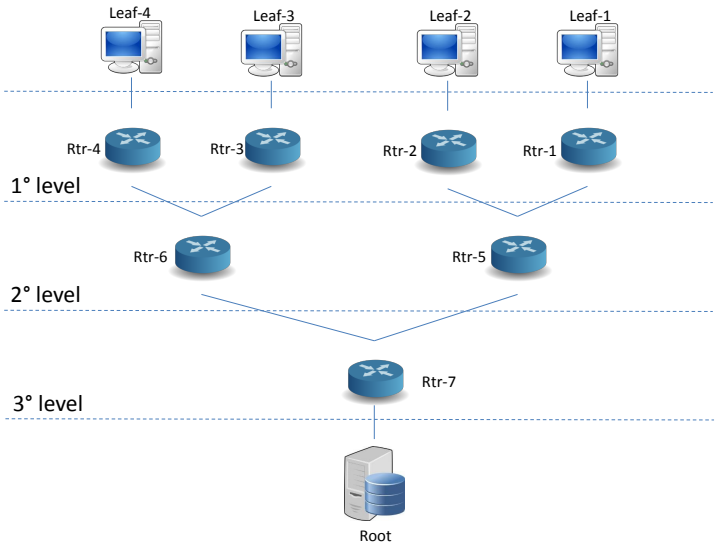

Figure 7.1: *The tree topology*

requested with a request rate $\lambda_{c,u} = \lambda p_{c,u}$, where $\lambda = 27600$ requests/sec. Moreover, we consider a tree topology with three levels of caches as shown in Figure 7.1. The link between each pair of nodes has a transmission delay equal to $2ms$. Each router node, rtr, has a cache (i.e. Content Store) of size s contents, which is equal to 207000, that is exactly the number of content of six classes.

7.5.1 Performance without Data Perturbation

First, we consider the mean round trip time perceived by users to retrieve a content depending on popularity classes. The results are shown in Figure 7.2 for dissimilarity $d = 100$ and in Figure 7.5 for $d = 100000$, where all the caching policies are compared.

As can be noted in Figure 7.2, where the dissimilarity between users is low, the prefetching policies achieve better performance in terms of Round Trip Time (RTT) than the classical LRU and LFU policies. Moreover, the three lines follow the same stepped trend because the caches contain the same content classes in the same level of the tree.

With increasing dissimilarity, as shown in Figure 7.5, the PxU policy guarantees the lowest latency for the six most popular classes of popularity

as seen by each user. For the least popular classes, this policy has worse performance than the other ones. However this is generally not an issue since the performance seen by the user is dominated by the most popular classes, which cover the majority of the content requests. The PxP policy is independent of the popularity classes because it does not take into account the users behavior. The PxR has a similar trend as LRU policy because it reflects the mean ranking of all users.

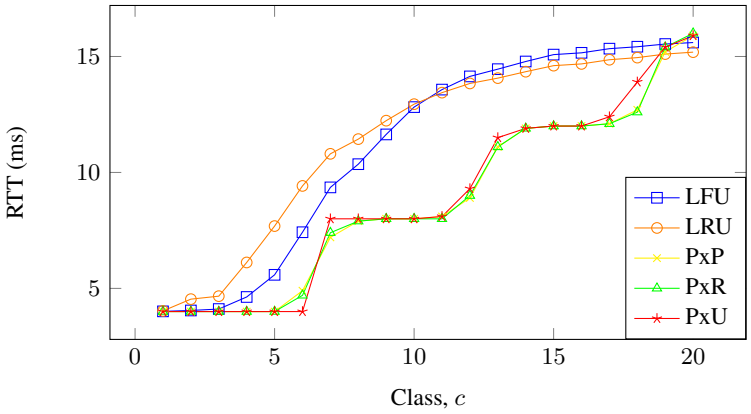

Figure 7.2: *Mean Round Trip Time perceived by users with* $d = 100$ *depending on popularity classes, c. Confidence 95%, Precision 10%*

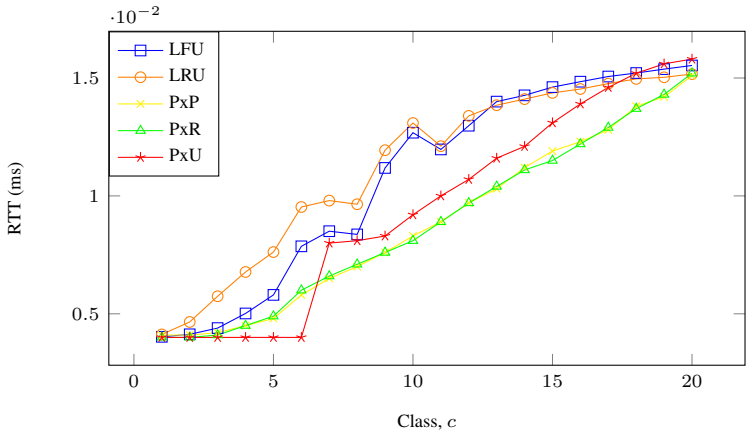

Figure 7.3: *Mean Round Trip Time [ms] perceived by users with* $d = 1000$ *depending on popularity classes. Confidence 95%, Precision 10%*

The results relative to $d = 1000$ and $d = 10000$ are summarized in Figure 7.6, where the mean delay for the first six popularity classes is depicted depending on growing dissimilarity. Notice that we consider only the first six classes because they cover 90% of the total number of requests.

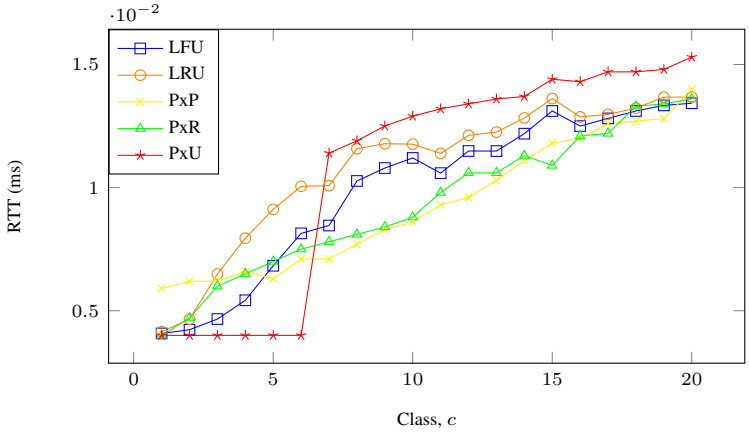

Figure 7.4: *Mean Round Trip Time [ms] perceived by users with* $d = 10000$ *depending on popularity classes. Confidence 95%, Precision 10%*

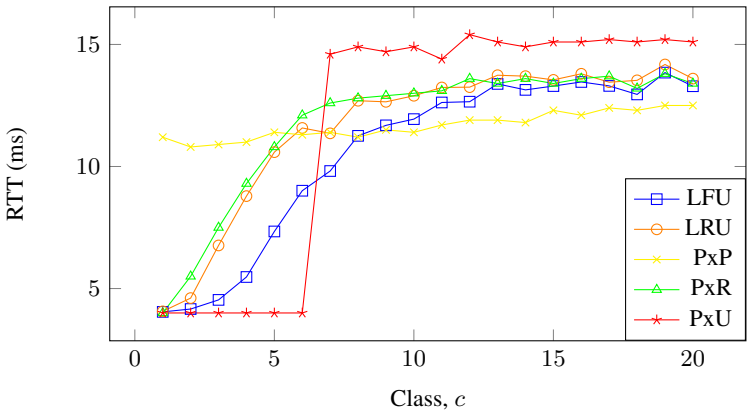

Figure 7.5: *Mean Round Trip Time perceived by users with* $d = 100000$ *depending on popularity classes, c. Confidence 95%, Precision 10%*

Figure 7.6 compares the mean delay for the first six popularity classes depending on dissimilarity. The prefetch by user policy achieves the lowest latency, which is constant for growing dissimilarity. The PxP and PxR performance decreases with more different users; while the LRU and LFU policies slowly increase the delay for the first classes with bigger dissimilarity. The more the performance are good, the more personal information are needed. This sentence poses a challenge that should be overcome: we need users preferences to gain the best but we should guarantee their privacy. Since the PxU policy guarantees lower latency than the classical LRU and LFU, the privacy analysis focuses on the PxU policy.

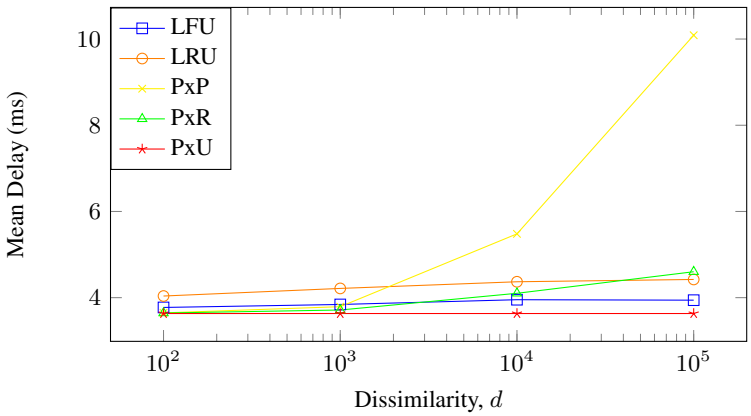

Figure 7.6: *Mean Delay perceived by users for retrieving a content of the first six popularity classes depending on dissimilarity, d.*

7.5.2 Performance with Data Perturbation

Now, let's see what happens if data perturbation is applied over the users' ranking. Figure 7.7 compares the mean delay for the first six popularity classes with growing dissimilarity depending on perturbation using a PxU policy. As can be noted, the performance are comparable to that without data perturbation with $p \leq 100$. While, the performance decreases with bigger perturbation: the mean delay grows. It can also be noted that the performance decreasing does not highly depend on dissimilarity.

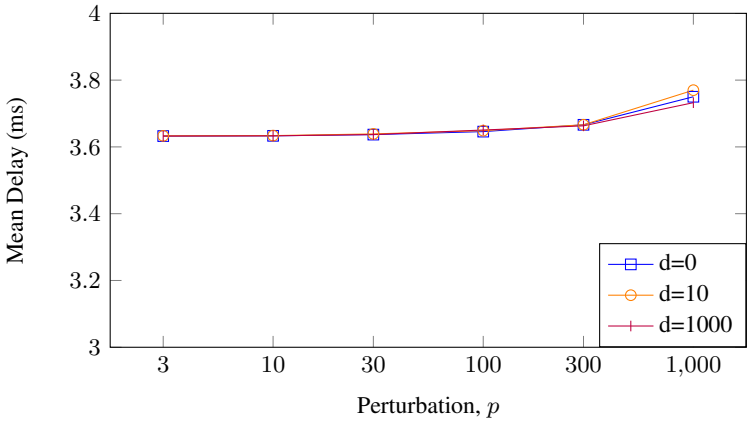

Figure 7.7: *Mean Delay perceived by users for retrieving a content of the first six popularity classes with growing dissimilarity, d, depending on perturbation, p.*

Finally, Figure 7.8 reports the advantage δ that the Adversary $\mathcal{A}$ has in winning the game previously defined, with $i = 6$, that indicates how many

of the k_c classes are given to $\mathcal{A}$. The results are obtained repeating the challenging game until a 95% of confidence is reached. We'd like to note that the advantage in case of no perturbation, $p = 0$, is exactly 1 that means the adversary always wins the game. While if we apply a perturbation, the advantage decreases till 0.05 with any perturbation $p \geq 100$. This means that if there is this kind of perturbation the adversary randomly chooses a user and wins the game with a probability:

$$Pr[T|u \in \mathcal{U}_k] \leq \frac{1}{4} + 0.05 \vee Pr[F|u \notin \mathcal{U}_k] \leq \frac{3}{4} + 0.05.$$

Thus, the user privacy is guaranteed with a perturbation higher than 100. However, we have seen in the previous figure that the performance are not worsened with $p \leq 100$. This means that we have to choose a tradeoff between latency and privacy, and we think that using a perturbation p=100 could be the perfect choice. Indeed, the performance are comparable to the case without data perturbation and the adversary advantage is low, so the user privacy is guaranteed.

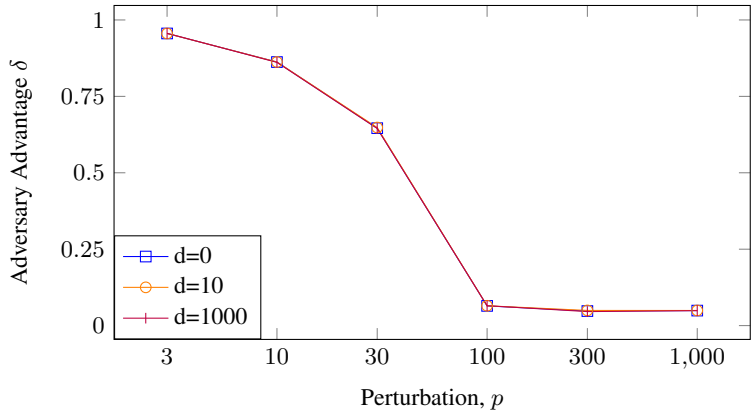

Figure 7.8: *Advantage that the Adversary $\mathcal{A}$ has in winning the $\mathcal{A}^{\text{CH}}$ game depending on perturbation, p, with growing dissimilarity, d*

7.6 Conclusion

This Chapter analyzes the tradeoff between network performance and user's privacy in a information centric scenario. On the one hand, low latency provides both clients and providers with advantages. On the other hand, user sensitive information are needed to gain better performance. This work poses the basis for finding a solution that simultaneously guarantees performance and respects users' needs. Indeed, we propose a prefetching pol-

icy based on users' ranking and a data perturbation technique to guarantee users' privacy.

We think that there is a lot of research to inspect in this direction. First, we need to provide a privacy analysis for LRU and LFU policies. Then, we can find the tradeoff between privacy and performance using these caching policies Moreover, we would like to extend our work considering users churn and different popularity of content chunks in order to provide better network capabilities. We believe that this change slightly modifies the optimal caching policy and also requires a deeper knowledge of user's behavior. On the other side, we would like to expand the solutions for guaranteeing user's privacy introducing an anonymization protocol.

CHAPTER 8

Optimal Content Placement in Information Centric Networking Vehicular Network

In this Chapter we show how the ICN architecture with content pre-distribution can maximize the probability that a user retrieves the desired content in a Vehicle-to-Infrastructure scenario. We give an ILP formulation of the problem of optimally distributing content in the network nodes while accounting for the available storage capacity and the available link capacity. The optimization framework is then leveraged to evaluate the impact on content retrievability of topology- and network-related parameters as the number and mobility models of moving users, the size of the content catalog and the location of the available caches. Moreover, we show how the proposed model can be modified to find the minimum storage occupancy to achieve a given content retrievability level. The results obtained from the optimization model are finally validated against a Name Data Networking architecture through simulations in ndnSIM.

[1]Part of the contents of this Chapter have appeared in: (i) F. Bruno, M. Cesana, M. Gerla, G. Mauri and G.Verticale, "Optimal Content Placement in ICN Vehicular Networks", in the *5th International Conference on Network of the Future*, Dec 2014; (ii) G. Mauri, M. Gerla, F. Bruno, M. Cesana, and G. Verticale, "Optimal Content Prefetching in a ICN Vehicle to Infrastructure Scenario", *submitted to Transaction on Vehicular Technologies*, May 2015.

8.1 Introduction

Today's Internet is focused on content retrieval instead of point to point communication. In addition, users are mostly mobile, continuously changing their location. One of the main problem of user mobility is the intermittent connectivity that causes loss of packets. An Information Centric Networking (ICN) framework is the solution to the previous problems. Indeed, the communication is based on a request/response model where the focus is on the content. This is the basis of the ICN framework, where the communication paradigm shifts from retrieving the content from a given location to retrieving the content from the network.

This paradigm blends well with specific issues of user mobility, such as intermittent connectivity and changing topologies. Let us consider what happens during a handover, e.g., when a user is in phone call in a moving car. As the car moves, the Access Point (AP), to which the user is connected, is no longer available and a new connection should be established seamlessly. Thus, handover procedures are used to keep track where the user is, that is, which mobile base station or access point she is currently attached, in order to re-route the phone call traffic to the proper access network device. Differently, in an ICN scenarios, when a consumer moves from one location to another, a seamless handover is easily achieved by reissuing the interest messages from the new location, and the network can then deliver the content from the best source for the new location. This means that the `DATA` packet can be sent from whichever network node that has the content in its local cache.

We focus here on exploiting the ICN paradigm in the context of vehicular networks. Namely, we analyze the beneficial effects of pre-fetching contents at static network nodes on the performance of Vehicular-to-Infrastructure (V2I) communication paradigms. In our setting, vehicular users move along pre-defined paths and get in touch with several Access Points (APs) along their travel. The users entering the area served by the ICN network ask for a content from a *content catalog*. Each content is segmented in chunks, so the vehicular user can retrieve different parts of the content from different Access Points. A content is successfully retrieved if the user moves out of the area with all the chunks of a content.

We define the problem of optimally pre-fetching the content chunks in the network nodes in order to maximize the average content retrieval probability. We show that the problem can be formulated as an Integer Linear Programming (ILP) problem which can be extended/modified to encompass the case where the minimum retrieval proabability (out of all the mo-

bile users) is maximized. The proposed formulations are then leveraged to study the impact of several parameters on the retrieval probability, such as the number of users in the system, the AP available bandwidth, the propagation latency, and the size of the available caches which are called throughout the chapter *Content Stores*. We also introduce another model that returns the optimal storage size of the whole network in order to reach a desired retrieval probability. Then, we also compare our results with theoretical results and we show the efficiency of our solution. Finally, we compare the performance of the optimal pre-fetching solution against the popular Least Recency Used (LRU) reactive caching policy, by means of simulations with ndnSIM.

The chapter is structured as follows:Section 8.2 explains the scenario considered in this work and the behavior of the ICN-enabled nodes. The optimization models for content placement are presented in Section 8.3. The numerical results are shown in Section 8.4. Conclusions are left for the last Section 8.5.

8.2 A Vehicle-to-Infrastructure Scenario for ICN

This Chapter considers a Vehicle to Infrastructure (V2I) scenario in which vehicular users move in a predefined area covered by Access Points (APs) and download contents from the network. The reference network scenario represented in Figure 8.1 is composed of a set of Access Points (APs) to provide connectivity to moving vehicles, and a backbone network which interconnect the APs.

The set of network nodes is identified by N where $\mathcal{I} \subset N$ is the set of Access Points. Each node in the network (vehicles, APs, backbone nodes) runs a simple version of the NDN protocol [57]. Namely, when a vehicle enters the area and connects to the first AP along its path, it issues an `INTEREST` message for the first chunk of content j from the content catalog, which is composed of S_j chunks, each of size $D^{(\mathrm{ck})}$. Each AP has a Content Store, capable of holding CS chunks. If the AP receiving the `INTEREST` message has the requested chunk of the object j, then it delivers it to the vehicle, otherwise it issues an `INTEREST` message upstream in order to retrieve the requested chunks. The same procedure is repeated by any node in the backbone. If they have the requested chunks, they send them downstream, otherwise they propagate the `INTEREST` upstream. The *Producer* node is a special node in the backbone which is assumed to have all the chunks of all the contents and always satisfies the requests. A vehicle keeps sending `INTEREST` messages as long as it receives chunks or

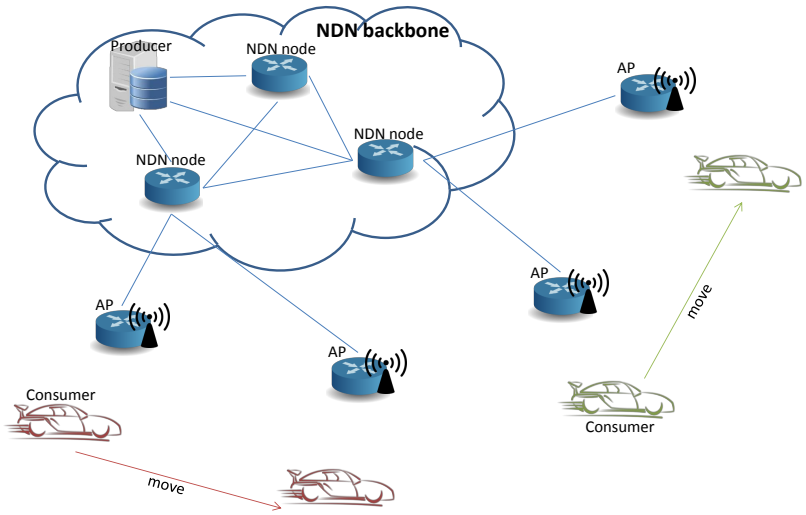

Figure 8.1: *The reference network scenario.*

it moves outside of the AP coverage area. When the vehicle connects to a new AP, it starts reissuing requests for the missing chunks until the content is fully received.

The reference scenario includes U_{tot} users moving at average speed s which may also change along the user's path. The overall mobility pattern is modeled as follows: each user is randomly assigned to one moving path out of V possible paths. The v-th path has a probability β_v of being chosen that follows a Zipf's law with exponent α_p:

$$\beta_v = v^{-\alpha_p} / \sum_{v=1}^{V} v^{-\alpha_p}.$$

Each moving path then determines the set and sequence of APs visited by the moving user along its way. The APs are assumed to have a transmission diameter d; consequently, the connection time available to one user at a given AP is defined as $T_{con} = d/s$. For the sake of presentation, the time needed to discover and associate to the APs is not considered; however, the model/scenario can be trivially extended by scaling down the T_{con} parameter by a factor which depends on the discovery/association time.

By letting the binary parameter m_{iv} be equal to 1 if a user along path v connects to AP i, and 0 otherwise, the number of users on each path can be written as $U_v = U_{tot} \cdot \beta_v$, while the average number of users connected to

the ith AP per path v is:

$$U_i = \sum_{v=1}^{V} U_v m_{iv} / \sum_{v=1}^{V} m_{iv}$$

The content catalog is composed of C content objects. The j-th content has a probability of being requested that follows the Zipf's distribution with exponent α_r. The probability that the jth content is requested is:

$$\sigma_j = j^{-\alpha_r} / \sum_{j=1}^{C} j^{-\alpha_r}$$

Let us define the set of links in the reference network $\mathcal{E} = E_b \cup E_a$, being E_a and E_b the set of access and backbone links, respectively; the available link bandwidth is c_e with $e \in \mathcal{E}$. We further assume a generic but known routing pattern in the reference scenario, that is, for every node in the networks it is known the set of edges (and ordered nodes) which constitute the shortest path to/from the reference node. Namely, it is defined the set SP_{ij} which includes the sequence of edges which belong to the shortest path from node i to node j.

The maximum number of content chunks which can be transferred over link e downstream towards the ith AP for the users in the vth path is called the Maximum Downloadable Burst, B_{eiv}, and depends on the available bandwidth, which must be shared among all the requests from the downstream nodes. Assuming that the bandwidth of the link is equally shared by all the users connected to the same AP i along the path v, we have:

$$B_{eiv} = \frac{c_e \cdot T_{iv}}{D^{(\mathrm{ck})}} \Big(\sum_{i \in \mathcal{I}: e \in SP_{h(e)i}} U_i \Big)^{-1} \tag{8.1}$$

where $T_{iv} = T_{con} \cdot m_{iv}$ is the duration of the connection between AP i and the user along the path v, T_{con} is the total duration of the connection between the user and the APs and $SP_{h(e)i} \subset E$ is the subset of links which belong to the shortest path towards access point i. The function $h(e)$ returns the tail of the edge e.

It is also worth noting that retrieving chunks from a backbone node "closer" to the *Producer* node incurs in a transmission and processing overhead. The parameter $\eta_{eiv} \geq 1$ represents the ratio between the time needed to retrieve at AP i a content chunk for a user along path v through a backbone link e and the time to retrieve the same chunk if the Content Store is available at AP i. This factor can be computed as:

$$\eta_{eiv} = \frac{\sum_{e \in SP_{ih(e)}} T_{iev}^{(\mathrm{ck})} + 2\tau \cdot |SP_{ih(e)}|}{T_{ie_a v}^{(\mathrm{ck})} + 2\tau}.$$

where $T_{iev}^{(\mathrm{ck})} = T_{con}/B_{eiv}$ is the transmission time for a single chunk using the available MDB through link e. A transmission latency τ accounts for the processing cost of the messages.

8.3 Optimal Content Placement

This section provides the Integer Linear Programming (ILP) formulation for the problem of content placement. Differently than the reference NDN protocol, we make the following assumptions:

1. The chunks in the Content Stores of the network nodes do not change over time according to a caching policy, but are pre-fetched according to the decisions of an off-line management platform;

2. The content objects are protected with a Forward Error Correction code, which encodes a file of size S_j chunks in H chunks. The jth content can be fully reconstructed if the consumer obtains any S_j chunks. Therefore, the Consumer does not issue `INTEREST` messages for specific chunks, but issues general `INTERESTs` for additional chunks.

8.3.1 Maximizing the Content Retrievability

The optimization objective is to distribute content chunks into the Content Stores in order to maximize their availability for the retrieval to a user that moves around the network. Let us define the following sets:

- Access Points: $i \in \mathcal{I} = \{1, \ldots, I\}$
- Contents: $j \in \mathcal{J} = \{1, \ldots, C\}$
- Content Stores: $k \in \mathcal{K} = \{1, \ldots, K\}$
- Paths: $v \in \mathcal{V} = \{1, \ldots, V\}$
- Links: $e \in \mathcal{E} = \{1, \ldots, E\} = \{E_a \cup E_b\}$

For the sake of presentation, we assume in the following that $N = \mathcal{K}$, that is, all the nodes in the network (access points and backbone nodes) can cache contents. Moreover, we further assume that the content *Producer* is assigned index 0.

The proposed formulation leverages the decision variables and parameters resumed in the following Table 8.1:

Table 8.1: *Model Variables and Parameters*

Variables	
$x_{jk} \in \mathbb{Z}$	number of chunks of content j cached into the content store k
$A_{jv} \in \{0, 1\}$	boolean variable that is 1 if the content j is retrievable along the path v, 0 otherwise
$y_{eivj} \in \mathbb{Z}$	number of chunks of content j retrieved through link e and meant for users associated to AP i along the path v
Parameters	
$\sigma_j \in [0, 1]$	probability of requesting content j
$\beta_j \in [0, 1]$	probability of choosing path v
$S_j \in \mathbb{Z}$	size of content j
$B_{eiv} \in \mathbb{Z}$	available MDB to the user along the path v connected to AP i from level l
$\eta_{ilv} \in \mathbb{Z}$	the highest cost in retrieving chunks from the furthest content store in the distribution tree
$CS_k \in \mathbb{Z}$	capacity of content store k, measured in number of chunks

The problem of pre-fetching content at network content stores such that the probability of retrieving contents is maximized can be formalized as follows:

$$\max: \sum_{j\in\mathcal{J}} \sigma_j \sum_{v\in\mathcal{V}} \beta_v A_{jv} \tag{8.2}$$

s.t.

$$\sum_{i\in\mathcal{I}} \sum_{e\in E} y_{eivj} m_{iv} \geq A_{jv} S_j \qquad \forall j \in \mathcal{J}, \forall v \in \mathcal{V} \tag{8.3}$$

$$\sum_{e\in SP_{i0}} (\eta_{eiv} \cdot y_{eivj}) \leq B_{eiv} \qquad \forall i \in \mathcal{I}, \forall v \in \mathcal{V}, \forall e \in E_a, \forall j \in \mathcal{J} \tag{8.4}$$

$$\sum_{e\in SP_{h(e)0}} y_{eivj} \leq B_{eiv} \qquad \forall i \in \mathcal{I}, \forall v \in \mathcal{V}, \forall e \in E_b, \forall j \in \mathcal{J} \tag{8.5}$$

$$y_{eivj} \leq x_{jh(e)} \qquad \forall i \in \mathcal{I}, \forall v \in \mathcal{V}, \forall e \in \mathcal{E}, \forall j \in \mathcal{J} \tag{8.6}$$

$$\sum_{j} x_{jk} \leq C S_k \qquad \forall k \in \mathcal{K} \tag{8.7}$$

$$x_{jk} \geq 0, \qquad y_{eivj} \geq 0, \qquad A_{jv} \in \{0, 1\} \tag{8.8}$$

The objective is to maximize the retrievability of the content j, A_{jv}, that is the probability of satisfying the request of the vehicular user. The objective function (8.2) depends on the probability of requesting the content j, σ_j and on the probability of choosing the path v, β_v. The first constraints (8.3) define the retrievability of a content. A content is retrieved if the overall number of chunks which are retrieved from any network node (AP and backbone node) is above the required threshold. Constraints (8.4) and (8.5) impose a limit on retrievability that depends on the maximum downloadable burst (MDB). In short, the overall number of chunks retrieved from network nodes closer to the *Producer* cannot exceed the capacity of the network links used to deliver them downwards. The parameter η_{eiv} emphasizes the fact that the further the chunks are in the tree, the higher is the cost to retrieve them. The number of chunks cached in each content store constrains the maximum number of retrievable chunks, as enforced by constraints (8.6). Equations (8.7) introduce budget-type constraints on the maximum number of contents/chunks stored at any network node. Finally, the equations (8.8) define the decision variables of the formulation.

8.3.2 Maximizing the Worst Content Retrievability

The model introduced in the previous section maximizes the average content retrievability, thus fairness among different paths/users may be low. In this context, it is also worth enforcing a more fair solution by maximizing

the "worst" content retrievability. We introduce a new variable, ϵ, that we would like to maximize in our objective function. The value of this variable is determined from the first constraint (8.10).

Objective function:

$$\max\colon \epsilon \tag{8.9}$$

Constraints:

$$\sum_{j} \sigma_j A_{jv} \geq \epsilon \quad \forall v \in \mathcal{V} \tag{8.10}$$

$$x_{jk} \geq 0,\ y_{eivj} \geq 0,\ A_{jv} \in \{0,1\}, \epsilon \in [0,1] \tag{8.11}$$

The constraints from (8.3) to (8.7) are unchanged. The variables and the indexes remain the same, as in Section 8.3.1.

The objective function (8.9) maximizes ϵ that represents a probability; indeed, ϵ is the smallest success probability within the success probability associated to the different paths v in the network, deduced from the constraint (8.10). Equation (8.11) adds the information that ϵ should be within 0 and 1.

8.3.3 Minimizing the Total Size of Content Stores

The previous two formulations target the maximization of the content retrievability metric under a given budget in terms of available Content Store size. Here, our aim is to minimize the total size of content store in order to guarantee a desired success probability. We introduce a new parameter that is the minimum content retrievability P_{succ} and it is chosen to be 0.95.

Objective function:

$$\min\colon \sum_{j,k} x_{jk} \tag{8.12}$$

Constraints:

$$\sum_{j,v} \sigma_j \beta_v A_{jv} \geq P_{\text{succ}} \tag{8.13}$$

The constraints from (8.3) to (8.8) are unchanged, except for constraint (8.7) that is deleted. The variables and the indices remain the same, as in Section 8.3.1.

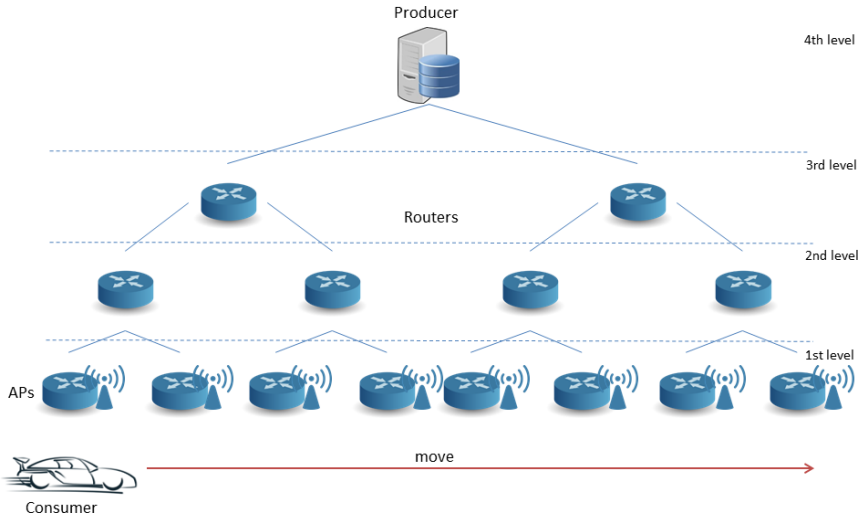

Figure 8.2: *The reference V2I scenario.*

The objective function (8.12) minimizes the sum of all the x_{jk} that represents the number of chunks stored in the network content stores. The new constraint (8.10) imposes a lower bound on the content retrievability.

For the sake of completeness, we evaluate and consider the models complexity, which is expressed as a function of the number of variables and constraints. We notice that the complexity is similar for all the three models.

The number of variables of the first model 8.3.1 is:

$$O(C \cdot (IVE + V + K))$$

While the number of constraints is:

$$O(CV + K + 2IVCE)$$

The second model 8.3.2 has an additional variable and $O(V)$ additional constraints.

Finally, the third model 8.3.3 has the same number of variables and $O(K)$ fewer constraints.

8.4 Performance Evaluation

In this section, we show the results obtained by solving the ILP model by means of AMPL with CPLEX. We start off by considering the network network topology reported in Figure 8.2 with 8 APs, 6 backbone nodes and

one content Producer. Routing to/from the APs happens along a shortest-path tree as represented in the figure. Table 8.2 further summarizes the scenario parameters and the values used in the optimization, except if stated otherwise.

Table 8.2: *Scenario Parameters*

Parameter	Description	Value
τ	link latency	0.25 ms
$c_e : e \in E_a$	access link capacity	9 Mbit/s
$c_e : e \in E_b$	backbone link capacity	1 Gbit/s
L	tree depth	4
I	number of access points	8
K	number of content stores	14
C	size of the content catalog	50
α_r	exponent of content popularity	1
V	number of paths	5
α_p	exponent of path popularity	1
U_{tot}	total number of users	700
S_j	content size	1000 chunk
$D^{(\mathrm{ck})}$	chunk size	1000 byte
CS_k	content store size	2000 chunk
d	diameter of AP coverage	250 m
s	vehicle speed	90 km/h
T_{con}	total duration of connection	10 s
m_{iv}	connection indicator	1

The results show how different network parameters influence the content retrievability. We define the success probability P_{succ} as:

$$P_{\mathrm{succ}} = \sum_{j \in \mathcal{J}} \sigma_j \sum_{v \in \mathcal{V}} \beta_v A_{jv}.$$

We compare different results in order to find the best tradeoff between success probability and network costs. The network costs depend on the content store size, the available bandwidth and the number of access points per path.

8.4.1 Maximizing the Content Retrievability

Figure 8.3 shows the success probability, P_{succ}, in content retrieval depending on the size of the content store, CS, and with different values of τ and, consequently, of η_{eiv}. Thus, this chart highlights the influence of the overall network latency, showing that the success probability is larger with smaller

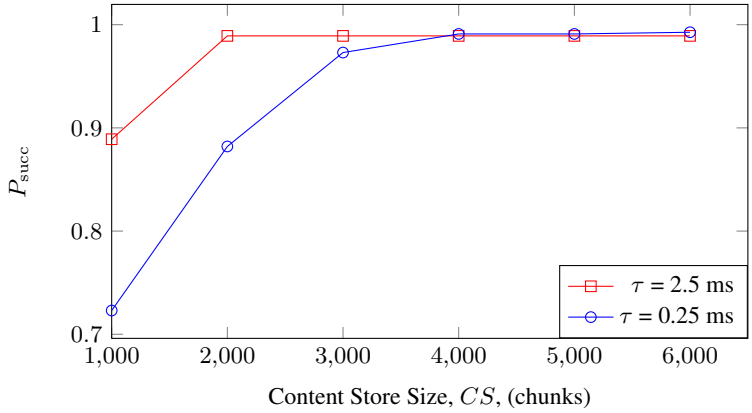

Figure 8.3: *Success probability, P_{succ}, in content retrieval depending on Content Store Size, CS, with different values of link latency, τ.*

τ. Therefore, if the links have a larger latency, the storage size must be increased in order to achieve the same success probability. Moreover, as can be expected, the storage size has a big effect: the bigger are the content stores, the higher is the success probability.

Then, we evaluate the importance that storage has at the different levels of the tree. Figure 8.4 shows the success probability in content retrieval depending on Content Store size assuming that storage is available only at the APs, at the APs plus the second level of nodes, or at all the levels. As the Content Store size increases, the success probability also increases. Having storage only at the APs however does not allow the network to achieve its full potential. Adding storage to the nodes of the second level increases the success probability. However, as stated in [22], the performance improvement is at most 17% relative to the first level. Adding storage also to the third level nodes slightly increases the success probability, but only to a limited extent, which becomes null as the storage in the outer nodes grows.

Figure 8.5 shows the success probability P_{succ} in content retrieval depending on the contact time ratio. We define the Contact Time Ratio, CTR, as the ratio between the longest and the shortest contact between the user and each AP. We assume that the user moves with different speeds along its path but the total time the user is connected with the APs in the network is 80 s. We notice that as the ratio grows, the success probability lowers. This happens because the connection with the APs is more intermittent: the user alternates very long and short connections. Moreover, also in this case, we show that the gain provided by the additional level of content stores is at

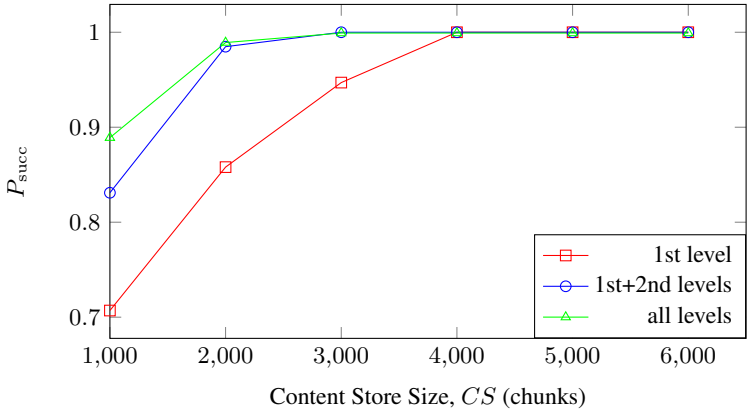

Figure 8.4: *Success probability, P_{succ}, in content retrieval depending on Content Store size, CS, assuming storage at different levels of the network.*

most 17% relative to the first level.

The results reported in Figures 8.4 and 8.5 lead us to the same conclusions of [22]. Indeed, by using a sensitivity analysis, the paper [22] shows that an ICN architecture with pervasive caching and nearest-replica routing can provide at most 17% of best-case improvement in network performance over a simple edge-based caching architecture. Moreover, it proves that if the edge caches are doubled, an edge-caching architecture performs even better than ICN. Thus, as concluded in [22], we can state that making pervasive use of content stores does not provide substantial advantages that could justify to add additional complexity to the network. While, we can think that the gain given from use of caching can be achieved in an incrementally deployable fashion.

Figure 8.6 depicts the success probability P_{succ} in content retrieval depending on content catalog size. We assume different sizes for the content stores. It can be noticed that the success probability is almost 1 for a catalog size ranging from 50 to 100 contents with CSs of 3000 chunks. Then, the success probability is more than 90% if the content stores can store 2000 chunks. The success probability is smaller as smaller is the content store size and as bigger is the catalog.

The results reported so far show that large Content Stores, high link capacity, non intermittent connections, and small content catalog, all increase the success probability, but the effect is different depending on the other system parameters. In particular, increasing the wireless capacity and avoiding intermittent connections have a beneficial impact, but at the same time are unlikely to be controlled and changed. The size of the Content

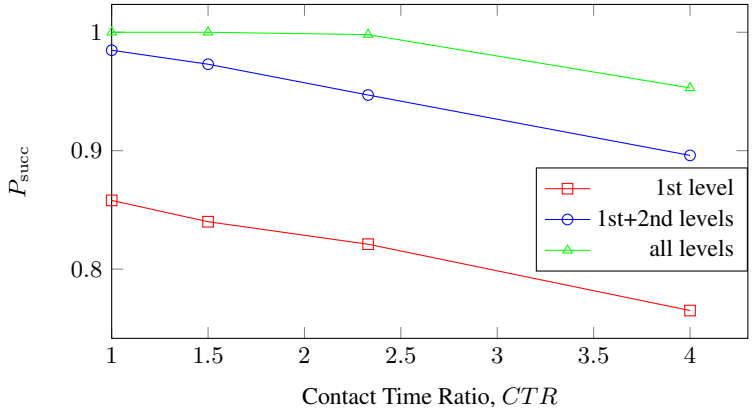

Figure 8.5: *Success probability, P_{succ}, in content retrieval depending on contact time ratio, CTR, assuming storage at different levels of the network.*

Store, on the other hand, is likely to be effortless to improve and can partly compensate a less dense network or a network with less capacity. However, we should take into account that spreading the content stores over a big network is far from being an easy task. Finally, we should consider that the bigger is the variety of content, the more important becomes the size of the content stores.

Figure 8.7 shows the number of users in the network versus the user speed. The continuous line represents the upper bound for the tradeoff between the number of users in the network and their speed to get a success probability equal to 1, assuming that all the contents chunks are stored in all the content stores of the first level.

The values are computed using the following equation:

$$U_{tot} \leq \frac{c_{e:e\in E_a} \cdot d \cdot I \cdot \sum_{i=1}^{I} m_{i1}}{S_j \cdot D^{(ck)} \cdot s}$$

where s and U_{tot} are the changing variables to get the chosen success probability. Note that m_{i1} represents the number of contacts with the access points along the first path. The value is the same for all the paths.

The dots in the Figure represents the number of users versus their speed to achieve 100% success probability, if it is exploited the solution given by our model. We depict values varying the content store sizes CS_k, that is the same in all the tree levels. The bigger are the content stores, the nearest are the values obtained from our model to the bound value.

Thus, this Figure highlights that the results provided by our model are comparable to the optimal bound. Moreover, also small content stores can

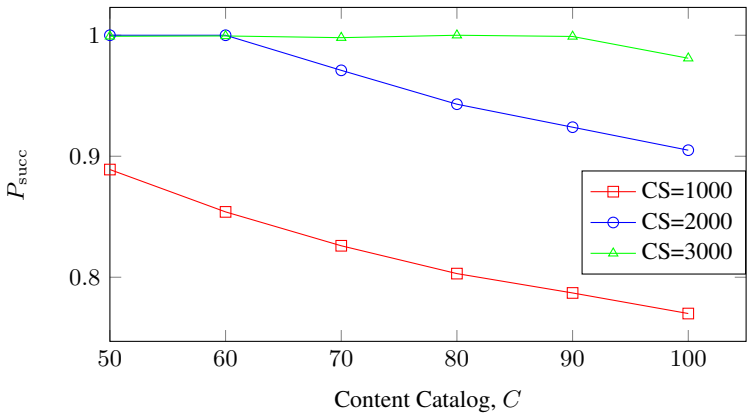

Figure 8.6: *Success probability, P_{succ}, in content retrieval depending on content catalog size, C, assuming different storage sizes.*

reach 100% of success probability paying only a small price in terms of decreased speed and diminished number of users.

Validation against Simulation

Now, we compare the results given by our mathematical model and by the ndnSIM simulator. The main difference is that our model optimally places the contents into the Content Stores using a prefetching approach, while the model used in ndnSIM exploits a Least Recency Used (LRU) caching policy.

In order to have a fair comparison, we compute the number of active users in the network using the Little's law, i.e. $U = \lambda \cdot W$, where U is the mean number of users in the system, λ is the mean number of entering users per unit time and, W is the mean time spent in the system by a user. Then, we evaluate the following:

$$U_{tot} = \frac{1}{f_{in}}[P_{succ}T_m + (1 - P_{succ})T_p]$$

where f_{in} is the frequency of user entering the system, T_m is the mean time to retrieve a content and, T_p is the time spent in the system. Finally, P_{succ} is the success probability in content retrieval, that we have previously defined

Figure 8.8 displays the probability of content retrieval versus the number of users in the system, U_{tot}. This Figure compares the results obtained from the solution of our model and the same scenario implemented by means of ndnSIM [4].

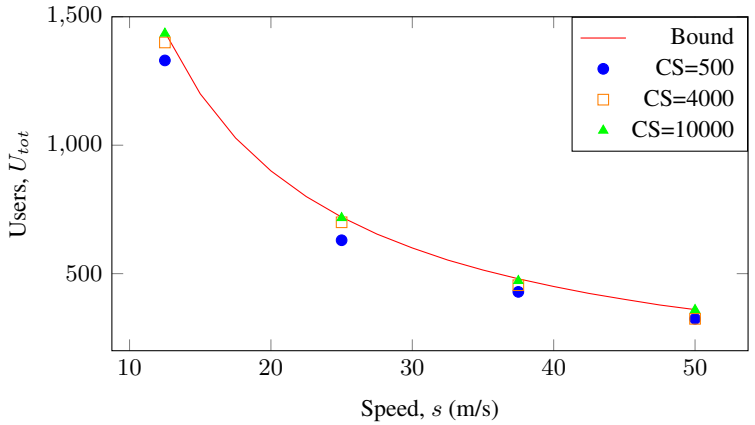

Figure 8.7: *Success probability, $P_{\text{succ}} = 1$, in content retrieval depending on user's speed and number of users in the network varying the content store size CS.*

It can be noticed that the success probability P_{succ} is 1 with less than 200 users and is 0 with more than 800 users in both cases. While, the gain in using the optimization model instead of the classical LRU policy ranges from 50% with about 400 users to a really big gain when the users are 700. Thus, we can say that the proactive placement of content into the content stores can provide big advantages over a reactive caching policy. However, using a prefetching policy requires to have information about contents and users.

8.4.2 Maximizing the Worst Content Retrievability

The second model tries to maximize the worst success probability among the possible paths. Thus, ϵ represents this success probability in content retrieval relative to the path with the smaller probability of being chosen. We assume that there are 480 users that connect with only six over the eight possible APs. The Figure 8.9 represents the values of ϵ as a function of the Contact Time Ratio assuming storage at different levels of the network. It can be noticed that the success probability lowers if the users move faster. Moreover, differently from Figure 8.5, the values of content retrievabilty are very close for all the three considered situations, this is due to the new *minmax* model that tries to balance the success probability of all the paths. However, we have to pay the price of decreased performance.

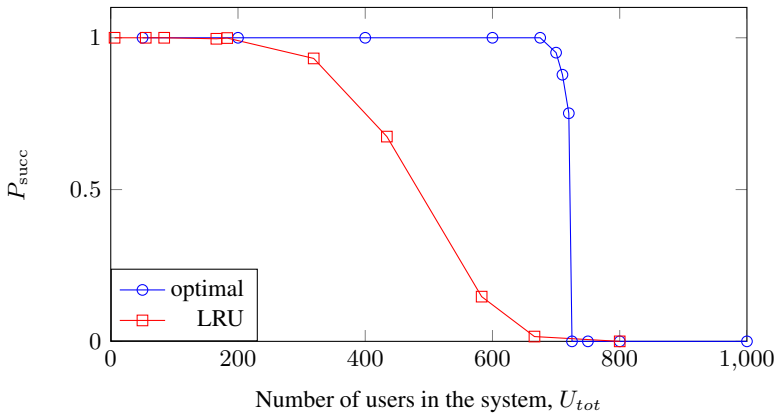

Figure 8.8: *Success probability, P_{succ}, in content retrieval evaluated using the proposed optimal content placement and ndnSIM with the LRU caching policy varying the number of users in the system.*

8.4.3 Minimizing the Total Size of Content Stores

The third proposed model allows us to evaluate the size of the content stores to install in the network in order to guarantee a chosen success probability, that we fixed at $P_{\text{succ}} = 0.95$. The total content store size in the network is represented as $CS_{tot} = \sum_{j=1}^{C} \sum_{k=1}^{K} x_{j,k}$. We study the number of users that our network with different content store sizes can serve in Figure 8.10. Moreover, we assume that content stores are available at different levels of the tree. We note that the content stores are useless with less then 660 users because the network can afford the load and retrieve the contents from the repository at the root of the tree. While, ranging from 660 to 720 users, it is possible to raise the content store sizes in order to achieve the fixed success probability. Then, if the users are more then 720, the network can not supply the requests due to lack of bandwidth. Thus, content stores cannot overcome the limit imposed by the link access availability. Considering the case with content stores in all the tree levels, we noticed that, as the number of users grows, the content stores are pushed dow towards the access network. On the contrary, the content stores in the upper levels are more effective when there are fewer users.

Figure 8.11 shows the total content store size as a function of the exponent of the path popularity to achieve a success probability of 95% in content retrieval. We assume that there are 480 users that connect with only six out of the eight possible APs. It can be noticed that when the users distribution across the available paths is more skewed, that is, when the congestion level of the more congested path increases, the size of the de-

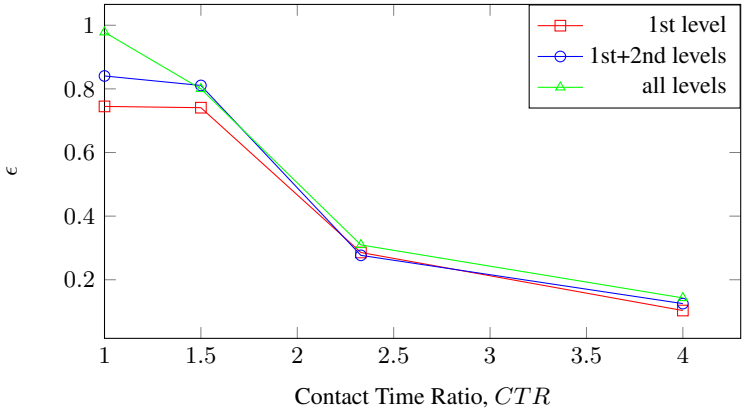

Figure 8.9: *Success probability, P_{succ}, in content retrieval depending on contact time ratio, CTR, assuming storage at different levels of the network. Number of contacts with the APs: 6; number of users in the network: 480.*

ployed content stores increases. Moreover, as it could be expected, if only the first level in the topology is available, it is necessary to install bigger content stores than the cases with also the second and the third level. This Figure also confirms that the third level of content stores is less useful for the scenarios considered.

Figure 8.12 represents the total content store size as a function of the chosen success probability in content retrieval. As it can be expected, the higher is the desired success probability, the bigger the content stores size. Furthermore, moving from 80% of success probability to 100%, we should more than double the total content store capacity. Thus, if we would like to guarantee an higher probability in content retrieval, we should evaluate whether to invest more in content store is worth the costs. Moreover, it can be noticed that by installing content stores not only in the Access Points, it is possible to reduce the total content store size and so to save in costs.

Finally, Figure 8.13 depicts the total content store size needed to achieve a chosen success probability, considering variable users' speed. The succession of long and short connections with the APs brings to install bigger content stores in the network. However, the investment for the storage to guarantee the same chosen success probability in case of different CTRs is reasonable and provides an infrastructure that is more suitable for heterogeneous scenarios.

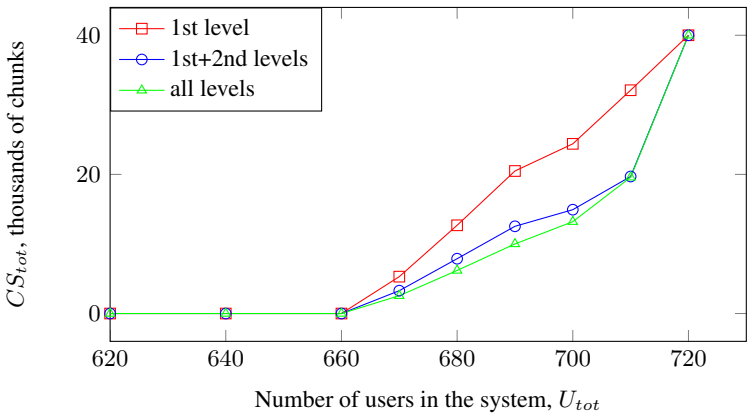

Figure 8.10: *Total content store size to achieve a success probability of 95% in content retrieval depending on number of users in the network, U_{tot}, assuming storage at different levels of the network.*

8.5 Conclusion

This chapter exploits the in-network memory, foundation of the ICN framework, to simplify the delivery of the contents in a vehicular network. Two are the critical aspects for the success of ICN: content placement and caching policy. Our work provides a modeling framework that is a starting point for solving these issues. We define a model that aims to maximize the probability that a vehicular node can obtain the requested content during its stay in the network. We evaluate the system in terms of success probability in content retrieval and investment on storage capacity in the network. By assuming that the Content Stores of the network nodes can be populated in advance, we provide an ILP formulation of the problem of optimally placing the content chunks in the network. The proposed optimization framework is then applied to realistic ICN scenarios to assess the impact of several network parameters onto the content retrievability. Such analysis provides insightful views on where to place content stores and which size of content store to place in ICNs. The following general guidelines/outcomes are sample results of the analysis carried out in this work:

- increasing the storage capacity at the nodes can improve the success probability especially when the APs are sparse, when the content catalog is bigger or when there are a lot of users in the network;
- the maximum gain is achieved by investing in storage capacity in the APs, that is, in the access network closer to the end-users;

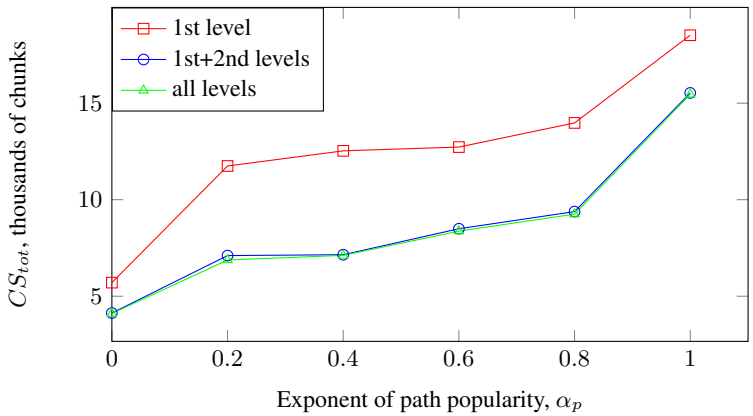

Figure 8.11: *Total content store size to achieve a success probability of 95% in content retrieval depending on the exponent of path popularity, α_p, assuming storage at different levels of the network. Number of contacts with the APs: 6; number of users in the network: 480.*

- investing in storage capacity in the second level of network nodes also improves the performance of about 17%, especially if the link latency is low;
- the access link capacity is often the bottleneck on content retrievability, even if infinite content store size is available;
- when users move with variable speed, bigger content stores are needed and adding levels of storage helps in performance improvement.

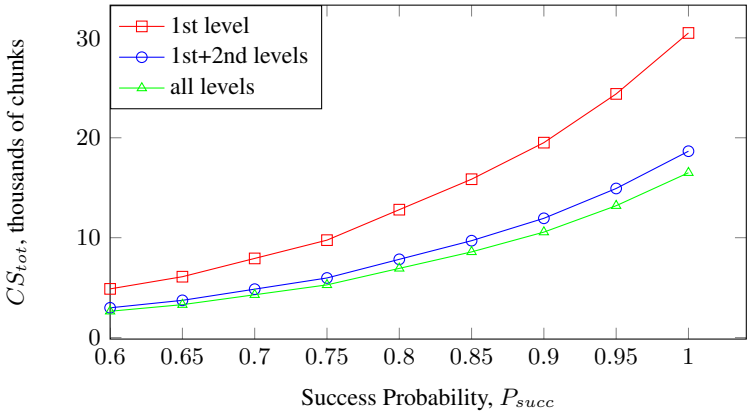

Figure 8.12: *Total content store size to achieve a chosen success probability in content retrieval, assuming storage at different levels of the network.*

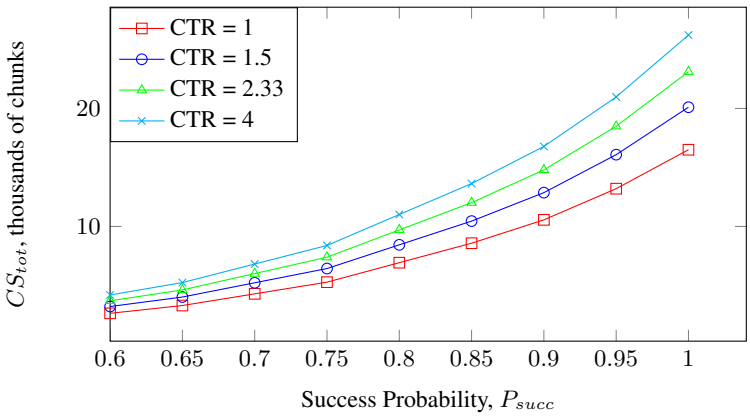

Figure 8.13: *Total content stores size to achieve a chosen success probability in content retrieval, assuming variable users' speed.*

CHAPTER 9

Conclusion

THIS work analyzes and proposes solutions for solving some of the major challenges in the Information Centric Networking framework. We first focused on security issues and suggested a trust management infrastructure for data object authentication. Thus, we provide a communication protocol to verify origin and integrity of content.

The main challenge was to define how to distribute or retrieve the publisher's public key to the end users, and we discussed four alternative methods in Chapter 5. We started with a proactive method inspired by the Certificate Revocation List protocol and based on the repository synchronization mechanism. Then, we suggested two reactive solutions inspired by the Online Certificate Status Protocol and based on the interest/data communication model. Finally, we considered a distributed method that allows trusted nodes to provide the public key instead of the publisher. The four proposals have been compared in terms of throughput and latency in order to show their usability for an ICN scenario. The results showed that the distributed solution is suitable for a network where the access network has spare capacities. While, a centralized protocol is better to choose when the access network is overloaded.

The work also discussed in Chapter 6 how it is possible to launch an attack such that the attacker can gain a large amount of storage for attacker-controlled content. The attacker is able to create a false locality into the end users caches by making them sending a high number of requests for specific content. In this way, it builds a large storage network that is very near to the end user and can be exploited to serve content with low latency. The attacker can also sell or rent the network under its control to those producers that cannot invest in the infrastructure. Thus, the attacker has an economic incentive and the attacker becomes very attractive. We, then, provided a possible countermeasure to reduce the interest in this kind of attack.

Moreover, in Chapter 7, we analyzed the trade-off between caching performance and user privacy guarantee. Firstly, we evaluated different proactive and reactive caching policies. Then, we chose the best proactive policy and we used it as reference for evaluating user privacy. We defined an adversary model and the user anonymity, thus evaluating it against the adversary. We showed that the adversary can break the user privacy but we also provided a solution to mask the user profile, a data perturbation technique.

In addition, we considered a different, not security-related problem, but particularly important for evaluating the possible impact of ICN in the wireless access networks. Finally, in Chapter 8, we considered a Vehicle-to-Infrastructure scenario showing that the ICN architecture with content prefetching can maximize the probability of content retrieval. We proved our intuition with an ILP formulation of the problem of content distribution, accounting the link capacity and the storage capacity in the network. We evaluated our model against various topology- and network-related parameters and validated it against the NDN architecture.

To the best of our knowledge, this work designs original solutions for the described Information Centric Networking challenges. It is the first that proposes a complete PKI-like framework for up-to-date key retrieval in ICN. The framework allows the network entities to choose which method is more suitable for the security constraints and the network conditions. Moreover, we show that all the proposed solutions are deployable for a ICN scenario. Then, in a content centric network, it is of paramount importance to control the content access and the content staleness. We have defined a solution that provides both the properties in a efficient way. Problems related to new possible attacks and privacy issues have been also analyzed and original solutions for those problems have been suggested. Finally, since the future is mobile, we have taken into account a Vehicle to Infrastructure scenario in ICN.

We believe that this work paves the way towards the design of a comprehensive communication architecture capable of ensuring better performance in content distribution. Some open challenges have been solved in this work, others could be input for a future research. In particular, future research efforts will be focused on enhancing the proposed solutions and on integrating them into the ICN framework. Moreover, we will continue our work for defining naming and routing schemes that could provide the needed properties.

List of Figures

List of Tables

Bibliography

[1] Gergely Acs, Mauro Conti, Paolo Gasti, Cesar Ghali, and Gene Tsudik. Cache privacy in named-data networking. In *Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on*, pages 41–51. IEEE, 2013.

[2] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and Lixia Zhang. Interest flooding attack and countermeasures in named data networking. In *IFIP Networking Conference, 2013*, pages 1–9, May 2013.

[3] Alexander Afanasyev, Ilya Moiseenko, and Lixia Zhang. ndnsim packet format, 2011-2013.

[4] Alexander Afanasyev, Ilya Moiseenko, and Lixia Zhang. ndnSIM: NDN simulator for NS-3. Technical Report NDN-0005, NDN, October 2012.

[5] B Ahlgren, E Axelsson, C Dannewitz, A Eriksson, J Frtunikj, B Gronvall, B Kauffmann, A Lindgren, A Lynch, B Ollman, KA Persson, O Strandberg, J Tuononen, S Weber, M D'Ambrosio, L Brown, Z Despotic, S Farrell, M Gallo, C Imbrenda, D Kutscher, H Lundqvist, L Muscariello, JF Peltier, P Poyhonen, P Truing, and V Vercellone. Scalable and adaptive internet solutions (sail), July 2011.

[6] Marica Amadeo, Claudia Campolo, and Antonella Molinaro. Content-centric networking: Is that a solution for upcoming vehicular networks? In *Proceedings of the Ninth ACM International Workshop on Vehicular Inter-networking, Systems, and Applications*, VANET '12, pages 99–102, New York, NY, USA, 2012. ACM.

[7] Marica Amadeo, Claudia Campolo, and Antonella Molinaro. Information-centric networking for connected vehicles: A survey and future perspectives. *IEEE Communication Magazine*, Feb 2016.

[8] M. Backes, A. Kate, P. Manoharan, S. Meiser, and E. Mohammadi. Anoa: A framework for analyzing anonymous communication protocols. In *Computer Security Foundations Symposium (CSF), 2013 IEEE 26th*, pages 163–178, 2013.

[9] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino. Modeling data transfer in content-centric networking. In *Teletraffic Congress (ITC), 2011 23rd International*, pages 111–118, Sept 2011.

[10] Palo Alto Research Center. Ccnx signature generation and verification, Nov 2009.

Bibliography

[11] Abdelberi Chaabane, Emiliano De Cristofaro, Mohamed Ali Kaafar, and Ersin Uzun. Privacy in content-oriented networking: threats and countermeasures. *SIGCOMM Comput. Commun. Rev.*, 43(3):25–33, July 2013.

[12] Kideok Cho, Munyoung Lee, Kunwoo Park, T.T. Kwon, Yanghee Choi, and Sangheon Pack. Wave: Popularity-based and collaborative in-network caching for content-oriented networks. In *Comp. Comm. Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 316–321, 2012.

[13] Nakjung Choi, K. Guan, D.C. Kilper, and G. Atkinson. In-network caching effect on optimal energy consumption in content-centric networking. In *Communications (ICC), 2012 IEEE International Conference on*, pages 2889–2894, June 2012.

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

[15] Mauro Conti, Paolo Gasti, and Marco Teoli. A lightweight mechanism for detection of cache pollution attacks in named data networking. *Computer Networks*, 57(16):3178 – 3191, 2013. Information Centric Networking.

[16] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile, May 2008. RFC 5280.

[17] Christian Dannewitz, Dirk Kutscher, BöRje Ohlman, Stephen Farrell, Bengt Ahlgren, and Holger Karl. Network of information (netinf) - an information-centric networking architecture. *Comput. Commun.*, 36(7):721–735, April 2013.

[18] Leiwen Deng, Yan Gao, Yan Chen, and Aleksandar Kuzmanovic. Pollution attacks and defenses for internet caching systems. *Comput. Netw.*, 52(5):935–956, April 2008.

[19] Pralhad Deshpande, Anand Kashyap, Chul Sung, and Samir R. Das. Predictive methods for improved vehicular wifi access. In *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, MobiSys '09, pages 263–276, New York, NY, USA, 2009. ACM.

[20] B. Etefia, M. Gerla, and Lixia Zhang. Supporting military communications with named data networking: An emulation analysis. In *MILITARY COMMUNICATIONS CONFERENCE, 2012 - MILCOM 2012*, pages 1–6, 2012.

[21] B. Etefia and Lixia Zhang. Named data networking for military communication systems. In *Aerospace Conference, 2012 IEEE*, pages 1–7, 2012.

[22] Seyed Kaveh Fayazbakhsh, Yin Lin, Amin Tootoonchian, Ali Ghodsi, Teemu Koponen, Bruce Maggs, K.C. Ng, Vyas Sekar, and Scott Shenker. Less pain, most of the gain: Incrementally deployable icn. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, pages 147–158, New York, NY, USA, 2013. ACM.

[23] Nikos Fotiou, Pekka Nikander, Dirk Trossen, and GeorgeC. Polyzos. Developing information networking further: From psirp to pursuit. In Ioannis Tomkos, ChristosJ. Bouras, Georgios Ellinas, Panagiotis Demestichas, and Prasun Sinha, editors, *Broadband Communications, Networks, and Systems*, volume 66 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 1–13. Springer Berlin Heidelberg, 2012.

[24] Nikos Fotiou, Dirk Trossen, and GeorgeC. Polyzos. Illustrating a publish-subscribe internet architecture. *Telecommunication Systems*, 51(4):233–245, 2012.

[25] G. Garcia, A. Beben, F.J. Ramon, A. Maeso, Ioannis Psaras, G. Pavlou, Ning Wang, J. Sliwinski, S. Spirou, S. Soursos, and E. Hadjioannou. Comet: Content mediator architecture for content-aware networks. In *Future Network Mobile Summit (FutureNetw), 2011*, pages 1–8, June 2011.

[26] Paolo Gasti, Gene Tsudik, Ersin Uzun, and Lixia Zhang. Dos and ddos in named-data networking. *CoRR*, abs/1208.0952, 2012.

[27] Cesar Ghali, Gene Tsudik, and Ersin Uzun. Needle in a haystack: Mitigating content poisoning in named-data networking. *Proceedings of NDSS Workshop on Security of Emerging Networking Technologies (SENT)*, 2014.

[28] Cesar Ghali, Gene Tsudik, and Ersin Uzun. Network-layer trust in named-data networking. *SIGCOMM Comput. Commun. Rev.*, 44(5):12–19, October 2014.

[29] Giulio Grassi, Davide Pesavento, Lucas Wang, Giovanni Pau, Rama Vuyyuru, Ryuji Wakikawa, and Lixia Zhang. Vehicular inter-networking via named data. *arXiv preprint arXiv:1310.5980*, 2013.

[30] Ying Huang, Yan Gao, Klara Nahrstedt, and Wenbo He. Optimizing file retrieval in delay-tolerant content distribution community. In *Proceedings of the 2009 29th IEEE International Conference on Distributed Computing Systems*, ICDCS '09, pages 308–316, Washington, DC, USA, 2009. IEEE Computer Society.

[31] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '09, pages 1–12, New York, NY, USA, 2009. ACM.

[32] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. A data-oriented (and beyond) network architecture. In *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '07, pages 181–192, New York, NY, USA, 2007. ACM.

[33] D. Kutscher, S. Eum, K. Pentikousis, I. Psaras, D. Corujo, D. Saucez, T. Schmidt, and T. Waehlisch. Icn research challenges, February 2015.

[34] Tobias Lauinger, Nikolaos Laoutaris, Pablo Rodriguez, Thorsten Strufe, Ernst Biersack, and Engin Kirda. Privacy risks in named data networking: what is the cost of performance? *SIGCOMM Comput. Commun. Rev.*, 42(5):54–57, September 2012.

[35] Jihoon Lee, Sungrae Cho, and Daeyoub Kim. Device mobility management in content-centric networking. *Communications Magazine, IEEE*, 50(12):28–34, 2012.

[36] Priya Mahadevan, Ersin Uzun, Spencer Sevilla, and J.J. Garcia-Luna-Aceves. Ccn-krs: A key resolution service for ccn. In *Proceedings of the 1st International Conference on Information-centric Networking*, INC '14, pages 97–106, New York, NY, USA, 2014. ACM.

[37] Giulia Mauri and Giacomo Verticale. Distributing key revocation status in named data networking. In *Advances in Communication Networking*, pages 310–313. Springer, 2013.

[38] Abedelaziz Mohaisen, Xinwen Zhang, Max Schuchard, Haiyong Xie, and Yongdae Kim. Protecting access privacy of cached contents in information centric networks. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, pages 1001–1003, New York, NY, USA, 2012. ACM.

[39] Marc Mosko. Ccnx 1.0 collection synchronization. Technical report, Palo Alto Research Center, Inc., 2014.

[40] Luca Muscariello, Giovanna Carofiglio, and Massimo Gallo. Bandwidth and storage sharing performance in information centric networking. In *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ICN '11, pages 26–31, New York, NY, USA, 2011. ACM.

Bibliography

[41] JoseL. Muñoz, Jordi Forné, Oscar Esparza, and Miguel Soriano. Cervantes – a certificate validation test-bed. In SokratisK. Katsikas, Stefanos Gritzalis, and Javier López, editors, *Public Key Infrastructure*, volume 3093 of *Lecture Notes in Computer Science*, pages 28–42. Springer Berlin Heidelberg, 2004.

[42] M. Myers, R. Ankney, Malpani A., S. Galperin, and C. Adams. X.509 internet public key infrastructure,online certificate status protocol - ocsp, June 1999. RFC 2560.

[43] Venkata N. Padmanabhan and Jeffrey C. Mogul. Using predictive prefetching to improve world wide web latency. *SIGCOMM Comput. Commun. Rev.*, 26(3):22–36, July 1996.

[44] John P. Papanis, Stavros I. Papapanagiotou, Aziz S. Mousas, Georgios V. Lioudakis, Dimitra I. Kaklamani, and Iakovos S. Venieris. On the use of attribute-based encryption for multimedia content protection over information-centric networks. *Transactions on Emerging Telecommunications Technologies*, pages 422–435, 2014.

[45] Andreas Pfitzmann and Marit Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, 2009.

[46] Ioannis Psaras, Richard G. Clegg, Raul Landa, Wei Koong Chai, and George Pavlou. Modelling and evaluation of ccn-caching trees. In *Proceedings of the 10th International IFIP TC 6 Conference on Networking - Volume Part I*, NETWORKING'11, pages 78–91, Berlin, Heidelberg, 2011. Springer-Verlag.

[47] Ying Rao, Huachun Zhou, Deyun Gao, Hongbin Luo, and Ying Liu. Proactive caching for enhancing user-side mobility support in named data networking. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2013 Seventh International Conference on*, pages 37–42, 2013.

[48] Dipankar Raychaudhuri, Kiran Nagaraja, and Arun Venkataramani. Mobilityfirst: A robust and trustworthy mobility-centric architecture for the future internet. *SIGMOBILE Mob. Comput. Commun. Rev.*, 16(3):2–13, December 2012.

[49] Giuseppe Rossini and Dario Rossi. Coupling caching and forwarding: Benefits, analysis, and implementation. In *Proc. of the 1st ACM SIGCOMM Conference on Information-Centric Netowrking*. ACM, sept 2014.

[50] Zuhua Shao and Yipeng Gao. Certificate-based verifiably encrypted rsa signatures. *Transactions on Emerging Telecommunications Technologies*, 26(2):276–289, 2015.

[51] Diana Smetters and Van Jacobson. Securing network content. Technical report, PARC, 2009.

[52] Gareth Tyson, Nishanth Sastry, Ivica Rimac, Ruben Cuevas, and Andreas Mauthe. A survey of mobility in information-centric networks: Challenges and research directions. In *Proceedings of the 1st ACM Workshop on Emerging Name-Oriented Mobile Networking Design - Architecture, Algorithms, and Applications*, NoM '12, pages 1–6, New York, NY, USA, 2012. ACM.

[53] Dan Wendlandt, David G. Andersen, and Adrian Perrig. Perspectives: Improving ssh-style host authentication with multi-path probing. In *USENIX 2008 Annual Technical Conference on Annual Technical Conference*, ATC'08, pages 321–334, Berkeley, CA, USA, 2008. USENIX Association.

[54] Mengjun Xie, Indra Widjaja, and Haining Wang. Enhancing cache robustness for content-centric networking. In Albert G. Greenberg and Kazem Sohraby, editors, *INFOCOM*, pages 2426–2434. IEEE, 2012.

[55] 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*, 16(2):1024–1049, Second 2014.

[56] L Zhang, A Afanasyev, J Burke, V Jacobson, KC Claffy, P Crowley, C Papadopoulos, L Wang, and B Zhang. Named data networking. Technical report, University of California, Los Angeles, 2014.

[57] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, Jim Thornton Diana K, Smetters Beichuan Zhang, Gene Tsudik Kc Claffy Dmitri, Krioukov Dan Massey, Christos Papadopoulos, Tarek Abdelzaher Lan, Wang Patrick Crowley, Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D. Thornton, Diana K. Smetters, Beichuan Zhang, Gene Tsudik, Kc Claffy, Dmitri Krioukov, Dan Massey, Christos Papadopoulos, Tarek Abdelzaher, Lan Wang, Patrick Crowley, and Edmund Yeh. Named data networking (ndn) project ndn-0001, 2010.

[58] Xinwen Zhang, K. Chang, Huijun Xiong, Yonggang Wen, Guangyu Shi, and Guoqiang Wang. Towards name-based trust and security for content-centric network. In *Network Protocols (ICNP), 2011 19th IEEE International Conference on*, pages 1–6, Oct 2011.

[59] Zhenkai Zhu and Alexander Afanasyev. Let's chronosync: Decentralized dataset state synchronization in named data networking. In *21st IEEE International Conference on Network Protocols (ICNP 2013)*, 2013.

[60] Zhenkai Zhu, Alexander Afanasyev, and Lixia Zhang. A new perspective on mobility support. *Named-Data Networking Project, Tech. Rep*, 2013.

Acknowledgments

FIRST, I would like to thank the Scuola Interpolitecnica di Dottorato for providing the financial support to my research activity and for offering the chance to take part in a high qualification PhD program, which gave me the opportunity to join an international and cosmopolitan research environment, greatly improving my professional and personal knowledge.

Foremost, I thank my advisor Dr. Giacomo Verticale for his valuable guidance during the last three years. My sincere thanks also to Prof. Mario Gerla for his precious advice during the seven months of collaboration with the Network Research Laboratory in the Computer Science Department at University of California, Los Angeles. A particular mention goes also to Prof. Giovanni Pau, whom I thank for his support and encouragement during my stay at UCLA. A wholeheartedly thanks goes also to Prof. Eduardo Cerqueira for his humanity and precious suggestions.

I would also like to thank all the professors and colleagues of my research group for their company, friendship and help during the daily work at university.

Finally, I wish to express the most sincere gratitude to my parents, Rita and Gianni, for their constant love, support and help during the whole period of my studies, to my brother, Stefano for being my family and my best friend and to both the Italian and the worldwide friends, who shared my everyday life during my PhD.

Last but not the least, my bigger gratitude is for my dearest Francesco, consultant and example for everyday life.