A multifold approach to address the security issues of stateful forwarding mechanisms in Information-Centric Networks

Salvatore Signorello

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

A multifold approach to address the security issues of stateful forwarding mechanisms in Information-Centric Networks

PhD THESIS

présentée et soutenue publiquement le 21 juin 2018

pour l'obtention du

Doctorat de l'Université de Lorraine

en co-tutelle avec l'Université du Luxembourg

(mention informatique)

by

Salvatore Signorello

Composition du jury

Rapporteurs :	Pr. Gabrijela Dreo-Rodošek- Professor at the Bundeswehr University Munich, Germany Pr. Maryline Laurent - Professor at TELECOM SudParis, France
Examinateurs :	Pr. Thomas Engel - Professor at the University of Luxembourg, Luxembourg Dr. Maria Rita Palattella - Senior R&T Associate at LIST, Luxembourg Dr. Jérôme François - Researcher at INRIA, Nancy-Grand Est, France
Encadrants :	Pr. Olivier Festor - Professor at TELECOM Nancy, France Dr. Radu State - Head of research at SnT, Luxembourg

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503
SEDAN research group at SnT, University of Luxembourg

Mis en page avec la classe thesul.

To all my family and friends, who believe in me much more than I do myself. You are my strength and oxygen.

Summary

List of Figures

Introduction

The Internet started as an experiment escaping the lab in the early 70s and in a few decades time it has become a disrupting technology deeply changing our everyday lives. The foundations upon today's Internet lay are the same ones set at the time of the original architecture design. On the one hand, the targeted simplicity of the original layered Internet architecture has fostered a smooth proliferation of services to be designed and integrated into it. On the other hand, the easiness in the introduction of additional features has sometimes embedded point solutions ignoring the overall architectural framework. The proliferation of such ad-hoc patches has increased the complexity of the original Internet architecture. Because of the struggling deployment of new features over the last decades (e.g., IPv6 or IP multicast), there is a widespread fear that the evolutionary patch-work of the Internet has contributed to its ossification. That is, it has become hard or impossible to create any further innovation at the core of the Internet. Innovation at the edge of the network works, but it constantly faces constraints imposed by the core which remains untouched. However, current Internet's issues with security and mobility models, network management, accounting, content distribution mandate infrastructural changes to be properly addressed. Therefore, a clean-slate design of the Internet architecture is required, that is, a redesign of the network based on a updated set of core principles. With that aim, a considerable amount of clean-slate research on future Internet architectures has been pursued since more than a decade.

Any future Internet design must acknowledge that today's Internet is largely dominated by multimedia content consumption. Users produce and consume several exabytes of contents permonth in a totally distributed and uncontrolled manner. The massive content distribution is supported by the current Internet via ad-hoc infrastructure overprovisioning and overlay orchestration mechanisms. In fact, Content Distribution Networks (CDNs), which represent the mainstream solution to the content distribution problem in the Internet, consist of dedicated servers placed in geographical proximity of users meant to hold content replicas and improve the users' download experience. Nevertheless, the widespread use of CDNs is a far from optimal solution to the content distribution problem. CDNs require agreements with ISPs and DNS redirections. Further, CDNs introduce considerable operational and capital costs which can only be afforded by a few big commercial players.

The need for a dedicated support to achieve efficient content distribution in the Internet stems from the absence in its original architecture design of content-oriented primitives at the network core. Internet was originally designed as a host-to-host communication network. Bidirectional data-exchanges in the Internet are achieved by explicitly including source and destination physical addresses in every IP datagram. Conversely, today's Internet usage is mainly focused on retrieving specific contents rather than on reaching specific sources providing those contents. Information-Centric Networking (ICN) is a clean-slate research paradigm aiming to design a more content-oriented future Internet. By introducing content-oriented routing primitives, widespread in-network caching and asynchronous publish/subscribe-oriented communication models, ICN aims to improve content retrieval and use of network resources in the Internet architecture.

Among the ICN instances, the Named-Data Networking (NDN) has generated a considerable momentum since its original inception in 2006. The NDN architecture envisions to replace the Internet hourglass model's tiny waist, i.e., the Internet Protocol, with a layer working on named-data. Name-based anycast routing and opportunistic in-network storage are included in the NDN layer-3 to find and fetch closest replicas of the required contents in the network. NDN implements an asynchronous communication model based on the exchange of two different network packets, one requesting a content, Interest, and the other one carrying back the content itself, Data. Interests travel through the network without specific destination addresses rather by following name-based forwarding tables' entries to any source of the specified content. For-

warded Interests are recorded by routers in dedicated Pending Interest Tables. Trails are left by Interests in traversed routers for potential Data packets to follow the reverse path to the original requester of a content.
In this dissertation we address the security issues introduced by the forwarding mechanism featured by the NDN architecture. The Pending Interest Table (PIT) in NDN networks allows final users' Interests to create state in the network and achieve an asynchronous address-less Interest-Data exchange. On one hand, the PIT stateful mechanism enables properties like Interests aggregation, multicast Data delivery and native hop-by-hop control flow. On the other hand, the PIT stateful forwarding behavior can be easily abused by malicious users to mount a disruptive distributed denial of service (DDoS) attacks, called Interest Flooding Attacks (IFAs). In IFAs, loosely coordinated botnets can flood the network with a large amount of hard-to-satisfy Interests with the aim to overload both the network infrastructure and the content producers. Countermeasures against the IFA have been proposed since the early attack discovery. However, a fair understanding of the defense mechanisms' real efficacy is missing since those have been tested under simplistic assumptions about the evaluation scenarios. Thus, overall, IFAs still appear easy to launch but hard to mitigate. This open security threat raises serious doubts about the worthiness of the PIT stateful forwarding mechanism in NDN networks.
The work presented in this manuscript shapes a better understanding of both the implications of IFAs and the possibilities of improving the state-of-the-art defense mechanisms against these attacks. The contributions of this work include the definition of a more complete and realistic attacker model for IFAs, the design of novel stealthy IFA attacks built upon the proposed attacker model, a re-assessment of the most-efficient state-of-the-art IFA countermeasures against the novel proposed attacks, the theorization of novel class of IFA countermeasures, the design of an instance of this new class of countermeasures which efficiently addresses the novel stealthy IFAs. Finally, this work also proposes to leverage the latest programmable data-plane technologies to design and test alternative forwarding mechanisms for the NDN which could be less vulnerable to the IFA threat.
The content of this manuscript is organized as follows. Chapter 1 gives an introduction to the research on future Internet architectures. Firstly, it summarizes some of the current Internet issues and outlines the future requirements for a global inter-networking architecture. Secondly, it provides an overview of the Information-Centric Networking (ICN) research paradigm. Finally, it describes the main building blocks of the Named-Data Networking (NDN) architecture. Chapter 2 focuses on the Pending Interest Table (PIT) component of the NDN forwarding plane. Chapter 2 details the PIT component structure and operations, it explores the possible data-structures for a PIT, it discusses the main design challenges for this component and finally it analyzes its directly and indirectly enabled properties. Chapter 3 introduces the Interest Flooding Attack (IFA), an NDN-specific Denial Of Service (DoS) attack exploiting the PIT stateful forwarding mechanism. Chapter 3 identifies flaws in the state-of-the-art countermeasures against the IFA and leverages those to define a steadier attacker model. Further, chapter 3 introduces a new set of IFAs built upon the novel attacker model. Finally, chapter 3 presents a re-assessment of existing IFA countermeasures against the novel IFAs. Results of the evaluation reported in chapter 3 shed a totally different light on the efficacy of the state-of-the-art defense mechanisms against realistic IFAs and so, call for a different set of countermeasures to protect the NDN against this threat. Chapter 4 defines proactive IFA countermeasures, which are novel defense mechanisms against IFAs inspired by the issues with the state-of-the-art ones outlined in Chapter 3. Chapter 4 introduces Charon, a novel proactive IFA countermeasure. Chapter 4 illustrates Charon's design and tests it against the novel IFA attacks. Chapter 4 shows Charon counteracts the latest stealthy IFAs better than the state-of-the-art reactive countermeasures. Chapter 5 explores alternative

forwarding mechanisms for the NDN architecture and illustrates the NDN.p4 design, that is, the first implementation of an ICN protocol written in P4. Firstly, chapter 5 classifies existing alternative forwarding mechanisms with respect to a set of PIT cardinal properties. Secondly, chapter 5 proposes to leverage novel programmable-networks technologies to test and evaluate different NDN forwarding plane designs.
To conclude, the last section of this manuscript summarizes the contributions described in this dissertation and outlines several research directions for future work. With regard to the future work, first, this dissertation paves the way for further improvements in the state-of-the-art defense mechanisms against IFAs. More precisely, findings emerged throughout this work show the potential for hybrid reactive/proactive and/or control-plane assisted countermeasures to be designed and counteract IFAs more effectively. Second, this dissertation proposes to leverage the latest technologies in the field of programmable data-planes for the design of NDN routers. The NDN.p4 design was proposed shortly after the P4 language vision had emerged, so it was somehow constrained by that technology's infancy. Since then, the P4 language ecosystem has considerably evolved. Today, P4 features a richer language feature-set and is supported by many hardware/software targets. Recent advances in P4 can be used to refine the NDN.p4 design and fairly compare it to the state-of-the-art NDN router designs.

Chapter 1

Future Network Architectures

The design principles for today's Internet had been developed in the 60s and 70s as the main outcome of a US defense research organization, the Advanced Research Projects Agency (ARPA), project. Less than twenty years later, the rationale that had shaped the design of the early Internet protocols was elegantly summarized in [3]. The first Internet design mainly aimed to interconnect existing packet switching networks to provide some larger services. On top of that, some secondary requirements were set, like reliability under transient network disruptions, support for different types of transport services, distributed network management tools, etc. The main goal of that early design was to provide a general and flexible infrastructure upon which to easily implement and run diverse services. The realization of Internet built upon that original abstract architecture has, over the years, surpassed the initial expectations. In fact, today's Internet accommodates a much broader range of services.

Since the explosion of the World Wide Web, a new set of requirements and trends of use compromise some of the Internet original design principles and strain the current network infrastructure. Most of the time, additional features have been added to the architecture as point solutions to meet specific, yet legitimate, requirements (e.g., mobility, security, IP addresses scarcity). Thus, some sort of absence of a broader "architectural thinking" has often embodied patches which, in the best case, have only broken some of the original design principles (e.g., several flavors of middlebox violate the end-to-end principle [4]), but which, in the worst case, have introduced undesired and conflicting feature interactions [5] (e.g., intrinsic IPsec/NAT incompatibilities). Indeed, this incremental refinement process has increased the architecture complexity and compromised its flexibility therefore harming the conception and the smooth deployment of any future feature. Ideally, overturning this evolutionary path would imply to only develop and introduce new features coherently with the existing architectural framework. Nevertheless, today's really different requirements would require to devise a completely new architecture. This latter process is usually referred to as "clean-slate design" [6].

A clean-slate approach implies rethinking the basic assumptions and design principles of the architecture to address simultaneously all the newly-emerged challenges that today's Internet is facing with. Many scientists question clean-slate research in networking as "impractical" since that does not lead to any short term deployment. Others think this intellectual exercise is first beneficial to the ongoing incremental Internet evolution and then plants seeds that will enable research experiments to consolidate in future networks [7]. For those reasons, several big research frameworks [8] have been launched worldwide to encourage clean-slate research on the design of future Internet architectures. Although the projects differ from each other, there seems to be a small set of common goals that they all pursue. Those goals raise from the Internet require-

ments of security, support for mobility of terminals, cost-effective and resource-efficient massive content-distribution, infrastructure scalability, need for fine-grained and seamless accounting and network management tools. Moreover, since the original design of Internet, many factors have changed like advances in technologies, stakeholders, new application requirements, role of and trust in involved parties, service differentiation at the network core. This produces a broader perspective on the design of any internetworking architecture including end-users and economics aspects too.
Beyond the peculiarities of each proposed design, there are a few core factors which any new architecture design should consider. These factors could be gathered in five main categories according to the domain of interest, although these domains are tightly interrelated:

- entities: at the beginning Internet was engineered for host-based communication between expensive mainframes. Since then, network participants have increased and, just to name a few, now include normal users, Internet Service Providers, government agencies, telco carriers, large commercial players (e.g., Google, Facebook, Microsoft, Apple). More actors bring more interests and so increase the conflict of interests space [9].

- communications models: there has been an evolution from the single original client-server model to multiple communication models. The World Wide Web (WWW) and other technologies like Email have dictated a usage trend based on content retrieval from specific sources and access of services. Moreover, peer-to-peer (P2P) networks have boosted a model where content can be fetched simultaneously by different hosts found through some middleware support. Yet, the emergence of platforms for sharing user generated content (e.g., Youtube, Instagram, Flickr) has also considerably increased the daily quantity of content produced and consumed via the global network. Finally, the adoption of the Cloud Computing paradigm has generated yet another kind of traffic which offloads computation and storage to resource pools distributed in the network.

- advances in technology: several technical challenges must be addressed when proposing any new architecture realization. Therefore, it is important to know the current, and to speculate on the foreseeable, hardware and software toolset which may be leveraged to address those challenges. For example, the scalability of a proposed routing approach or a new naming system may look unfeasible according to the state of art. However, a combination of technology advances and new algorithmic solutions may overcome this barrier in the short term (e.g., when IPv6 was proposed, there existed no algorithms to compute prefix lookups in microsecond for those longer IP addresses, later on, a binary search on prefix lengths algorithm was invented [10]).

- multi-requirements network: the conception of the original Internet architecture had envisioned a limited-size research network rather than a globally deployed one. Hence, a unique list of requirements was considered for the whole network. However, the diversity of today's Internet created by the multitude of actors and trends of use inevitably leads to a scenario where no fit-all list can be conceived. Some requirements will apply to some portions of the network meanwhile they will strain in others.

- trends of information consumption: today's Internet traffic is dominated by the access of contents. According to the latest Cisco's traffic forecast [11], the expected traffic growth will increase threefold over the next 4 years, raising the bar to 127-fold from 2005 to 2021. The busy-hour Internet traffic grows more rapidly than the average one, bringing up

the phantom of the flash crowd effect. By 2021, traffic from wireless and mobile devices will account for more than 63% of the the total Internet traffic. When looking at the consumer traffic by subsegment, "Internet Video" and "Web, email and data" largely drive the consumption.

Among the above factors, two issues deserve a particular attention since these have to be constantly addressed to keep the Internet alive and operational: i) the increase of the global traffic and ii) the prominent use of mobile devices to consume/provide data. Both factors fear the Internet collapse because of the poorly scalable and resource demanding solutions, namely, infrastructure over-provisioning (as done, for example, with Content Delivery Networks) and network handover mechanisms (e.g., mobile IP), which have been so far devised to deal with them.
On one hand, the same forecast [11] says that 70% of all Internet traffic will cross Content Delivery Networks (CDNs) by 2021. CDNs consist of strategically deployed data centers to serve users in the geographical vicinity. The main aim of CDNs is the reduction of the latency experienced by end-users by bringing content copies closer to them. However, the service is a complex overlay solution which brings additional capital and operational costs, requires an agreement with a CDN provider and a DNS redirection mechanism. On the other hand, Mobile IP [12] is today's de facto standard to guarantee continuous connectivity despite mobility of host terminals. The Mobile IP protocol is based on multiple software agents which perform either tunneling or address translation. Although mobility is provided transparently to end-users, this approach has several security issues, introduces triangulation routing problems and, of course, additional unpredictable latency.
Many of the clean-slate Internet research proposals use a holistic approach to deal with the above factors. That is, their design aims to fulfill several of the requirements (security, mobility, reliability and availability, problem analysis, scalability, quality of service, economics [13]) for a global network at the same time. However, factors and requirements are variegated and so generate a complex interaction space. Hence, research projects usually put the main focus on one specific issue and then build around that to address some other primary requirements. For example, the Mobility project [14], funded by the NSF Future Internet Architecture (FIA) program, envisions a network design to ease the mobility of terminals. Then, as direct consequence of centering the design on the mobility requirement, other factors deserve immediate attention, likewise security and trust models for the open wireless access, name resolution and storage-aware routing protocols. Or yet another FIA project, the Expressive Internet Architecture (XIA) [15], mainly targets simultaneous support for diverse communication entities (e.g., users, services, content), called principals, meanwhile it enforces other properties like intrinsic security and flexible addressing.
According to the actual large deployment and consequent usage of CDNs, any future Internet architecture will mainly need to deal with large-scale content distribution. Yet, this mainstream usage trend is sometimes either neglected or treated as secondary by most of the related research projects. Instead, a content-oriented approach is taken by another branch of clean-slate Internet research, called Information-Centric Networking (ICN). The ICN paradigm promotes the foremost role of the information in future network architectures. Among other design principles, ICN designs aim to empower directly the network architecture with mechanisms (e.g., ubiquitous in-network caching, content-oriented routing primitives) to efficiently support massive information distribution. The resulting content-oriented networks feature properties to promisingly address future Internet requirements and factors. Therefore, throughout the last decade, research on ICN has gained considerable traction and produced important scientific insights into the design of future networks.
The motivation and the general idea behind the ICN vision is presented in Sec. 1.1 where related

research projects, tenets and research challenges of this paradigm are also outlined. Next, in Sec. 1.2 a specific instance of ICN is illustrated since it has been extensively target of study in this work. Finally, Sec. 1.3 explains how some other latest research on networking is (and will be) used to ease the deployment of and further experimentation on ICNs.

1.1 Content as first-class citizen of the future Internet

Information-Centric Networking (ICN) is a clean-slate approach to networking which has emerged to better support the most prominent content distribution models in the Internet. In fact, the current Internet architecture and the related set-of-protocols struggle to serve efficiently today's massive production and consumption of multimedia contents through the global network. The key cause for such inefficiency can be better explained by looking at the architecture hourglass model depicted in Fig. 1.1. The Internet Protocol (IP), which lays at the waist of the hourglass, offers a basic connection-less best-effort service to the upper layers and accommodates many lower different layer-2 technologies. The basic service offered by IP was meant in the early design to inter-connect a variety of existing packet switching networks. Thus, for simplicity, the IP narrow waist constitutes a datagram network, where packets are routed independently based on the destination address carried in their header. The basic IP packet processing logic mainly deals with hop-by-hop datagram forwarding according to pre-populated routing tables from one source address to a destination one. Therein, in its simplest form, a datagram header is agnostic to the payload content and, by consequence, does not provide information to perform any application-specific forwarding optimization.
At the dawn of the Internet, the intended simplicity of the original IP datagram did not seem too constraining. Moreover, additional datagram labeling mechanisms had been later devised to meet potential generic traffic classification requirements (e.g., the DiffServ protocol [16] to differentiate quality of service in network traffic). However, despite the further refinements, the original datagram unit design still implies that sometimes thousands or millions of identical (i.e., containing the same application layer payload) IP packets may be carried needlessly on the same path. Moreover, transferred data are hardly reusable after the end of a transport session, since they have not been stored anywhere on the path and their security, if any, was tied to the specific recipient providing them. Further, if either one or both communication endpoints move during the data transfer, the transport session has to be kept or re-established for the transfer not to be disrupted (by no surprise, the intricacies of protocols to support host mobility, like MobileIP, stem from this last requirement).

The above observations seem to represent solid arguments for the design of a different Internet whose main focus is information more than communication endpoints. As the definition may suggest, the main goal of ICN is to shift the focus of the current Internet architecture from 'where' a certain information is stored to "what" the information is actually about. Hence, content[1] itself becomes the first-class citizen of the network. This would imply a shift in the semantics of the datagram unit because that would need to convey some sort of indication about the packet payload. This shift mainly aims at more efficient content localization and distribution at the network-level. Nevertheless, decoupling contents from specific sources also opens novel promising perspectives to rethink solutions to deal with other collateral issues: mobility of terminals, IP addresses scarcity, security of data-exchanges and fast recovery mechanisms from network disruptions.

[1] The terms Content, Information, Data are often used as synonyms in related literature, so we do that here too.

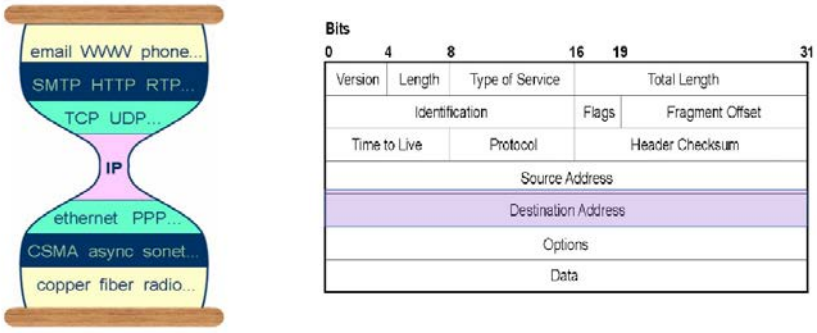


Figure 1.1 – The hourglass model of today's Internet and the IP datagram structure.

A network architecture describes a "set of abstract principles for the technical design of protocols and mechanisms for computer communications" [5]. Hence, an architecture defines design principles which serve as guidelines for the technical design of the network itself. Obviously, an architecture is a design philosophy and may lead to several technical instantiations. The design of an ICN architecture is an open research question which has so far led to different realizations. An overview of some past and current research projects on ICN is given in Sec. 1.1.1. Common goals of ICN proposals are listed in Sec. 1.1.2. Finally, advantages of deploying an ICN are advocated in 1.1.3, while the major open challenges in the field are illustrated in Sec. 1.1.4.

1.1.1 Research on Information-Centric Networking

The genesis of the research on ICN is hard to track back precisely since it should be fairly attributed to ideas converged from several domains into a unique research area. A tentative timeline of ICN research from the TRIAD project onwards is presented in [17]. However, the author does believe that a chronology should be differently sketched outlining the birth and growth in parallel of some seminal ideas which made the soil fertile for the emergence of further more structured research frameworks. So, he rather proposes classifying existing related literature and research in three distinct epochs. The classification is illustrated in Fig. 1.2.

The first generation of research on ICN, whose research projects are here named "the ancestors", includes four main works on the design of a more content-oriented Internet architecture. In a certain sense, those works laid down the foundations for most of the further related research. In fact, they had identified the primary goals (improving security and performance of content distribution) and some key challenges (e.g., naming, routing and forwarding, security and trust models) in the design of such a future network as well as suggested some preliminary solutions later leveraged and refined by the next ICN architectures. The ancestors are:

- Standford's TRIAD project [18]: in the early 2000s the TRIAD work [19] is often credited to having pioneered the field with the first seminal proposal for a complete content-oriented Internet architecture. TRIAD's authors feared the inadequacy of transparent web proxies, CDNs and load balancing switches at web sites to implement content routing at the Internet scale. Those existing solutions are often proprietary and not efficient, further, violate some architecture principles. Therefore, TRIAD proposed an IP-compatible "content layer" to perform efficient content routing to content replicas in the network.

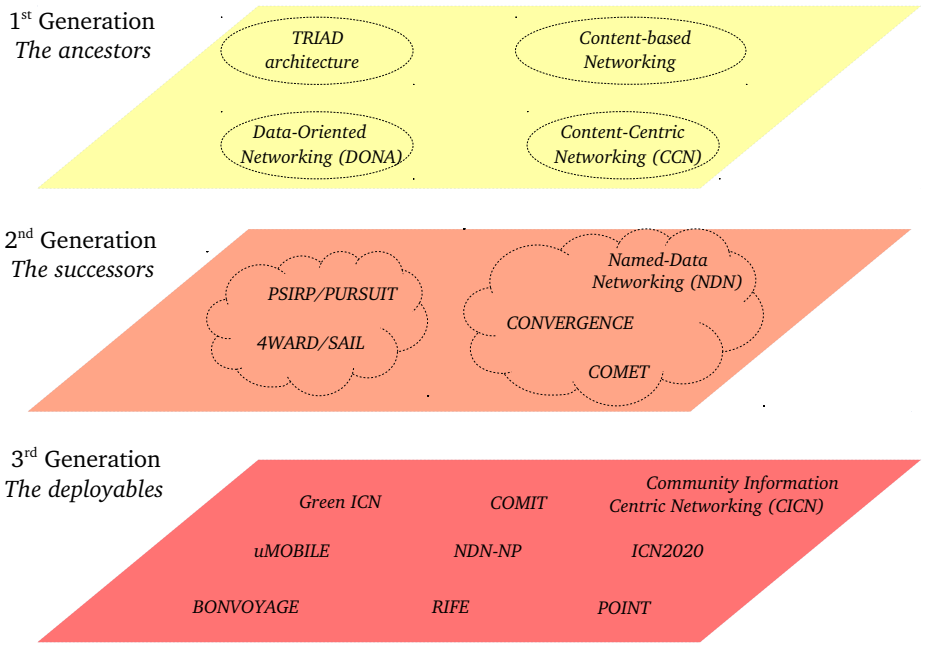


Figure 1.2 – The genealogy of the research on ICN from TRIAD to the current days.

- PARC's Content-Centric Networking (CCN) [20]: this work was first advertised at Google [21] by Van Jacobson, and later on published in [22]. CCN treats content as the main network primitive, so departing from the address-based routing model of IP. CCN aims to improve content availability and security by devising a content-oriented level for the thin waist of the Internet's hourglass model. The authors illustrate the new CCN layer-3 protocol proposing packet formats and the router's forwarding plane. Further, they implement an open source prototype of the CCN stack and show its efficiency for both file transfer and point-to-point network protocols.

- Berkeley's Data-Oriented Named Architecture (DONA) [23]: DONA proposes a clean-slate Internet to achieve persistence of names, availability and authentication for data and services. The architecture mainly focuses on naming (using Self-Certifying Names) and name resolution (made by anycast routing on names) to establish the three aforementioned properties. DONA also uses ubiquitous caching and names lower in the network stack to achieve routing to nearest copies. The related work presented in [23] acknowledges that this proposal merges several early ideas in an coherent architecture.

- Carzaniga et al.'s Content-Based Networking (CBN) [24]: a theoretical framework for CBN, which is proposed in [25], defines the required building blocks and primitives for a content-oriented communication infrastructure. Content-based networks are meant to offer better support for several classes of applications where data-transfers are triggered by receivers' interests without specific sender addresses. In modern networks, that communication model

Table 1.1 – The ancestors' problem definition and goals.

Project	Main targets	Main challenges
TRIAD	to scale content delivery	efficient content routing
CCNx	to improve current network scalability, security and performance	naming and content-based security
DONA	to achieve persistence of names, availability and authentication of data	naming and name-based network primitives
CBN	to support the services of a publish/subscribe middleware with a new communication infrastructure	efficient routing and forwarding on novel datagrams

is often achieved via content-oriented publish/subscribe middleware which, however, does not scale well on large communication infrastructures [26]. Therefore, CBN mandates new specific datagrams and functions on datagrams to be designed. As most pressing milestone, the work on CBN tackles the imminent need to explore the design space of content-based forwarding and routing algorithms [27, 28].

It should be recognized that at that early stage these works were not fully aligned on the same set of goals. The different aims of the *ancestors* projects are shown in Table 1.1. Nevertheless, they had presented many similarities (e.g., the exigence of some kind of in-network storage, name-based routing algorithms, name-resolution mechanisms) which were leveraged later by a second generation of projects.

The "successors" are projects which build upon the directions and the problems identified by the ancestors by analyzing technical solutions in more mature architectural frameworks. They are not presented anymore as separate ellipses in Fig. 1.2 since they do not stand anymore as fully separate projects. In fact, they can rather be grouped in macro categories which have taken mutually exclusive decisions on some core design principles like naming and routing (for a definition of those design principles see Sec. 1.1.2). On one hand, the PURSUIT project and its predecessor PSIRP [29] propose a clean-slate Internet architecture implementing a receiver-driven Publish-Subscribe paradigm. PURSUIT/PSIRP leverages the flat identifiers naming system proposed in DONA, but it features rendez-vous nodes to match publications to subscriptions and topology-manager nodes to build forwarding paths for content delivery. Further, it enhances the flat identifier names with the notion of scopes in order to organize the information distribution in the network and improve its reachability. Or, the Network of Information (NetInf) architecture proposed by the 4WARD/SAIL project [30] is based on a hybrid name-based routing and name resolution scheme. NetInf provides access to named Data by either forwarding requests by name or by querying a Name Resolution service or both. As PURSUIT/PSIRP, NetInf uses a flat namespace to achieve persistent and location-independent naming of contents. On the other hand, architectures like NDN [31] (at the time of its inception, the academic fork of the PARC's CCN) and Convergence [32] use a hierarchical human-readable namespace. Routing is done by names in both architectures. NDN assumes routing tables are populated with aggregated name prefixes by means of name-based variants of intra-AS and inter-AS routing protocols. Convergence includes a hierarchical name resolution service, like in DONA, which registers prefix names

advertised by content providers and can be queried to get routes to contents.
Finally, no precise naming schema has been defined in CURLING which is the ICN architecture proposed by the COMET project [33]. In CURLING, names are provided by a Content Resolution System (CRS) where publishers previously registered this information. With regard to the name resolution and data routing, COMET uses a hybrid approach. Name resolution and information localization use the CRS nodes which in turn set up the data routing path from the source of information back to the original requester.
Yet, all the *successors* projects contributed to advancing the understanding of important technical challenges in the design of Information-Centric Networks. For example, they did research on the scalability of the routing tables with different naming schemas, on routing protocols and forwarding mechanisms, on security properties and trust models, on novel content-oriented network APIs.
The contributions made by the *successors* projects have fueled a third generation of projects that are hereby named "The deployables". These projects (some of which are still ongoing at the time of writing this manuscript) focus on both large-scale deployment of ICN prototypes and the application-driven iterative refinement of the respective architectures. Application scenarios of this third generation include distribution of critical information in the aftermath of a disaster (the Green ICN project [34]), commercial viable IP-over ICN deployment (the POINT project [35]), the optimization of the transport of passengers and goods (the Bonvoyage project [36]), mobile healthcare and building automation and management systems (the NDN Next-Phase [37]). This newest generation of projects pursues an application-driven evolution of the ICN architectures designed so far. The goal is testing ICN solutions/architectures against selected scenarios of use and enhancing them accordingly in an iterative fashion. Further, this research phase aims at two more targets: i) helping industry partners to develop Proofs of Concept of products and services exploiting ICN, ii) deploying more experimental facilities, ideally test-beds at global-scale, to advance the state of the research on ICN.

1.1.2 Design tenets

As illustrated in the previous section, past and ongoing research projects on ICN differ in the architectural solutions proposed, meanwhile they share several assumptions and goals. Therefore, it is reasonably possible to identify a set of design principles which are valid across different ICN architectures. The author summarizes the principles in one sentence before providing an in-depth definition for all of them: "Everything is information, Information is named, Names are scoping/forwarding hints".
First, as the paradigm name suggests, the Information is at the core of every ICN architecture. The term is used generically to identify any kind information which is exchanged among network participants. Second, information needs persistent and unique names. Third, the names are to be used across different layers to improve the content localization and distribution.
Core principles of ICN are:

Naming of Information: in ICN, identifiers are associated to every piece of information traversing the network. Those identifiers hold some important properties. They are unique. They are independent of location, of source (e.g., original producer, in-network replica, repository) providing them, and of the way the related data is retrieved. In general, their semantics, if any, is opaque to the network layer.
There have been mainly two naming schemes adopted by existing ICN architectures: hierarchical *human-readable names* (HRN) and flat *self-certifying names* (SCN). An example of both

name types is given in Fig. 1.3. On one hand, HRNs are often represented as URIs made of a sequence of variable length (mostly semantically meaningful) strings separated by a delimiter. The advantages of HRNs is that they are easy-to-understand, semantically meaningful, easy to aggregate (aggregation at large-scale of a similar naming scheme is done in today's DNS). On the other hand, SCNs are meant to provide with the name itself the necessary information to verify some Data's security properties. Hence, SCNs are often made out of the hash of the content provider identity (e.g., the public key) concatenated by the hash of the content identifier.

Figure 1.3 – An example of both HRN and SCN.

The schemes hold different properties and suffer from different issues. Naming properties for ICNs have been brilliantly outlined in [38] where HRNs and SCNs are compared in terms of security, scalability and flexibility, which are important requirements for any Internet architecture.

Communication Model: ICN architectures pursue publish/subscribe models as the main communication model in their networks. Thus, the data exchange between consumers and producers of information is asynchronous. Among other properties, this model provides two main advantages which are leveraged by the architecture to implement a content-oriented data-exchange. First, the model decouples senders and receivers in space and time. Second, the publish/subscribe model gives more control both to the network, on how the information is stored and delivered, and to the consumers, which can better specify what information they are interested in fetching.

Self-Secured Data or Data-Centric Security: ICNs data are stored independently from specific network locations. Therefore, today's end-to-end security mechanisms (e.g., the Transport Layer Security protocol) cannot be used to implement secure communication in those networks. Instead, security must be ascertained through the content itself since that can be stored in and retrieved by any location in the network. To achieve that, every data must carry some additional meta-data information which can be used to assess its security properties. This security model usually requires a cryptographic technique applied to the name and the data itself. For example, cryptographic signatures binding names and data can be used as small in-packet anti-tamper integrity proofs. It must be told that the choice of a certain naming schema has direct implications on the security.

In-Network Storage/Caching: ICN promotes to store popular contents in the network. For that purpose, some ICN routers are supposed to be equipped with local storage to perform application-independent opportunistic caching of data. This capability will allow routers to serve requests locally if they hold a copy of the related contents in their caches. Obviously, in-network caching contribute both to improve data-availability and to reduce data-delivery delay. The benefits of using in-network caches seems to be further supported by the evidence that content requests in today's Internet exhibit a high spatial and temporal locality [39].

Application-Program Interfaces: an ICN mandates a new communication API for applications to interact with the network layer. In fact, today's most dominant interface to the network, BSD's Sockets, does not fit a data-oriented model. A socket binds an application to both a specific protocol and an endpoint. As advocated by previous work, a perfect network abstraction should break spatial and temporal bounds [40]. Any ICN layer-3 protocol should expose to upper-layers applications primitives to publish and retrieve data. Different proposals have been advanced so far. They differ yet they usually refer to contents by name and may also take additional parameters to define scope of publications and/or filtering criteria for searches.

Routing and Forwarding: in ICN, data-oriented algorithms and protocols are used to route requests and responses in the network. The related mechanisms are strictly tied to the naming system adopted by the architecture and may demand additional components, e.g., a name resolution system which converts object identifiers to network locators. A cornerstone element of these protocols for ICNs is that they should be designed to allow the network to locate and fetch nearby copies of the requested contents. Sometimes this requirement is achieved on the fly inside the network by adaptive forwarding mechanisms, while other times there are omniscient network components which translate content identifiers to network locators.

The above principles have been unanimously stated in several related works [41, 17], however, the author defines two more novel principles worth outlining:

- *In-Network Capabilities*: all the ICN architectures envision a "more sophisticated" processing done at layer-3 which may also require the existence of additional components, e.g., local caches in routers or name resolution agents.

- *Cross-Layer Information*: ICNs have architecture components like names or identifiers which are carried across different layers. The components' semantics serves different purposes according to the layer where that is processed. This stems from the ICN objective to implement generic network services yet meeting application-specific requirements. This is a shift from the current IP model where, for example, the BSD socket abstraction requires applications to specify endpoints and select specific transport protocols. Those two pieces of information do not allow applications to specify any requirement beyond a small set of transport layer options. Conversely, in ICN architectures, names often specify requirements the network layer can build services upon. For example, names can embed access control scheme semantics in NDNs, or yet, they can scope information in PSIRP networks.

1.1.3 Benefits

On the one hand, detractors of this network revolutionary paradigm often claim that indeed ICN does not offer many advantages beyond the ones produced by in-network caching of contents [42, 38]. On the other hand, advocates of the ICN postulate and research on scenarios to showcase the potential manifold benefits of a data-oriented architecture. Hereby follow some of the improvements that an Information-Centric Network could bring:

Scalable vendor-free cost-effective content delivery: the delivery of content across the Internet has been improved by means of several content distribution mechanisms like Web caching, Content Delivery Networks, carrier network Transparent Caching. These solutions have led to the deployment of storage resources owned by different operators across different network segments. Hence, they are suboptimal in terms of required resources, management costs and further they show low or no interoperability. Instead, ICN "democratizes" the content distribution freeing the content providers from buying specific services from the CDN operators and from relying on the ad-hoc infrastructure deployed by the ISPs in order to achieve an efficient distribution of their contents. By implementing content distribution features at the network level, ICNs breaks the current monopoly of CDN operators, ISPs and big content providers. Though this scenario calls for new accounting schemes for ISPs and content providers [43].

New security/mobility models: by decoupling senders and receivers in space and time, ICN opens new security and mobility scenarios. With regard to the security, ICN finally untangles Internet from the authentication of every sender, receiver and channel between them. In fact, ICN delineates a security model where every content has to be secured independently of the source providing and/or storing it. The ICN content-oriented security model theoretically avoids specific nodes to be target of an attack because of the absence of addresses. With regard to the mobility, ICN applications do not necessarily need to track down mobile terminals since every data-exchange is only based on a named content. Freeing the applications from the notion of topological locators radically changes the layer in the stack where to deal with the mobility of consumer and producer nodes [44]. In ICNs, content availability is determined by the network.

Reduction of middleware: most of today's applications are data/service-oriented and based on a publish/subscribe interaction scheme [45]. Those applications do not deal with the security, the performance and the reliability of the data transfer which are delegated to the lower layers in the network stack. This implies that from the application layer to the network one there often exists some middleware mapping between those different layer abstractions. The adoption of an Information-Centric Networking would reduce the intricacies of those in-between translation mechanisms by means of i) novel content-oriented APIs for application developers to access network services, ii) semantics-rich content identifiers whose expressiveness may be leveraged down to the network layer to provide several services beyond efficient content delivery.

Glue for the Internet fringes: it seems that a unified system to inter-connect different Internet-of-Things (IoT) *silo* solutions is still missing. The proliferation of ad-hoc solutions has been partially confined by the emergence of a few unifying standards, e.g., the 6LoWPAN protocol. However, the IoT ecosystem still proliferates with diverse solutions as a proof that "the TCP/IP stack is not a good fit to the IoT environment" [46]. The data-centric communication models of ICNs offer promising support to address the challenges of data retrieval delay, resource-constrained devices and intermittent network connections, in many IoT use cases [47].

1.1.4 Open challenges

Beyond potential benefits, the design and the deployment of Information-Centric Networks raise numerous challenges too. The main technical challenges for an Information-Centric Networking are hereby outlined, while a more comprehensive summary can be found in [48].

Naming: the main purpose of a name in ICNs is to provide persistent unique location-independent identifiers for contents. Meanwhile, a name can be used to create a verifiable binding between the content and the name itself. This binding can be used to verify that the content received corresponds to the one requested by that name (the literature on ICN refers to this property as *content integrity*). Therefore, the choice of a naming scheme for an ICN influences design choices related to security and trust models. Sec. 1.1.2 reported about the two main approaches to naming in ICNs: hierarchical human-readable names (HRNs) and flat self-certifying ones (SCNs). Those schemas exhibit different properties in terms of the implications on the security and the routing. Both schemas enable data-integrity and data-authentication mechanisms, even though, they do this by means of different solutions.

Scalability of the routing system: as raised in [38], any ICN architecture should be able to deal with a large number of objects. Conservative estimates suggest 10^{12} objects. That order of magnitude would have a direct impact on the size of the routing tables in ICN-enabled routers. Today, routing tables must be relatively smaller, roughly 10^6, for IP routers to operate at packet speeds in certain network segments. By consequence, the mismatch between size manageable by today's routing technologies and foreseeable ICN scenarios must be solved somehow. Classless InterDomain Routing (CIDR) has taught us that aggregation could be a viable solution to reduce the size of routing tables. Along this line, a similar aggregation of domain names in Domain Name Servers suggests HRNs could be shrunk to generate relatively small ICN routing tables. Nevertheless, the choice to adopt HRNs could limit caching effectiveness by always routing requests towards the original providers. This argument was first raised in [38], however, since then, adaptive forwarding techniques[2] have been included in some ICNs [50], which could potentially neutralizes this side-effect. In fact, an adaptive forwarding behavior would contribute to contents be fetched through and cached across different network paths to a set of content sources.

Security and privacy: ICN promotes a content-oriented security model where no secure channels exist between users and content providers. Implications on the security and privacy models naturally arise from this design choice. This novel security model immunizes the network against some well-known vulnerabilities which were caused by the Internet's host-centric communication model. However, ICN introduces novel components and features which open new attack surfaces. For example, this manuscript thoroughly analyzes the security vulnerabilities of a class of ICNs performing stateful forwarding. Furthermore, neither source nor destination addresses are included in ICN packets, hence, parties involved in a data-exchange benefit from anonymity. However, the name of and other information about every requested content are available to the all the nodes processing the related packets. This generates privacy issues on multiple dimensions: in-network caches, contents, names and signatures [51]. The revealed information can

[2] An adaptive forwarding plane [49] can explore multi-path routes in response to specific network conditions or as a simple probing mechanism to find more efficient paths.

be exploited for different purposes like censorship, tracking users' behaviors, non-repudiation. Therefore, ICN proposals have to be carefully designed to support better privacy. A good survey of the related work identifying risks of and proposing mitigation mechanisms for privacy issues in ICNs can be found in [52].

Caching: there are four main challenges to achieve ubiquitous in-network caching. The first one relates to the decision of which nodes are eligible to cache contents, which is referred to the *cache placement* problem. The second one relates to the distribution of contents to deployed caches, which is referred to *content placement*. The third one deals with the efficient routing of content requests to the closest caches, which is referred to *request-to-cache* problem. The fourth one relates to the policy to empty and/or replace contents when there is no more space in cache, which is referred to the *cache replacement policy*. The above challenges are usually addressed according to performance/gain the caching techniques aim to achieve. Among the goals, delivery latency, cache utilization and network load are usually considered. Some of the performance indicators are conflicting, thus, some caching techniques prioritize one metric over another one.

Deployment: although they may be considered of secondary issue in the early investigation of a clean-slate networking paradigm, activities like prototyping, simulating, testing are of foremost importance to validate and explore the implications of a network architecture design. The goal is even more challenging since the needed experimentation should happen at large scale and under realistic traffic conditions. A definition of such an experimental facility for clean-slate research on networking is provided in [13]. Current deployment challenges for ICNs include the design of ICN-compliant forwarding elements, the set-up of large-scale testbeds, standardization activities, solutions to achieve integration with and backward compatibility to the existing infrastructure.

1.2 A primer on Content-Centric/Named-Data Networking

Among the available ICN realizations, CCN and its academic twin NDN have achieved a certain degree of maturity since their inception. Today, both ICN platforms and protocol specifications are widely used as reference by the research community to further investigate challenges and opportunities in the Information-Centric Networking arena. This kind of architecture has been mainly investigated in the context of this work, hence, it is better illustrated in this section to help the reader familiarize with the architecture components and details which will be often referenced in the rest of the manuscript. The rest of the section is organized as follows. Sec. 1.2.1 illustrates a timeline of both projects to give some historical perspective about the development of the architecture. Sec. 1.2.2 presents the architecture and describes core components like naming, the routing and forwarding plane, security and privacy, applications. Finally, Sec. 1.2.3 outlines the current status of the technology with current and future research directions working on the refinement of the architecture.

1.2.1 Historical notes

The Content-Centric Networking (CCN) architecture is the outcome of a research project on ICN launched at the Xerox's Palo Alto Research Center (PARC) in 2007. The idea behind CCN was anticipated through a speech given at Google in 2006 [21] by Van Jacobson, who had

joined PARC to lead this project. PARC's CCN proposes a communication architecture based on named content chunks rather than on named hosts. The vision and the design principles were stated in the seminal paper [22]. PARC made the project public on a website in 2009 when it also released an open-source software implementation of the proposed protocols to build content-centric networks, called CCNx. Since then, the PARC's CCN project has devoted a certain effort to document the CCNx reference implementation (by periodically releasing protocols specifications and software bundles) and promoted its adoption to advance the status of research on content-oriented networks.
The original CCN architecture has been iteratively refined over years of research on architectural challenges and applications design done by PARC and the NDN's consortium. A precise timeline of the CCN/NDN project is out of the scope of this work. Nevertheless, the rest of this section tracks the evolution of the two siblings architectures through some milestones which are illustrated in Fig. 1.4.

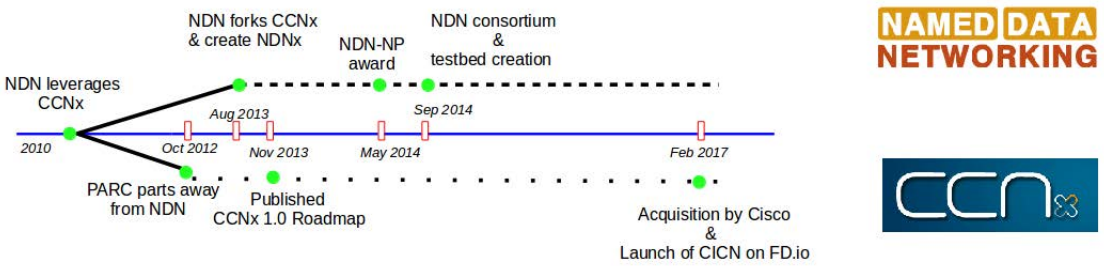


Figure 1.4 – Timeline of Named-Data Networking and Content-Centric Networking projects.

Shortly after the release of the open-source software implementation, the ccnx source code was leveraged for research by a large consortium of US universities running one of the five successful NSF Future Internet Architecture (FIA) projects, i.e., the Named-Data Networking (NDN) [53]. Around October 2012, the CCNx and the NDN projects got separated. First, a few months after the separation, PARC parted away from the NDN team and stated its design goals in a roadmap document for the release of CCNx 1.0 [54]. Then, in August 2013, the NDN consortium created a fork of the ccnx code. The separation has been most likely driven by irreconcilable divergences in design choices (see the Appendix A for the "Differences between CCN and NDN"). Since the separation, CCNx and NDN have diverged on several features and there exists no comprehensive report documenting those[3].
On one hand, PARC has contributed to the creation of important drafts which have been adopted as working documents from the IETF ICNRG [56, 57]. In 2017, Cisco announced the acquisition of the CCN platform from PARC with the goal "to foster convergence of various dialects of ICN (namely CCN and NDN) into a single harmonized version of ICN, promoting wider and faster adoption of ICN-based solutions required to solve future networking needs" [58]. Cisco has merged the acquired CCNx code with other of its contributions and integrated all of it into the Community ICN (cICN) [59] open-source project hosted by the Linux Foundation.

[3]Since July 2016 the ICN research community has committed to precisely document and harmonize those differences via the creation of an IETF-ICNRG work item [55].

On the other hand, the NDN project has so far gathered together in a consortium roughly 30 members equally spread across founding universities, industrial partners and other academic institutions. Undeniably, the NDN consortium has been able to foster a vibrant ecosystem of research on future Internet architectures beyond the NDN architecture itself. Since 2014 the NDN consortium has been pursuing more research on and validation of the NDN architecture over the key areas of Health IT and Cyberphysical Systems under the umbrella of a next-phase NSF's award [37].

1.2.2 The Named-Data Networking Architecture

Communication in NDN is achieved asynchronously by means of two different packets: Interest and Data. An Interest represents a request from a certain consumer for the network to fetch some content or access some service, while a Data contains the content/service itself. Neither Interests nor Data carry any address, since both packets are named instead and the names are used to route them across the network. The communication is receiver-driven; hence, Interests trigger Data to be fetched and forwarded downstream to consumers. Every NDN router holds the three following main data-structures:

- Content Store (CS): the router's local cache used to opportunistically store Data packets.
- Pending Interest Table (PIT): a table recording forwarded Interests which have not been consumed by any Data yet.
- Forwarding Information Base (FIB): a name-based forwarding table performing component-wise longest prefix matching and determining output interfaces to forward Interests.

An illustration of an Interest-Data exchange in NDN/CCN routers is depicted in Fig. 1.5 where the content of a router's data-structures during an Interest-Data exchange is showed too. The consumer Alice expresses an Interest to fetch the file "ndn-protocols.pdf" held by Bob. Alice's Interest traverses in sequence the routers R1-R2-R3. At each router the following operations take place. First, the router checks if it holds a local copy of the requested content by looking up its CS. If so, a copy of the content is sent downstream through the interface where the Interest was received. If not, the Interest is looked up to the PIT to check if there are similar pending requests. If a PIT entry for the same name exists, then the Interest is dropped while the PIT entry is updated if the interface has not been seen yet. If the router does not have any PIT entry, then the FIB is looked up to forward the Interest to a next hop upstream. The FIB lookup consists of a Longest-Prefix Match (LPM) on the name components. This LPM selects the FIB entry with more name components over all the matching entries. The matching FIB entry provides one or more output interfaces out of which the Interest can be forwarded. If there is a matching FIB entry, the Interest is forwarded and a new PIT entry tracking the Interest name and the incoming interface is created. The status of R2's PIT after the Interest is processed is reported in Fig. 1.5 and it consists of one entry with the Interest name as key and the incoming interface as value. Once a Data packet is found, e.g., when Alice's Interest reaches Bob, that is forwarded downstream on the reverse path by following the respective PIT entries. At each router, processing of Data consists of checking if the Data is solicited (that is, corresponding to any PIT entry), possibly forwarding downstream and/or caching. Unsolicited Data are dropped. Solicited Data are forwarded out the interfaces recorded in the corresponding PIT entries. Further, a solicited Data consumes the related PIT entry, meaning that the PIT

entry is deleted after the Data packet is forwarded. Caching of the data packet may happen, however, it depends on the router's caching policy and available resources, e.g., free storage space. So, R2 will have a new cache record for the Data content and the PIT entry previously set by the Interest is deleted since the related Data has been forwarded downstream.
The basic workflow described above can be amended by the presence of a Strategy Component. A forwarding strategy is a piece of software logic which may perform adaptive forwarding decisions after the FIB lookup.

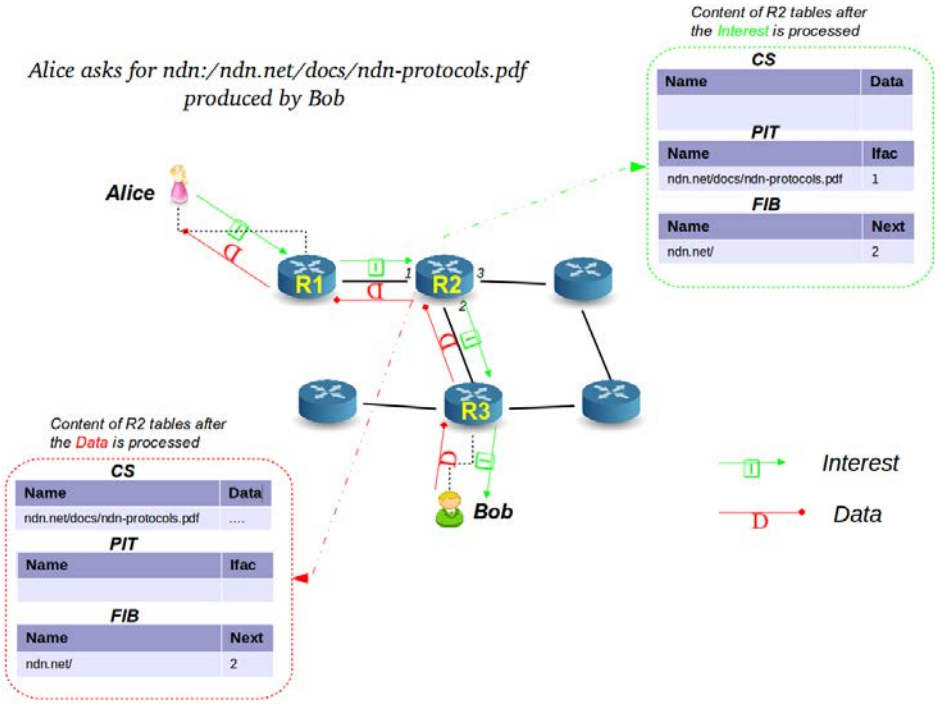


Figure 1.5 – Interest-Data exchange in an NDN network.

Packet format

The current packet format for NDN Interest and Data packets is the outcome of an iterative refinement process old as the research project itself. The design of the NDN packet format is driven by a set of requirements: universality/elasticity, extensibility and process efficiency [60]. When requirements conflict, they are sorted by priority. For example, the packet format is made of many variable length fields which provide flexibility but at the price of an increase of the packet processing time.
The general structure of Interest and Data packets is illustrated in Fig. 1.6. All the packet fields in Interest and Data follow the variable-length Type-Length-Value (TLV) encoding specified in [61]. The Name field, present in both Interest and Data, is made of a variable number of TLV fields each containing a single name segment (see the following "naming" subsection). The full name of a Data packet should also contain a final digest component within the *ImplicitSha256DigestComponent* field. However, this implicit digest may appear in an Interest and is never included in Data packet explicitly. An Interest must contain a randomly-generated 32 bit nonce too. In fact, the combination of the Name and a Nonce uniquely identifies an Interest and

thus detects possible looping ones. Between the Name and the Nonce, an Interest may include Selectors. Selectors are optional fields which can be used to either fetch or avoid to fetch specific contents. At the end of the packet, an Interest may also include two other optional fields called Guiders. Guiders indicate how far, in the *Scope* field, and for how long, in the *InterestLifetime* field, an Interest may propagate.
A Data packet may contain some meta information following the Name in the MetaInfo field. That meta-information specifies the type of content (e.g., simple payload or public key or link object), the validity time for this Data and the identifier of the final block in a sequence of data chunks. The MetaInfo field is immediately followed by the payload. Finally, a Data always includes a Signature block made of two TLVs. Those fields, namely the *SignatureInfo* and the *SignatureValue*, contain respectively a signature description and the actual bits of the signature.

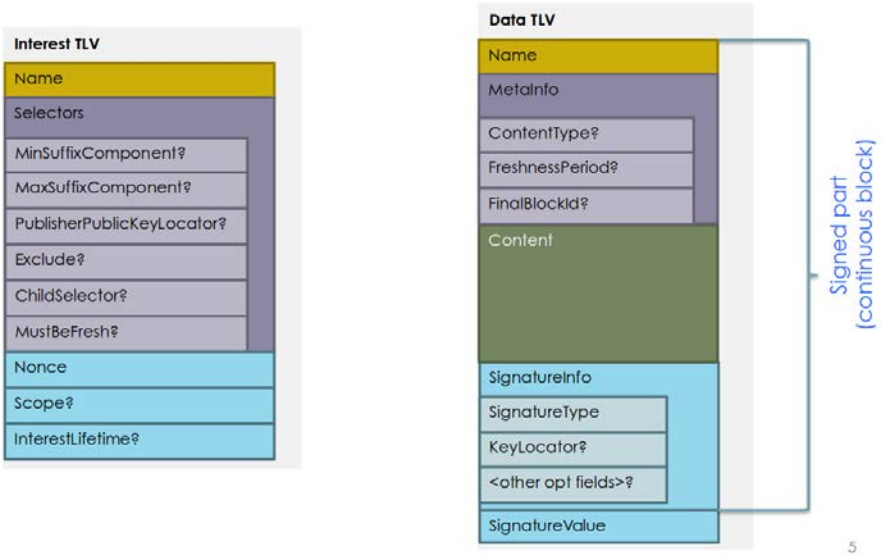


Figure 1.6 – Format of Interest and Data packets from the slides presented by A. Afanasyev at the Jan17 ICNRG Interim Meeting.

Naming

Names are a cornerstone of NDN. Names exhibit two main properties by design: network opacity and structured hierarchy. First, NDN names are opaque to the network. In fact, routers understand names structure for forwarding purposes but they are agnostic to any upper-layer semantics. This property was meant to guarantee names and the network layer to evolve independently. This desired opacity allows applications to define specific name conventions without affecting the basic forwarding operations in the network. Second, NDN names are hierarchically structured. This has a twofold objective. Hierarchies can be exploited to achieve the scalability of the routing system through aggregation, as already discussed in Sec. 1.1.2. Further, the structure of a name can be leveraged to embed conventions in it, e.g., in the order of the name components. NDN names are often represented likewise today's omnipresent Web URLs. They are human-readable variable-length strings separated by instances of a certain delimiter (e.g., the forward-slash '/'). The name of some headlines on ccn (a), of a segment of a video clip on

youtube (b), of the command to switch off some lights in one of the computer science department rooms of the university of Luxembourg (c) could respectively look like:

(a) ndn:/cnn.com/2018/04/13/news/economy/us-tpp-trade-trump/index.html
(b) ndn:/youtube.com/videos/DaveMatthewsBand-IfOnly/1/32
(c) ndn:/uni.lu/cs-building/2nd/A103/switch-off/light/sensor2

There are two more clarifications often made with regard to names in NDN. The first one is about name uniqueness. Indeed, names must be unique only when they refer to globally addressable data. For example, the ccn news and the youtube video in (a) e (b) may need to have globally unique names. Otherwise, when names only refer to objects within a certain scope, their global uniqueness is not required anymore. For example, the name of the command in (c) could be simply understood by routers within the University of Luxembourg campus network.
Finally, since the research on NDN started, the architecture was meant neither to advocate nor to manage any specific namespace. Nevertheless, the exploration of namespaces driven by different applications' needs has suggested name conventions which may improve the content localization and delivery. Therefore, a set of generic naming principles have been finally implemented in software libraries for the NDN.

Routing and Forwarding

In NDN, routing and forwarding is based on names. At a first glance, this approach solves a series of problems affecting the current IP routing architecture: NAT traversal, mobility, address exhaustion and management. However, it also introduces notable challenges. The rest of this section tries to summarize both aspects.
NDN guards the separation of routing and forwarding planes. Name prefixes are exchanged through routing protocols and kept in forwarding tables (FIBs) by NDN routers. Further, at the data-plane, NDN leverages per-Interest state in routers and specific Interest packet fields to support loop-less multi-path routing. Security of routing is improved with NDN for three main reasons: i) all data packets, including routing announcements, are signed so preventing the dissemination of fake information in the routing plane, ii) the multipath capabilities allow routers to detect and circumvent security threat in the network, e.g., prefix hijacking , iii) because of the absence of addresses, no single specific recipient can be attacked.
Nevertheless, routing in NDN raises some challenges too. First, an unbounded namespace can generate a large number of routing entries which must indeed fit in routing tables whose size must stay limited for scalability reasons. Second, it is not clear how a multi-path routing protocol should intelligently support forwarding on different paths. In theory, this capability would be beneficial; yet, the implications are not fully understood.
NDN radically changes today's forwarding plane logic. In IP networks, all the forwarding decisions are taken by routing protocols and forwarding in the data-plane is merely done by following the routing table entries. While NDN routers keep datagram state which enables adaptive behaviors in the forwarding plane and exonerates routers from certain tasks. More in detail, routers may use every PIT entry to store information about the related forwarded Interest (e.g., Interest arrival time) and measure some relative performance (e.g., the time for the Data packet to be received). Those measurements can possibly help adjust the forwarding behavior to the network conditions. Beyond the basic components described in Sec. 1.2.2, each NDN router has also a

Forwarding Strategy module containing customizable forwarding behaviors. This module eventually makes forwarding decisions for every Interest packet.
The adaptive forwarding capabilities of NDN routers constitute a promising tool to detect, among the others, packet losses, network congestion, broken links and to measure packet delivery performance [50]. Nevertheless, forwarding variable length names instead of fixed size IP addresses requires a re-engineering of today's forwarding devices. In fact, NDN prototypes must be able to sustain line rates over tables supporting millions and even billions of names.
It must be said that such an enhanced forwarding plane implies a rethink of the role of the routing protocols in NDN networks [62]. The adaptive forwarding plane may overcome short term network churns without relying on the convergence of the upper routing protocol. Therefore, a less stringent requirement for quick convergence in routing protocols allows to explore routing protocols which were thought to be unsuitable before.

In-Network Storage

NDN Data packets carry names and signatures, so their integrity and validity can be assessed independently from the source producing them. This implies they can be cached in routers' Content Stores (CSes) to serve further requests for the same data. An NDN router's CS may be considered analogous to an IP router's local packet buffer, except that buffered IP packets cannot be reused while buffered Data packets can be.
On one hand, in-network caching of both static and dynamic contents (e.g., a live-stream video) provide advantages. Cached static content can be leveraged to reduce the delivery time to consumers as well as the load on content producers and network links. Cached dynamic content can be leveraged by network for multicast communications or retransmissions after packet losses. On the other hand, cached contents raise several privacy issues, since any router can infer sensitive information about those [51].
Furthermore, the NDN architecture supports an additional form of shared persistent storage, namely a repository, which has been developed to answer a common need among different distributed applications. A repository represents a shared data-set which needs to be synchronized among distributed participants. A repository, which includes a list of namespaces, is essentially a data-set of contents. This storage, which is usually more persistent and larger-volume than a CS, has been used so far to support files or keys sharing.
Finally, the NDN community is currently investigating different storage models, file-system or database like, to design standard method for applications to discover, trust and write to different kinds of persistent storage in the network.

Security and Privacy

The NDN comes with built-in security since a mandatory signature binds the name to the content in every Data packet. Therefore, once a Data is emitted in the network and is signed by the producer private key, its integrity is guaranteed and the content cannot be tampered. In fact, the signature and some additional information about the publisher identity (see the Data packet format illustrated in Sec. 1.2.2) can be used by consumers to verify the Data provenance too. In this model, the trust is decoupled from the network and is left to applications to authenticate entities and grant those with privileges to perform operations.
The robustness of the NDN architecture can be theoretically evaluated against Internet's well-known Denial-Of-Service (DoS) attacks. A seminal analysis of both resilience to existing attacks and architecture-specific vulnerabilities was presented in [63]. NDN is resilient to the attacks

targeting specific network endpoints, since it has no notion of address. Hence, attacks which flood victims with traffic to saturate their resources, e.g., TCP SYN flooding attacks, cannot be carried out. Furthermore, attacks carried out by compromised routers advertising invalid routes, e.g., black-holing via prefix hijacking [49], can be detected and circumvented by the NDN's adapative forwarding plane.
Nevertheless, some inherent NDN features can be leveraged by potential attackers to mount novel DoS attacks. In fact, the architecture has so far showed to be vulnerable to attacks which capitalize on per-packet state and storage in NDN routers. In particular, routers can be overflowed with malicious requests aiming to stop them from serving legitimate requests. Those malicious requests aim to occupy routers' available resources, mainly the PIT space. This attack is commonly identified with the term Interest Flooding Attack and is extensively covered in Chapter 3. A second kind of NDN-specific threat relates to contents cached by routers. Data packets are all signed, so their validity can be ascertained by signature verification. However, line-rate signature verification of every Data is technically impracticable in routers. Moreover, even though consumers can verify signatures and so detect fake content, they do not have mean to ask routers to flush some contents from their caches. The two above conditions favor attackers to pollute routers' caches with fake content which is both fetched by consumers and stored in network caches. This attack is commonly referred to Content Poisoning [64].
With regard to privacy, some NDN features enhance some privacy aspects. For instance, the absence of addresses makes impossible to identify source and destination entities of a communication in any point of the network. Nevertheless, the presence of caches and the use of human-readable names may reveal information about the requested Data. Indeed, NDN suffers from the same general privacy issues other Information-Centric architectures are affected with, which have been briefly described in 1.1.4. In fact, the semantic correlation between names and contents can be exploited to unveil or guess the payload of Data packets by only looking at names. Then, this information can be benignly used to profile users' requests or maliciously to censor some traffic. Further, cached contents are not encrypted by default. Thus, it is the responsibility of the content producer to prevent the content from being eavesdropped. However, the latter task is not easy, since plain cryptography would hinder the usefulness of the in-network caching for the encrypted contents.

Transport Layers?

The NDN architecture does not foresee any transport-layer protocol since most of the services provided by that layer have been spread across different other components. Usually a transport protocol builds upon a network protocol to provide delivery from a process on a machine to another process on a different machine with a certain reliability. Further, a transport layer provides abstractions to the applications to use the implemented delivery services. In the following, typical services implemented in a transport protocol (see Chapter 6 of [65]) are analyzed in the context of the NDN architecture.

Addressing - In IP networks, there are Transport Service Access Points (TSAPs), i.e., port numbers, which work as transport addresses software processes can listen to. Port numbers are meant to distinguish multiple transport processes on the same network address. This functionality in NDN can be achieved via names. In fact, conventions can be applied to name components so that Interests are delivered to the appropriate applications. For example, *ndn://uni.lu/CSF/s1/pandora/* and *ndn://uni.lu/CSF/s1/navi/* could be used to demultiplex the communication to two different processes, named *pandora* and *navi*, hosted on the *s1* server

in the Computer Science Faculty (CSF) of the University of Luxembourg. Similarly, NDN would also allow programmers to replicate the idea of a portmapper process which already exist in today's operating systems. That is, a dedicated process which is reachable by a well-defined standard name and can be contacted by requests that do not know a TSAP in advance.

Connection Establishment and Release - there are no fixed destinations in NDN, so no persistent connection between two endpoints must be established and then released to achieve data exchanges. This alleviates all the related problems faced by the design of the TCP protocol, to name a few, the wrapping of sequence numbers in the establishment of a connection or the two-army problem of releasing a connection.

Error and Flow Control - Error control refers to a mechanism verifying Data are delivered without errors. Flow Control is a mechanism meant to impede fast senders to overload slow receivers. In NDN, the former is guaranteed by signatures carried by every Data packet which can be verified at the receiver to assess the packet integrity. The latter is somehow natively smoothed by the receiver-driven communication pattern and by PIT entries which can be consumed only by a single Data packet.

Congestion Control - Congestion control mechanisms deal with machines sending too many packets and too quickly in the network since this sort of behavior can cause network congestion and by consequence performance degradation. Although congestion occurs at routers, it is under the responsibility of both transport and network protocols to avoid it. Therefore, in IP networks, mechanisms to control congestion are implemented at both layers (for example, the Explicit Congestion Notification (ECN) flag bits in the IP header and the Additive Increase Multiplicative Decrease (AIMD) control law of the TCP protocol). Network congestion is a phenomenon that persists even in NDN networks. Nevertheless, since traditional congestion control mechanisms are designed for host-to-host connections, those cannot be directly leveraged by the connectionless NDN architecture. Moreover, the design of congestion control mechanisms for NDN has to take into account additional factors like in-network caching and multi-path forwarding. There have been so far several proposals of congestion control algorithms for NDN, a summary of the related work can be found in [66].

API to the application layer - A proposal of a Consumer-Producer API for NDN exists and was first presented in a technical report in 2014 [67] and then refined in [68]. Although the definition of this API is an ongoing process driven by the design of new applications, the current one already sets out two main requirements. First, the data transfer is controlled by properties of the namespace. Second, some Consumer and Producer requirements may be really different. So, the current NDN API defines two different programming abstractions: a Consumer context and a Producer one. The consumer context assists consumer applications to fetch contents under a certain prefix name. While the producer context assists producer applications to perform data publishing under a certain prefix name. Both contexts have options which can be used by applications to define the parameters of the data transfer. For example, consumers may want to specify Interest selector fields or Interest retransmission policies; instead, producers may want to specify algorithms and key size for the signature of Data packets. Both contexts also supply callback functions to notify about events they may trigger.

1.2.3 Ongoing and future development

During the first four years of the NDN's life, the research community had focused to address more fundamental questions like naming, scalable routing and forwarding on names, content-oriented security models [69]. Those different research questions were addressed within a consistent architectural framework. After that, throughout the next two years of the Next-Phase NSF award [37], the research on NDN focused on the validation and the refinement of the conceived architecture on different network environments through the design of several key applications.
The explored network environments were enterprise building automation and management systems (as test-case, the large Siemens building monitoring system deployed at the UCLA university facilities [70]) and mobile health (as test-case, the Open mHealth non-profit patient-centric open platform to share clinical data [71]). Those environments had been selected to address two of the national priorities of the NSF, namely Health IT and Cyber-physical systems. With regard to the application design, the fruition of digital media through mobile devices like smartphones, laptops, vehicles, a.k.a. Mobile Multimedia, was selected. Both network environments and applications have been chosen since they characterize environments and communication patterns that a data-centric architecture could serve better than the current host-based TCP/IP architecture. Those use cases have improved the status of the NDN architecture on different aspects with a special focus on namespace design, name-based access control, trust management, distributed synchronization of data-sets, routing and forwarding, congestion control. In the meantime, this phase of research has also outlined open issues which still deserve more investigation, e.g., producer mobility support [72], encryption-based access control [73], a distributed always-on lookup service [74].

1.3 ICN enablers

Running ICN protocols in today's networks would require either to have all the network equipment to be "ICN aware" or to design an overlay network upon the Internet's TCP/IP protocol suite. The former scenario is quite unlikely to happen in the near future since a forklift upgrade of a large fraction of the deployed network equipment on a global scale is literally no feasible. The second scenario requires statically configured IP tunnels necessary to implement ICN services in the overlay network which, by the way, imposes a certain shift from the original vision of a native in-network support of the ICN functionality. Overall, this barrier to upgrade or replace the existing infrastructure, and the consequent frustration of a possibly very long, even never-happening, deployment contribute to generate some strong fear of impracticality about the ICN paradigm.
Nevertheless, over the last decade research on computer networks has produced technologies which can enable ICN deployments in a short term. In particular, the Software Defined Networking (SDN) [75] paradigm has changed the way today's networks are designed and managed. The SDN introduces the separation of the control plane from the data-plane in forwarding elements like switches and routers[4]. The former runs the logic which decides how to handle the network traffic. The latter operates the forwarding of packets based on the decision made by the former. With SDN, a single software running in the control-plane can control the data-plane of multiple devices via standard Application Program Interfaces (APIs).
Today, the de facto standard SDN API is OpenFlow (OF) [76], which is supported by many products across different network vendors. The OF API follows a programming model where switches hold tables of packet forwarding rules. In the most basic form, each rule has a matching pattern and a list of actions which can be associated to the table entry. Actions implement protocol-specific packet editing operations and forwarding decisions. Actions are executed on a packet when that matches the corresponding rule. OF does not dictate how control plane programs should be written. However, SDN and OF have allowed operators to write control-plane programs so to address disparate control and management issues in their networks [77].
In theory, SDN could facilitate the deployment of ICN features in today's networks without requiring any network equipment upgrade. In practice, today's SDN standards like OpenFlow mostly work with standardized host-oriented protocols. This implies that those SDN standards can be directly leveraged to program networks to handle novel content-oriented ICN protocols. Two approaches can be followed to overcome this issue. The long-term one is to envision extensions of the current SDN standards to accommodate the ICN protocols [78]. The short-term one is to use the existing SDN solutions to offer the ICN functionality [79, 80]. The former solution is hardly applicable because of the long extension process of the existing SDN standards and of the ongoing evolution of the current ICN protocols. The latter seems the viable approach since it only relies on commercial SDN-enabled elements compliant to established standards.
Several works [79, 80] have already investigated how ICN functions could be supported via today's OF switches. Since OF defines matches and operations on TCP/IP protocols' header fields, an ICN protocol's fields must be mapped to the available OF fields. In the existing approaches this is usually done by either specialized nodes (border nodes in [80]) or a control plane logic as in [79]. The mapping is mainly meant to i) identify ICN traffic in the networks, ii) carry information related to the ICN protocol, e.g., content names, iii) define flow identifiers. Flow identifiers are used to set-up forwarding paths for both requests and responses in SDN-enabled

[4]The term switch is used from now on to identify any network element performing forwarding decisions on traffic.

elements. Those SDN-based approaches may also allow ICN traffic to traverse non-SDN network segments [79]. Despite the design differences, the solutions in [79, 80] achieve ICN deployments over traditional IP-based networks by leveraging the SDN paradigm.
In addition to the existing SDN standards, another set of recently emerging solutions to program the data-plane of the network forwarding elements represents a promising enabler for the ICN. In fact, nowadays, the design of switching chips undergoes a transition from Application Spefic Integrated Circuits (ASICs) to fully-programmable chips. ASICs were believed to be the only way to achieve the latency required to perform packet processing operations at high network speeds. Yet, ASICs produced fixed-function switches, i.e., switches able to operate on a predetermined set of protocols fields with a small set of predefined actions. However, recent studies [81] showed that fully-programmable switching chips can be designed for the same purpose. The data-plane of those chips can be programmed to support non-standard protocols and packet processing operations. The emergence of these new switching chips has also fueled the dialogue about the need for a common high-level language to program different kinds of forwarding elements (e.g., software and hardware switches, Network Processor Units, FPGAs) [82]. The design of such a language targets protocol- and target-independence to write portable and modular network programs.
We believe that the latest generation of programmable switches and the related high-level languages will both i) in a long-term ease the deployment of ICN over existing networks, ii) in a short-term enable experimentation of ICN protocols at an unprecedented pace. Both technologies will overcome the limitations posed by today's SDN standards. First, the new switching chips could be configured to understand and process ICN protocols without devising cumbersome translation mechanisms to map those protocols to standard protocols as done with today's OF-based elements. Second, ICN designs could for the first time be written in a easy and portable way. This latter characteristic is absent in the state-of-the-art ICN designs, which are tailored to specific hardware and software platforms instead. Existing designs are difficult to share, modify, improve and so, use for further research.
To promote this line of research, we have implemented NDN.p4 [83], the first NDN router design written in the high-level language for packet processors P4, which is illustrated in chapter 5. Although NDN.p4 features a partial implementation of an NDN router, because of the early P4's language specification limitations, its design is simple and can be ported over multiple P4-compliant platforms. The NDN.p4 showcases for the first time a promising direction to advance the deployment in research and production networks of ICN forwarding devices.

1.4 Summary

This chapter has introduced the clean-slate research on future Internet architectures with a specific focus on the Information-Centric Networking (ICN) paradigm. The main ICN design principles, the research challenges and the opportunities in the deployment of such architectures have been presented. Among the ICN realizations, the Named-Data Networking (NDN) architecture has been described in more depth since it has been the main target of the research of this dissertation.
The next chapter focuses on the description of the Pending Interest Table (PIT), a core component of the NDN's forwarding plane. The PIT enables the basic data-exchange in the NDN architecture. Yet, the PIT exposes the NDN architecture to a severe security threat this work has studied and addressed from different perspectives. Therefore, Chapter 2 provides the background to understand the role of the PIT in the NDN and the vulnerability it exposes.

Chapter 2

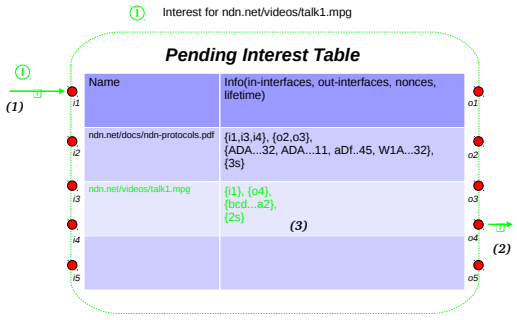
Pending Interest Table

The original main purpose for a table of the pending interests has been stated in the seminal paper on Content-Centric Networking (CCN) [22]. A table recording Interests forwarded upstream to potential content sources is used to send Data packets downstream to nodes requesting them. That table, called Pending Interest Table (PIT), performs exact-match on names for those Interests which cannot be served at first by a node's local cache. First, Interests traversing the network are recorded at routers by creating PIT entries. Afterward, potential Data can follow and consume those PIT entries. Lately, the PIT entries, which are eventually not consumed by any Data, expire after a certain timeout set independently by each router.
The current Pending Interest Table is still a key component of the NDN forwarding plane. Beyond the originally-stated asynchronous Interest-Data exchange, the PIT enables an important control flow principle. That is, a Data packet is accepted for forwarding only if there exists a related PIT entry and, once that Data is forwarded downstream, the related PIT entry is deleted. The PIT's 1-to-1 Interest-Data mapping, initially conceived as a control flow mechanism inspired by the ancestor data-ack mapping in TCP, offers several additional benefits. In fact, state in PITs can be used to offload complexity in the design of multipath routing protocols (e.g., loops avoidance) to the data-plane. Or yet, measurements over PIT entries can be leveraged directly by the data-plane to track the performance of every Interest-Data exchange and define adaptive forwarding behaviors. For example, unusually large RTTs or expired PIT entries can be used to detect and react to anomalies in the network like broken links, congestion or, worse, attacks [49].
The software and hardware design of the PIT in NDN routers has been an open challenge since the early inception of the NDN architecture. Several design choices must be taken, e.g., the choice of the data-structure, the sizing of the table and, when in hardware, the placement on the forwarding element's line cards. Further, PIT designs must also consider the requirements of a possible Internet-scale deployment. According to the research done so far, it seems that the design of a PIT working at common link speeds is feasible with the current technology. However, the security threat opened by the PIT per-Interest state seriously questions the existence of this component in the data-plane of NDN routers. Therefore, alternative forwarding mechanisms are more and more recently emerging, aiming to preserve some of the main PIT properties while avoiding the related security issues.
The content of this chapter is organized as follows. Section 2.1 describes the generic data-structure by focusing on the information contained and the requirements. Section 2.2 focuses on the software and/or hardware requirements for the design of the data-structure. Section 2.3 discusses the main other design challenges. Finally, section 2.4 analyzes the direct and indirect properties of the PIT.

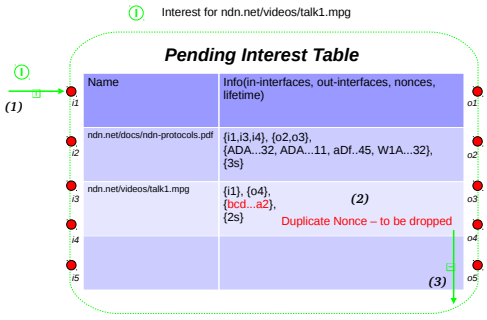
2.1 PIT Description

The operations performed by the NDN forwarding plane have been described in the Sec. 1.2.2 of the first chapter. The PIT is the functional block responsible to record forwarded upstream Interests which wait to be consumed by potential Data downstream. Abstractly, the PIT can be seen as an associative memory binding Interest names to supplementary information about forwarded Interests. At each node, each forwarded Interest is recorded through a PIT entry. Generally, a PIT entry associates the Interest name to the following information at least: an expiration time, two lists of interfaces, and a list of nonces. The expiration time indicates the longest duration for which the entry lasts in the PIT before being deleted if no Data consumes it. A list of the interfaces where a certain Interest has been received is needed to perform a correct multicast delivery when the related Data packet is received. A PIT entry also records the outgoing interfaces, that is, where the Interest has been forwarded to. A track of outgoing interfaces is used to i) differentiate between solicited and unsolicited Data, ii) do performance measurements. A list of seen nonces unambiguously tracks different Interests.
In theory, a PIT performs a lookup through an exact match of Interest and Data names. That is, the full Interest or Data name is used to lookup the table for existing records. In practice, the NDN protocol defines two more prefix name matches performed by the PIT (see Sec. 2.2 of [84]). Further, a finer search in the PIT can be activated through optional Interest header fields. Overall, an Interest matches existing PIT records according to its name and the selectors fields (see Packet format in Sec. 1.6 of chapter 1). While, a data packet matches all the records sharing a common name prefix. Thus, for example, the entry whose name is "/ndn.com/file/docs.zip" would be matched by any of the following: '/ndn.com/file/docs.zip/v1', '/ndn.com/file/docs.zip/annexes', '/ndn.com/file/docs.zip/errata'. By consequence, there could be many *similar* Data packets consuming the same PIT entry. Nevertheless, for sake of generality, the exact match lookup for both Interests and Data is mostly considered when reasoning about the logical structure of the PIT.
The most common operations done by the PIT are summarized in Fig. 2.1. For the sake of generality, any outcome of the *Forwarding Strategy* module, which indeed makes the final forwarding decision for every Interest, is not considered. At the reception of a new Interest, a successful forwarding operation generates a new PIT entry as showed in Fig. 2.1a. The generated entry records incoming and outgoing interface, the Interest nonce and the entry lifetime (e.g., respectively the values $\{i1\}$, $\{o4\}$, $\{bcd...a2\}$, $\{2s\}$ of the third PIT entry generated in Fig. 2.1a). Because of multipath routing protocols which do not feature loop avoidance, in NDN the same Interest could be received twice at the same router. However, a duplicate is easily detected and dropped thanks to the recorded Nonce values in the related PIT entry as shown in Fig. 2.1b. When an Interest for an existing PIT entry is received at a different input interface, the new interface and the nonce are added to the existing entry as shown in Fig. 2.1c. An Interest can also be retransmitted either by a consumer or by a router's forwarding strategy module. Nonces are used to distinguish retransmissions from the original Interests. A retransmitted Interest carries a new Nonce, so it updates an existing entry as shown in Fig. 2.1d. A Data packet is said to be solicited when there exists a related PIT entry whose list of outgoing interfaces contains the interface where the Data has been received. Copies of a solicited Data packet are forwarded downstream to all the incoming interfaces recorded by the related PIT entry as shown in Fig. 2.1e. A solicited Data also consumes the related PIT entry. When the interface where a Data packet is received is not in the list of the outgoing interfaces of the related PIT entry, the Data is considered unsolicited and so dropped as shown in Fig. 2.1f. Unsolicited Data do not alter the related PIT entries.

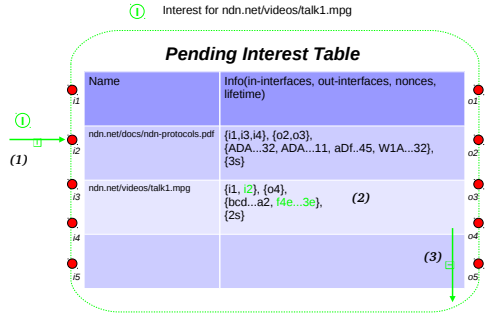
In summary, the PIT provides two main services: Interests aggregation at Interest reception and multicast Data delivery at Data reception. The Interests aggregation happens when a new Interest is received and a PIT entry with that name already exists (e.g., in Fig. 2.1b, 2.1c, 2.1d). The outcome of the aggregation depends on whether or not a similar Interest was either previously received on a different interface. If so, the related PIT entry has to be updated to record the new interface (e.g., in Fig. 2.1c, 2.1d). If not, the new Interest can be simply dropped without taking any further action (e.g., in Fig. 2.1b). The multicast Data delivery happens when a new Data packet is received and there exists a related PIT entry. Multiple copies of the same Data packet are forwarded downstream on all the interfaces recorded in the corresponding PIT entry (as in Fig. 2.1e). Overall, the PIT keeps a natural flow balance between Interests and Data since, at every router, it does neither transmit duplicate Interests nor accept duplicate Data.
The PIT records per-Interest state in the data-plane. The PIT state is also per-hop, that is, it is only local to a certain router. Therefore, if a router crashes, only its state gets lost and this does not directly affect any state in neighbor routers. The PIT state is cardinal to Data being delivered over the reverse path to consumers, yet it may also be used to dynamically perform real-time adaptation of the forwarding behavior to the network conditions. Although the soft state in PIT can be leveraged to design an adaptive forwarding plane, e.g., for robust packet delivery [49] and simplified routing algorithms [62], NDN does not mandate any specific forwarding behavior per-se. The design choice of generality on the functions of the data-plane state was driven by past experiences with IP networks. In fact, there had been several previous attempts to introduce some state in the network, however, their coarse granularity did not make the state reusable for different purposes. For example, state stored to perform congestion control (e.g., the state needed by the eXplicit Congestion Protocol) was not compatible with the one required by a DDoS defense mechanism (e.g., the state needed by the Pushback mechanism for IP traceback) [50]. Conversely, NDN stores datagram state, so achieving the finest possible granularity. Storing datagram state requires more resources yet gives the flexibility to implement more functions.
Among the three main functional blocks of an NDN router, the PIT is the one which has the most stringent and critical update requirements. The FIB is updated with coarser frequency by routing protocols running in the slow path. Thus, modifications to the FIB can tolerate a certain delay before becoming effective. The CS is updated by the forwarding plane. However, if any insertion or removal of content from the local cache is deferred, the consequences are either unnecessary upstream forwarding of a few Interests or replies containing outdated content. Conversely, PIT state changes must be performed almost instantaneously at line-rate since the processing of the next received Interest or Data packet may depend upon the updated state. Therefore, the design of the PIT is not trivial. Beyond the per-packet state update at line rate, the design also requires determining the size of the table according to different link loads and traffic distributions.



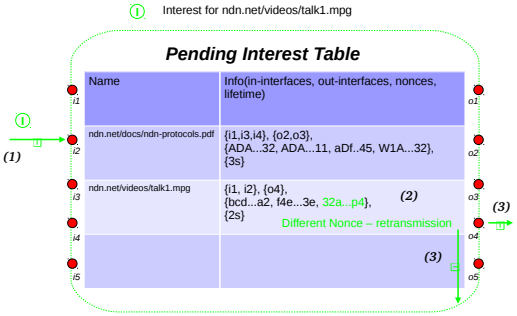
(a) New Interest.



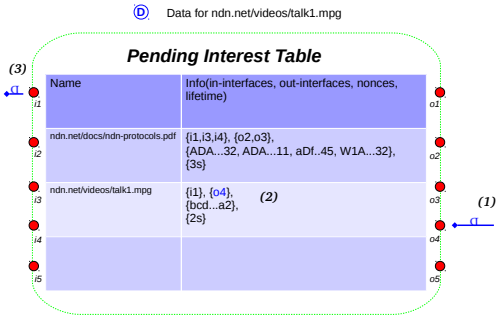
(b) Duplicate Interest.



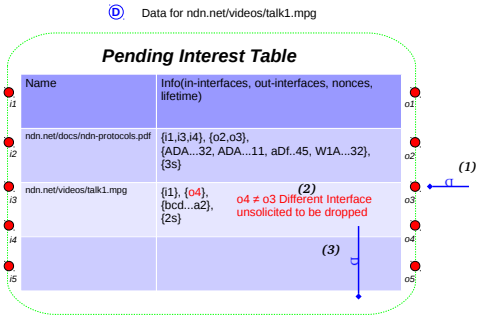
(c) Existing Interest, different interface.



(d) Existing Interest, different nonce.



(e) Solicited Data packet.



(f) Unsolicited Data packet.

Figure 2.1 – Illustrations of the different PIT operations at Interest and Data reception, numbers in parenthesis indicate the temporal sequence.

2.2 Data structure

The key issue in the design of a data-structure for the PIT is the "exact string matching with fast updates" [85]. This problem refers to a membership verification for a string, the Interest name, in a set of strings, the names of the existing PIT entries. Further, the problem definition includes also i) insertions, when no PIT entries match incoming Interests, ii) removals, when Data packets consume existing PIT entries, iii) updates, when new nonce or interface or lifetime values have to be recorded in existing PIT entries.
High-speed exact matching on strings has been thoroughly investigated by the research on intrusion detection and filtering systems in IP networks. However, those systems do not consider the requirement of possible per-packet update. On the contrary, per-packet updates at line-rate are fundamental for the correctness of the NDN forwarding plane. Furthermore, content names may be very long and so require a lot of storage to be recorded together with additional forwarding information. For all the aforementioned reasons, some research has been done to address the more stringent requirements of exact string matching for the PIT in the NDN data-plane.
In summary, the design of a data-structure for the PIT is challenging because of the three following requirements:

- fast exact-match on variable length names: this has required a re-engineering of the state-of-the-art techniques which were mostly designed to deal with fixed-length IP addresses. Instead, NDN uses hierarchical human-readable names (HRNs) (see Sec. 1.1.2 of chapter 1 for a recap of this naming scheme). HRNs have variable length with no upper bound. Those factors could cause a lookup time linear to the name length which could easily not meet the requirements of the NDN forwarding-plane for certain link speeds (several tens or hundredths of Gbps).

- per-packet lookup, insertion or update: all of these operations have to be performed with almost no delay since the processing of a next packet could be affected by the state change caused by the current one.

- per-content state information: a PIT table may require a considerable amount of memory to be stored. If URLs are considered in analogy to NDN names, today's URLs require several tens of bytes of storage per name. In addition to that, as detailed in Sec. 2.1 each entry records additional information about interfaces, nonces and performance measurements. Furthermore, the PIT size increases with the data-rate since more packets may be received on faster links. For example, at 100 Gbps, packets may have an arrival rate of a few nanoseconds. Those tight packet inter-arrival times mandate PIT designs to target data-structure which all or partially reside in small on-chip memories, like SRAM or RLDRAM. Those high-speed memories are overly expensive and so space-limited.

Generally, a PIT design aims at two main improvements: i) shrinking the data-structure size and ii) reducing the average access time by also determining exactly the worst case one. PIT design should be guided by some additional observations as seminally outlined in [85]. First, the structure of NDN names can be leveraged in the design. Names are made of components and matches are defined at the component-level rather than at the character-level. This point can guide the refinement of existing techniques which were originally meant to work at the character granularity. Second, even though there is no theoretic limit to the number of components in a name, it is foreseeable that in real networks there would be such an upper bound. Therefore,

the maximum number of components for most names (for example, some related works report that most URLs have less than 30 name components) can be used to define heuristics which achieve constant times with certain name distributions. Third, it is better to consider simple data-structures because of the requirement of the possible very fast per-packet update. A very little time from one packet to the other seems to preclude the utilization of any complex highly-nested data-structure. Fourth, the role and position of a forwarding device in the network can be used to relax the PIT requirements. In fact, while backbone routers must sustain higher loads than edge routers, the former ones can perform more aggregation without risking to break the Data packet delivery [86]. Three different kind of data-structures have been so far proposed in the state-of-the-art works: name prefix trie based, bloom-filter based, hash table based. The classification reported in [87] has been hereby extended.

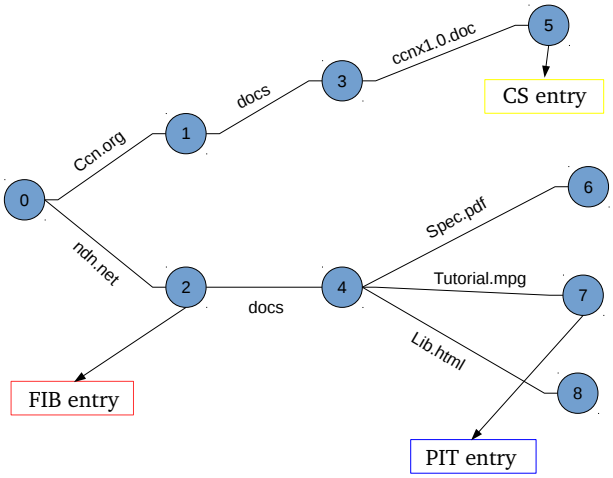
2.2.1 Name Prefix Trie (NPT) based

The Name Prefix Trie (NPT) is a data-structure inspired by the binary trie for IP prefix lookup, which capitalizes the hierarchical structure of names and the component-level granularity of the longest prefix matching in NDN. The design of a PIT based on an NPT data-structure was first proposed in [88] as further refinement of an encoding technique for Name Prefix Trie previously proposed in [89].

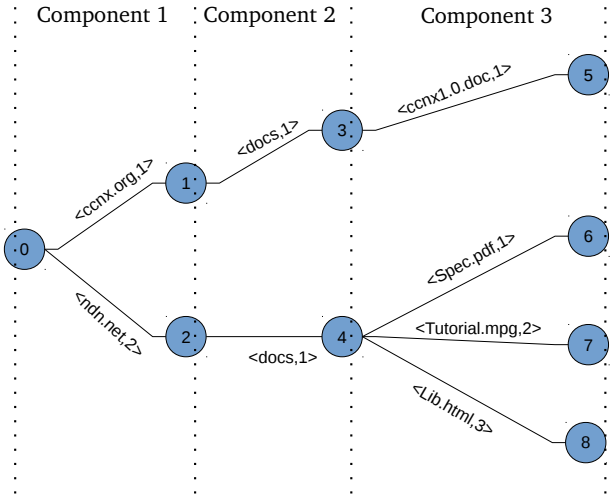
The main idea of NPT is to represent NDN names through trees like the one in Fig. 2.2a. Every edge represents a component and every node represents a lookup state. The reader could be wondering about the rationale to use a data-structure giving visibility of the full name structure when only an exact match is required to lookup the PIT. Indeed, NPTs have been proposed as solution to implement a single index for the three main functional blocks of the NDN data-plane, i.e., CS, PIT and FIB. In fact, Fig. 2.2a shows that a non-empty lookup state may correspond to either a FIB entry or a CS one or a PIT one. The advantage is a reduction of the memory space to store a single index rather than three. The downside is the additional complexity of the data-structure required to support different kinds of read/write operations.

An NPT can be represented with a name character trie (NCT) [90]. Nevertheless, character tries require a lot of memory to store states and transitions. Besides, they have an $O(mn)$ computation for the LPM lookup in average (where m and n are the average number respectively of children per node and name length). Compression techniques can be applied at the name-component level to reduce the space needed to store the prefix trie. Encoding rather than hashing techniques are preferred for compression by existing literature since the latter ones affect the correctness of routing because of collisions. Thus, for example, the prefix trie in Fig. 2.2a could be encoded in a new trie showed in Fig. 2.2b, called Encoded Name Prefix Trie (ENPT), where codes are assigned to name components. The representation with codes is already more compact than the one storing the characters. However, it can be further improved as detailed in [88, 89] where a Name Component Encoding (NCE) is proposed. Improvements of an NCE relate to the code allocation mechanism which aims to reduce the total number of codes, to the trie representation which is implemented by means of State Transition Arrays (STAs), to the algorithm to perform components' code lookup and to the management of the STAs for the insertion/removal/update operations.

In terms of complexity analysis, the NCE reduces the memory space to store an NPT and speeds up the lookup in the average case compared to the NCT. Experiments reported by [89] compare the two data-structures on three data sets built using domain information from the D-MOZ, ALEXA and Blacklist public websites. For the biggest data set composed of around 3 million



(a) Name Prefix Trie



(b) Encoded Name Prefix Trie

Figure 2.2 – Examples of the Name Prefix Trie (NPT) in Fig. 2.2a and of the Encoded Name Prefix Trie (ENPT) in Fig. 2.2b.

names, the average name length in bits is reduced by 75% after encoding the names with NCE. The complete NCE representation through STAs saves around 20% over the NCT one. NCE memory requirements also grow more slowly compared to the NCT one. Thus, NCE turns to be more memory efficient on both small and large data-sets. In a followup work [88], a mapping method to infer size and access frequency required by a PIT on a 20Gbps link is proposed. The trace translation suggests some requirements in terms of number of entries and lookup/insertion/deletion frequencies which are all met by the NCE.
Despite the high compression rate and the lookup performance, the NPT has not been leveraged in any complete design of an NDN router yet. In fact, related work features either Bloom Filter based or Hash-based PIT designs whose characteristics are illustrated in the rest of this section.

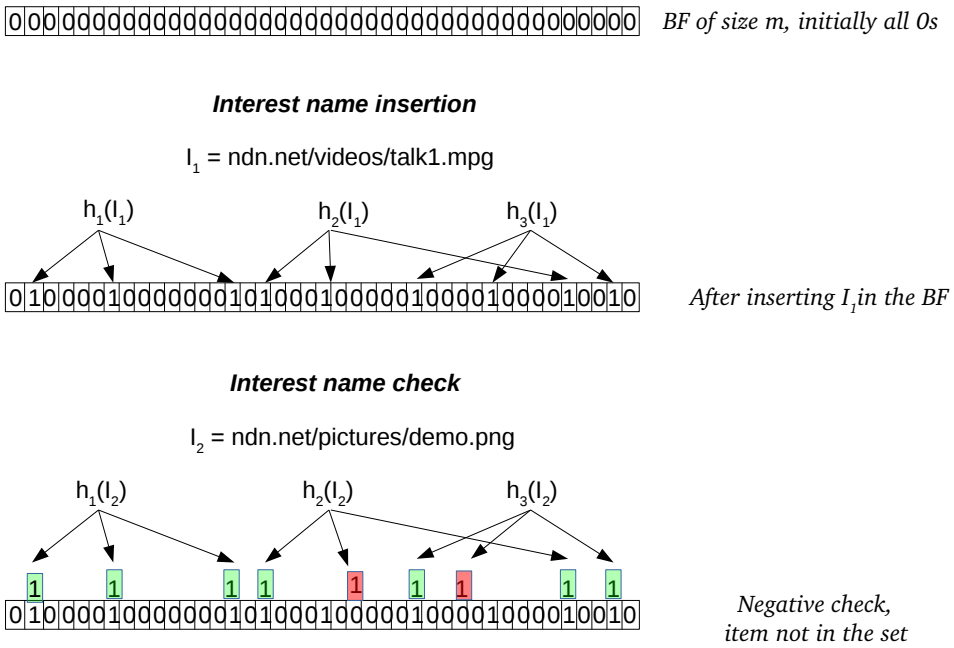


Figure 2.3 – An example of bloom filter used to record pending Interests. The filter is initially all set to 0. At every Interest reception, the Interest name is hashed k times, with each hash producing several bit locations in the filter which are set to 1. To check if an Interest/Data is in the filter, its name is hashed k times and the corresponding bits in the filter are checked. If at least one corresponding bit in the filter is not set, then the filter does not hold this Interest/Data.

2.2.2 Bloom Filter-based (BF-based)

A bloom filter (BF) [91] is a memory-efficient data-structure for membership queries which suffers from false positives. A standard or binary BF is represented by an array of m bits initially all set to 0. The BF uses k independent hash functions to map items of a set to the array m. An

item is inserted in the filter by hashing it k times and setting the resulting 1 bits of every hash in the array m. To check if an item is in the filter, the item must be hashed k times and every time the 1 bit locations in the computed hashes must correspond to the ones in the array m. If at least one 1 bit position in any of the hashes corresponds to a 0 in the filter, then the item is surely not part of the filter. If all the 1 bits in the hashes correspond to the 1 bits in the filter, then either the item is in the set or it is a false positive. The false positive probability of a BF depends on the size m of the filter, the number n of elements in the set and the number k of hashing functions, so it can be tuned.
BFs are an excellent candidate for exact matches on Interest names. In fact, Interest names, which work as keys for PIT entries, can be added to a bloom filter and then the filter can be queried to check if a new packet, Interest or Data, corresponds to any existing set element as illustrated in Fig. 2.3. Further, there exists a considerable amount of literature about the use of BFs for packet processing algorithms [92]. In fact, BFs have two appealing properties i) they are compact, ii) they do not generate false negatives, that is, if the query result is negative, the element is surely not part of the set. Those properties make possible to design compact filters which can be queried quickly before performing any more cost-intensive operation. For example, BFs have been used to speed up IP longest prefix matching algorithms [93]. Therein, BFs stored in on-chip memories are used to footprint the set of IP prefixes which are stored in hash tables on off-chip memories. This idea reduces drastically the number of hash table lookups with the assumption that those are slower and more costly than the BF ones.
Although bloom filters are able to compactly store sets of elements which can be queried quickly, they cannot be applied straightforwardly to represent a table of the pending Interests for three main reasons. Firstly, a PIT entry contains more information than the Interest name itself. Interest names can be stored in a bloom filter, but the other entry's information (see Sec. 2.1) cannot. Therefore, when BFs are used to represent a PIT, a subsidiary data-structure is required to store the full entries. Secondly, traditional binary BFs do not support deletions of elements where PITs needs to be updated when either Data consume existing entries or those expire after a timeout. To overcome this second issue, a BF variant, which is called Counting Bloom Filter (CBF) and requires more space in memory, can be used. CBFs replace binary digits in BFs with integer counters. Last but not least, even though the error probability is tunable and can be kept very low, BFs can always generate false positives.
With regard to false positives, it is important to understand their consequences and take the necessary precautions in any BF-based PIT design. A BF-based PIT is queried both at Interest and Data reception. The verification process is identical, but the consequences of a false positive in those two cases are different:

(A) a false positive at Interest reception means that an Interest which indeed was not seen before is wrongly considered to be duplicate. As result, the Interest is dropped and not forwarded. Although there could be Interest retransmission mechanisms either at the client side or in the network, this is definitely an undesirable critical effect.

(B) a false positive at Data reception creates two issues: i) *incorrect downstream Data forwarding* where no related Interest had been previously received; ii) *incorrect deletion* since the BF is updated by an incorrect Data forwarding and, thus, state relative to other Interests is irreparably lost. The former issue is most likely naturally solved by other downstream routers, where the Data are detected as unsolicited and so dropped. The consequence of the forwarding is some unnecessary consumption of network data rate. As for the issue with false positive at Interest reception, the latter issue can be mitigated by regular Interest

retransmissions, but it is considered more critical than the former one.

An early idea to leverage BFs to compress the PIT, named United Bloom Filter (UBF), was sketched in [94]. Therein, a BF is associated to each interface and the PIT is seen as the union of all those BFs. That design accepts False positives at Interest and Data reception assuming retransmissions deal with those. UBF uses two binary BFs rather than CBFs since the latter ones cause incorrect deletion when false positives happen at Data reception. Regular deletions are implemented through a mechanism of aging between different epochs where the two BFs are swapped and one of them is cleared. UBF assumes all PIT entries exist for a short amount of time and so they can be deleted after that anyway.
DiPIT, a PIT design based on BFs, was proposed in [95]. The DiPIT design aims to tackle the issues generated by false positives, which affect the regular Interest and Data forwarding as detailed in points (A) and (B), by lowering the related probability. DiPIT is made of per-face counting bloom filters complemented by a single shared binary bloom filter. DiPIT features separate BFs not to explicitly keep track of Interests incoming interfaces. The single shared BF is queried when any of the CBFs generates a positive response for an Interest name. The additional BF is meant to lower the probability of the A issue as that is the product of the false probability errors of the two bloom filters. So, received Interests are checked first against the per-face CBF. If the check is positive, then a second check is done against the shared bloom filter. If the second check is again positive, then the Interest is considered duplicate and so dropped. If not, the Interest is regularly forwarded and later added to the shared bloom filter. Instead, the forwarding of Data only goes through the per-face filters. Data packets are supposed to be checked against all the per-face BFs except the one those are received at. The design includes an additional control-plane triggered mechanism to update the shared BF, which being a binary filter does not feature an automatic deletion mechanism.
Despite the positive evaluation results reported in [95], the DiPIT design fails to address some other PIT requirements which would make impossible to apply it without further refinements. For example, the DiPIT approach assumes PIT entries only store names and incoming interfaces when, as outlined in the list of general requirements for a BF-based design, there is additional information about nonces, outgoing interfaces and lifetime timers to be stored too.
Another BF-based PIT design, called MaPIT, is proposed in [96]. MaPIT mainly aims to reduce the on-chip and off-chip memory cost while keeping up with line-speed processing without incurring in high false positive rates. MaPIT is based on a data-structure, called Mapping Bloom Filter (MBF), made of two main components: an on-chip Index Table and an off-chip Packet Store. The index table records the presence and the address of entries in the set respectively through a BF and a Mapping Array. The packet store records information about the entry except the name, which is omitted. MaPIT achieves higher compression rates for tested data-set compared to hash table-based designs and to the trie-based NCE of [88]. Nevertheless, the description of the design presented in [96] does not detail how the critical issues above described in points (A) and (B) are addressed.
Overall, BF-based PIT designs pose some constraints on the placement of this data-structure on a device's line-cards, a design issue which is explained in Sec. 2.3. Moreover, any BF-based design must be complemented by subsidiary data-structures. In fact, nonces and timers must be stored in each entry to guarantee loop detection and entry expiration.

2.2.3 Hash Table based (HT-based)

Since more than a decade HT-based systems and algorithms have been extensively investigated for wire speed network processing. The reasons for such an increasing research interest are some of the promising conclusions drawn by related theoretical research. A foremost theoretical finding is that there exist simple hash functions which perform similarly to theoretical fully random hash functions and that can be implemented efficiently. That and other theoretical results have finally led to efficient software and hardware implementations for practical applications in areas like packet forwarding, load balancing and network monitoring.
In hash tables, the time to lookup a particular item is expected to be constant on average. However, hash tables suffer from collisions, i.e., multiple items can be assigned to the same bucket. Hence, some lookup performance may differ from average till a certain worst case time. Luckily, collisions lead to worst case lookup times whose upper bound can be known with high probability. More in detail, collisions are usually solved by two techniques: chaining and open addressing. Chaining hash tables store more elements per bucket usually in a linked list. Chaining avoids collisions, but it increases the number of memory accesses when a collision happens, since multiple elements within the same bucket need to be checked. There exist techniques to improve the performance of hash-table with chaining, which exhibit better probabilistic worst case lookup guarantees (for example, multiple chain hash tables are hash tables where every item is stored in a bucket according to one of several hash functions). Yet another method to resolve collision are the open-addressed hash tables whose buckets store a fixed constant number of items. Usually, this number is fixed so that all the items in a bucket can be read in parallel (e.g., the size of a cache line). In open-addressed hash-tables, all the items are stored in the table and there are no external lists. So, the advantage of open-addressing is that it eliminates the need for pointers and this extra memory can be used to increase the size of the table itself. The downside of open addressing is that to perform an insertion and a search, different slots may need to be examined.
With regard to the PIT, HT-based designs maps keys (Interest names) to values (PIT entries). An advantage of any HT-based design over to the BF-based ones is that the former can store full information about PIT entries and so preserve PIT properties, like loop detection. Further, authors of [97] also motivates why HT-based PIT designs are better amenable to different placements on hardware forwarding devices (see Sec. 2.3). The clear disadvantage is that HT-based designs have a bigger memory footprint than the BF-based counterparts.
In what follows three different HT-based PIT designs are presented with the aim of giving an overview of the state of the art. Those designs had been driven by slightly different goals. On the one hand, the schema proposed in [97] by Perino et al. targets an hardware implementation of the single PIT component. On the other hand, the schema in [98] targets a software design for a full NDN data-plane. Finally, the schema proposed in [86] targets an Internet scalable design, which is motivated by the infeasibility of state-of-the-art pure HT-based PIT designs for core router speeds.

In [97], the authors analyzed two HT-based solutions to implement the PIT: linear chaining hash-table (LHT) and open-addressed d-left hash-table (DHT). Both solutions store full PIT entries and can support deletions. The space needed by each kind of data-structure is illustrated in Fig. 2.4. In particular, under the assumption to allocate 48 bits to store an entry's subsidiary information, LHT requires 112+l bit per entry, while DHT requires 80+l bits, where l is the content name length in bits. The additional 32 bits required by the LHT schema are due to the pointer needed to implement the chains as linked lists. In [97], the two HT-based designs are compared in terms of memory footprint under some assumptions about the length of the content names and the number of PIT entries. The number of entries is theoretically computed for 10 and

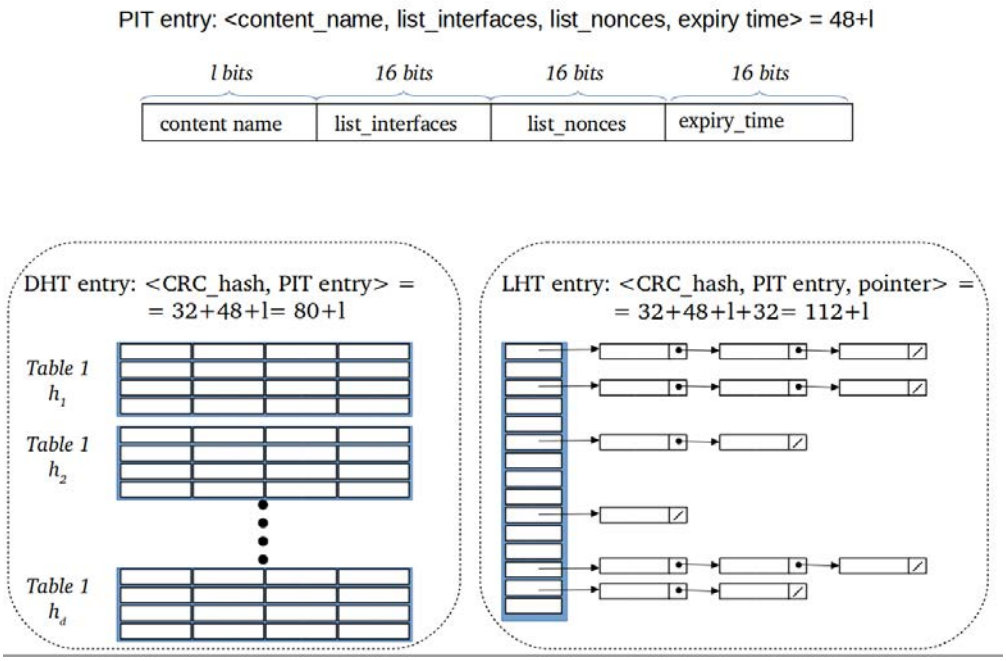


Figure 2.4 – Illustration of the data-structure and the table entry for the linear chaining hash-table (LHT) and the open-addressed d-left hash-table (DHT) PIT designs.

40 Gbps in both average and worst case packet arrival rate. According to the findings, none of the state-of-the-art data-structures for the PIT can fit on on-chip memory, i.e., on SRAM (access time 0.45 ns, maximum size ~210 Mb), for worst case scenario. Furthermore, the two HT-based solutions would not fit on existing RLDRAMs (access time 15 ns, maximum size ~2 Gb) either, so nowadays those must be implemented on DRAM (access time 55 ns, maximum size ~10 GB) [5]. As a further result, DHT seems to be more resilient to the traffic load growth. While, other data-structures degrade the more the load approaches the worst case. The bottom line in [97] is that DHT seems to be the more promising solution among the data-structures compared to (CBF, DHT, LHT, ENPT). Moreover, DHT suits to different placements on a device's line-card while sustaining traffic up to 10 Gbps as confirmed by a prototype implementations on a Cavium network processor.

In [98], a software design for a complete NDN forwarding engine is proposed and tested on a target platform. The proposed HT-based design has been optimized to perform operations for all the main data-plane functional blocks, i.e., CS, PIT and FIB. This design is based on a compact representation of a chained hash table. Further, it only uses a single hash function because that must be computed on all the prefixes of each name for forwarding purposes. An illustration of that schema is shown in Fig. 2.5. The chaining hash table is implemented as a compact array of buckets rather than as regular linked list. The objective of this representation is to minimize data cache misses during lookup since those introduce memory stall cycles. Every bucket contains seven compact entries, made of the hash of an entry name and the pointer to the full entry. The name hash is stored instead of the full name in the bucket so to avoid string comparison to all the entry in a bucket. String comparison to the full name is only done when the stored

[5] All the information about different memory technologies' access time and maximum size reported in [97] refers to the previous estimates in [99].

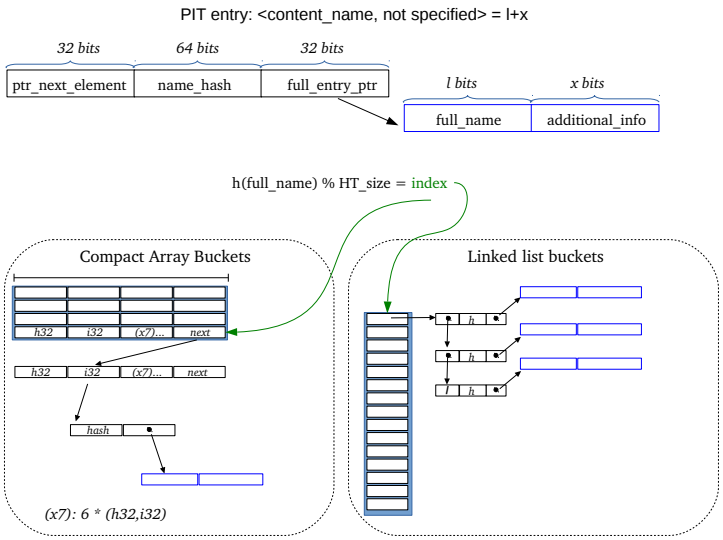


Figure 2.5 – Illustration of the data-structure and of the table entry for the Cisco's hash table-based PIT design.

hash matches the content name one. Moreover, some pre-fetching of the HT buckets is done to further reduce the number of data cache misses. This design envisions further optimization of the PIT lookup operation. In fact, the PIT can be partitioned and partitions can be assigned to different cores/threads so for those do not conflict on resources and fully exploit the parallelism on multi-core/multi-threaded systems. For the second optimization to work, Interests and Data related to the same entry must be assigned to the same PIT partition. This last requirement is met by assigning packets to a partition based on full-name hashes. From the reported evaluation, the throughput of the NDN forwarding engine is minimally affected by different PIT sizes, yet its performance drops when the HT load factor (#entries/#buckets) is greater or equal to 8. The full router design of [98], which is implemented on the Integrated Service Module (ISM) of a Cisco ASR 9000 router, achieves an average throughput of 8 Mpps when running on multiple cores (The reader should be aware that some of the latest NDN router designs proposed [100, 101] can achieve higher throughput respectively of 10 and 40 Mpps on off-the-shelf hardware).
Another HT-based schema is proposed in [86], where the PIT is implemented as a d-left hash table. That design achieves a PIT compact representation at core routers by storing fixed-length fingerprints instead of full names. Assumptions on Interest aggregation at edge routers are made to preserve the correctness of packet delivery (this design choice is driven by scalability issues in PIT design which are discussed in Sec. 2.3). The data-structure is illustrated in Fig. 2.6 and is based on the d-leftCBF (dlCBF) construction proposed in [102]. A d-leftCBF is a hash table based data-structure which replicates the functionality of counting bloom filters but requires less space. In dlCBF, cells stored in d buckets contain a fixed number of bits for fingerprints and counters instead of full elements. The d tables have B buckets which can contain up to E entries. The d hash functions compute the index of the bucket and the fingerprint. The design in Fig. 2.6 also includes an overflow table since d-left hashtables suffer from overflow when all the possible buckets where an element could be placed are not empty. Every PIT entry is made of five fields. The Occupy bit indicates if the entry is in use and enables the implementation of lazy deletion mechanisms by flagging/unflagging this bit. The collision bit indicates the presence of one or

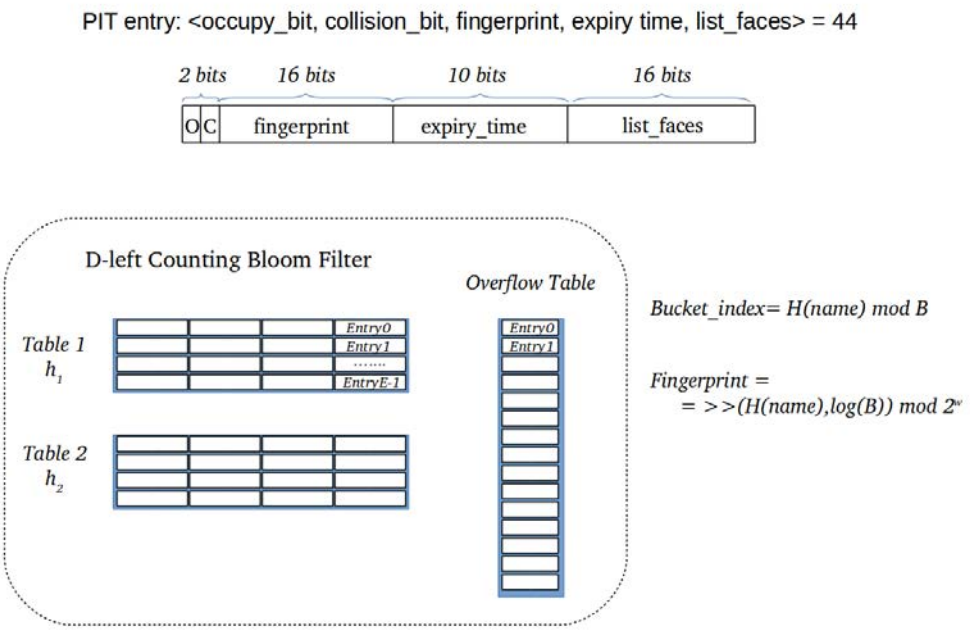


Figure 2.6 – Illustration of the data-structure and of the table entry for the d-leftCBF (dlCBF) PIT design.

more Interests received for this entry. The fingerprint is computed by the hash function for a certain table based on the content name. The expiry time is the timeout validity for the entry. The list of faces is a bit vector of the same size of the number of available interfaces on a certain device.
The analysis of the memory size of this PIT design is highly dependent on the fingerprint distribution. In particular, it is important to derive an upper bound on the number of duplicate Interests since those will increase the lifetime of the relative PIT entries. Such an upper bound is computed in [86] and is used to perform a worst case analysis on the memory size for this PIT design. According to the analysis in [86], the dlCBF design's size seems to be of the order of 40MiB in the ideal case, which would fit in on-chip SRAM memories, while it seems to be around 250 MiB in the worst case, which would instead mandate the use of RLDRAM memories.
Nevertheless, storing fingerprints instead of full names in PIT entries compromises the basic services at NDN routers, that is, Interest aggregation and multi-cast Data delivery (see Sec. 2.1). Therefore, any fingerprints-based PIT design should be complemented with additional mechanisms so as to guarantee the correctness of the above basic PIT services.

2.3 Design challenges

Beyond the choice of a data-structure, there are other factors which must be considered when designing a PIT:

1. First and foremost, the appropriate PIT size should be determined since insufficient PIT space leads to Interests to be dropped. A PIT has to be large enough to sustain the traffic rate carried on the device's links across static and dynamic cases in different network settings. Because of the tight resource constraints in the design of architectures for NDN-compliant network equipment, overprovisioning PIT space is not considered a viable option. Further, PIT has preferably to reside on fast memory to cope with certain link speeds. This problem is usually referred to *sizing* or *dimensioning*.

2. Second, although NDN is a clean-slate proposal, it has not been assumed yet to redesign current routers architecture. By consequence, the NDN data-plane must be implemented on existing routing/switching architectures. The abstract NDN data-plane assumes the existence of globally accessible tables which, however, are not present in the majority of current routing/switching architectures. In fact, most routers have several line-cards, usually connected through a high-speed crossbar switch-fabric, containing the interface logic for a data link. Then, every line card has a forwarding engine CPU with non-shared memory resources leading to the question of where to store information about the PIT so that it can be accessed consistently by several line-cards' CPUs. This problem is usually referred to *placement*.

3. Third, assessing the *scalability* of a certain PIT design could become critical when NDN has to be deployed either at Internet scale or in resource-constrained networks. At a first glance, this issue could be considered analogous to the PIT *sizing*. However, dimensioning the PIT does not yield any universal design, rather it provides general guidelines and resource bounds. Once boundaries are known, moving "from principles to practice" [86] could require additional design trade-offs, e.g., the choice of a certain data-structure over another and/or changes to the normal data-plane operations to achieve packet processing at a certain line-speed.

4. Last but not least, security of any proposed PIT design must be thoroughly assessed since the PIT constitutes a functional block of core network elements. In NDN, normal user's requests create and modify PIT entries in routers. Unfortunately, user's misconducts can be really disruptive. They cannot be predicted a priori and can be achieved via legitimate means. Thus, the possible attack surface has to be limited at design time.

2.3.1 Sizing

Compared to literature on the design of a data-structure for the PIT, the understanding of the required PIT size as a function of the system parameters has received much less attention. Theoretically, such size could be estimated as a function of two components, the data-rate of the router's links and the average time entries last in the PIT. This simple formula would need, of course, to be complemented with assumptions on packet sizes for Interest and Data to estimate the number of packets which may create and consume PIT state. Practically, there are other factors like content popularity distributions, application request patterns, routers network position which ask for a much more difficult characterization of the resources required for a PIT.

Related works on PIT sizing account to only three works [103, 88, 104] leading to contradicting findings. Seminary, the work in [104] analyzed the content state in content-centric networks mainly to identify possible attack vectors. Therein, the router's forwarding state evolution is modeled over time as a function of the link utilization and the average network Round Trip Time (RTT). Nevertheless, content-centric networks are expected to exhibit highly variable RTT measurements because of their intrinsic characteristics (e.g., in-network caching, multipath forwarding). Therefore, differently from what has been done in the past with TCP, good estimates of average RTT values will be hardly predictable by routers in content-centric networks. By consequence, large RTT fluctuations would make it difficult to provision properly resources on routers. In any case, assuming a steady-state model in the absence of retransmissions and packet loss, the lower bounds are estimated respectively as $16 * 10^4$ and $16 * 10^6$ for the number of PIT entries on 1Gbps and 100Gbps links (assuming maximum link capacity, an average packet size of 1000 bytes and the Internet RTT mean and standard deviation values measured by the PingER project). A further strong, yet apparently premature according to some later work [103], assertion in [104] is that the estimated lower bound for the number of PIT entries should hold for any router where it is only a function of its interface capacities and of the average time an entry remains in the table (similar assumptions were held in [99] which, however, led to a different estimate of the number of entries).
A more empirical analysis is presented in [88], where a one-hour trace of a 20Gbps link is used to estimate PIT size and access frequency. That work details a method to map TCP/UDP connections for a common set of Internet applications (e.g., HTTP, FTP, P2P applications, email services) to NDN Interest/Data exchanges. The mapping method is used to identify and group flows in the trace, which are related to the same applicative data exchange, and then to convert them to PIT operations. In terms of size, the proposed mapping method produces 1.5 M PIT entries for the main applications (that is, excluding network management protocols and domain name resolution messages). Furthermore, read, insertion and deletion frequencies are estimated through flows behavior too. So, for example, the emergence of a new flow is considered as a PIT insertion, while the disappearance is considered as a PIT deletion. The computed lookup, insert and delete frequencies account respectively for 1.4 M/s, 0.9 M/s and 0.9 M/s (Million per second). Although the work lacks a theoretical model and the mapping method is highly arguable, it still makes a valid point about the different nature of Internet services which generate different PIT access patterns.
Finally, the work in [103] presents an analytical model of average and medium PIT size over time at steady-state in a single bottleneck scenario. The proposed model is validated on an ad-hoc high-speed platform by using synthetic and trace-driven traffic workloads. The model applies a deterministic fluid model consistently to what was done on previous well-accredited literature on buffer and queue sizing in IP routers. Furthermore, a major contribution is that the model takes into account queue oscillations and dynamics, which have been neglected by previous works. PIT dimensioning is modeled in the very worst case, without in-network caching, Interest aggregation and expiration of PIT entries.
Hereby some of the main findings of [103] are reported:

- in the static case (constant number of parallel flows), there is a strong correlation between the status of the queue on a bottlenecked link and the PIT sizes on the nodes downstream it.

- the PIT size in routers downstream a bottlenecked link does not depend on the bottlenecked capacity rather on the number of parallel flows.

- experiments suggest that proper settings in a congestion control algorithm can be used to reduce the PIT size while fully exploiting the link capacity. In the dynamic case (number of flows dynamically changing during the experiment), correlation between queues at bottlenecked links and PIT size in routers is still present. Moreover, the PIT size at a certain router grows as a function of the load on the upstream bottlenecked link.

In a realistic ISP topology, the model is used to estimate the PIT size for three network equipment with very different links fan-out in different networks: an optical line terminator (OLT) in the access network, an IP edge router in the backhaul network and a router of the backbone. Overall, the estimated PIT size never seems to make the design of the component unfeasible. OLT's PIT stays very small (less than one thousand entries in the very worst case where all the flows are bottlenecked upstream the OLT) and depends on the flows bottlenecked upstream. Backbone routers are estimated to need around 2 million entries in the very worst case. The required number would account for a PIT size of 400 MB in the considered implementation (d-left HT-based PIT in [97]). Such a PIT size would easily fit on off-chip DRAM technology yet it could only sustain a few tens Gbps data-rate, which would be a bit less of the requirements for that portion of the network. No indication about PIT size at the edge router are reported, where only a statement about the absence of any concern for dimensioning the component at that stage is made.
More importantly, the latest state-of-the-art work on PIT sizing [103] has generally dismantled concerns and estimates raised by previous related works. Further, that work has undoubtedly advanced the general understanding of the memory requirements for this NDN data-plane component. Nevertheless, for the sake of simplicity, the derived analytical model is based on assumptions that may not be necessarily true in every network scenario and, so, whose impact should be better studied too. For example, client requests are expressed assuming there is an optimal congestion control mechanism regulating them, or, no other management schemas, e.g., to mitigate DoS attacks, are considered either.

2.3.2 Placement

The work in [97] provides a good overview of the different PIT placement options on router's line-cards. Therein, an analysis of the pros and cons of each approach is presented and, moreover, a novel placement schema is conjectured to address issues with the state-of-the-art solutions. Hereby those options are recapitulated assuming, for the sake of simplicity, that device line-cards can be logically separated in two categories: input and output.

Input-only placement [94, 95]: PIT tables are only placed on each input-line card. By consequence, an Interest creates a PIT entry only on the card where it is received. By contrast, Data packets must be broadcasted to all the line cards except to the one those are received to in order to find possible related PIT entries. Hence, this placement requires additional switching and multiple PIT lookups at Data reception. Even worse, this placement does not preserve Interest aggregation and loop detection since similar Interests yet received at different line-cards generate PIT entries on different tables.

Output-only placement [88] PIT tables are only placed on each output-line card. Interests do not create PIT entries at input line-cards where those are just looked up to the FIB. Rather PIT tables are stored on output-line cards. This approach fixes issues with Interest aggregation

and loop detection the input-only placement is affected with. However, it requires a FIB lookup per-Interest and it shows limitations in case of multi-path. The former issue matters since it may introduce unnecessary FIB lookup when a PIT entry is already present on some output card. The latter issue is present if there exist multi-path FIB entries. In fact, those routing entries create separate entries on different output line-cards which then will possibly receive and switch internally multiple times the same Data packets. This latter issue can be further worsened by cases where a forwarding strategy module decides upon the available forwarding options and may alternate output cards leading to much more inconsistency among the PITs on different output cards. This last scenario affects the ability to detect loops and retransmissions since different nonces may be stored in different output line-cards' PITs.

Input-Output placement [88]: this placement suggests PIT entries should be created on both input and output line-cards for every forwarded Interest. This placement does not create any unnecessary FIB lookup. However, it suffers from a multi-path Data replication issue analogous to the one described for the *output-only* placement. Thus, multiple Data packets may be internally switched through the input line-cards if PIT entries were previously created at different output line-cards. Furthermore, this placement increases of a factor of two the required number of PIT lookup operations.

Third-party placement [97]: this placement steers every received Interest to a dedicated line-card based on the content name hash. The selected line-card holds the PIT for all the received Interests. At Data reception, the output-line card identifies the related line-card by using the same hash function on the Data name. The Data is forwarded to the designated card. On the one hand, this placement introduces additional in-router switching operations both at Interest and Data reception. On the other hand, it does not experience the issues with loop detection and multicast delivery the other placements do.

2.3.3 Scalability

The deployment at the Internet scale of NDN requires routers being able to operate NDN packet processing at high line-speeds. Above certain link speeds, the inter-arrival packet translates to a very few nanoseconds to process a single packet. As known in routers, memory access affects the packet processing more than combinatorial logic. Thus, tight execution times make the number of allowed memory accesses very limited. This imposes additional constraints to the PIT design too.
As evidence of this issue, the work in [86] shows how the state-of-the-art HT-based PIT implementations may not meet the requirements of backbone routers. In fact, the required memory to sustain tens of interfaces at 100 Gbps is estimated to be around 500MiB, which would oblige to implement HT-based PIT on off-chip DRAM memory. However, the required packet processing speed would not tolerate multiple read/write operations on off-chip memories. Storing compressed filters on small SRAM chips has been showed by the related work on high speed packet processing to be a valuable resource to save DRAM accesses. Along the same line, the work in [86] proposes a PIT design which stores fingerprints instead of full content names at core routers and reduces the number of required DRAM accesses per PIT operation. However, storing fingerprints incurs two main issues, namely, collisions and duplicates. Collisions happen when two different Interest names produce the same fingerprint and hash bucket index. Interest duplicates are a normal phenomenon, which, however, become difficult to detect since those are

identical to collisions. The colliding Interest should be forwarded, while the duplicate one should be dropped. So, as can be guessed, this design needs some refinements to deal with those two side effects. For example, the work in [86] describes as the aggregation of requests for popular contents in edge routers, caching and adaptive forwarding behaviors can be leveraged to address and/or limit the aforementioned issues. Further, it identifies the consequences in terms of network and memory overhead generated by those collateral effects and measures via simulation the fingerprint collision rates.

Security: security in PIT design entails many aspects. First, the PIT is subject to user-driven modifications. So, in theory, PIT designs should be resilient to any possible users' misbehavior. However, in practice, the PIT design can already be driven by some well-known security threats. For example, a DDoS aiming to overload PIT resources has already been identified and characterized by some related research [105, 106, 107], so resilience to those existing threats should be considered at design time. Second, with regard to the data-structure, HT-based PIT implementations seem to be the mainstream choice. Therefore, a careful choice of the hashing function is vital to reduce the hash computation and bucket lookup time. Otherwise, certain hash functions could expose routers to hash flooding attacks aiming to generate many collisions so as to degrade the average lookup performance. Finally, the PIT represents the status of the requests processed by a certain router at anytime. Thus, snapshots of the PIT state can be misused for censorship or monitoring purposes. This naturally raises questions about the information revealed through this component and the degree of privacy which should be guaranteed.

The above-discussed four issues and the choice of the data-structure covered in Sec. 2.2 account for some mainstream problems in the PIT design space. Additional secondary requirements, e.g., the PIT entry timeout and the Interest retransmission policies, may further influence the design of this component.

2.4 Properties

The previous section has outlined challenges in the design of a Pending Interest Table for NDN. Meantime, the PIT state in the data-plane establishes properties which go far beyond the original conception of the component. A PIT entry represents network state of the lowest granularity, that is, per-Interest per-hop. Further, the semantics of the PIT state is left undefined by the NDN architecture design and so can be leveraged to implement very different features. In what follows, some of the mainstream uses of the NDN datagram state are briefly illustrated. Nevertheless, for the sake of completeness, it also must be said that potential benefits of the NDN stateful forwarding plane are still vividly debated among the ICN research community members. Further, there have been strong claims about "supposed benefits" in face of real problems [108].

Anonymous communication: NDN packets contain neither source nor destination addresses. The lack of addresses enables anonymous communication between different parties where only the content is revealed. Forwarding Interests upstream is guaranteed by an anycast name-based routing which aims to identify nearby sources of a certain named information. Forwarding Data downstream is made possible only by the so-called "breadcrumbs trail" left in routers' PITs by the related Interests. The PIT state is used to implement Reverse Path Forwarding and deliver Data to consumers who requested them.
Further, host anonymity in NDN shields specific recipients from being targets of Denial-of-service

attacks. The address-less Interest and Data packets do not carry the identity of a host or a group of hosts. This characteristic makes unfeasible to launch a large set of DoS attacks commonly used in today's IP networks, e.g., TCP flooding, replay attacks.

Interest aggregation/collapsing: PIT aggregates duplicate requests for the same content by merging information about similar Interests in a single table entry and by suppressing exact duplicate ones. Merging similar Interests reduces both upstream traffic and load on content providers. Overall, Interest aggregation saves network bandwidth.

Multicast delivery: PIT entries record incoming interfaces of similar Interests which have been aggregated. This information is used at Data reception to multicast out on different downstream interfaces copies of the same Data packet.

Multi-path loop-less routing: FIB entries may include multiple next-hops for the same prefix name. Therefore, multiple paths can be explored seeking Data for Interests under such prefix names. Multiple Data traveling downstream do not create concerns since once the first received Data consumes the related PIT entry, all the next Data are considered unsolicited and so dropped. Nevertheless, NDN Interest packets have no IP-TimeToLive analogous header field to limit their lifespan[6]. Meaning that some Interest could loop indefinitely (though an application can still specify an Interest's approximate maximum lifetime, see the description of the *InterestLifetime* packet field in Sec. 1.2.2 of chapter 1). Such race conditions are avoided by means of Nonce information recorded by PIT entries. Nonces allow routers to distinguish between looped Interests and retransmissions or new requests.
The multi-path ability enabled by the PIT enables probing multiple interfaces in parallel and possibly discovering new "best" paths. Further, offloading loop-detection mechanism in the data-plane also relaxes complexity in the design of routing algorithms leading to the exploration of solutions considerable unfeasible before [62].

Hop-by-Hop flow balancing: the PIT guarantees that on each link one Interest brings back no more than one Data. This strict 1-to-1 Interest-Data mapping naturally allows the routers to control the load over their links. Furthermore, this flow balance guarantee can be leveraged to design in-network congestion control mechanisms [109] by assuming the content/interest size ratio on a certain link can be estimated (as done by the Interest shaper mechanism in [110]).

Adaptive forwarding capabilities: the presence of per-packet state in the NDN data-plane can be leveraged to implement adaptive forwarding behaviors as forwarding strategies (see Sec. 1.2.2 of Chapter 1) at the network layer. In IP, routing protocols exchange information about available routes and compute best paths from sources to destinations according to link costs and policies. The routing information is later made available as table entries that the data-plane forwarding logic dumbly follows. Whenever path changes happen because of planned routing updates or unexpected network failures, routing protocols converge to a new set of valid paths by exchanging update messages and adapt to the topological network changes. However, routing convergence takes a certain time and may yield to some side effects like routes oscillation and destination unreachability, particularly in the presence of short-term connectivity losses [111]. Instead, the NDN architecture leaves the semantics of the PIT state undefined. The granularity of the for-

[6]It should be noted that since the version 0.3 of the NDN packet specification, the optional field *HopLimit* has been added to the Interest packet to indicate the number of hops an Interest is allowed to be forwarded.

warding state in PITs enables the implementation of a very different range of services. Basically, storing information about each Interest enables routers to precisely measure performance of the data-plane (e.g., by measuring at each hop the RTT of every Interest-Data exchange). Those measurements can be leveraged to test forwarding paths as well as to dynamically react to sudden network problems like congested links. The related work in [49] has already showcased how data-plane designs can leverage the PIT state to achieve robust packet delivery. The forwarding strategy designed in [49] emits Interests retransmissions, performs proactive interface probing and implements Interest NACKs by using the information stored in the PIT. According to the evaluation, the designed forwarding strategy deals efficiently with network congestion, security attacks, broken links.

2.5 Summary

This chapter has described the Pending Interest Table (PIT) component of the NDN's data-plane. The description details the PIT operations in the forwarding of Interest and Data packets across the network. The mechanics of the PIT impose certain requirements to the design of a data-structure for this table to operate at high link-speeds. Those requirements have been sorted and the state-of-the-art PIT designs have been analyzed. Additional design requirements about the size of table, the scalability, the placement on a hardware forwarding device, the security, have been illustrated too. Among the design requirements for a PIT, the security is the one which stays unsolved at the time being and, so, deserves immediate attention. In fact, by design, the PIT can be easily abused by malicious users to carry out very disruptive Denial-of-Service attacks. For the above reason, the next two chapters address the security of the PIT by reviewing the state of the art on this subject, reassessing existing solutions and proposing novel ones.
To conclude, this chapter also illustrates some ancillary properties enabled by the per-Interest state kept in the PIT. The importance of the PIT state is outlined since it is leveraged today by several mechanisms present in the NDN architecture. Therefore, a clear understanding of those properties is mandatory if the PIT existence has to be rethought. For example, as it is advocated in Chapter 5 where research on alternative forwarding mechanisms is pursued to address the current PIT security issues.

Chapter 3

Interest Flooding Attacks

Interest and Data packets carry hierarchical human-readable names which are made of globally-routable prefixes and provider-specific content identifiers. NDN packet names are used by a stateful forwarding plane in routers. In fact, processed Interests are temporarily stored by their names in routers' pending interest tables for homonymous Data packets to be routed on reverse paths. On one hand, such status in the network can be leveraged to implement useful features at the data-plane level like loop-free multipath forwarding, flow balance, real-time recovery from network anomalies, etc. [49]. On the other hand, this state exposes NDN networks to novel Distributed Denial-of-Service (DDoS) attacks. Interest Flooding Attacks (IFAs) are NDN-specific DDoS attacks issuing a huge quantity of unsatisfiable Interests with the aim to exhaust network's and content providers' resources [105, 106, 63]. The severity of IFAs for NDN-like networks and the questionable efficacy of the proposed solutions threaten the stateful forwarding plane existence in ICNs [108].
The aim of the work presented in this chapter was to avoid premature conclusions on the efficacy of the state-of-the-art countermeasures against IFAs. In fact, while several countermeasures had been proposed to mitigate this kind of attack, overall, simplistic assumptions widely used in the evaluated attack scenarios risked to provide erroneous, either promising or deceptive, findings. First, the quasi-totality of the existing countermeasures are reactive, i.e., the mitigation of an attack inevitably starts after a time interval in which routers collect statistics about the processed traffic. Second, the evaluation scenarios for those countermeasures often assume malicious and legitimate Interests belong to trivial disjoint prefix sets and/or use randomly-generated content identifiers to generate requests for non-existent contents. The former makes these countermeasures vulnerable to attackers which may adjust their Interest generation pattern to influence collected statistics or exploit routers' monitoring time windows. The latter makes it easy to drop malicious traffic once a certain prefix has been detected as infected. None of the two factors was considered by attacker models in related work. Therefore, this work proposed a stealthy attacker model where attackers would mimic real content names and vary the attack characteristics over time in order to remain undetected by routers.
Sec. 3.1 introduces the IFA security threat and a novel related attacker model. Sec. 3.2 provides an overview of the state-of-the-art IFA countermeasures by outlining weak points both of the existing defense mechanisms and in the related evaluation scenarios. Sec. 3.3 proposes some new IFAs based on the novel attacker model, which capitalize on the pitfalls of existing defense mechanisms. The efficacy of those novel IFAs against the most effective state-of-the-art countermeasures has been proved by extensive experiments which are summarized in Sec 3.4. Overall, according to the results presented in this chapter, the exploration of the design space for coun-

termeasures against this NDN-specific security threat is still incomplete. Therefore, alternative research directions are presented in the next chapters.

3.1 IFA in NDN networks

The NDN data-plane makes the architecture resilient to many of the DoS attacks affecting the current Internet [112, 49]. NDN packets do not contain physical addresses yet content names. Therefore, attackers can not aim to exhaust specific endpoints' resources like in today's application-level flooding attacks. Further, routers accept Data only when solicited by previous related Interests. Thus, prospective attackers can hardly flood unsolicited Data through the network like in today's reflection attacks in IP-networks.
Meanwhile, the NDN has been shown to be vulnerable to some totally new attacks which capitalize on architecture-specific features [112]. Among other features, the presence of per-Interest PIT state in NDN introduces some security threats. In fact, since PIT modifications are packet driven, tailor-made packets and packet generation frequencies may be misused to overload routers and content providers with unnecessary processing or to exhaust their resources, e.g., tables memory or CPU cycles. As consequence, regular traffic will suffer from a degradation of performance at best or from a denial of service at worst.
An Interest Flooding Attack (IFA) consists of a large number of closely time-spaced Interests emitted by a pool of nodes to overload the network infrastructure and/or content providers. It is generally an attack which capitalizes on the possibility of raising computation at providers and consuming resources at routers by emitting a large amount of content requests.

3.1.1 Attack Genealogy

Preliminary evidence of the IFA vulnerability, as well as of the possibility to easily exploit it, was shown in [106]. Among several vulnerabilities identified therein, the authors showed that resource exhaustion on routers can be achieved by issuing a large number of Interests for non-existent content. However, the term *Interest flooding* was coined for the first time in [69] to identify an attack mirroring traditional DoS attacks by sending large numbers of Interests hard to aggregate and be served by caches. A more precise definition was later given in [63], where the attack goal to overflow routers' PIT is also clearly stated. Finally, the first seminal proof that IFAs can severely disrupt network services was proven empirically in [107] where a modest number of attacking nodes sharply decreased the throughput delivered to legitimate consumers.

3.1.2 General Characteristics and IFA types

The kind of Interests used to generate an attack can be used to characterize an IFA in terms of attacker's goals and consequences for consumers, producers and network infrastructure. A preliminary tentative taxonomy was presented in [112] and then resumed in [113] with the aim of classifying unambiguously the Interest types and the respective IFAs.
Interests whose aim is to fetch Data for a legitimate purpose are commonly referred to Legitimate Interests (LIs). While, Interests aiming to achieve network or producers service degradation are commonly referred to Malicious Interests (MIs). A Malicious Interest can be satisfiable, when it refers to an existent content or to a dynamically generated content, or Fake, when it refers to a non-existent content (FI). FIs can be easily created by trailing randomly generated strings to

valid routable prefix names, e.g., "*ndn:/en.wikipedia.org/wiki/technology/media/512h3jh10u*". A valid routable prefix name, e.g., "*/en.wikipedia.org/*", allows FIs to create PIT entries in routers across the paths to content sources in the network. Further, assuming there is no specific eviction policy in PITs, FIs maximize related PIT entries lifetime, since those cannot be consumed by any Data packet and so last until expiration (although PIT entries persistence depends on different expiration timers set by routers, a PIT entry lifetime is supposed to last longer than the average Round Trip Time)[7].
An IFA consists of the generation of a large number of closely time-spaced MIs targeting one or few prefix names. The attack rate (which is the number of MIs per second) is important since MIs saturate router's PIT if their frequency is greater than the one at which the Interests are erased from the PIT, either consumed by Data or expired. Before the work presented in this manuscript, the following IFA variants had been investigated in the related scientific literature: i) IFAs mounted by a weakly-coordinated botnet of consumers issuing FIs, as illustrated in Fig. 3.1, which is the attack variant mainly studied by other works in literature; ii) IFAs performed by a botnet of collusive consumers and producers where consumers issue malicious yet satisfiable Interests, while collusive producers delay at maximum the delivery of the corresponding Data packets [114]; iii) IFAs performed by a botnet of malicious users issuing malicious yet valid interests for dynamic content, called DoS-signing attack in [115], so to overload content producers with an excessive generation of content signatures.

3.1.3 Attacker model

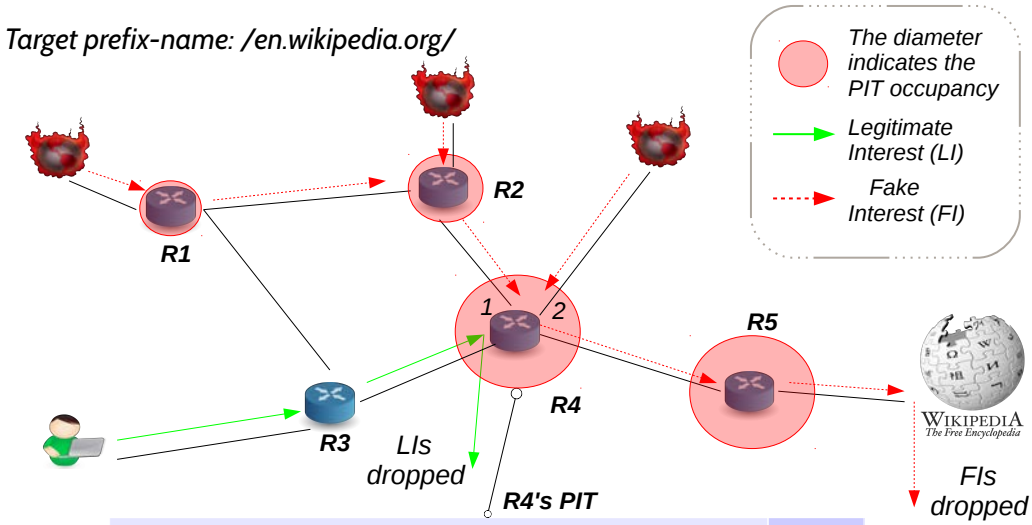
This work introduces a robust attack model with attacker capabilities and goals described in the following. Furthermore, Table 3.1 summarizes the attack impact on different targets and the Interest type that can be used to achieve it. The summary also outlines that an attacker may decide to shape the attack with different Interest types in order to achieve different results.

Attacker capabilities - Interests are regular requests expressed by users without any specific privilege. Hence, controlled end-hosts are sufficient to launch an IFA and no control over the network infrastructure is required. However, according to [116], a botnet of infected end-devices generating FIs can maximize the attack efficacy. Wherein, no router is compromised nor misused to intensify or make more efficient the attack.
In general, IFA attackers have the following capabilities:

- ability to produce Interests for existent and non-existent content,
- some knowledge about detection techniques operated in routers,
- ability to affect multiple targets at the same time by means of certain Interest types and different prefix names,
- ability to influence traffic metrics monitored by routers.

Attacker goals - IFAs may be targeting either content providers or the network infrastructure or both. So far it has appeared that they can achieve the following effects:

[7] As outlined in [63], some Interest packet header fields can also be used to create MIs. For example, different Nonces can be misused to refresh the expiration time of the same PIT entry; while the *PublisherPublicKeyLocator* field can be used not to fetch cached contents and direct all the requests to target content sources. However, the analysis of the feasibility and the efficacy of IFAs based on the above packet fields has not been investigated in literature yet.



Interest Name	Iface
/en.wikipedia.org/wiki/technology/media/abracablah	1
/en.wikipedia.org/wiki/technology/media/simsalabim	2
/en.wikipedia.org/wiki/technology/media/tatabtatab	1
/en.wikipedia.org/wiki/technology/media/patpattata	2

Figure 3.1 – Illustration of an IFA targeting a Wikipedia content provider on an NDN network. Malicious users produce closely-spaced Interests for non-existent contents (alias Fake Interests) which persist on routers' PITs until expiring. Affected routers are across the path to the content provider. Thus, as a result, Legitimate Interests (LIs) may be frequently dropped by some router whose PIT is full (e.g., routers R4 and R5).

- the exhaustion of routers' resources. Especially, they tend to fully occupy the PIT space causing further Interests to be dropped.
- the overload of requests on target providers. In order to be more efficient, such requests should ask for existent contents since those require more computation from the provider, e.g., signatures generation. Instead, FIs can be more easily filtered out by content producers.
- the saturation of network links due to a certain asymmetry in size between Interests and Data packets. Although this requires colluding parties performing the attack or asking for many different existent contents, flooding with Interests upstream links may cause Data packets received downstream to saturate the bandwidth.

3.2 IFA countermeasures and good practices

Since the identification of the IFA, a considerable amount of research has been devoted to the assessment of the severity of the attack and to the design of countermeasures to prevent it. The majority of the countermeasures against IFAs feature a two-phase, detection and mitigation, defense mechanism. The detection phase aims to identify the attack source (generally, a specific interface) and/or target prefix names. While the reaction, which is triggered by a successful

Table 3.1 – Summary of the IFA impact on different targets and of the Interest types used for each purpose.

Attack target	Attack effects	Interest Type
Content providers	overwhelmed with requests demanding computation. Some Interests may require more computation on the provider, e.g., for the signature generation [117], while FIs can be filtered out by lightweight mechanisms.	MI
Routers	suffering from resources exhaustion, especially of PIT space, which causes further incoming Interests to be dropped	MI
Network infrastructure	saturated links due to a certain size asymmetry between Interests and Data packets.	$MI - FI = \{x \in MI : x \notin FI\}$

identification, tries to either stop the attack or reduce the attack's impact.
The rest of this section introduces the state-of-the-art IFA countermeasures with a focus respectively on detection techniques in 3.2.1, mitigation mechanisms in 3.2.2, evaluation settings in 3.2.3. Finally, Sec 3.2.4 outlines pitfalls of the presented techniques with regard to those three aspects.

3.2.1 IFA detection

The detection of an attack is based on the periodic observation of some metrics that can suggest anomalous traffic patterns. Although, the metric definition and the granularity of the monitored time window may vary from a countermeasure to another, the construction of the attack indicators is based on roughly the same set of information. In fact, so far, none of the previous works on IFAs fetches information from the Content Store (CS) to monitor the router's status, a few of them [118] propose to use modified FIB entries, but most of them analyze PIT activities for the same purpose.
Both percentage of expired Interests and PIT usage may be periodically observed to detect IFAs. On one hand, routers know precisely whether every forwarded Interest is satisfied by a Data or expired after a timeout because of the strict NDN flow balance principle ("in absence of packet losses, one Interest packet results in one Data packet on each link" [119]). Therefore, an *unusual* number of expired Interests over forwarded ones along a certain time window certainly indicates the presence of anomalies on certain paths in the network. In fact, metrics like the Interest Satisfaction Ratio (ISR) in [105], which is defined as the ratio between expired Interests over satisfied ones, or the ratio of incoming Interests over outgoing Data in [120] are monitored to detect an IFA. Less precisely, counters on expired Interests can work as similar indicators [118, 121]. On the other hand, information about the PIT usage provides an indication of the traffic load managed by a router at a certain time. Therefore, abnormal utilization of the PIT space is also employed to detect network anomalies. For example, PIT size in bytes is monitored in [122, 116], while, PIT utilization rate over the entire PIT or per prefix name are monitored respectively in [123] and in [1, 114].
Overall, both ISR-oriented and PIT-oriented metrics are good indicators of anomalies in the network, yet assuming the related ratios are computed over meaningful time windows and interpreted conjunctly. Nevertheless, both detection metrics require to define threshold values to trigger alarms when exceeded. Ideally, thresholds are dynamically adjusted to reflect the net-

work status like in [116]. In reality, most of the times threshold values are based on empirical observations which are inevitably biased by a very few topologies and traffic distributions used in the evaluation settings [1].
Lastly, a very different approach is taken in [124, 125] where strong assumptions (i.e., Interest/-Data arrival rates on the router's interfaces and packet-loss rate) on the traffic distributions of Interest and Data packets are made to build a statistical framework based on the hypothesis testing theory. Although those results seem promising, since they would allow to have a detector whose precision would only depend on the number of samples and the desired probability of false alarms, those assumptions seem to make that approach unsuitable for any real deployment.

3.2.2 IFA mitigation

With regard to the mitigation mechanisms, different reactions can be performed locally at each router:

- rate limits of accepted Interests can be enforced on interfaces based on different parameters, e.g, expired Interests per FIB-record [118], at access routers [122], explicit alerts received on neighbor routers [116, 105], physical link capacity [105].
- detection and mitigation can be consolidated in a single phase, e.g., the satisfaction-based mechanisms of [105] use the ISR as the probability to forward or drop Interests.
- those prefix names, which are detected as target of an attack, can be processed differently to prevent them from generating state in the PIT, as done, for example, in [121].

Nevertheless, forwarding decisions taken independently at every router suffer from two issues as empirically proven by related work in [105, 116]. First, they cause *overreaction* [105], that is, Interests are unnecessarily checked multiple times by the same defense mechanism on all the routers along the path from a consumer to any producer (for example, in Fig. 3.1, the routers R3-R4-R5 along the path from the user to the wikipedia content provider may independently apply their own defense mechanism). Overreaction increases the probability of dropping malicious Interests over a certain network path, meanwhile it lowers the probability of forwarding legitimate Interests from one source to a destination [105]. Second, a decision local to a router does not necessarily *detect or mitigate the attack globally.* In fact, the effect of the attack is expected to be stronger in the proximity of producers of contents for a targeted prefix name (see the diameter of the circles around routers R4 and R5 in Fig. 3.1). While, routers which are more distant from those target producers (e.g., router R1 in Fig. 3.1) may carry malicious traffic as well but not be able to detect it properly.
For the above reasons, collaborative[8] defense mechanisms [122, 116, 123, 1, 105] result to be more efficient. In fact, in collaborative techniques, the dissemination of routers' local status allows defense mechanisms to be triggered even where the attack effect is not sufficiently strong to be detected (e.g., in Fig. 3 alert messages produced by router R4 may trigger mitigation at R1). Three different collaborative techniques are hereby briefly described, while a summary of their characteristics and differences is provided in Table 3.2. These countermeasures have been tested against some novel IFAs based on the attacker model presented in 3.1.3. Related results are

[8] The terms coordinated, distributed and collaborative can all be found in literature to refer to countermeasures where routers collaborate to detect and mitigate attacks. Therefore, those terms are used as synonyms throughout this manuscript.

Table 3.2 – Summary of the Collaborative IFA Countermeasures re-evaluated in this work.

Technique	Detection	Reaction	Prefix name	Content Identifier	Protocol changes
SBP [105]	none	Probabilistic Forwarding based on ISR values advertised by neighbor routers for their interfaces	trivial disjoint sets	random integers	appending information to the name of an Interest with a local scope; queuing and scheduling mechanisms at output interfaces
DP [116]	local ISR & PIT usage values & neighbors' alerts reception	rate limit on the infected interface & dissemination of alerts to neighbors	unreported	unreported	reserved name-space for the communication between neighbor routers; processing of unsolicited Data packets
CNMR [1]	locally at router: per-prefix ISR & PIT usage values. Globally at the controller: ISR per-prefix name	Probabilistic Forwarding based on ISR values locally at each router	non-disjoint sets	random integers	i) placement algorithm for the selection of monitoring nodes; ii) ad-hoc routing and forwarding algorithms to maximize traffic coverage by the monitoring nodes; iii) ad-hoc messages and namespaces between the controller and the monitoring nodes; iv) custom changes to the Interest packet format

presented and discussed in Sec. 3.4.

Satisfaction-Based Pushback (SBP) [105] implements a three-fold strategy to reduce the impact of an IFA. First, SBP uses the Interest Satisfaction Ratio (ISR) of an interface as probability to either forward or drop incoming Interests to penalize malicious traffic. Second, SBP implements separate queues on output interfaces to establish fairness among different input interfaces. Third, SBP adjusts forwarding rates on interfaces according to rate limits pushed back by upstream routers.

In ***Distributed Poseidon*** (DP) [116], routers monitor the PIT usage and ISR values per incoming interface and issue alarms to their downstream neighbors when observed values exceed their respective thresholds. DP aims to propagate downstream information about a possible attack as close as possible to its source. An alarm message announces a new rate limit on a certain interface and an affected namespace. Propagated alerts are used by traversed routers to lower their respective detection thresholds. The idea of pushing back notifications, which is also leveraged by the TraceBack countermeasure proposed in [122], is inspired by previous work on DDoS detection in IP routers [126, 127].

Coordination Monitoring (CNMR)[9] [1] envisions the deployment of specialized routers which are meant to closely monitor the forwarding of Interest packets within the same Autonomous System. Those monitoring routers may either detect an attack through the analysis of local statistics or rely on explicit indications provided by a centralized controller, which collects reports about suspicious activities from all the monitoring routers. At each monitoring router, detection is triggered by anomalous PIT usage and ISR values per-prefix name, while reaction is implemented as a probabilistic forwarding based on the related local ISR values.
Moreover, the monitoring routers avoid *overreaction* issues by flagging previously monitored Interests for no further inspection. Globally, a ratio of expired Interests is computed over all the monitoring nodes by the controller per every reported prefix name. If the overall ratio exceeds a certain threshold, then the controller alerts the monitoring routers that certain prefix names could be the target of an ongoing attack.

3.2.3 Evaluation

Evaluation metrics - The metrics commonly observed to validate both the impact of an attack and the efficacy of a countermeasure are the amount of resources consumed in routers and the average quality of service perceived by final consumers. The former usually relates to PIT space and computation resources like CPU cycles. The latter usually relates to both download time and number of contents retrieved. A successful attack increases sharply the load on routers and degrades clients experience. While, an effective countermeasure shields routers from being overloaded and guarantees a certain quality of service to clients.
An indicator of the quality of service perceived by final consumers can be obtained by looking at the percentage of satisfied Interests over the expressed ones. Among the existing literature works, this metric seems to be considered more than the average download time. As regards the routers, both consumed resources as PIT space, and throughput as number of content packets forwarded in the unit of time, are measured.

Content names - Legitimate consumers and attackers issue Interests with specific content names which, however, either are sporadically reported in or whose importance is underestimated by related literature. The name sets used for the evaluation of countermeasures in previous works are reported in Table 3.3. Those works are classified in three different categories. The largest portion of the previous works, which are listed in the row "unreported", provide no information about the content names. A further division in two subcategories is applied to the works which report about used content names, either in their publications or in the related publicly available implementations. On one hand, the subcategory "disjoint sets" lists the works using different prefix names for attackers and legitimate consumers. On the other hand, the subcategory "non-disjoint" lists the works assuming malicious Interests and legitimate ones may be expressed with the same prefix name.
As for the content name composition, in the vast majority of works, where reported, content names consist of only two name segments, one segment for the routable prefix name and one for the content identifier. The prefix name is usually a short string, whether mimicking some original domain name, like "/google.com/", or aiming to be self-explanatory, like "/good/" to indicate that Interests with that prefix are legitimate. As regards to the content identifier, the consumer applications provided by the widely used ndnSim simulator [128] use integers.

[9] There is no such an acronym in [1], which indeed was found in the related implementation.

Table 3.3 – Summary of prefix names used in literature.

Unreported	[124, 125, 116, 107, 122, 114]	
Reported	disjoint	[105, 118, 121]
	non-disjoint	[1]

Simulation Environment - Throughout the literature on IFAs, several network topologies have been used to evaluate the proposed countermeasures on a network simulator (for the sake of completeness, it must be said that there are also some techniques which have been evaluated on real network topologies [107]). Among the simulated topologies, there are simple ones and more realistic ones. On one hand, testing on a simple topology constitutes a canonical first step towards a more complete evaluation and can help convey general ideas about the characteristics of a proposed approach. Instead, larger and more complex topologies explore more realistic network scenarios. Countermeasures against IFAs have been often evaluated on a set of topologies inferred by the [129] work which can be imported in the ndnSim network simulator environment [128].

3.2.4 Pitfalls

Some of the design flaws in the state-of-the-art defense mechanisms, which are outlined in this section, inspired the design of more effective IFAs which are described in Sec. 3.3. Most importantly, the here-reported findings constitute a general handbook of good practices for the design and evaluation of future countermeasures.

Detection - As explained in Sec. 3.2.1, the presence of FIs has so far been identified by monitoring the ISR and/or PIT usage in routers, yet both detection metrics present some downsides:

- ISR-based metrics are influenced by all types of Interest (even at the prefix name granularity), e.g., legitimate Interests satisfied, legitimate Interests expired, fake Interests expired, legitimate, yet issued by the attacker, Interests satisfied. Therefore, this sort of metric can be intentionally polluted to circumvent the detection. For example, the first IFA variant presented in Sec. 3.3.3 employs a certain percentage of legitimate interests with the aim of keeping the IRS-based metrics observed by routers under the respective detection thresholds.

- abnormal PIT usage values may be also caused by network conditions where the load grows more easily, e.g., bursts of Interests or congestion; therefore, an overloaded PIT itself does not provide any indication about the kind of Interests which populate the PIT.

The above issues might be overcome by conjunctively analyze both metrics. For example, in [116] routers assume low ISR values may avoid false alarms when traffic peaks generate a high, yet legitimate, PIT usage. However, attackers can still influence the ISR-based metrics and stay undetected even under heavy PIT load conditions, as achieved, for example, by the first IFA variant of Sec. 3.3.3.
Finally, detection metrics are observed over a certain time window and, beyond that, should be ideally applied to a prefix name granularity to penalize less legitimate traffic. This implies respectively that i) any reaction happens with a certain delay, ii) specific prefix names have to be first identified as infected. Both conditions can be potentially exploited by attackers to remain undetected. For example, attackers in the second attack variant presented in Sec. 3.3.3 achieve

this goal by changing target prefix name over short time intervals.

Reaction - As outlined in Sec. 3.2.2, distributed defense mechanisms have so far proved to be more efficient in the detection and mitigation of IFAs, yet they open up other issues:

- they often require reserved prefix names and ad-hoc data-plane modifications (e.g., see the column "Protocol changes" of Tab. 3.2). For example, routers in [116] exchange unsolicited Data packets or routers in [122] generate spoofed Data to trace an attack back to its closest source. Those practical requirements limit any straightforward deployment of proposed solutions and impede their interoperability.

- they introduce overhead, e.g., for inter-router communication, in-router storage and analysis of detection metrics, which is not always considered and reported in the evaluation of the countermeasures.

- they may counter-productively disseminate wrong information when detection metrics are purposely polluted by the attackers.

Evaluation - The evaluation of works in the subcategory "disjoint sets" of Tab. 3.3 assumes attackers and legitimate consumers issue Interests with different prefix names. In those conditions, mitigation techniques applied at the prefix-level result extremely effective since they do not penalize any legitimate Interest. Indeed, as specified by the steadier attacker model described in Sec. 3.1.3, attackers have the ability to precisely produce Interests both for existing contents and non-existent ones. Therefore, attackers aim to masquerade their malicious Interests with the legitimate ones to remain undetected by routers. Therefore, those mitigation techniques dropping Interests with an infected prefix name inevitably affect some legitimate traffic too.
Moreover, using simplistic names as content identifiers in the simulation environment represents a risk of underestimation of several operations done for packet processing [85]. In fact, although the size of the PIT entries, which will be eventually determined by a specific PIT implementation, could be independent of the Interest name size, longer content names are definitely going to influence many other processing tasks performed in routers and by content providers. Overall, at the time being the precise implications of the content name composition on the evaluation of countermeasures against IFAs stay unexplored. Nevertheless, the IFA using Wikipedia page titles, which is presented in Sec. 3.2.3, has shown to be more efficient compared to the more simplistic models previously adopted by the related literature.

3.3 Flooding an imaginary NDN-based wikipedia

This section describes a novel IFA which targets large websites whose lists of contents are publicly available and are only subject to incremental changes. By accessing lists of available content names for target providers, prospective attackers can more effectively calibrate Interest types in attacks according to their goals. The attack i) is built upon the attacker model which has been detailed in Sec. 3.1.3, and ii) exploits drawbacks of existing defense mechanisms which have been illustrated in Sec. 3.2.4. The choice of a specific target website used for the evaluation of the attack impact is motivated in Sec. 3.3.1. The main attack is presented in Sec. 3.3.2, while two attack variants are illustrated in Sec. 3.3.3.

3.3.1 NDN content names for Wikipedia pages

As outlined in Sec. 3.2.4, related works on IFAs have so far neglected the importance of the name sets used for the evaluation of their proposed countermeasures. In contrast, more realistic sets of content names are important for two reasons. First, those allow research to better quantify the resources needed by routers and the impact of an attack whose main purpose is to abuse them. Second, they help simulate and test more realistic attack scenarios where attackers produce difficult-to-detect fake Interests (FIs) to dissimulate their attack.
In the attack model presented in Sec. 3.1.3, prospective attackers can refer to publicly available content lists to spoof content names so those look very similar to valid ones. Moreover, attackers can precisely issue Interests for existent and non-existent contents. This latter ability allows the attackers to dose the generated Interests for specific purposes, like in the attack variants presented in Sec. 3.3.3.
For the purpose of the evaluation of the novel IFAs presented in this work, a specific target website has been selected, i.e., the free online encyclopedia Wikipedia (however, the attack is widely applicable to any large website whose list of contents is available). A daily updated list of Wikipedia's page titles is provided by the Wikimedia Foundation [130]. All the content names used in the evaluation of this work share the same valid routable 1-component-long prefix name, which is "/wikipedia.org/". Then, legitimate Interests include one of the English page titles as 1-component-long content identifier, which is a variable-length alphanumeric string; while, malicious Interests have a 1-component-long fake content identifier generated as described in Sec. 3.3.2.

3.3.2 IFA on an NDN-based Wikipedia server

A novel Interest Flooding Attack targeting a specific NDN content provider has been proposed in [113]. The attack is named *pure IFA* (pIFA) and some more efficient attack variants (described in 3.3.3) can be also easily mounted by slightly varying the attackers behavior. The pIFA attack assumes NDN contents holding similar naming property of DNS names. Namely, content names from the same content provider follow a naming scheme which make them semantically related, as shown in [131]. A pIFA models the structure of content names for a targeted provider using the available related set of existent content names. The model is later used to generate new and non-existent content names following the provider naming scheme.
By choosing Wikipedia as target content provider, pIFA attackers can download the full list of existent contents since that is publicly available. Wikipedia page titles are composed as follows: *"*$word_1\_\, word_2\_ \cdots \_\, word_n$*"* with $n \in [1, x]$ where $word_i$ is meaningful. New names for Wikipedia pages can be created via DISCO [132], which is a generator of semantically related words. Given an input word w and a number m, DISCO returns a maximum of m most related words to w. Using an existing seed content name w, attackers generate the list l_1 of 200 most related words to w, e.g., the words "*computing, hardware, desktop, etc.*" are generated from the existing Wikipedia content name *"computer"*. The word generation process is launched a second time on each word obtained in l_1 with parameter $m = 200$ to produce a second word list l_2. Each obtained word in l_1 and l_2 can be used as content name for a malicious Interest by the attacking nodes in a pIFA. For sake of completeness, it must be said that this generation process produces less than $200 + 200 \times 200 = 40,200$ new words since seed names and duplicates are deleted from l_1 and l_2 and only unique words are considered.
The pIFA in [113] uses a fixed number of attacking nodes holding each a list of 5,000 seed Wikipedia content names. Each node generates off-line content names for malicious interests

using the above-described generation technique. Generated existent content names are discarded since the full list of existent contents is publicly available for the Wikipedia use-case. The off-line generation process produced 180,800 new content names per node on average, from which 50.3% were non-existent.

3.3.3 Attack variants

Two simple variants of the pIFA have also been proposed in this work. The *blended IFA* (bIFA) includes attackers generating Interest for both existent contents and non-existent ones. The percentage of Interest for existent contents is a fixed portion of the attack frequency per second, called *purity level*. The purity level is set at a certain value at the beginning of the attack by all the attacking nodes and kept invariant for all the duration of the attack. The main aim of a bIFA is to pollute the detection metrics observed by routers, i.e., the ISR-based ones and the PIT usage. By lowering the ratio of expired Interests over forwarded ones, bIFA attackers try to remain undetected and/or lower the probability for their malicious Interests to be dropped. The *chameleonic IFA* (cIFA) includes attackers which change the target valid prefix name after a certain time window. The decision about the change of a target prefix name may be probabilistic or driven by the observation of the content retrieval latency. The goal of a cIFA is to avoid prefix name based mitigation which is generally applied by countermeasures after an observation window where certain name prefixes are marked as infected.

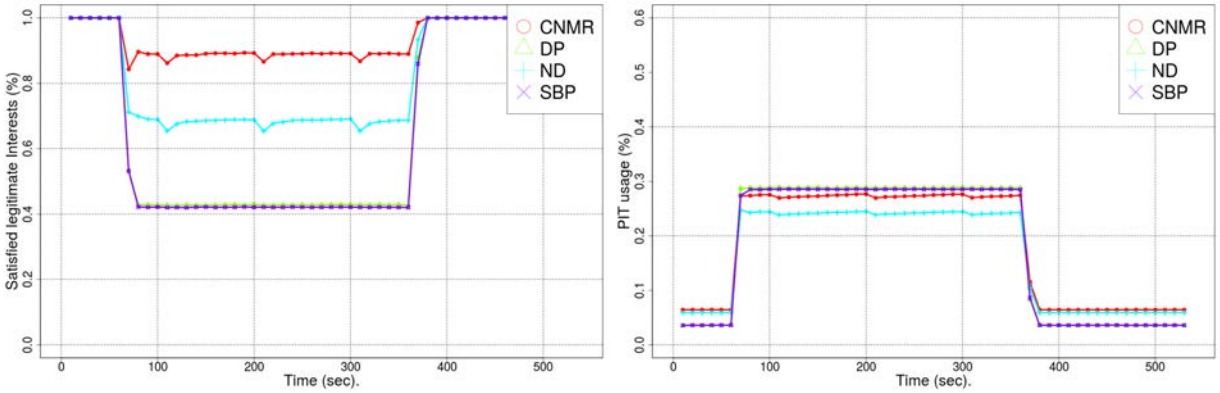
3.4 Evaluation

This section reports results obtained from the evaluation of three collaborative state-of-the-art countermeasures against the IFAs introduced in Sec. 3.3.2 and Sec. 3.3.3. The evaluation results indicate that existing defense mechanisms show poor robustness against a steady IFA attacker model. These findings suggest to reconsider results reported by previous related works where proposed countermeasures were shown to efficiently protect NDN networks from this kind of attack.
The simulation settings reported in Sec. 6B of the work in [1], which are briefly summarized in Sec. 3.4.1, have been replicated for the sake of comparison to related work. In particular, the experiments in [1] have been taken as reference since the CNMR countermeasure, therein proposed, happens to be the latest defense mechanism overperforming previous state-of-the-art countermeasures. The countermeasures have been evaluated against the pIFA, the bIFA and the cIFA of Sec. 3.3.2. All the experiments have been conducted on version 1.0 of the open-source ndnSIM [128] module to fairly compare to the state-of-the-art countermeasures which were implemented and evaluated on that version of the simulator. The ndnSIM extensions implementing the novel attacks and all the tools developed for the sake of this work have been made available at [133] in order to reproduce the reported experiments.

3.4.1 Set-up and Metrics

Set-up

The network topology simulated is the same as in [1], that is, the Exodus ISP (AS 3967) inferred by the Rocketfuel project [129]. In every simulation scenario, 25% of the total number of client nodes are selected as attackers. Normal clients generate 100 Interest packets per second (ipps), while attackers Interest generation frequency varies from 500 ipps to 10000 ipps. Hereby mainly



(a) Average ISR of legitimate clients

(b) Average PUR of all the routers

Figure 3.2 – Interest Satisfaction Ratio (ISR) in 3.2a and PIT Utilization Ratio (PUR) in 3.2b for the experiments presented in [1] and reproduced by this work.

results for the 1000 ipps attack frequency have been reported since this frequency is representative enough for the conducted experiments (further, it helps the reader visualize more clearly results in each plot). Each simulation lasts nine minutes with the attack starting at 60 seconds and ending at 360 seconds. All the scenarios have been simulated 10 times and average values are reported each time. With regard to each technique's specific parameters, settings reported in the related papers and/or implementations have been exactly replicated where possible. The curves labeled with 'ND' correspond to a baseline scenario with no defense mechanism. The curves labeled SBP, CNMR, DP correspond to scenarios with selected countermeasures, [105], [1] and [116] respectively on.

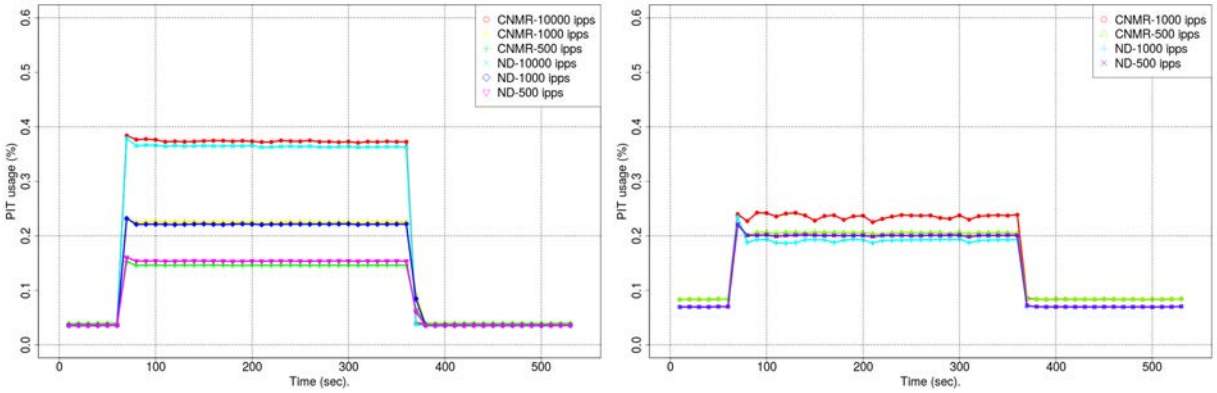
Metrics

The impact of the novel IFAs of Sec. 3.3.2 and 3.3.3 is evaluated on two metrics which have been widely used in the related works (the rationale behind the evaluation metrics for IFAs was already introduced in Sec. 3.2.3). First, the Interest Satisfaction Ratio (ISR) at clients, which is computed as the ratio between satisfied Interests and issued ones over a time window, gives an indication of the quality of service perceived by end-users while the network is under attack. Second, the PIT utilization ratio (PUR) reports about the available capacity on routers to process legitimate traffic during an attack. The PUR is computed as the ratio between the number of PIT entries and the maximum PIT size in entries over a time window.

State-of-the-art (SoA) results

As first evaluation step, the experiments presented in [1] have been replicated, where only the countermeasures CNMR and SBP were evaluated[10]. Surprisingly, although the general trend for both the evaluation metrics is almost consistent with what reported therein, different results have been obtained.

[10]DP is not considered in [1], most likely because the implementation has never been made available since that work was published. For the sake of this work, DP has been implemented as forwarding strategy in ndnSim according to the description in the original work [116] and made available at [133].



(a) PUR values for all routers

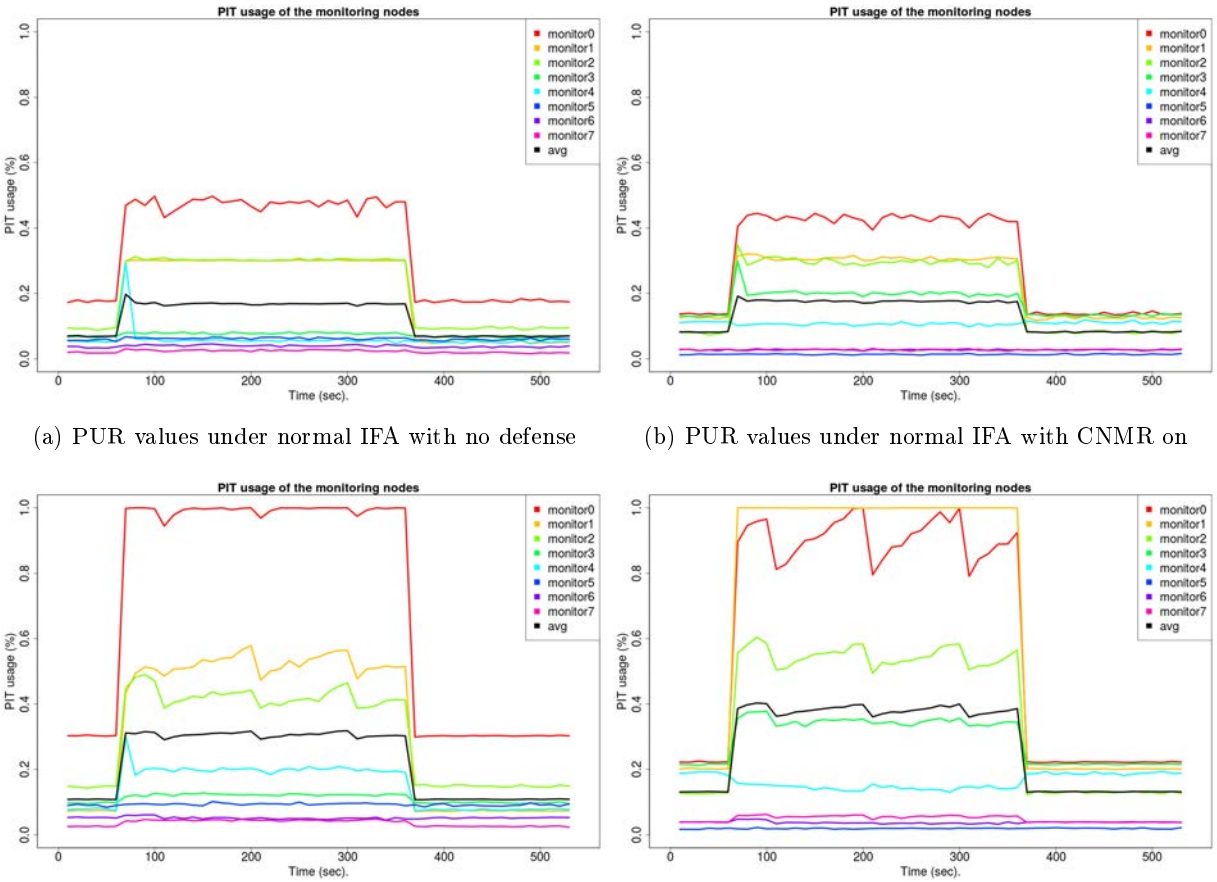
(b) PUR values for monitoring routers

Figure 3.3 – PIT Utilization Ratio (PUR) average values for all the routers in Fig. 3.3a with different attack frequencies and for monitoring routers in Fig. 3.3b in the SoA experiments.

The ISR values for the ND and the CNMR scenarios, plotted in Fig. 3.2a, are 10 to 20% higher than the ones in [1]. Fig. 3.2a also shows that SBP and DP worsen the ISR compared to the baseline. This last behavior was only reported in [1] for higher attack frequencies. Indeed, this result was expected since SBP and DP apply rate limits of accepted interests on interfaces detected as infected independently from the affected prefix name. Meaning that, when multiple prefix names are used in the evaluation, as done for the experiments in [1], SBP and DP may accidentally drop many Legitimate Interests. CNMR, differently, mitigates only the prefix names which are detected as infected. Therefore, CNMR overperforms SBP and DP in this evaluation scenario because the former never drops legitimate Interests with other prefix names except the one under attack[11].

With regards to the PUR plotted in Fig. 3.2b, none of the countermeasures alleviates the average PIT consumption in routers contrary to what reported in [1]. This would be expected for SBP and DP, which would allow more malicious Interests to occupy PIT space across the network since they do not mitigate the attack efficiently as already shown in Fig. 3.2a (low efficiency of SBP in large network topologies with many attackers was already reported in [105]). Nevertheless, as additional outcome of an aggressive dropping policy which highly affects the ISR perceived by clients, SBP/DP also reduces the number of Interests gaining PIT space in the network. As can been seen in Fig. 3.2b, SBP/DP curves only slightly increase the PUR compared to the baseline. As regards CNMR, the average global PIT usage is slightly higher (less than 5% more) than the one measured in the ND scenario. It must be said that results in [1] reported a lower average PUR compared to the baseline under attack when the countermeasure is on. The thereby-reported behavior was not seen in any simulated scenario by this work nor the authors of [1] were able to provide the correct information to replicate those results when contacted about this. A consistent behavior of CNMR with regard to the PUR has been observed by this work with all the tested attack frequencies (e.g., the plot in Fig. 3.3a shows PUR values for 500, 1k and 10k ipps with

[11] Almost all the curves reported in this section present a periodic artifact occurring at around 100, 200 and 300 seconds of the simulation time. It is extremely difficult to precisely pinpoint the root cause of this effect because of the many variables involved in the simulation scenario. Nevertheless, we have measured a strong correlation between the total number of duplicate Interests produced and this periodic variation of the ISR and PUR curves. Although the content identifiers for the Interests are generated by using random functions in the network simulator fed with different seeds, there is an overall decrease first and increase after of the total number of duplicates around those time windows.



(a) PUR values under normal IFA with no defense

(b) PUR values under normal IFA with CNMR on

(c) PUR values under pIFA with no defense

(d) PUR values under pIFA with CNMR on

Figure 3.4 – PIT Utilization Ratio (PUR) values for all the CNMR's monitoring routers under an the attack mounted in [1] and the pIFA introduced by our work. Results for the 1000 ipps attack frequency are plotted. The simulation scenario includes 8 monitoring routers whose PURs are reported on 8 different colored curves. The curve in black plots the average values.

and w/o defense mechanism on). The same pattern holds if only monitoring routers' PITs are considered, as shown in Fig. 3.3b. Indeed, it must be said that an increase of the average PIT utilization ratio through the activation of the CNMR defense mechanism is not unexpected. Yet, the higher PIT usage is due to a notable path stretch increment introduced by CNMR's ad-hoc forwarding strategy. In fact, CNMR steers every Interest first towards the closest monitoring router in order to maximize traffic coverage (see description of the MR-Aware Routing in Sec. V-B of [1]).

The plots in Fig. 3.3, as well as all the other PUR plots of this section, report the PUR average values. This was meant to ease the comparison to the results reported by the CNMR work in [1]. Nevertheless, the average PUR values computed over either all routers of the topology or all the CNMR's monitoring routers may not visualize clearly an attack's impact. In fact, the PIT usage of the reported experiments oscillates between 0.2 and 0.6. That is, the attacks seem to never saturate any router's PIT, yet the attacks are able to affect the ISR measured by the clients. Indeed, the tested attacks saturate several routers' PITs and this can be seen more easily by observing the PIT load of the monitoring routers under attack in the following plots.

The plots in Fig. 3.4 show the impact of an IFA with 1000 ipps on the PIT load of the monitoring

nodes in a single experiment (not average reported this time, since every round of simulation corresponds to different positions in the network for both clients and attackers). The 1000 ipps attack frequency is already sufficient to saturate the PIT of some of the monitoring routers as better shown by Fig. 3.4c and 3.4d (see red and yellow curves) where the network is under a *pIFA*. Further, the monitoring routers are often some hops away from the clients. So, those clients' Interests may need to traverse first other routers before reaching any monitoring router. Those routers' PITs could be already heavily affected by the attack. Similarly, this is observed also with the state-of-the-art experiments in [1], as reported in Fig. 3.4a and 3.4b, where, however, the measured impact is lower.

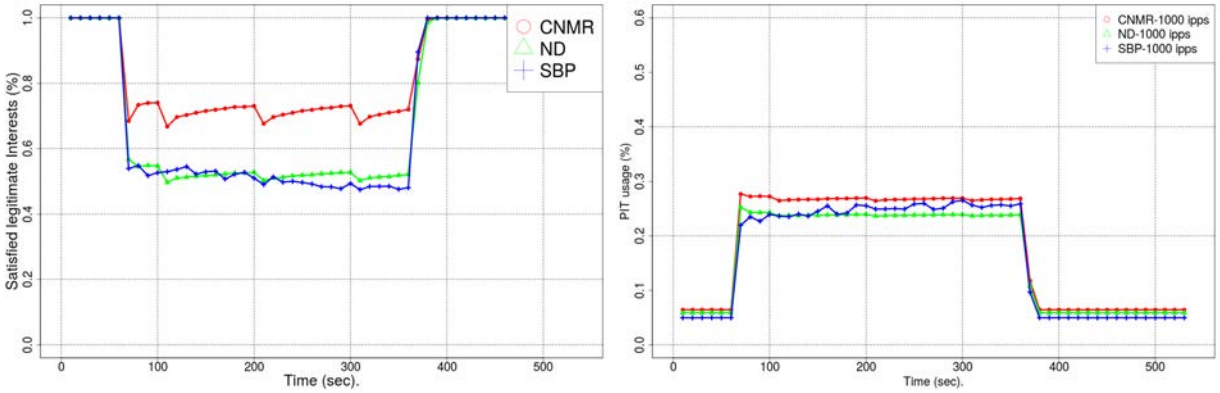
3.4.2 Attack results

Pure IFA - pIFA

In the evaluated pIFAs, attackers generate Interests only for non-existent contents by fetching invalid content identifiers from off-line generated lists of spoofed names. Meanwhile, legitimate users only produce Interests for existent contents referring to the original Wikipedia page titles list. Furthermore, all the content consumers, legitimate and malicious ones, use the same single routable prefix name. Both above conditions did not hold in the SoA experiments of [1].
The measured ISR values are reported in Fig. 3.5a (for the sake of clarity, from now on the DP curves are omitted since the relative values are very similar to the SBP ones). As in the SoA experiments, SBP poorly mitigates the attack, while CNMR improves this metric compared to the baseline. It also important to see how the SBP countermeasure (blue curve) worsens the attack effect compared to the baseline (green curve). The low ISR average value caused by SBP is due to its aggressive dropping policy applied to the affected interfaces. Comparing the SoA results in Fig. 3.2a to the ones in Fig. 3.5a, the pIFA introduces a 17% decrease of the ISR values for both the baseline scenario and the CNMR one. This result confirms that the novel attacker model leads to more effective attacks compared to the ones used in the evaluation of the related works.
Fig. 3.5b also shows a 10% increment of the PUR in CNMR's monitoring routers compared to the SoA results for both the baseline and the CNMR scenario, consistently with the observed ISR decrease. It is worth noticing that SBP achieves lower PUR values compared to CNMR because of the higher amount of Interests dropped to mitigate the attack.

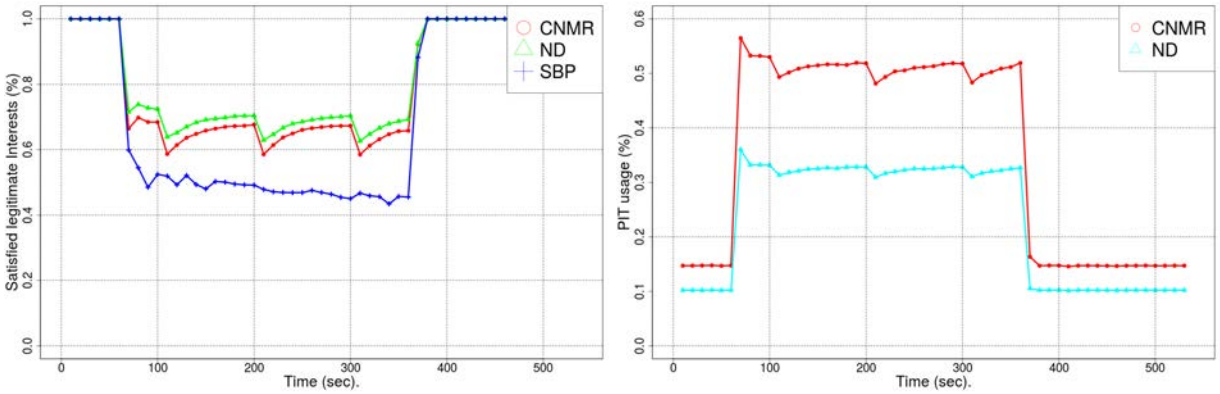
Blended IFA - bIFA

The ISR values of the bIFA scenario are reported in Fig. 3.6a. In bIFAs, attackers also generate Interests for existing contents (a fixed percentage of the attack frequency per second, called *purity level*) with the aim to reduce the number of expired Interests monitored by routers. Fig. 3.6a shows a 22% ISR decrease of the SoA results and a 5% decrease of the pIFA ones for CNMR. Moreover, pIFA attackers' behavior totally neutralizes this countermeasure. In fact, the CNMR curve always lays above the baseline in Fig. 3.6a. The efficiency of the SBP technique is reduced too in case of bIFA, since the detection metric monitored by the routers is polluted and allows more fake interests to create entries in the PITs. In Fig. 3.6b the PUR values for CNMR are plot, the attack succeeds at occupying more space across monitoring routers' PITs, as a consequence of the poor detection achieved by the applied countermeasure.
It is also important to see how the efficacy of a bIFA increases with the increment of the number of legitimate Interests used by the attackers. Fig. 3.7a shows a decrease of the ISR for CNMR



(a) ISR values under pIFA (b) PUR values under pIFA

Figure 3.5 – Interest Satisfaction Ratio (ISR) and PIT Utilization Ratio (PUR) under the pure IFA (pIFA). The PUR is computed over all the routers in the network.



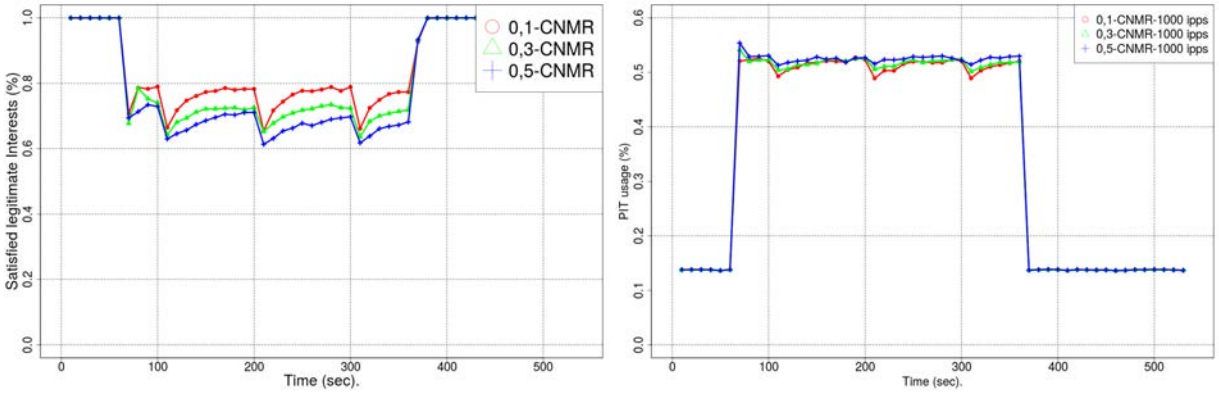
(a) ISR values under bIFA (b) PUR of CNMR under bIFA

Figure 3.6 – Interest Satisfaction Ratio (ISR) and PIT Utilization Ratio (PUR) under the blended IFA (bIFA).

when the bIFA's purity level increases, while Fig. 3.7b shows a small increment of the PUR values when the attack's purity level is increased.
The efficacy of a bIFA is a consequence of the forwarding decisions taken independently at each monitoring router. In fact, although the successful detection made by the CNMR controller node, the monitoring routers' local statistics (see definition in Sec. 3.2.1) are used as probability to either forward or drop every Interest with the detected prefix name. Therefore, the legitimate Interests issued by pIFA attackers improve routers' local statistics and, by consequence, the probability for Fake Interests to occupy PIT space and be forwarded to the next hops.

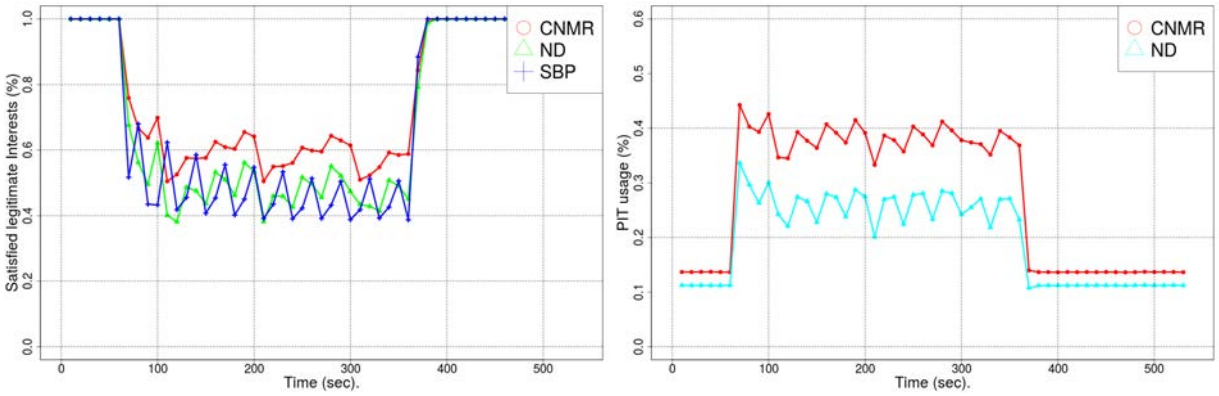
Chameleonic IFA - cIFA

In cIFAs, the attackers either know or guess (e.g., by emitting and measuring the RTTs of some legitimate Interests) the observation time window used by detection techniques in routers. The attackers' goal is to avoid prefix name-based mitigation techniques (e.g., the CNMR of [1]). For that purpose, attackers switch the target prefix name at every observation window.
The impact of cIFAs has been evaluated in scenarios where content producers serve several prefix



(a) ISR of CNMR (b) PUR of CNMR under bIFA

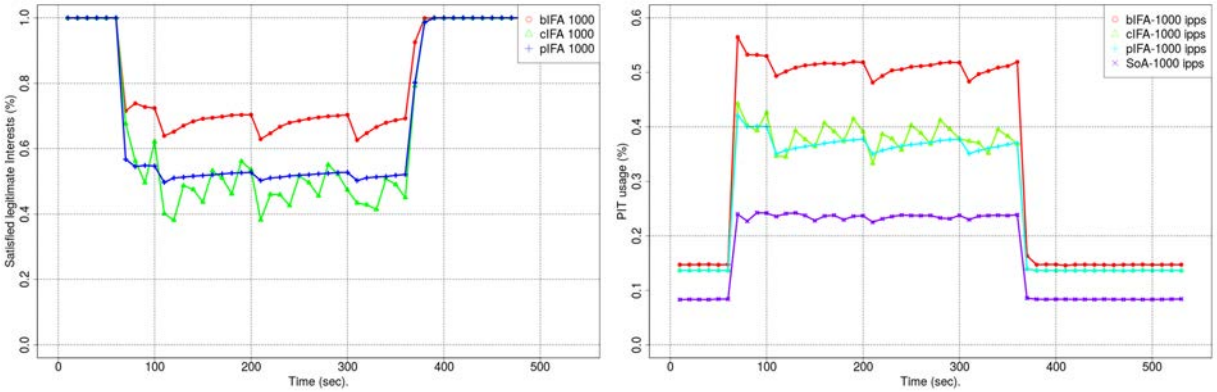
Figure 3.7 – Interest Satisfaction Ratio (ISR) and PIT Utilization Ratio (PUR) for different bIFA's purity levels.



(a) ISR values under cIFA (b) PUR of CNMR under cIFA

Figure 3.8 – Interest Satisfaction Ratio (ISR) and PIT Utilization Ratio (PUR) under the chameleonic IFA (cIFA).

names and CNMR monitoring routers used a fixed observation time window of 10 seconds as in [1]. Hence, attackers switch target prefix name at every such time window (this justifies the periodic oscillations seen in Fig. 3.8a and 3.8b during the attack). Meanwhile, legitimate clients generate traffic with all the prefix names served by the producer nodes. As expected, this attack reduces the ISR of CNMR compared to the SoA results as shown in Fig. 3.8a. Among the three novel attacks, the cIFA is the attack variant which degrades more the client's performance in the evaluated scenarios. The ISR baseline values measured for the three different attacks when no countermeasure is on are summarized in Fig. 3.9a. As similarly observed for the previous scenarios, the SBP still poorly detects the attack.
As regard to the PUR of CNMR's monitoring routers, shown in Fig. 3.8b, the values are consistent with the ISR ones, showing an increase of the former metric when the latter one is reduced. Nevertheless, cIFA generates a PIT increment of the baseline which is smaller than the one caused by the bIFA reported in Fig. 3.6b. The impact on the average PIT utilization of monitoring routers under the different attack scenarios is summarized in Fig. 3.9b.



(a) ISR values for baseline scenarios

(b) PUR of CNMR under different attacks

Figure 3.9 – In Fig. 3.9a, the Interest Satisfaction Ratio (ISR) is measured in the baseline scenario under the three novel IFAs. In Fig. 3.9b, PIT Utilization Ratio (PUR) for monitoring routers under all the different attacks compared to the SoA results.

Summary

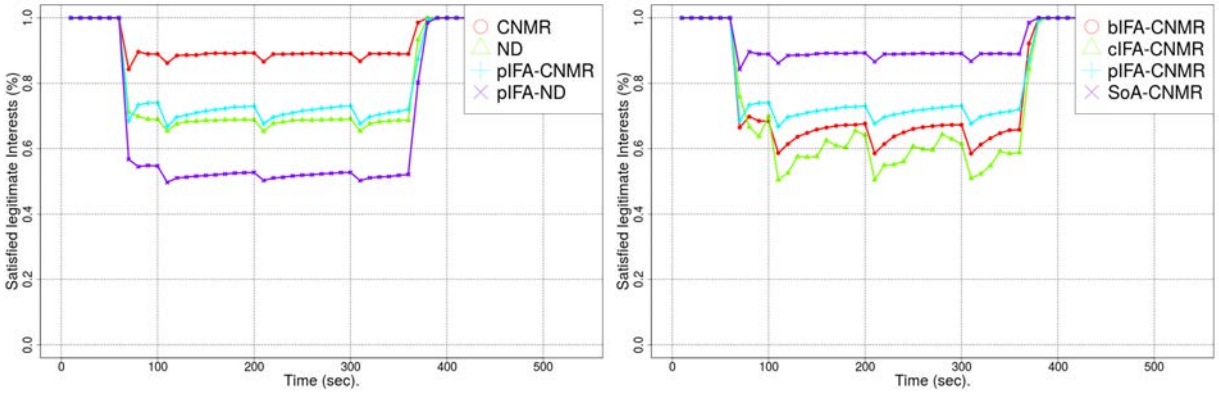
The IFAs, built upon the novel attack model proposed in this work, degrade the performance of the best state-of-the-art countermeasures. In particular, the attack model reduces the quality of service perceived by clients (i.e., lower ISR values mean less client requests satisfied) and increases the load on network's nodes (i.e., higher PUR values correspond to more computational and memory resources used in routers). Fig. 3.10b shows how the model can be leveraged to mount increasingly strong attacks which notably affect the most effective state-of-the-art countermeasure. The observed trend holds for all the tested attack frequencies and the three introduced attack variants. The main findings are recapitulated in the following points:

- by using non-disjoint prefix name sets all the tested defense mechanisms drop some legitimate traffic too. In fact, no state-of-the-art countermeasure can distinguish between Fake and Legitimate Interests, rather, when either a prefix name or an interface is detected as infected, all the related traffic is mitigated.

- It is possible for attackers to pollute metrics collected by defense mechanisms in routers and increase the efficacy of their attacks. This was successfully achieved by the bIFA variant in our experiments.

- It is possible for attackers to circumvent detection operated by defense mechanisms in routers and stay undetected. This was successfully achieved by both bIFA and cIFA variants in our experiments.

- Under all the tested attacks, CNMR generates a higher load on routers compared to SBP. This means that SBP better protects the network infrastructure (see the last two columns of Tab. 3.4) while CNMR better protects the clients (see the first two columns of Tab. 3.4). Yet, SBP performs aggressive mitigation on infected interfaces which inevitably affects many legitimate Interests; thus, overall, clients experience an ISR degradation. This behavior would be beneficial to legitimate traffic coming from non-infected interfaces, which would find the necessary PIT space along the path to contents.



(a) ISR in SoA and pIFA scenarios

(b) ISR in the different attack scenarios

Figure 3.10 – ISR values for the baseline and with CNMR on for both the state-of-the-art experiments and for a pIFA in Fig. 3.10a. ISR values for the novel IFAs compared to the state-of-the-art results in Fig. 3.10b.

Table 3.4 recapitulates the impact achieved by the different attacks on both the evaluation metrics considered. Overall, the cIFA causes the worst ISR at clients as reported in the first two columns of Tab. 3.4. However, the cIFA is not the attack variant which generates the highest load on routers for all the countermeasures. In fact, the bIFA variant causes higher PUR when CNMR is on, as reported in the third column of Tab 3.4. Therefore, the ultimate choice between bIFA and cIFA could be driven by knowledge available to the attackers (e.g., ability to guess observation windows used by routers' defense mechanisms) and desired targets (either clients or network infrastructure or both).

Table 3.4 – Summary of the results achieved by our attacks compared to the values in the SoA results. Negative values in the ISR-related columns indicate a decrease of the average quality of service perceived by the clients. Positive values in the PIT-related columns indicate an increase of the average load on routers.

	CNMR-ISR	SBP/DP-ISR	CNMR-PUR	SBP/DP-PUR
pIFA	-17%	-36%	+10%	+2%
bIFA	-22%	-38%	+18%	+1%
cIFA	-28%	-40%	+11%	+3%

3.5 Summary

This chapter has introduced the Interest Flooding Attack (IFA), a Denail-of-Service attack which capitalizes on the per-Interest PIT state created by regular Interests in the NDN. The IFA characteristics and goals have been illustrated before proposing a novel and steadier attack model. The state-of-the-art IFA countermeasures have been dissected. Flaws in the current mainstream defense approach and in the evaluation settings have been outlined. The proposed attacker model and the analysis of the state-of-the-art countermeasures have been used to build new stealthy IFA attacks. These attacks have revealed the inefficacy of the state-of-the-art countermeasures

to protect the clients and the network infrastructure under more realistic attack scenarios.
The findings reported in this chapter show the limitation of the existing IFA defense mechanisms against a novel set of steadier IFAs. Therefore, these results call for different and more robust countermeasures to be designed in order to protect the NDN architecture from this attack. For that reason, a novel approach to the defense against IFAs is presented in the next chapter.

Chapter 4

Proactive defense against Interest Flooding Attacks

Interest Flooding Attacks (IFAs) represent a major open security threat to the Named-Data Networking architecture. In IFAs, loosely coordinated botnets of attackers may exhaust routers' resources and deny service to legitimate users. Countermeasures may be applied to mitigate this attack, yet those suffer from several drawbacks making their potential deployment either non-efficient or unfeasible. First, most of the proposed countermeasures introduce ad-hoc protocol changes and/or make unrealistic assumptions about the control of the network infrastructure so impeding any seamless integration in existing deployments. Second, the best state-of-the-art countermeasures introduce monitoring overhead and perform a delayed reaction because of their intrinsic reactive nature. The reactive nature of a countermeasure can be exploited by the stealthy attacks proposed in Chapter 3 to circumvent defense mechanisms active in routers and achieve the attacker's goal. Finally, all the state-of-the art countermeasures are not designed to discern between legitimate and malicious Interests, so they inevitably penalize legitimate traffic too. This latter issue is usually underestimated by the related literature which only considers the average performance at clients achieved with the application of a certain countermeasure. Though an improvement of the average quality of service perceived by clients could not be sufficient for a certain class of applications which do not tolerate too many closely spaced packet losses, e.g., the stream of multimedia content.

The design of existing countermeasures against IFAs is based on the assumption that an Interest packet itself does not contain the necessary information to distinguish between legitimate and malicious Interests. Therefore, other factors have to be analyzed to determine the nature of an Interest. For example, the NDN flow balance principle ("one Interest packet results in one Data packet on each link" [119]) is used to detect anomalous percentages of unsatisfied Interests [105]. Or yet, assumptions about the traffic distribution are used by statistical models to define the conditions for an ongoing attack [125]. Instead, this chapter shows that under some circumstances an Interest name is sufficient to determine its either malicious or benign nature. This assumption, though restricted to a certain case, eliminates the vulnerability exposed by reactive countermeasures since no monitoring of previous traffic is required to make forwarding decisions. Further, this work also shows that a specific proactive design is able to preserve legitimate traffic from being erroneously dropped. In fact, this specific design leverages bloom filter-based content summaries which generate no false negative, that is, no legitimate Interests can be mistaken for a malicious one and so dropped. Both factors are showed by extensive simulations against the latest steady attacks proposed in [113] which neutralize previously proposed IFA countermeasures.

4.1 Issues with Reactive Countermeasures

Chapter 3 has already discussed the state of the art on countermeasures against the IFA. The main aim of that first survey was to highlight pitfalls in existing defense mechanisms which could be leveraged to build stealthy attacks. The survey presented this time in Sec. 4.1.2 outlines an intrinsic characteristic, the *reactiveness*, of all the most effective state-of-the-art countermeasures since that is indeed the root cause of the pitfalls identified in those defense mechanisms. Before, Sec. 4.1.1 describes the main Interest type used by attackers in IFAs, which has oriented the design of the related countermeasures.

4.1.1 A core strength of IFAs: Fake Interests

An effective IFA DoS consists of flooding the network with a large amount of Interests with the aim to overload routers and content sources with unnecessary processing. Attackers' means and goals may vary as clearly summarized in [113] and reported in the previous chapter. A botnet of loosely coordinated clients generating requests hard to be aggregated and served by caches can be used to maximize the attack's efficacy. For that purpose, attackers mainly generate Fake Interests (FIs). A FI is a unsatisfiable Interest since it does not corresponds to any existent content. FIs names can be easily generated by appending random identifiers to valid routable prefix names as shown in Fig. 4.1. A valid globally routable prefix name allows FIs to traverse the network towards content sources and create state in routers' PITs. The spoofed content identifiers are easily discarded by the content providers which however do not notify network about those invalid requests (at the time being the definitive role and adoption of negative acknowledgments in CCN/NDN-like networks stays unclear [134]). Nevertheless, FIs create long-lived state in PITs since the related entries last until the PIT entry expiration time.
FIs' strength resides in being easy to generate and hard to detect. In fact, endless different invalid content identifiers can be trailed to valid routable prefix name. A valid prefix name is the only information looked up by routers to decide about an Interest forwarding. While no check is done on the content identifier.

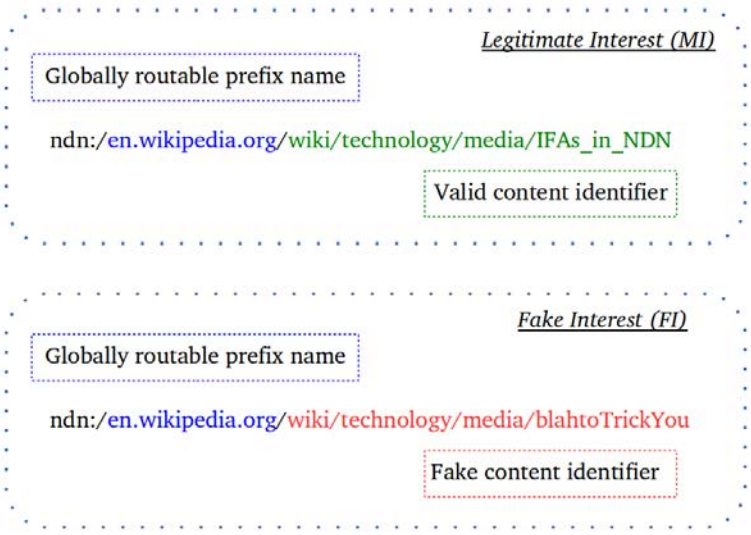


Figure 4.1 – Example of a Legitimate Interest in the topmost box and of a Fake Interest in the bottommost box.

4.1.2 A novel Taxonomy for IFA countermeasures

The classification of approaches to defend against IFAs presented in Sec. 3.2 of Chapter 3 mainly focuses on a certain class of countermeasures which, in reality, represent a subset of all the existing defense mechanisms. A more comprehensive classification of the countermeasures against the IFA DoS has been proposed by [52]. The hereby presented taxonomy leverages some terminology from the work in [52], but proposes a different classification of the existing approaches. More specifically, two main sub-root categories are introduced in the taxonomy of Fig. 4.2 to classify existing IFA countermeasures: reactive and proactive. The adjective reactive is used to identify countermeasures which respond to attacks only after those have already taken place. Potential symptoms of an IFA are an excessive load on PITs, an anomalous number of expired Interests, an unexpected increase of the number of incoming Interests. Therefore, monitoring of any or all of these factors is required by reactive countermeasures to detect potential ongoing attacks. While, the adjective proactive is used to identify countermeasures which anticipate an attack. The nature of every Interest, either legitimate or malicious, has to be determined before forwarding it to anticipate an attack. So, forwarding decisions have to be taken only by looking at the Interest packet fields.
As it can be seen in Fig. 4.2, among the related works the above-definition of proactiviness cannot be applied to any existing countermeasure. In fact, the majority of the existing countermeasures rely on the observation of some previous traffic and so those are considered reactive according to the above-defined classification criteria. There exist a few other approaches which are not considered reactive since they do not perform any monitoring: i) alternative forwarding mechanisms, ii) client's proof-of-work ones. The "client's proof-of-work" category identifies

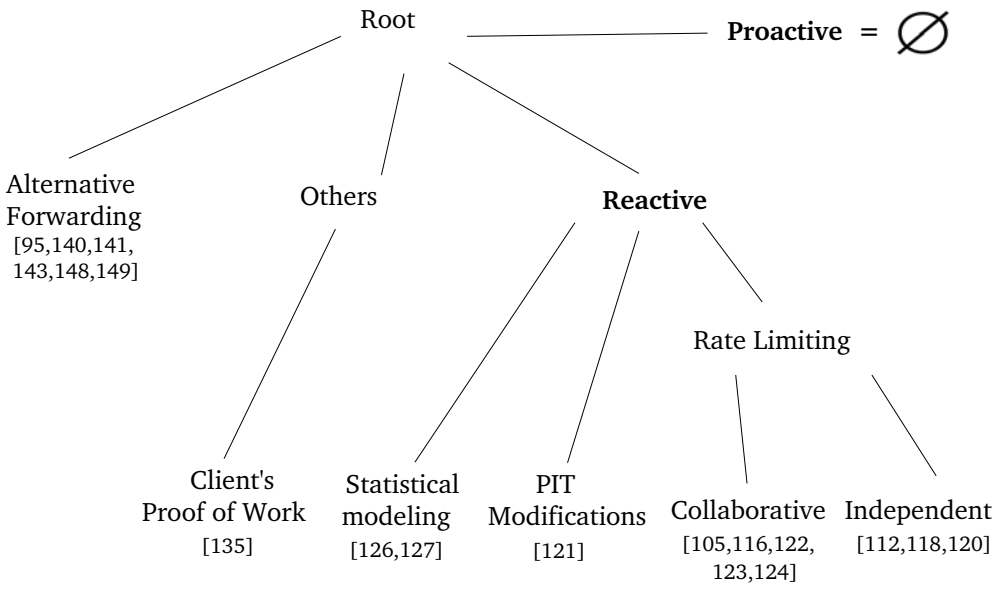


Figure 4.2 – Taxonomy of reactive and proactive IFA countermeasures.

works proposing application-based solutions to defend against IFAs. That is, clients are required to insert subsidiary information in their Interests, so as their nature can be assessed by other network entities, e.g., routers and/or content providers. For example, the InterestCash schema of [135] requires consumers to solve puzzles to generate Interests for dynamic contents checked at the reception by content providers. Overall, InterestCash reduces the amount of effective malicious traffic that single attackers can generate in the unit time. However, the nature of an Interest is determined at the content producer side. Hence, this schema does not prevent malicious Interests to occupy PIT space across the network and so does not protect the network infrastructure. A proactive variant of this mechanism could be designed assuming the information inserted by consumers in their Interests would be validated in routers. Though, no such work has been proposed yet. The category "alternative forwarding" refers to work addressing the IFA security threat by proposing alternative PIT designs. Some propose stateless designs just in order to avoid the vulnerability of the PIT stateful forwarding plane [108]. These alternative forwarding mechanisms outline insightful and promising research directions for the design of less vulnerable NDN/CCN networks. Thus, the related works are covered in depth in the Chapter 5.
The majority of the IFA countermeasures proposed so far is reactive. Reactive countermeasures feature a canonical two-phase, detection and mitigation, defense mechanism. In the detection phase, either interfaces or prefix names are identified as infected. The methodology used to perform the detection can be either an analysis of traffic metrics observed over certain time windows [105, 116, 1] or a statistical guess [125, 124]. Then, in the mitigation phase, a strategy is adopted to reduce the attack's impact. A further characterization can be applied based on the mitigation technique in two main subcategories: PIT-modifications approaches and rate-limiting ones.
There are very few works proposing PIT-modifications to the data-plane operations to block IFAs from saturating the PIT. For example, the Disable PIT Exhaustion (DPE) approach in [121] detects an attack by monitoring some PIT-based metrics, but it applies a different packet processing to the traffic detected as suspicious. The DPE approach proposes not to record Interests in PIT when their prefix name is detected under attack. Instead, the reverse path is appended hop-by-hop to suspicious Interest packets when those are forwarded upstream for eventual Data to be forwarded downstream.
Applying a rate limit to input/output routers interfaces is a widely adopted mitigation mechanism by IFA countermeasures. Those countermeasures are classified under the "rate limiting" category and are subject to a further distinction between collaborative and independent. Independent countermeasures are applied at each router by leveraging only local detection information. In collaborative countermeasures, routers communicate to exchange information about their respective states. Communication between routers is of foremost importance to both avoid overreaction on the same Interests and rate-limit an attack far from the final targets, where it may be harder to detect. The form of collaboration among routers vary from IP-traceback inspired techniques [116, 122, 105], which push-back downstream notifications asking routers closer to the attack sources to scale down their rates, to ad-hoc deployed specialized nodes [1], which through a global network view can localize and mitigate more precisely ongoing attacks.
Until very recently, the works under the reactive/rate-limiting/collaborative category had been showed to be the more effective defense mechanisms against IFAs. However, the canonical reactive approaches have been shown by recent literature to be susceptible to stealthy attacks [113] and their efficacy to be highly-variable with network topologies [136]. In particular, the latest attacker model proposed in [113] capitalizes on the reactive nature of those existing countermeasures. In fact, two intrinsic characteristics of these state-of-the-art reactive countermeasures are to react after an observation time window and to rely on collected metrics. Both factors can be easily exploited by potential attackers to stay undetected, in particular when the attackers have

the necessary knowledge to dose with precision the kind of generated Interests.
The pitfalls of reactive countermeasures are not present in proactive countermeasures as the latter are conjectured by this work. Thereby, it is worth to investigate the conception of a novel class of countermeasures exhibiting intrinsically different characteristics since those have the potential to improve the state of the art. The cryptographic route token approach of [115] introduces the notion of proactive defense and postulates novel evaluation metrics, similar to the ones defined in this work, for the first time in literature. However, in [115] no precise definition of proactiveness is given. Therein, the adjective proactive is rather used to differentiate an in-packet cryptographic mechanism proposed in opposition to the reactive nature of existing countermeasures. Conversely, this work precisely conjectures this novel class of defense mechanisms, delineates general characteristics and proposes concrete implementation options for the first time.

4.2 Proactive Countermeasures

A proactive IFA countermeasure is here defined as a defense mechanism aiming to make routers able to detect an IFA without monitoring any prior traffic. Thereby, proactiveness eliminates the need for routers to collect and analyze traffic statistics to infer detectors of possible attacks. Attack detectors are affected by network conditions and attacker strategies, so their absence reduces the risk of unnecessary reaction to regular traffic peaks and of manipulation of routers' decisions by misleading attacker's behaviors. Moreover, proactiveness does not introduce any delay in the reaction to ongoing attacks at the price of some additional delay in the Interest processing.
Proactive countermeasures rely on no additional information except the one contained in the Interest packet to determine the either benign or malicious nature of an Interest. Mainly this revolves around the analysis of the Interest name. How this latter task is performed and the kind of delay introduced in the Interest processing further characterize the proactive technique. First, a list of generic requirements for the design of IFA countermeasures is presented in Sec. 4.2.1. The framework is used in Sec. 4.2.2 to showcase the advantages of an ideal proactive countermeasure over the existing reactive ones. Finally, a possible declination of the ideal proactive model, named Charon, is concretely explored is Sec. 4.2.3, 4.2.4 and 4.2.5.

4.2.1 Requirements for IFA countermeasures

To clarify pros and cons of the different kinds of defense mechanisms, this work sets out a list of ideal goals for IFA countermeasures. This list aims to be a common framework, missing at the time being, for a fair comparison of existing and future defense mechanisms. It would be desirable for IFA countermeasures to exhibit the following characteristics:

- G1 - **low overhead**: refers to packet processing operations and resources, computation and storage.
- G2 - **high protocol compatibility**: refers to avoiding possible required ad-hoc protocol changes, e.g., the introduction of new packet fields or a change in the packet processing workflow.
- G3 - **low infrastructure deployment**: refers to the deployment of ad-hoc infrastructure, e.g., nodes with additional monitoring capability.

Table 4.1 – Ideal proactive defense versus existing reactive countermeasures against IFA compared with respect to the *Gx* requirements. The rating scale includes five values ranging from very low to very high. The score *none* is only applied when referring to the infrastructure deployment.

Technique	Overhead	Protocol Compatibility	Infrastructure Deployment	Efficacy	Impact on Legitimate Traffic
Reactive - SBP	high	mid	none	mid	very high
Reactive - DP	high	low	none	mid	very high
Reactive - CNMR	very high	very low	very high	high	high
Proactive	very low	very high	none	very high	very low

- G4 - **high efficacy**: refers to the performance which is usually measured in terms of the average quality of service given to clients and the amount of resources, network bandwidth and/or routers' resources, consumed by the attack in the network.

- G5 - **low impact on legitimate traffic**: refers to the percentage of legitimate traffic affected by the application of the countermeasures itself.

Certainly, priority among the above goals is not absolute rather driven by design requirements. For example, the design of a countermeasure to be deployed at Internet scale would require to introduce a few or almost no overhead because of the high link-speeds, nor infrastructure deployment and the highest protocol compatibility because of the possibly involved different administrative network entities. In that case, the requirements G1, G2 and G3 are targeted, while the requirements G4 and G5 may be relaxed. Conversely, a countermeasure operational in an ICN-island at the network edge aims to block malicious traffic mainly to guarantee a good experience to the clients (goals G4 and G5) accessing the mainstream network via that edge-network segment. At the network edge, low overhead is not of foremost importance since an edge-router handles slower network links, infrastructure deployment neither since the interested infrastructure could be all part of the same network operator. Therefore, the requirements G4 and G5 may be prioritized over the G1, G2, G3 ones.
In Table 4.1 three of the most effective reactive state-of-the-art IFA countermeasures have been classified according to the above-defined target *Gx* requirements: the Satisfaction-Based Push-back (SBP) of [105], the Distributed Poseidon of [116], the CoordiNation MonitoRing (CNMR) of [1]. The values in the columns *Overhead*, *Protocol Compatibility*, *Infrastructure Deployment* of the reactive techniques have been inferred by the technique descriptions in [105, 116, 1]. The *Efficacy* column values are based both on the evaluation reported in previous related work [105, 116, 1] and on the latest results reported in chapter 3. The *Impact on Legitimate Traffic* column values have been measured by this work as described in Sec. 4.3. Table 4.1 illustrates how:

- all the existing reactive techniques introduce considerable monitoring overhead. Monitoring overhead is also highly proportional to the efficacy, since a more fine-grained observation guarantees a more precise detection.

- the techniques showing higher protocol compatibility and requiring no infrastructure deployment (e.g., SBP [105] and DP [116]) exhibit mid efficacy and highly impact legitimate traffic.

- reactive countermeasures can be improved with regard to efficacy and impact on legitimate traffic at the price of largely increasing the overhead, the protocol compatibility and the infrastructure deployment. This is exactly what is achieved by the CNMR [1] technique. CNMR employs additional ad-hoc monitoring nodes and a centralized controller (infrastracture deployment), it performs a very fine-grained monitoring per each prefix name on each interface (overhead), it introduces Interest labeling to mark monitored Interests as well as a protocol to communicate reports to the centralized controller unit (protocol compatibility).

- the impact of the mitigation applied by a countermeasure on legitimate traffic has never been estimated by the related work. However, all the three countermeasures are very susceptible to the issue of dropping legitimate traffic if that either is issued with the prefix names identified as detected or comes from the same infected interface or both.

Overall, a reactive countermeasure seems to have several downsides. First, it introduces overhead for monitoring, protocol changes and infrastructure deployment proportionally to the desired efficacy. Second, despite the invested resources, it affects considerably legitimate traffic too since it cannot discern Legitimate and Fake Interests issued under the same prefix name. Third, this kind of countermeasure can still be circumvented with moderate attack capabilities.

4.2.2 Ideal Proactive Countermeasure

The last row of Table 4.1 shows how an ideal Proactive Countermeasure (PC) sets with respect to the *Gx* target requirements. First, an ideal proactive countermeasure does not require collecting and analyzing any traffic metric. Therefore, it is expected to introduce very low overhead. Per-Interest analysis must be done to distinguish between Fake and Legitimate Interests. However, this task can be accomplished by looking at the Interest packet fields and at a few more resources built off-line. Second, an ideal proactive defense mechanism exhibits full protocol compatibility. Forwarding decisions are made locally at each router and can be implemented in the form of an NDN forwarding strategy module. Third, such a countermeasure requires no infrastructure deployment since i) forwarding decisions are taken independently by each router, ii) no communication between routers is required. With regard to the requirements *G4* and *G5*, those highly depend on the attacker model and on the Interest packet analysis technique applied by the proactive countermeasure. In general, assuming an attacker model where attackers are able to precisely generate both forged content identifiers and valid ones, a proactive countermeasure could leverage the attacker's same knowledge (e.g., publicly available content lists for a certain provider as shown in Chapter 3) to effectively discern those (i.e., factors which lead to very high efficacy and no impact on legitimate traffic). Conversely, proactiveness is ineffective when it is not possible to establish a priori the validity of content identifier (e.g., providers dynamically generating content identifiers for multimedia streaming content). In this last case, neither the attacker nor the router could precisely ascertain a priori the validity of a content identifier (i.e. factors which lead to low efficacy and high impact on legitimate traffic).

4.2.3 From Principles to Practice

The ideal behavior described in Sec. 4.2.2 faces additional constraints in realistic settings. First, pure proactive countermeasures cannot detect valid Interests yet generated by attackers. So, they

are totally inefficient against attacks where collusive consumers and producers collaborate to delay the delivery of valid Data packets [114]. Second, proactive countermeasures assume routers are able to distinguish spoofed content identifiers from valid ones. This can be achieved either by knowing all the valid content identifiers or by learning them through prior traffic observation. The former method applies for static contents whose lists are made available by the respective producers. The latter method applies for both dynamically generated contents or contents whose lists are not available. In the former case, several possibly very large lists of content identifiers must be stored by routers and checked at run-time. In the latter, learning models must be built and maintained on-line. The former case is of notable interest because of the latest disruptive IFA attacks [113] using large providers' publicly available content lists. The latter case requires traffic to be monitored first in order to build a model, hence, it exhibits the same issues seen with existing reactive countermeasures.
The *Gx* requirements framework has been introduced to guide the design of future countermeasures. Therefore, the design of a proactive countermeasure starts from sorting the target *Gx* requirements. With that aim, the design is assumed to address at first the main issues raised by the latest attacks with respect to the state-of-the-art reactive countermeasures [113]. That is, to avoid using a countermeasure which performs unefficiently and penalizes legitimate consumers despite a large monitoring overhead and the deployment of an ad-hoc infrastucture to fine tune the monitoring operation. Hence, low overhead (G1), high efficacy (G2) and no impact on legitimate traffic (G3) are the main target requirements. While, the requirements of high protocol compatibility (G2) and low infrastructure deployment have lower priority.

4.2.4 Content summaries

Traditionally, Legitimate Interests and Fake ones have been considered indistinguishable at routers. In fact, those only differ in content identifiers and routers are not supposed to have any notion about valid and spoofed ones. Indeed, the assumption that routers do not have any knowledge about valid content identifiers is not always true and so it may be relaxed in some scenarios. For example, in the steady attack model proposed in [113], attackers have access to public lists of valid identifiers for target prefix names, so routers do too.
Assuming lists of valid content identifiers are available, those could be stored and checked by routers at the Interest reception. Such a scenario imposes three main requirements for the design of a proactive countermeasure which should not:

- alter the correctness of the Interest forwarding operation, that is, the countermeasures should affect neither the forwarding of legitimate Interests upstream nor the forwarding of the respective Data downstream.
- introduce large memory requirements to store the lists of contents.
- introduce intolerable additional latency in the Interest processing since the validity check of Interest names is expected to be done at line-rate per every Interest.

Bloom filters (BFs) [91] are a perfect candidate to create space-efficient summaries of content identifiers under a certain prefix name. A BF is a data-structure for membership query which can represent a set of n elements with an array of m bits where $m << n$ (the mathematics

behind the bloom filters has been summarized in Sec. 2.2.2 of Chapter 2 where the BloomFilter-based PIT designs have been illustrated). With regard to the three above requirements, first, BFs guarantee a constant membership test time, so they introduce negligible latency in the processing of Interests. Second, BFs provide a very compact representation for a set of elements, so the related memory requirement is kept low. Third, although BFs introduce false positive (whose error probability can be finely tuned), this does not alter the correctness of the forwarding operation. In fact, a false positive leads to the router marking as valid a content identifier which indeed is a spoofed one. The outcome of this error is the forwarding of a Fake Interest. Knowing this probability can be tuned, the impact on the network load will be negligible.
This work proposes Charon[12], a proactive countermeasure against IFAs where NDN routers use BF-based content summaries at run-time to check the validity of the content identifier in every Interest name. Charon, except for a few false positives, is able to drop Fake Interests as close as possible those enter the network.

4.2.5 Addressing Scalability

The basic design of Charon assumes routers always hold the content summary for a certain prefix under attack. However, this may be unlikely to scale in real settings for several reasons. First, routers would need to store content summaries for a large number of content providers, e.g., all the prefix names available in the router's FIB. Still, this would be reasonable, assuming those were mostly stored on secondary memory and moved to main memory at run-time accordingly. Second, those summaries may need to be updated by content providers or new summaries may be needed due to routing updates. For the above reasons, it seems reasonable to foresee a way for routers to occasionally fetch and/or regularly update content summaries. The basic design of Charon enforces no specific solution for that purpose rather highlights a few possibilities. Content summaries could be made available to routers via either a trusted centralized authority (as it happens for today's public key certificates) or could be directly asked to the respective content producers. In the latter case, encryption should be used to avoid prospective attackers to fetch the content summaries and generate set of forged content identifiers which generate false positives. Furthermore, once the summaries are issued by the content providers, they would be cached in the network as ordinary Data packets. Therefore, the latency necessary to retrieve the summaries is reduced and the risk of relying on a single centralized point of failure too. In both cases, a secure communication channel between the router and the source of the summaries should exist to prevent attackers from getting the information required to obtain a clear version of a summary. The architecture envisioned by Charon is illustrated in Fig. 4.3.

With regard to the scalability of the countermeasure, Charon offers a considerable advantage over existing reactive countermeasures. In fact, while reactive countermeasures must be deployed in several network areas and orchestrated so for the attacks to be properly detected, the check performed by a proactive countermeasure like Charon is independent from the position in the network where that is deployed. In fact, Charon can be selectively deployed in a few specific points of network without compromising its overall efficacy. In particular, edge-routers are perfect candidates since they block the fake Interests as soon as those enter the mainstream network. Additional deployment of the countermeasure in inner network tiers would not bring any tangible benefit assuming the content summaries are mostly identical, that is, having the same false error probability. The deployment at the edge also reliefs core network routers to perform additional per-packet processing which could not be sustained at certain link speeds.

[12]In Greek mythology, Charon is the Hades' ferryman who carries souls across the rivers dividing the world of the living from the world of the dead.

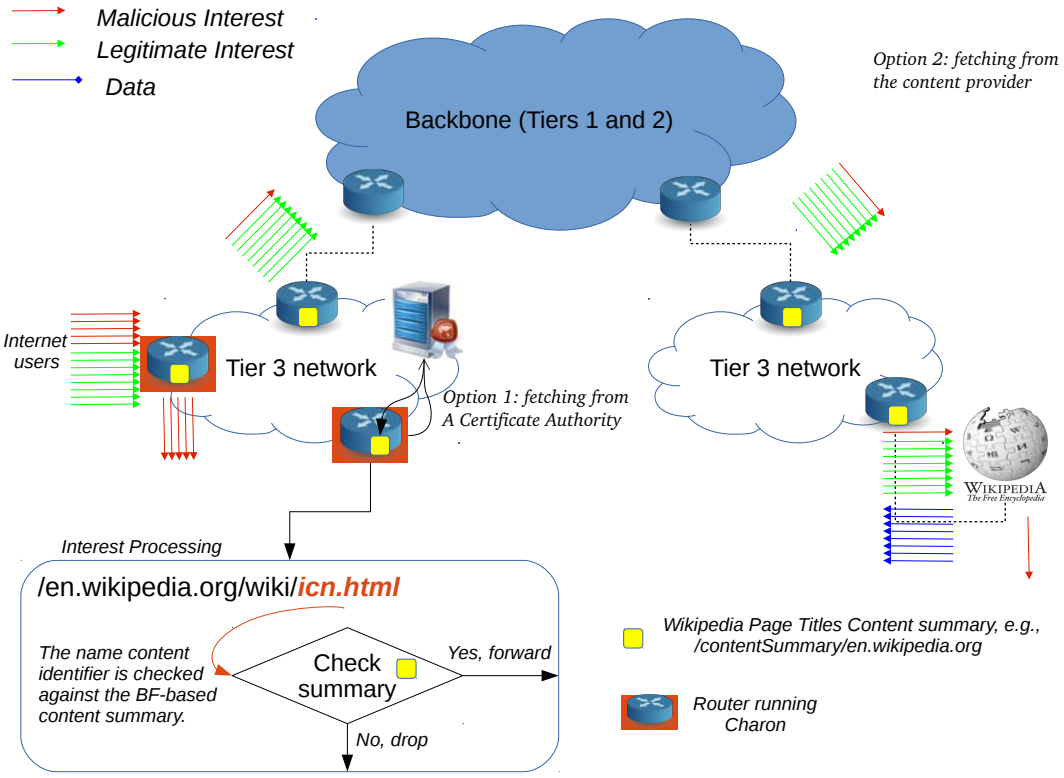


Figure 4.3 – An overview of the architectural components for the deployment of the proactive countermeasure Charon.

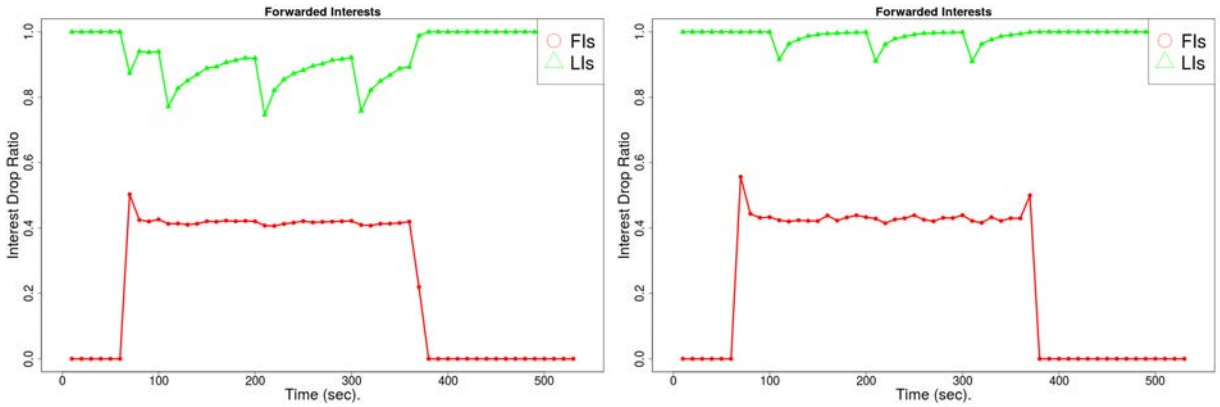
Finally, Charon envisions the possibility for routers to ask for different summaries of the same content list. The reason for this feature is to allow routers to decide about the size of the filter to store and the computational overhead to sustain. The false probability error in BFs depends on three factors: the number of elements in the set n, the size of the filter m, the number of hash functions k. Thus, for example, a smaller filter could be required by a provider whether a higher false probability error would be considered acceptable.

4.3 Evaluation

The evaluation reported in this section concerns two main factors. First, it shows that the proactive Charon design is more robust than the best state-of-the-art countermeasures against all the latest stealthy attacks proposed in [113]. In fact, Charon entirely preserves the quality of service perceived by clients when the network is under those attacks. Moreover, Charon shields the network infrastructure from the overload of state in PITs created by the attackers flooding with malicious Interests. Second, it outlines the importance of evaluating the impact of the application of the IFA countermeasures on legitimate traffic. Indeed, by precisely tracking the forwarding decisions taken on both malicious and legitimate interests in the network simulations, it is possible to see two different concerning behaviors: i) a considerable amount of legitimate traffic is always dropped as a consequence of the mitigation and the effect of this behavior on some kind of traffic, e.g., video streaming, stays unclear, ii) the portion of legitimate traffic dropped does not necessarily depend on the router load rather on whether or not Malicious Interests and Legitimate ones enter the router from the same downstream interface.
The rest of the section is organized as follows. The evaluation settings and metrics are summarized in Sec. 4.3.1. A novel evaluation metric which measures the impact of a defense mechanism on legitimate traffic is introduced in Sec. 4.3.2. Finally, the performance of Charon compared to the state-of-the art countermeasures is discussed in Sec. 4.3.3.

4.3.1 Evaluation Environment and Metrics

For the sake of a fair comparison to the related work, the efficacy of the basic Charon design has been assessed by replicating the experiments in [113]. Therein, three state-of-the-art countermeasures are evaluated with respect to different IFA attacks on the ndnSIM [128] simulation environment. Charon has been implemented as an ndnSIM's forwarding strategy module and deployed on all the routers of the simulation scenario. The experiments in [113] are conducted on a single AS-like topology (the Exodus ISP topology of the Rocketfuel project [129]) containing less than one hundred nodes. The nodes running Charon hold a bloom filter-based (BF-based) content summary of the valid content identifiers for a certain prefix name. The BF-based content summary is built off-line before starting the simulation.
In each simulation, attacks are carried out by a fixed percentage of randomly selected nodes in the network topology. All the attackers generate different kinds of Interest with a fixed attack frequency of 1000 Interest packets per second (ipps), while all the clients issue 100 legitimate Interest packets per second. Attacks start at 60 seconds and end at 360 seconds of the total simulation time. The performance of each defense mechanism has been evaluated by measuring two different metrics widely used by the related work: the Interest Satisfaction Ratio (ISR) at clients and the PIT Utilization Ratio (PUR) at routers (the rationale of both metrics has been described in Sec. 3.4.1 of Chapter 3). Each simulation scenario has been repeated 10 times, where different seeds generate different attackers and clients network positions (while Interest generation frequencies stay fixed for both kind of entities), and average values are reported. Charon has been tested against all the three IFA variants proposed in [113].
Charon's source code and all the necessary software to replicate the hereby reported experiments have been made available as open source at [133].



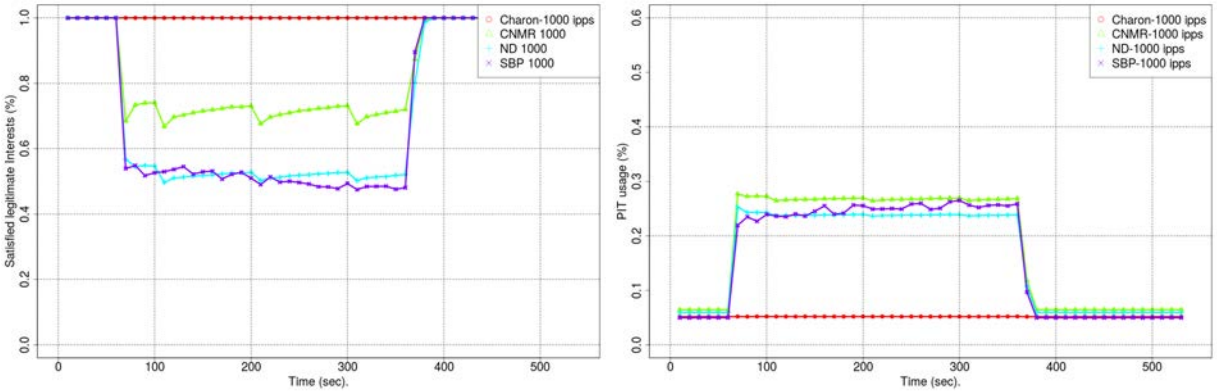
(a) IDR at the monitoring node 1

(b) IDR at the monitoring node 3

Figure 4.4 – Interest Drop Ratio (IDR) for two different monitoring nodes, monitor1 in 4.4a and monitor3 in 4.4a, of the CNMR technique when the network is under a pIFA with attack frequency equals to 1000 ipps. The IDR is shown for both Legitimate Interests (LIs) and Fake Interests (FIs).

4.3.2 Interest Drop Ratio

In IFAs, attackers generally strive for generating Malicious Interests with popular prefix names. By that, attackers have two main objectives: i) to masquerade an attack in-between legitimate traffic, ii) to cause mitigation techniques to also penalize legitimate traffic too, so to further degrade the service provided to other normal users. This assumption about the attacker model has been proved to be effective for building stealthy IFAs in [113]. Those attacks have a higher impact on both clients and the network infrastructure. However, at the time being there exists no evaluation metric in the literature on IFAs to quantify precisely the impact on legitimate traffic neither of this kind of attack nor of the countermeasures applied in defense to it. Indeed, a precise understanding of this effect could be beneficial to improve existing defense mechanisms. This work introduces the Interest Drop Ratio (IDR) as evaluation metric to quantify the impact of the application of a certain IFA countermeasure on the legitimate traffic in the network. The IDR is defined as the ratio of forwarded Interests over the received ones across a certain time window because of the countermeasure's forwarding decisions. The motivation behind the introduction of a new evaluation metric for IFAs is mainly to show that proactive countermeasures bring the additional advantage of not affecting any legitimate traffic over the existing reactive countermeasures. This hypothesis was confirmed by measuring the IDR in scenarios where Charon was applied as defense mechanisms to counteract ongoing attacks. In fact, Charon never drops any Legitimate Interest by design. On the contrary, both SBP [105] and CNMR [1] drop some legitimate traffic as effect of their mitigation techniques.
Besides, even at an early stage of the adoption of this metric, the evaluation of IDR in other simulation scenarios has already revealed unknown behaviors of other defense mechanisms. For example, Fig. 4.4 plots the IDR measured at different CNMR's monitoring nodes for both Legitimate and Fake Interests. The plots in Fig. 4.4a and in Fig. 4.4b show that the portion of legitimate traffic affected by a monitoring node in CNMR is not proportional to the load of received Fake Interests. In fact, while the monitoring node in Fig. 4.4a and the monitoring node in Fig. 4.4b hold the same load of Fake Interests (the IDR represented by the red curve), the amount of legitimate traffic they affect (the IDR represented by the green curve) is different. This result suggests that indeed monitoring nodes exhibiting a lower IDR for Legitimate Interests receive part of those Interests on interfaces which are not subject to any mitigation.



(a) Average ISR of legitimate clients under pIFA

(b) Average PUR of all the routers under pIFA

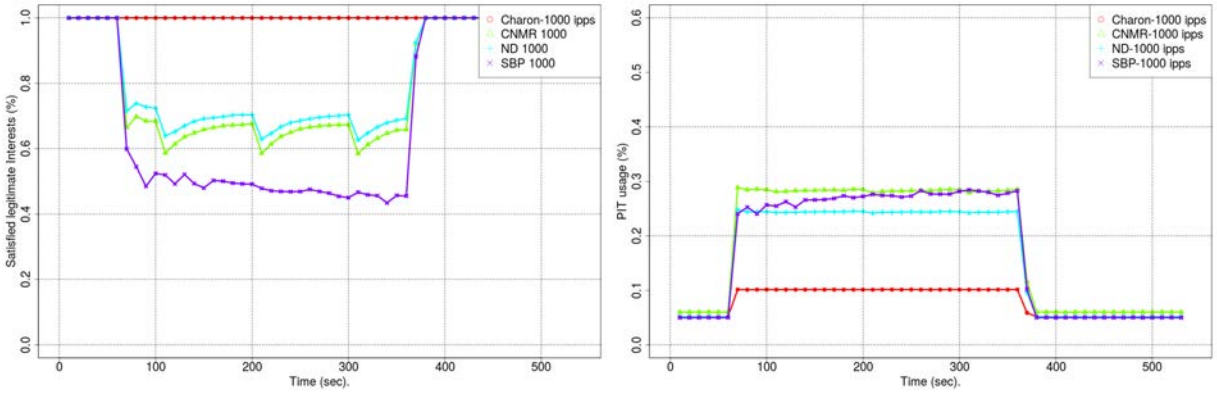
Figure 4.5 – Interest Satisfaction Ratio (ISR) in 4.5a and PIT Utilization Ratio (PUR) in 4.5b under a *pIFA* with attack frequency of 1000 ipps.

4.3.3 Charon performance

The proactive countermeasure Charon has been compared to the most efficient state-of-the-art reactive countermeasures in scenarios where the latter ones fail to counteract efficaciously stealthy IFAs. The evaluated reactive countermeasures are the Satisfaction-Based Pushback (SBP) [105] and the CoordiNation MonitoRing (CNMR) [1]. Both SBP and CNMR are collaborative countermeasures which monitor traffic metrics over observation time windows to detect attacks. On one hand, SBP applies a probabilistic forwarding to each Interest where the forwarding decision is based on locally measured ISR values and rate-limits announced by upstream neighbors. On the other hand, CNMR relies on an ad-hoc set of monitoring routers coordinated by a centralized controller to identify prefix names potentially under attack. Then, each CNMR router mitigates the attack according to the locally measured ISR. Overall, both defense mechanisms suffer from some weaknesses which can be exploited by attackers to remain undetected. More precisely, metrics collected at routers can be tainted by dosing different Interest types and by periodically varying target prefix names, as done respectively by the blended IFA and the chamaleonic IFA variants proposed in [113].

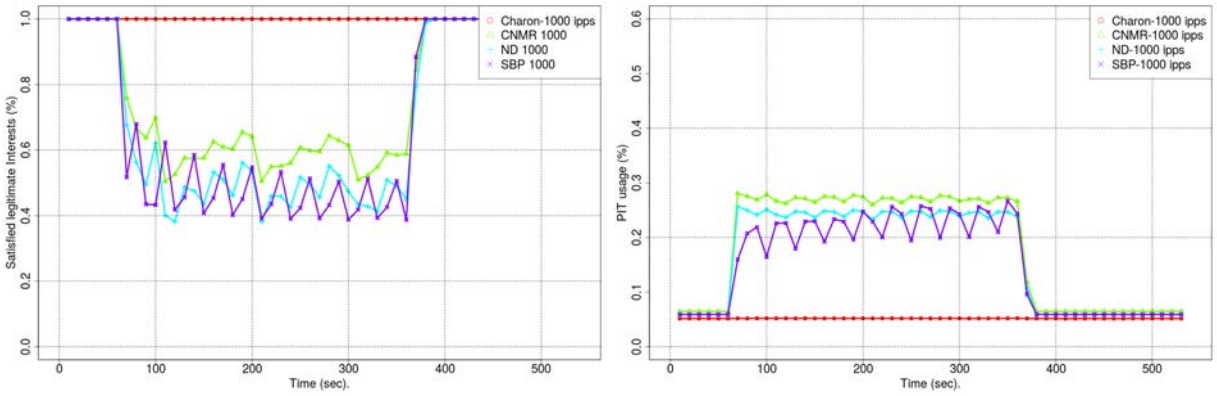
Fig. 4.5a and Fig. 4.5b illustrate respectively the average ISR and PUR values in a scenario where the network is under a pure Interest Flooding Attack (*pIFA*) (see Sec. 3.3.2 of chapter 3 for a description of the attack). In a *pIFA*, attackers use off-line generated list of spoofed content identifiers to precisely issue only Fake Interests, that is, Interests for non-existent contents. Fig. 4.5a shows that SBP performs closely to the baseline ND (which is the scenario where no defense mechanism is on). In the same plot, CNMR improves compared to the baseline yet still suffers from around a 30% loss of the original ISR measured at clients. In the same scenario, Charon totally preserves legitimate traffic from any degradation of service since it easily detects Fake Interests by querying the BF-based content summary held by every router. With regard to the load on the network generated by a *pIFA*, Fig. 4.5b shows Charon blocks the attackers to generate any additional load on routers since their Fake Interests are almost (except a few false positives whose effect is negligible) never forwarded farther than the first network hop.

Fig. 4.6a and Fig. 4.6b illustrate respectively the average ISR and PUR values in the same evaluation scenario where the network is under a blended Interest Flooding Attack (*bIFA*). The



(a) Average ISR of legitimate clients under bIFA (b) Average PUR of all the routers under bIFA

Figure 4.6 – Interest Satisfaction Ratio (ISR) in 4.6a and PIT Utilization Ratio (PUR) in 4.6b under a *bIFA* with attack frequency of 1000 ipps.



(a) Average ISR of legitimate clients under cIFA (b) Average PUR of all the routers under cIFA

Figure 4.7 – Interest Satisfaction Ratio (ISR) in 4.5a and PIT Utilization Ratio (PUR) in 4.5b under a *cIFA* with attack frequency of 1000 ipps.

main aim of a *bIFA* is to pollute detection metrics computed by routers and stay undetected. *bIFA*'s attackers achieve this goal by issuing a certain percentage of valid Interests too. Valid Interests allow attackers to maintain observed metrics below the respective detection thresholds set by the routers. In this way, a *bIFA* remains undetected. As shown in Fig. 4.6a, a *bIFA* results very efficient against reactive countermeasures since it is never mitigated (the CNMR and SBP curves lay below the baseline ND curve). Charon (the red curve represents the average ISR when Charon is active), instead, is immune to this kind of attacker's strategy since it does not rely on any detection metric. The only evidence of an ongoing attack is the little raise of the average PUR value illustrated in Fig. 4.6b. In fact, although the attack fails to achieve its purpose, the valid Interests emitted by the attacking nodes are regularly forwarded by all the routers in the network and so generate additional PIT state.
Fig. 4.7a and Fig. 4.7b illustrate respectively the average ISR and PUR values in the same evaluation scenario where the network is under a chamaleonic Interest Flooding Attack (*cIFA*). In *cIFAs*, attackers periodically change target prefix name to avoid prefix name-based mitigation techniques activated after a certain observation time window. The change of the target prefix name nullifies the rate-limit which is applied to that prefix name after the detection. If attackers

change target with the same periodicity used by routers to trigger the detection (as done in the scenario illustrated by the plots in Fig. 4.7), then the mitigation is applied to a prefix name which does not carry anymore malicious traffic. Among the attacks proposed in [113], the *cIFA* is the one degrading the most the performance of the tested reactive countermeasures. Conversely, Charon is also immune to this IFA variant since its forwarding decisions are not driven by any detection threshold. Therefore, Charon keeps average ISR and PUR values close the baseline even during the attack.
In summary, the attack variants proposed in [113] revealed the inefficacy of existing defense mechanisms to neutralize a certain type of stealthy IFAs. Charon, instead, neutralizes those attacks as soon as they enter the network, so it preserves the quality of service perceived by legitimate clients and offloads the rest of the network infrastructure from dealing with loads of unsatisfiable requests.

4.4 Charon: size of the summary, optimizations and future work

This section introduces some design aspects of Charon which deserve immediate attention to assess the feasibility of the proposed approach and target specific improvements as subject of any future work.
First, it is important to estimate the size of a BF-based content summary since that could be queried by routers at link-speed per every Interest packet. For that purpose, a BF-based content summary is required to be stored on a memory technology fast enough to sustain the necessary network throughput (the reader should remember that Charon is supposed to be deployed at the edge of network whose throughput requirements can be read in [86]). The size m of a BF-based content summary may be tuned according to the desired false probability rate p and the number of hash functions k, where the number of elements n in the set is known. Considering a content list of one million elements (this is not uncommon since, for example, the wikipedia page titles content list used for the evaluations presented in Sec. 3.2.3 of chapter 3 and in Sec. 4.3.3 of this chapter holds almost 13 million of different content identifiers), a false positive probability of 0.01 and using between 2 and 5 hash functions would require a bloom filter of a few MB [137]. A filter of that size could be easily stored on today's SRAM memory technology (whose today's maximum capacity is 210MB [138]). Hence, multiple filters could be contemporaneously stored in fast on-chip memory.
Second, even though large size BF-based content summaries can be stored in fast main memories and meet the packet processing speed requirements, the architectural framework of Charon in Fig. 4.3 envisions the content summaries to be fetched by routers and potentially cached in the network as Data packets. Concerning the implementation of these system features, it is also important to understand whether bloom filter compression techniques [139] could be used to shrink the size of a summary and ease its transmission over and caching in the network.
Third, even if a pure Charon design turned to be prohibitive because of the impossibility of storing all the filters in a router's main memory, content summaries could still be loaded and queried only when under attack. This would translate into a hybrid proactive/reactive design where the detection of an attack still relies on the monitoring of traffic statistics, while the mitigation loads in main memory and queries the content summaries. The mitigation based on content summaries has the advantage of never dropping any legitimate traffic compared to the canonical reactive rate-limiting applied to name prefixes and/or interfaces. By design, this hybrid technique is expected to respond efficaciously to a *pIFA* but not to the *bIFA* and *cIFA* attacks which circumvent the classical detection metrics. However, in those latter cases the detection

could be simply triggered by an unusual load on the router since loading and querying the filter would not introduce the same overhead as the canonical monitoring techniques.

4.5 Summary

This chapter has introduced a novel approach to the defense against Interest Flooding Attacks (IFAs) in NDN. This new approach stems from the observation of certain pitfalls in the state-of-the-art countermeasures, which can be exploited by potential attackers to mount very effective IFAs as shown in Chapter 3. The reactive nature of the existing defense mechanisms is the root cause of the inefficacy to defend against stealthy IFAs. Thus, the here-proposed defense approach departs from the usual reactive defense approach and proposes a proactive strategy to identify and counteract an IFA instead. Proactiveness is based only on the analysis of an Interest packet fields to determine its either malicious or benign nature. Therefore, an ideal proactive defense requires no monitoring of previous traffic to be performed to make forwarding decisions. The absence of monitoring, typically used across reactive defense mechanisms, eliminates the vulnerabilities which are exploited by the stealthy attacks introduced in Chapter 3.
This chapter has also designed a specific proactive defense schema, called Charon, and shown its efficacy against the latest stealthy IFAs. Nevertheless, Charon, specifically, and proactive defense, in general, are only applicable under certain attack scenarios. Thus, the IFA threat persists in the NDN. Therefore, the next chapter advocates for the investigation of different forwarding mechanisms for the NDN with the aim to preserve the main PIT properties without exposing the IFA-related security issue.

Chapter 5

Alternative "defense" mechanisms

The Interest Flooding Attack (IFA) represents a severe security threat to Named-Data Networks (NDNs). In fact, the state-of-the-art IFA countermeasures prove to be effective only against naive attackers. Yet, as thoroughly analyzed in chapter 3, effective IFAs built upon realistic attacker models can be easily mounted to disrupt even the most robust defense mechanism. Chapter 4 has introduced the concept of proactiveness in the design space of IFA countermeasures. Later, that theoretical framework is leveraged to instantiate a defense mechanism of that sort to counteract efficaciously the latest proposed IFAs [113]. Proactive countermeasures eliminate serious drawbacks experienced by their reactive counterparts. Neither monitoring overhead is required nor delayed reaction is introduced by proactive countermeasures. Nevertheless, proactive defense is not the panacea for IFAs, since this class of defense mechanisms applies well under certain conditions of static content names and publicly available content lists. Provisioning of dynamic contents, a very common scenario nowadays, breaks the assumptions proactive countermeasures are designed upon. Overall, the uncertainty about the efficacy of existing defense mechanisms as well as the poor understanding of the real implications of IFAs for an operational network raise the legitimate question whether or not different forwarding mechanisms should be considered at this early stage of design of the NDN architecture.

The root cause of the IFA attack is the per-Interest state kept by the Pending Interest Table. As a consequence, both works proposing more lightweight PIT designs [140, 141] and works questioning the PIT existence by proposing alternative forwarding mechanisms [142, 143] have been recently emerging. Nevertheless, replacing the PIT forwarding mechanism in NDNs has several implications. In fact, the PIT guarantees several properties which may be inevitably lost with a different data-plane design. Beyond the basic Interest-Data design, the PIT enables, among others, Interest aggregation, Data multicast delivery, sender anonymity and adaptive forwarding behaviors. Further, the PIT has been leveraged to detect network problems [49] and has influenced the design of routing protocols for the NDN architecture [62]. Or yet, the PIT has been used for the design of hop-by-hop Interest shapers to achieve congestion control [144] and for tracking producers in mobility schemas [72].

Moreover, the lack of a common easy-to-access platform for testing and comparing alternative forwarding plane designs makes hard to get a solid understanding of pros and cons of the different available approaches. For the above reasons, this chapter's contribution is two-fold. First, a taxonomy of the main alternative forwarding mechanisms for CCN/NDNs is presented in Section 5.1. The presented taxonomy classifies the existing different approaches with respect to a set of PIT's cardinal properties here identified. Second, by acknowledging that the current state of affairs does not contribute to compare and evaluate alternative forwarding mechanisms, emerging

technologies for programming the data-plane have also been explored in the context of this work. Thus, Sec. 5.2 introduces the P4 language, a novel high-level programming language for packet processors, which has been used to implement the high-level description of an NDN data-plane, NDN.p4, which is presented in Sec. 5.3.

5.1 Alternative PIT designs

The PIT is a core component of the NDN data-plane. As widely explained in Chapter 2, the PIT enables the basic Interest-Data packets exchange. Beyond the stateful address-less forwarding mechanism, the presence of per-packet state in the PIT has interesting implications for the role of routing protocols, the design of adaptive forwarding behaviors, the regulation of network traffic. Thus, alternative forwarding mechanisms for NDN should not be only considered with respect to the basic Interest-Data exchange but rather by looking at their impact to all the secondary PIT properties.
Since the inception of the NDN architecture, several alternative forwarding mechanisms have been proposed with the aim to reduce the amount of necessary state in the PIT. Those works have been mainly driven by concerns about the scalability of and the security threat introduced by the PIT. The former is investigated since the estimate of the amount of resources needed by the PIT (see Sec. 2.3 in Chapter 2) guarantees this component can sustain certain link speeds. Therefore, a compression of the PIT size could mean the component would fit into faster memories. The latter represents a serious concern since, by design, normal users can easily abuse PIT resources in the network to create denials of services. With regard to the PIT scalability, the latest work about PIT sizing [103, 86] suggested available PIT designs can be implemented in fast memories and sustain edge and core network link speeds. Instead, the uncertainty about the efficacy of existing defense mechanisms against IFAs fosters the research on alternative forwarding mechanisms immune to this Denial Of Service attack.

5.1.1 Summary of the cardinal PIT properties

The main goal of the NDN's stateful data-plane design is that Data can be routed back to the origins of the requests from any point in the network without the notion of the source address in Interest packets. This mechanism is often referred to as Reverse Path or Breadcrumb Forwarding. Other properties, like the multicast data delivery, the interest aggregation and the adaptive forwarding, can be considered the outcome of the NDN-specific stateful forwarding design. Most of the time those properties are stated differently in the related works which seek for PIT replacements. Thus, the terminology is hereby clarified before the state-of-the-art alternative PIT designs can be compared.

Reverse Path Forwarding: Interests dynamically generate or update forwarding state in traversed routers for Data to follow the trails back to the origins of the Interests. Thus, the presence of PIT state allows Interests to be issued without source address. Further, PIT entries allow routers to detect those Interests which are possibly looping because of the multipath feature offered by FIBs. However, the NDN's reverse path forwarding assumes path symmetry dwells in the network, which is in sharp contrast with the path asymmetry observed in the Internet backbone [145].

Interest aggregation: at the first common router, Interests for the same content are aggregated

in the same PIT entry. Interest aggregation has been promoted in NDN to reduce network and server loads in combination with in-network caching. There have been so far few works analyzing the likelihood of Interest aggregation in CCN/NDN networks. Further, the related findings are contradictory. In fact, the theoretical models derived in [108, 146] suggest that indeed aggregation may appear at the network edge with very low probability for popular contents. Meanwhile, the latest large trace-driven study in [147] reports that there could have been a considerable reduction of server hits with Interest aggregation in the network for the analyzed traffic trace.

Multicast Data delivery: Interests for a content traveling from origins to content sources build a temporary multicast forwarding tree which is made of the PIT entries left in the network. Different consumers may join the "multicast group" at any time and their requests are aggregated at the first router where a similar Interest has already been received and forwarded. Data packets are delivered downstream by following the tree and get multicasted where the tree branches. Data consume related PIT entries. The multicast tree construction and deletion happens entirely in the data-plane without any support from the control-plane.

Infrastructure security and robustness: the absence of end-host addresses in the NDN communication nullifies some popular kinds of DDoS attack which affect today's Internet. For example, bandwidth depletion attacks, which flood the network and target victims with unsustainable traffic rates, cannot be carried out in NDNs because of the absence of addresses and of the presence of intrinsic mechanisms like Interest aggregation and in-network caching meant to smooth excessive loads. Moreover, the PIT state may be leveraged to design adaptive forwarding behaviors to react against different sort of anomalies, e.g., network problems like broken links and/or congestion, malicious behaviors like prefix hijacking.

Sender anonymity: the absence of the source address in Interest packets guarantees consumer privacy. That is, a third party cannot determine the source of an Interest only by looking at the Interest packet fields. It is the state in routers' PITs which allows Data to track back the origin of a request. An origin could only be determined if routers collaborated, for example, as postulated in [122] to trace back sources of an Interest Flooding Attack.

5.1.2 Taxonomy of PIT alternatives

At a first glance, there seem to be only two viable approaches, namely stateless or stateful, to design the forwarding mechanisms in Information-Centric Networking (ICN) architectures. Indeed, several attempts have been made to either reduce the PIT size or replace the PIT component in NDN/CCN networks. Hence, the categorization is broader and includes at least the following categories:

- aggregation techniques [95]: these techniques aim to consolidate multiple PIT entries in one single record as to reduce the maximum amount of forwarding state in the PIT. However, since the compression is not loss-less, some forwarding information is lost.
- hybrid techniques [140, 141]: this refers to techniques which are neither entirely stateless nor fully stateful. A portion of the Interests is generally forwarded without storing any related status in the PIT. The nature of an Interest, either stateless or stateful, may be chosen by different factors, e.g., content popularity [141], probabilistic or deterministic

Table 5.1 – Alternative forwarding mechanisms in NDNs evaluated in terms of cardinal PIT properties.

Technique	RPF	Interest aggregation	Multicast	Security	Sender Anonymity
Aggregation	YES	YES	YES	YES	YES
Hybrid	YES	Some	YES	Some	YES
Routing-based	NO	NO	NO	NO	NO
State-shifting	YES	NO	YES	YES	YES/NO

functions applied by routers [140]. Those techniques generally require a mechanism to track the Interests when those travel through stateless part of the network. In general, routers encode the reverse path in the Interest header which is used to forward the Interest across network portions where no related state has been stored. If the mechanism is not fully deterministic, then the Data delivery may cause overhead since Data are forwarded downstream on unsolicited paths.

- fully routing-based techniques [143]: this refers to stateless techniques using routing tables to forward Interest upstream and Data downstream. Additional routing information to reach the origins of Interests is needed in order to route Data packets downstream. Therefore, these methods increase the size of the FIB tables in routers to contain routes for both consumers and producers of contents. Furthermore, if the source address is a topological address, this kind of approach violates the consumer privacy by revealing the sender's identity. Nevertheless, an identifier local to a router can be used to denote the source of Interests and preserve sender identity, as done with the anonymous datagrams proposed in [142].

- state shifting techniques [148, 149]: this class refers to techniques where the state necessary for a potential further data delivery is moved to packet headers. The state may be either collected by Interests traversing the network (for example, in the CONET architecture [148]) or source-routed by the origin by means of a topology-omniscient centralized entity (like done in the LIPSIN forwarding architecture [149]).

Table 5.1 summarizes whether these forwarding mechanisms keep or lose the PIT properties previously described in 5.1.1.
On the one hand, aggregation and hybrid techniques seem to be promising solutions since those preserve many of the original PIT design properties. For example, all the cardinal properties stay unaltered for the fraction of Interests which are forwarded statefully by hybrid techniques, since the basic PIT forwarding mechanism is neither replaced nor eliminated. However, those techniques only partially eliminate the exposure to Interest Flooding Attacks, which, by the way, is the main concern driving the research on alternative forwarding mechanism designs. In fact, those techniques bring only a reduction of the overall forwarding state in the network and do not eliminate the possibility for malicious users to generate that.
On the other hand, routing-based and state-shifting techniques eliminate the issue with the state in the network, though as a consequence they lose some of the related PIT properties. Generally,

those techniques completely lose the ability to perform in-network Interest aggregation. However, it is interesting to analyze the mechanisms those techniques put in place to preserve some of the other cardinal properties which are lost because of the absence of the PIT state.
Overall, all the proposed techniques introduce some modifications to the forwarding plane, e.g., new packet headers [148, 143], additional PIT state per entry [140], different forwarding data-structures and operations [140, 143]. Therefore, all the current proposals may face deployment issues because of incompatibility with the existing architectures and protocols.

5.1.3 Questioning PIT benefits

Some of the latest proposed alternative forwarding mechanisms for NDN/CCN networks [143, 142] present strong arguments against the cardinal PIT properties. Those works try to demonstrate that the PIT may be infeasible in practice because of actual network ISP agreements and of the unsolved security issues which are inherent to its stateful design. Moreover, many of the claimed benefits, like reduced end-to-end latency via Interest aggregation and consumer privacy via sender anonymity, may either not subsist or be achieved by means of other forwarding mechanisms. If indeed the benefits are overrated and the actual gain is considerably marginal, but the security issues are real and harmful, then the use of the PIT should be reconsidered.
With regard to the *Reverse Path Forwarding* tenet, that seems to be based on the common belief among the research community that communications flow bidirectionally across the same physical links in the Internet. This symmetry has been embedded for a long time in traffic classification tools and then adopted in the design of the NDN architecture where every Data packet follows a reverse path to one or more origins of the request. Nevertheless, recent studies [145] based on data collected over different Internet links (e.g., different tiers and links between those) show that moving from the network edge to the core Internet path symmetry is rarely observed. A dominant cause of this asymmetry is the "hot-potato routing" [150] practice employed by ISPs to minimize the transit of some traffic within their networks. Besides that routing policy, traffic engineering techniques, e.g., load balancing, may also cause packets sent to the same destination to be forwarded over different paths. Therefore, taking into account today's ISPs routing policies and economic settlements, assuming path symmetry across the future Internet seems unfeasible. As a conclusion, the reverse path forwarding is a property which should be either entirely rethinked or only enforced at the network edge where there could be the conditions to realize it.
As regards the *Interest Aggregation* property, some works [108, 146] have tried to determine the likelihood of Interest collapsing via theoretical models and network simulations. Interests are aggregated when they exhibit temporal correlation. That is, if the Interests arrive at a router within a very small time window, which is expected to be of the order of tens of milliseconds. Further, Interest aggregation will bring the most gain when that happens the most closest to the consumer. Hence, it is important to understand how likely both conditions hold. Both the independent studies [108, 146] highlight the high interplay between in-network caching and Interest aggregation. According to their simulation results, it appears that the former plays a fundamental role in determining the fraction of Interests which could benefit from aggregation. To summarize the findings therein, the percentage of aggregated Interests decreases with even a small amount of caching budget (e.g., 1% of the total number of published objects) available in the network and when consumers are less likely to ask for similar content (e.g., low temporal correlation among Interests). Moreover, most of the Interests are aggregated closer to the content sources. This last behavior would drastically affect the benefits of reduced end-to-end latency and network load advocated by this structure. These findings seem to demystify the

need of Interest aggregation, since in-network caching could be applied to achieve similar benefits without exposing the NDN data-plane to IFAs. It is assumed that Interest aggregation also prevents multipath routing to generate loops. In fact, PIT entries store Interest nonces and so are able to distinguish the reception of a duplicate Interest which may be looping in the network. However, there exist conditions (for the first time presented in [151]) where Interest aggregation in the PIT may prevent Interests loops to be detected and cause Data to never be delivered to consumers. The same work [151] proves no correct forwarding can be designed simply by uniquely identifying with a different Nonce each Interest. Therefore, the current NDN Interest aggregation mechanism does not guarantee correct interest forwarding. Alternative forwarding mechanisms, which are immune to the undetected loops problem, have already been proposed [152].

5.2 Programmable Data-planes

The first generation of IP routers were software programs running on general purpose processors [153]. The use of generic platforms eased the design and implementation of packet processing functions at a high level at a time where network throughput was not a prime requirement. In fact, general purpose chips and architectures were not designed to optimize few specific tasks, rather to be generic enough to perform a vast set of operations. Today, network functions are mostly implemented with Application Specific Integrated Circuits (ASICs) to meet strict latency and network throughput requirements imposed by higher link-speeds. Meanwhile, more versatile chips, like Network Processor Units (NPUs), can be used in less time-critical network environments where, however, changes of the network behavior may be required on the field.
Wisely looking at the design of forwarding devices, i.e., switches and routers, over the past decades, two important facts emerge. First, new protocols have always been first standardized and then implemented in network equipment (among them, the recent proliferation of encapsulation protocols like VxLAN, PBB, GRE). Second, today's forwarding devices offer more functionalities (QoS, security, etc.) in addition to the simple L2 and L3 forwarding. In either case, a new protocol or a new functionality may lead to the design of a new chip whether that is not supported by the latest hardware. The latter operation is a few-year long process which slows down the adoption of a technology at best, or discourages it at worst. Since the introduction of new protocols and functions seems to be more and more commonplace, several chip designs for high-speed reconfigurable packet processors have been proposed [154, 81, 155]. Novel programmable packet processors come together with new programming languages to express the network behavior into an easy and reusable way [154, 82]. Among those languages, the Programming Protocol-independent Packet Processors (P4) [82] has gained notable traction since its inception in 2014.
As explained in detail in the previous chapters, an ICN network layer routes content requests toward nearby content sources using their content identifiers rather than specific host addresses. In NDN/CCN-like networks, routers feature the three functional blocks, namely PIT, CS, FIB, working on named data. The NDN data-plane involves new and unconventional processing to be performed at line-speed. For each NDN packet, several lookups on variable length names may be required, further, some related state has to be stored and/or updated. The design of NDN-enabled routers turns to be challenging because of two main issues: i) the algorithmic challenges that still need to be solved [156, 157, 86], ii) the evident lack of adequate hardware support to describe this data-plane behavior [158, 98]. Existing designs of an NDN router have been engineered by leveraging specific features of the target software/hardware platform. As result, those designs cannot be easily shared and leveraged for either further research or operational deployment.
We believe that the emergence of a new generation of fully programmable forwarding elements represents an opportunity to advance the research on ICN. In fact, if the next generation of programmable networking devices has the right feature set to fully describe any ICN packet processing, then there will be more chances to showcase the potential benefits of ICN architectures. More broadly, we consider the advent of this technology as an inflection point [159] for a confluence between the Software-Defined Networking stream of thought and the ICN one.
To promote the description of ICN protocols in a unified open-source way, which would made the related designs sharable and re-usable, the work NDN.p4 in [83] has first been conducted. NDN.p4, which is presented in this section, has been the first attempt to implement an ICN protocol using the P4 language for packet processors. Sec. 5.2.1 introduces the P4 language, the main syntax and the related programming model. Sec. 5.3 describes the implementation of the

NDN protocol in P4. Finally, Sec. 5.3.6 recapitulates the main lessons learned during the design of the NDN.p4 solution.

5.2.1 The P4 language

P4 [82] is a programming language to configure the data-plane of network forwarding elements. The aim of the P4 language is to abstract how packets are processed by different devices in a high-level target-independent description. Then, target-specific compilers are supposed to map and optimize the high-level description in a P4 program to the low-level intricacies of the respective target architectures. The idea behind the language was first proposed in [82] and since then collaboratively developed by a rich community made of academics, network equipment vendors, telco providers under the umbrella of an open non-profit consortium [160].
Since the first language specification, called P4_14, the language has considerably evolved and become mature in a new language version, called P4_16. P4_16 is not backward compatible with P4_14. Both language specifications are available at [160]. However, NDN.p4 was implemented in P4_14, so the description of the language which follows refers to that early version.
A P4 program configures the supported protocols and the packet processing operations in a forwarding element. A forwarding element which can run P4 programs is called a P4 target or simply a target. The P4_14[13] language specification is based on a specific abstract model of a forwarding element. The forwarding model is composed of a programmable parser, a buffer for packet headers and metadata, an ingress and an egress pipelines of match-action tables as shown in Fig. 5.2.1 (the processing pipeline of this model mimics the commonplace design of some of the latest fully-programmable forwarding devices [81, 161]).

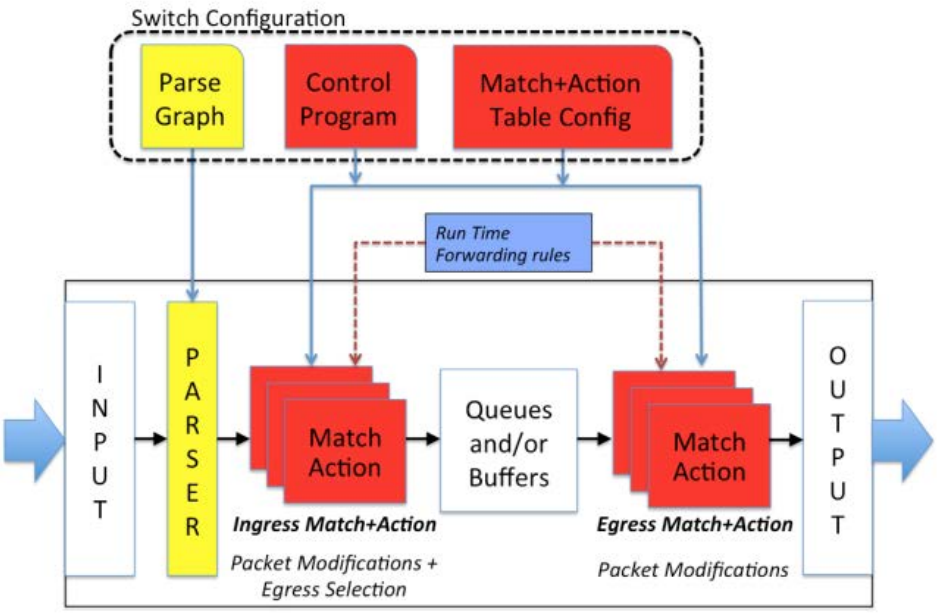


Figure 5.1 – The abstract forwarding model for the P4_14 language specification [2].

[13]For the sake of clarity, from now on until the end of this section, the abbreviation P4 is used whenever the text refers to the P4_14 language specification.

Usually five parts are needed to describe the behavior of a forwarding device in P4: the headers, the parser, the actions, the tables and the control flow. Those parts compose the yellow and red blocks in the abstract model of Fig. 5.2.1. A small snippet of code for each program block is illustrated in Table 5.2 where a simple P4 program is dissected. The illustrative program in Table 5.2 describes the behavior of a simplified IPv4 router which parses Ethernet and IPv4 headers in incoming packets and then forwards those based on their IP destination addresses. Finally, the P4 program rewrites the packet's MAC source and destination addresses based on the output port the packet is transmitted to.

The **headers** are the definition of protocols that the P4 target uses to parse the incoming bit stream to identify the protocols in a packet and extract the fields necessary to the processing. A P4 header contains a list of fields and field widths in bits arranged in a C struct-like layout, as shown in the A cell of Table 5.2 where the structure of an ethernet header is specified. Headers represent types that can be instantiated and initialized once found and parsed in packets, as shown in the B cell of Table 5.2.

The **parser** is a finite-state machine that determines and extracts the protocols seen in a packet according to the headers definition. The parser is made of parser states whose transitions can be determined according to header field values (e.g., the Ethernet's ethertype field in our example). A simple parser for the ethernet and IPv4 protocols is shown in the cells B and C of Table 5.2.

Actions are language primitives to express common packet editing tasks, like modifying header fields, selecting output ports, adding/removing headers, etc. Primitive actions can be combined in bundles to create compound user-defined functions. For example, the *set_nexthop* action in the D cell of Table 5.2 combines the two primitive actions, *modify_field* and *subtract_from_field*, to perform the packet editing necessary to forward the IP packet to the nexthop and decrement the TTL count. Compound actions are meant to be linked to table entries.

P4 **tables** are logical tables used to trigger the activation of actions for packets. The table in the F cell of Table 5.2 specifies a longest prefix match type on the IPv4 destination address to activate either the *set_nexthop* action or the *on_miss* one. The table in the G cell translates the information about a next-hop to an output port and re-writes the packet's MAC addresses based on the selected output port. Table definitions in P4 programs only specify the logical skeleton of the table. Table entries are then populated at run-time by a control plane logic. A run-time insertion command fills a table entry with specific values for the matching fields, an indication of the action to associate to the entry and possibly some optional parameters.

The **control flow** defines the program execution path. The programming model in Fig. 5.2.1 includes two pipelines, one for the ingress stage and another one for the egress. The ingress pipeline contains a sequence of match-action tables to be executed sequentially and/or according to some branching conditions. The ingress pipeline produces an egress specification in terms of output ports and number of packet copies. The egress specification is later used in the egress pipeline to correctly forward the packets. The illustrative P4 program of Table 5.2 has a very simple control flow, illustrated in the H cell, which consists of the application of the two defined tables. First, the table *ipv4_fib_lpm* is applied in the ingress pipeline to select the next-hop address. Then, the table *forward* selects the output port and rewrites the MAC addresses in the egress pipeline.

Table 5.2 – Snippets of code showing some of the P4_14 syntax.

A - Header Definition and Instantiation

```
header_type ethernet_t {
  fields {
    dstAddr : 48;
    srcAddr : 48;
    etherType : 16;
  }
}

header ethernet_t ethernet;
```

B - Parser part 1

```
parser start {
  return parse_eth;
}
parser parse_eth {
  extract(ethernet);
  return select(ethernet.etherType) {
    0x0800 : parse_ipv4;
    default : ingress;
  }
}
```

C - Parser part 2

```
parser parse_ipv4 {
  extract(ipv4);
  return ingress;
}
```

D - Action definition 1

```
action set_nexthop(nexthop) {
    modify_field(standard_metadata.egress_spec,
    nexthop);
    subtract_from_field(ipv4.ttl, 1);
}

action on_miss(){
    no_op();
}
```

E - Action definition 2

```
action rewrite_macs(smac, dmac) {
    modify_field(ethernet.srcAddr, smac);
    modify_field(ethernet.dstAddr, dmac);
}

action no_rewrite(){
    drop();
}
```

F - Table definition 1

```
table ipv4_fib_lpm {
    reads {
        ipv4.dstAddr : lpm;
    }
    actions {
        set_nexthop;
        on_miss;
    }
    size : 5000;
}
```

G - Table definition 2

```
table forward {
    reads {
        standard_metadata.egress_port : exact;
    }
    actions {
        rewrite_macs;
        no_rewrite;
    }
    size : 64;
}
```

H - Control Flow

```
control ingress {
    apply(ipv4_ fib_lpm);
}

control egress {
    apply(forward);
}
```

5.3 NDN.p4

NDN.p4 is the first implementation of an ICN protocol in P4. The NDN.p4 contribution was two-fold: i) showcasing for the first time how ICN protocols and the related network behaviors could be implemented and tested on the next-generation of programmable data-planes, ii) highlighting some of the limitations of the P4 language in describing non-standard packet processing operations. NDN.p4 had to face the constraints of the P4's early language specification, so it does not provide a fully functional NDN router rather a partial implementation featuring the basic NDN Interest-Data exchange. Additional features, e.g., in-router caching and Data multicasting, have not been implemented because those did not meet the two following requirements:

- a P4 program fully compliant with the language specification, that is, no new constructs have been added to the language,
- no assumptions about the existence of target-specific components.

As a result, the NDN.p4 program can be used to program any P4_14-compliant target. For example, the P4 software switch at [162] has been the reference target to validate the NDN.p4 design. The target is programmed as a forwarding element able to perform the basic NDN Interest-Data exchange. More in detail, at the Interest reception, the programmed target parses the Interest name and forwards the Interest according to its FIB-like records. For every forwarded Interest, the target registers a pending request in a PIT-like table. At Data reception, the target parses the Data packet and extracts the name. If the PIT-like table contains any pending request for the same content name, then the Data is forwarded downstream to the registered output interface; if not, the Data is simply dropped.
The correctness of the program logic has been evaluated through native NDN applications communicating over the software target programmed by the NDN.p4 code. The experiment simulates two NDN applications, ndnpeek and ndnpoke, exchanging Interest and Data packets through the P4 software-switch [162] as if it was a real NDN forwarding daemon. The ndnpeek application is a basic consumer of Data packets which can generate Interests with specific content names. The ndnpoke application is a basic producer of Data which can serve requests for contents under a specific prefix name. The experiment has also been made available on the mini-NDN [163] fork of the Mininet platform for network emulation. The source code of the NDN.p4 router and the instructions to reproduce the experiments are free available at https://github.com/signorello.
The NDN.p4 router program consists of less than five hundred lines of code written in P4. In the following, the main components of this P4 program are described through the illustration of partial snippets of code where needed to highlight distinctive features of the implementation. Finally, this section also presents some insightful lessons learned throughout the implementation of the NDN.p4 solution.

5.3.1 Headers & Parser

NDN packets are made of nested variable-length Type Length Value (TLV) blocks. Those blocks contain the header fields and the payload according to the packet specification [61]. For example, a dissected Interest packet is shown in Fig. 5.2. The outmost TLV of the Interest, here named T_1, contains all the Interest fields. T_1's Value contains two more TLVs, T_N for the name and T_{No} for the nonce. In turn, T_N's Value contains a sequence of four TLVs, denoted T_{Nc}, each containing

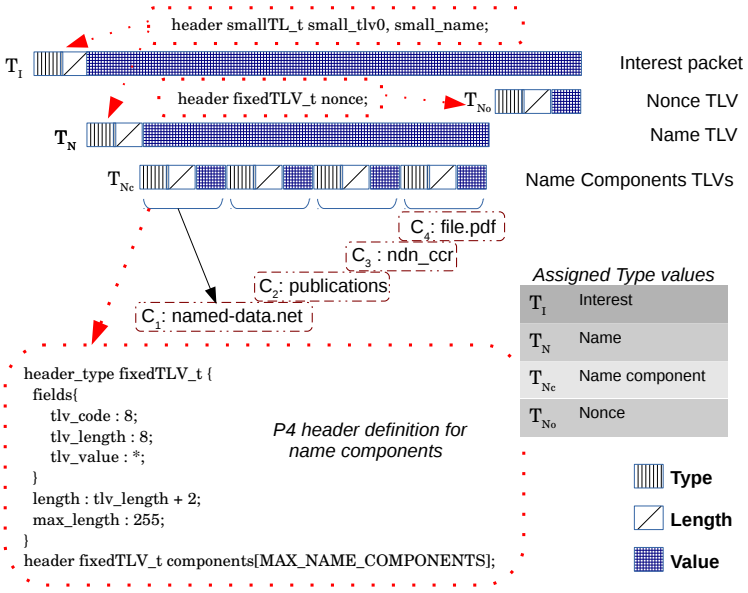


Figure 5.2 – Interest for ndn:/named-data.net/publications/ndn_ccr/file.pdf

a name component C_n . NDN TLVs follow a variable length encoding. Type and Length may be encoded in either 1, 2, 4 or 8 bytes while the Value size is indicated by the Length value. All the Type codes currently used by the NDN protocols are encoded in a single byte, so NDN.p4 assumes to deal only with TLVs of fixed-size Type.
P4 header definitions can contain one variable-length field at most per header. However, NDN packets feature several variable-lenght TLVs. Yet, some TLVs serve as containers for further nested header fields (e.g., the Name TLV is the container of all the name components TLVs). Therefore, NDN.p4 uses several headers to reproduce the structure of the NDN packets, which could not be described by a single header definition in P4. First, four header types for all the possible Length sizes are defined assuming the Type encoding is fixed (smallTL_t, mediumTL_t, bigTL_t, hugeTL_t). Those headers extract Type and Length of TLV containers leaving the Value for further inspection. Second, the TLVs with only variable-length Value are represented by the fixedTLV_t header type. For example, the Type and Value fields of the T_1 and T_N TLVs in Fig. 5.2 are parsed as a separate headers of the type smallTL_t (see this and all the other available header definitions for Type and Value blocks in Fig. 5.3). Instead, name components TLVs are instances of the fixedTLV_t header type (hence are of header type described in the lower fine-dotted box in Fig. 5.2). Name components TLVs have a 1-byte Length field encoding and a variable-length Value according to the component size.
Further, NDN.p4 assumes both Interest and Data packets contain a maximum number of name components C_n, which must be known at compile-time. For that purpose, P4 provides a fixed-size array-like construct, namely the header stack. A P4 header stack allows the programmer to have a concise representation of multiple consecutive headers of the same type. A header stack is used in NDN.p4 to define an array of fixedTLV_t headers for the name components, as shown in the lower fine-dotted box of Fig. 5.2. The header stack maximum size must be a constant value (e.g., the constant 'MAX NAME COMPONENTS' in Fig. 5.2).
The parser code inspects incoming packets and extracts header fields according to the header definitions. State transitions in the NDN.p4's parser occur either by : i) reading the Type value of the next TLV when the length of the current TLV block is known (as done in the

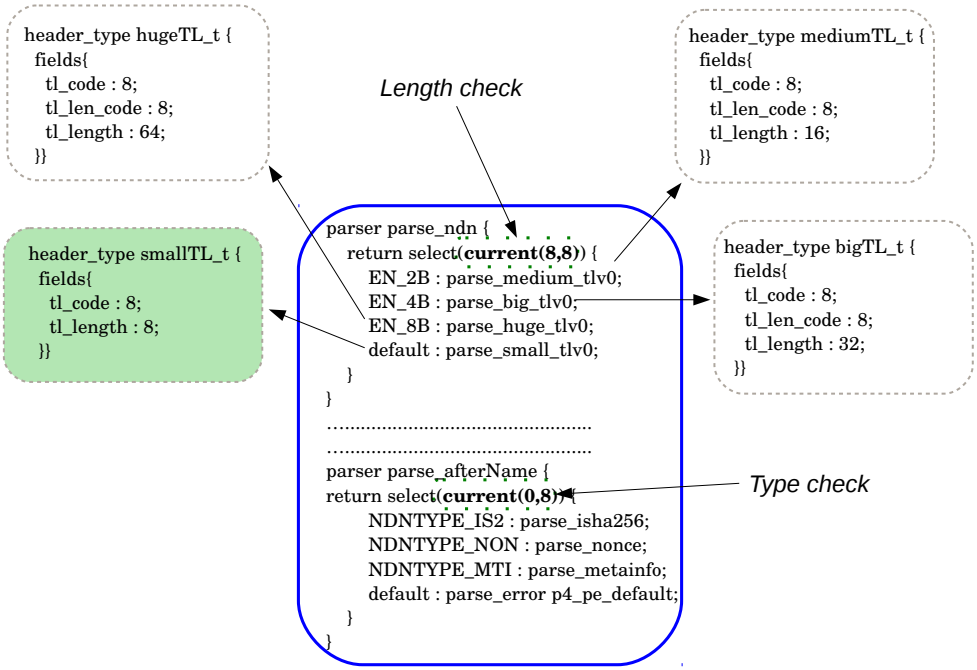


Figure 5.3 – Some parser states and header definitions in NDN.p4

parser_afterName state of Fig. 5.3) or ii) determining the Length value of the current TLV block to select the header to be extracted in the next parser state (as done in the parse_ndn method of Fig. 5.3). For example, the value of the Length of the outmost T_1 in Fig. 5.2 is read before proceeding with the extraction of a header of type smallTL_t, as shown in the upper part of the central box of Fig. 5.3. If the Length value indicated a bigger encoding, a different header would be extracted among the available header definitions: mediumTL_t, bigTL_t and hugeTL_t.
The parser also includes a special state to extract the name components, which loops on a counter initialized with the total name length. After a name component is extracted, its total length is subtracted from the counter; the parser loops until all the components have been extracted.

5.3.2 Pending Interest Table (PIT)

PIT entries are created and updated by incoming Interest and Data packets. By consequence, PIT modifications happen entirely in the data-plane. Nevertheless, P4 tables can just be edited by the control-plane (as better explained in the Subsection 5.3.5). Hence, the NDN.p4's PIT table is implemented on registers. P4 registers represent stateful memories which keep state across multiple packets and can be updated in the data-plane. The complete PIT behavior is simulated by also using three match-action tables: the hashName_table, the pit_table and the updatePit_table. The hashName_table computes the index of the PIT entry as hash of the packet name and writes that to the *name_hash* metadata field. Then, the pit_table uses

name_hash to read the indexed register location[14]. Finally, the update_Pit either creates a new PIT entry or records a new incoming interface in the existing PIT entry. In summary, the actual PIT table is stored in registers, while the match-action tables are responsible for addressing and eventually updating that register block.

5.3.3 Forwarding-Information Base (FIB)

FIB entries associate prefix names to one or many output interfaces. The algorithm that determines the FIB entry matching a content name is a Longest-Prefix Match (LPM) on name-components. An LPM on name-components prioritizes the entry whose prefix name has the longest-prefix in common with the Interest name over the entries having smaller ones.
P4 tables feature standard match types like exact or ternary match. Of course, an LPM on name-components is not natively supported by the P4 table construct. So, NDN.p4 implements this kind of match by using some other native match types and a specific set of table entries. The designed algorithm is better explained through an example which is reported in Fig. 5.4. The FIB of the example has the two entries A="/a/b/c eth1" and B="/a/b eth4", made of three and two name components respectively. At the reception of an Interest named ndn:/a/b/c/d, the A entry must be selected.
At that stage of the control flow, the full hash of the Interest name and the hashes of all the smaller name-prefixes are already available as metadata, as illustrated in the top-left box of Fig. 5.4. The fib_table (in the top-right box in Fig. 5.4) includes the total number of name components, the full hash of the name and the hashes of the shorter prefixes as matching fields. The route A for the prefix "/a/b/c" is made of a number of table entries equals to

$$n - m + 1 \qquad m \in [0, n],$$

where 'm' is the number of components in the prefix and 'n' is the maximum number of name components that the target is programmed to parse. Those entries allows the program to match Interests with a number of name components equal or greater to 'm' for that FIB entry. Since multiple table entries correspond to a single FIB entry, then a further selection mechanism has to be applied. Thus, ternary matches on the metadata fields "comp_metadata.cx" are used to match on the right hash combination for the prefix-name. For example, to match against a FIB entry that is made of three name components, e.g., route A in Fig. 5.4, the fourth match-field of the related entries in the fine-dotted green box , that is, 'h(a,b,c)&&&1', contains a full mask to perform an exact match on this field. All the other hashes have a zero mask, so not match is performed to those. However, Interests for ndn:/a/b/c/d would match the second and the fifth entry of the rule-set that correspond to route A and route B, if only the number of components and the hashes of the route prefix-names were considered. Hence, these FIB entries include an additional value that specifies the priority with which they must be applied, i.e., the last value of each entry of the rule-set. A lower value gives higher priority to a matching entry. For example, the 2nd entry has higher priority than the 5th entry, and so route A is finally the matching entry for the Interest /a/b/c/d.
Two shortcomings of this algorithm are: i) the number of records to be inserted for a single FIB entry depends on the number of name components of the related prefix name, ii) the priority assignments have to be predefined by a control plane logic.

[14]The reader should be aware that the name is produced as a hash and so the register suffers from collisions. However, in this preliminary implementation, no mechanism has been designed to prevent this issue.

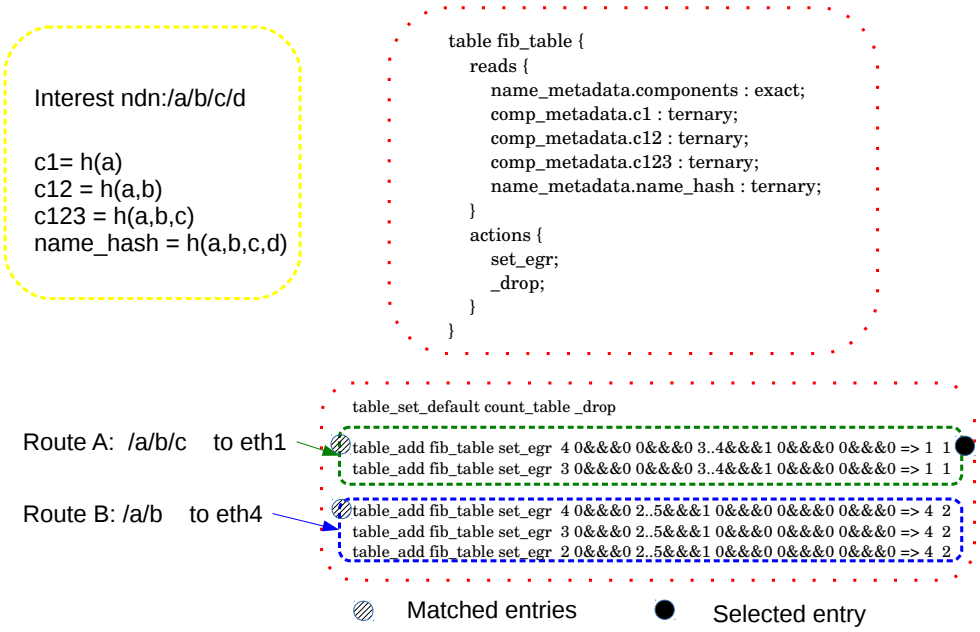


Figure 5.4 – Definition of the fib_table and FIB entries for two prefix names.

5.3.4 Control Flow Code

Fig. 5.5 schematizes the program execution path of the ingress pipeline where Interests and Data are processed. That ingress pipeline contains six match-action tables, a register block to simulate the stateful PIT behavior, some temporary metadata. In P4, an action cannot be directly called in the control flow program rather its activation goes through the application of a match-action table. Thus, some of the tables in the ingress pipeline are only meant to perform computation on the packet fields and store the results in metadata for further operations. Those tables are executed independently from any matching condition on the packet fields.
After the parsing stage, the count_table is applied to count the total number of name components in the current packet. As explained in the section describing the FIB implementation, this number is later used by the LPM algorithm to select the right FIB entry. The hashname_table is applied to compute the hash of the full packet name, i.e., h(a,b,c,d), as well as the hashes of all the shorter prefixes, i.e., c1=h(a), c12 = h(a,b) and c123 = h(a,b,c). The full name hash is needed to address the PIT, while the hashes of the shorter prefixes are required to perform the Longest Prefix Matching (LPM) on the FIB table. In sequence, the pit_table reads a *register* indexed by *name_hash* and copies it in the *isInPIT* metadata field. If the register value indexed by the full name hash equals zero, the target has no other pending Interests for this name and so a FIB lookup must be done to determine the next-hop. If the register is not empty, the Interest is dropped and eventually the register is updated. After applying the pit_table, the control flow has a conditional branch based on the packet type, that is, either Interest or Data. In case of an Interest packet, *Flow_metadata.isInPIT* contains the list of the interfaces where the target has already received an Interest for this content. A PIT entry is represented as a

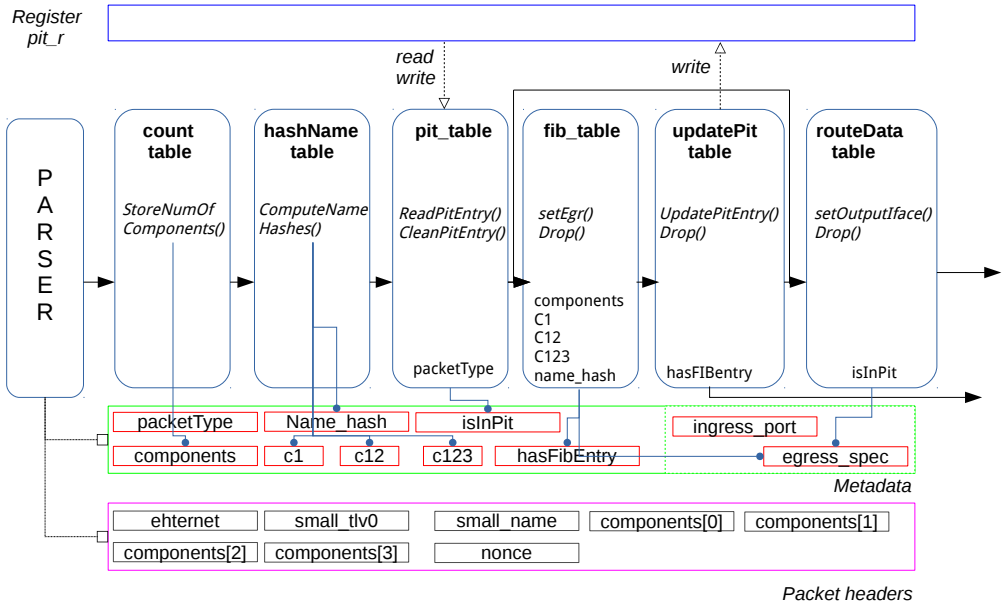


Figure 5.5 – Illustration of the control flow of the ingress pipeline of the NDN.p4 program.

bitmask. For example, the PIT entry '00000100' encodes information about 8 interfaces whose interface number 2 has already seen an incoming Interest for this content name. So, if any of the *Flow_ metadata.isInPIT* bits is set to 1, the target has a pending entry for that name and does not forward this Interest upstream. If not, then the fib_table is applied to lookup the next-hop. The fib_table performs the LPM algorithm described in Sec. 5.3.3. If the LPM returns an entry for the Interest, then the updatePit_table is applied to create the related PIT entry. The PIT entry is created by marking the incoming interface in the bitmask. The so-created entry is finally stored in the register cell indexed by the hash of the full Interest name.
The execution path for a Data packet is similar to the one for an Interest except for the code executed after the conditional branch on the packet type. The action associated to the pit_table entry for Data packets reads the register value to load the interfaces out of which this Data packet must be forwarded downstream. The same action cleans the read register since the current Data packet is supposed to consume the related PIT entry. In case of a Data, the routeData_table is applied last. This table reads the downstream interfaces this Data must be forwarded to from the *isInPIT* metadata field. The actions associated with the routeData_table's entries set the output interfaces on the standard metadata *egress_ spec* that will be read by the egress pipeline to forward the packet out of the selected switch port.

5.3.5 Control Plane

All the tables defined and used in the NDN.p4 program are initially empty. These tables must be filled by a control plane application to perform the defined packet processing logic. In P4 targets, the control-plane communicates with the data-plane via some interfaces which are partially defined by the P4 programs too. The P4 software switch in [162] offers an application programming interface (API) to the P4-programmed data-plane for managing entries in tables. Software tools are available at [160] to write control plane programs for the P4 switch too.

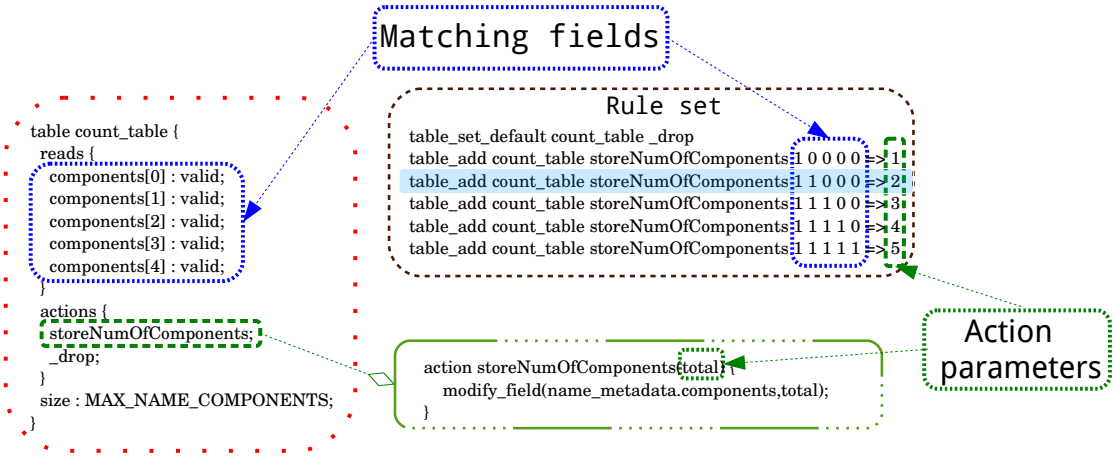


Figure 5.6 – Definition of the count_table and related table entries

A glimpse of the syntax to modify the match-action tables in NDN.p4 is provided in Fig. 5.6 where the count_table of the ingress pipeline and the rules used to fill it are reported. The count_table counts the number of name components in a name. More precisely, the count_table counts the name component TLVs in a packet through a validity match. A validity match returns false if a header is not present, true otherwise. Therefore, the table contains as many matching fields as the maximum number of name components that a packet may have. For example, an Interest name with two name components will have the validity bit set for both the component[0] and component[1] headers and clear for all the other header components. In this case, the table needs a rule that translates the two valid headers in the packet to a counter value, that is, the third rule of the rule-set presented in Fig. 5.6. Further, every table needs a default rule to specify the action to be performed in case of table-miss. The default rule for the count_table is the first rule of the rule-set in Fig. 5.6, which calls an action to drop the packets without any valid name component header.
A rule has the following syntax:

command_type table_name [associated_action] [match_fields_list] [=>] [parameters_list].

The command_type indicates the kind of operation, like add, delete or modify a table entry. The associated_action indicates the name of the action bound to this table entry. The match_fields_list are the specific value of the table matching fields for this entry. Finally, an action may or may not accept parameters. When present, the action's parameters follow the symbol '=>' as parameters_list.

5.3.6 Lessons learned and future development

When the NDN.p4 design was first tackled, the P4_14 language seemed to have a limited set of constructs to fully describe the NDN data-plane. This could have led to the premature conclusion that the language was not complete enough to express certain network behaviors and so language extensions were strictly required. Although enhancing the language syntax could have been easy and convenient in a first stage, on a long term it could have hampered the program portability. Further, it would have been a worthless effort since the language specification was

still premature and a clear direction for the language evolution had not been set yet. This latter observation was later confirmed by the release of the next language specification named P4_16 where, instead, the basic syntax has been reduced to a small core set to be complemented with libraries to express operations and components which could be more target-specific.
NDN.p4 overcomes some of the language limitations by making specific assumptions on some packet fields. For example, some assumptions about the NDN's headers length have made it possible to parse a variable-length TLV header although the language only allows one variable-length field per header.
The P4_14 language version has turned to be very tied to the reference abstract forwarder model. Indeed, the language was made to encompass a wide range of different targets, while the control flow of P4_14 programs allows developers to express execution flows by mainly stacking match-action table calls. Although architectures featuring match-action table pipelines are widely used to implement fully-programmable line-speed switching chips, those do represent neither the only target of the language nor a generic enough architecture. As a further evidence of this language restriction, the "architecture-language separation" has become a key concept of the P4_16 language specification.
With regard to potential language extensions, NDN.p4 has taught us the importance of approaching the language design community with "compelling" use-cases for new features to be incorporated into the language. By "compelling", we refer to standard protocols implementing a behavior similar to the targeted one. This kind of example better works as incentive for the community members to discuss about and work on the design of prospective language features. Here follow some examples:

- *Parsing NDN TLVs*: variable-length TLVs could be a common-place for the next-generation of network protocols, since their flexibility has already been adopted for optional fields of well-established protocols, e.g., the IPv4's options [164] and the IPv6's extension headers [165], or yet, the optional parameters in the BGP4's open messages [166].
- *PIT table updates*: there already exist packet operations which require table updates to be performed in the data-plane because of strict latency requirements. For example, a MAC learning algorithm stores the Ethernet source address and the incoming port of a received packet in a table that the target uses to forward later incoming packets based on their destination addresses. This table update operation might not bear the delay introduced by an eventual communication with a control plane application as the current P4 specification envisages [167]. A solution for updating a tiny status in a predefined table has been already raised by the work in [168].
- *Reading Data signature*: Data packets carry signatures trailing the packet payload. Routers may optionally verify signatures. Therefore, the signature field may need to be parsed. The latter operation would imply to skip a packet portion and move the parser pointer to the trailing signature field. However, P4 does not allow to skip the parsing of some packet portions [169]. Analogously, this feature would be needed by the IPSEC protocol [170] in tunnel mode, which is the protocol used for the traditional widespread setup of Virtual Private Networks (VPNs).

5.4 Summary

This chapter explores two different subjects: alternative forwarding mechanisms, at first, and design platforms, later, for the NDN data-plane. The need for this exploration originates from

the conclusions drawn by the contributions presented in Chapters 3 and 4. Those findings about IFA countermeasures exacerbate the fear about the impossibility to protect effectively the NDN architecture from all the possible variants of this attack. Therefore, it is worth exploring, as already witnessed by several recently-emerged related works, the design of different forwarding planes for the NDN. For that purpose, it is of foremost importance to establish both the cardinal PIT properties to be kept by such an alternative forwarding plane and the ones that can be relaxed or lost instead. Along this line, this chapter has introduced a proper evaluation framework and classified existing alternative forwarding proposals according to it.
Nevertheless, the comparison of alternative NDN data-plane designs is made difficult by the lack of open-source NDN router designs. In fact, existing NDN designs are tailored to specific software/hardware platforms and sometimes closed, hence, difficult to share, evaluate and modify. Hence, this work seminally has proposed to leverage the latest advances in the field of programmable data-planes to produce high-level and portable NDN router designs. Although the NDN.p4 data-plane design was constrained by the P4 technology's infancy at the time it was proposed, it has in any case outlined the potential of those emerging technologies to be leveraged to ease the deployment and test of ICN solutions. As better explained in the last section of this manuscript that presents the opportunities for future work, today's languages and platforms for packet processors can be leveraged to improve the NDN.p4 design and finally transform that in the originally envisioned platform meant for researchers to experiment with different data-plane designs for the NDN.

Conclusions and Future Work

The Named-Data Networking (NDN) concept has emerged as a promising clean-slate future Internet architecture. The NDN retains the same Internet's hourglass model, but it replaces the hourglass' narrow waist to achieve more efficient content localization and retrieval. The NDN's narrow waist supports content retrieval via a receiver-driven communication model, offering data-centric security and in-network storage. Therefore, not only content distribution, but content-centric security and consumer mobility are also naturally well served by the NDN design. Despite the many potential benefits, a set of open challenges still slows down a wide adoption of the NDN architecture for further experimentation. Among the NDN-related open issues, the security of the proposed architecture represents a major concern. The NDN architecture is immune by design to several well-known attacks today's IP-based networks are affected with, e.g., application flooding attacks, reflection attacks and prefix hijacking. Nevertheless, NDN suffers from novel architecture-specific vulnerabilities. Among the security threats to the NDN, the Interest Flooding Attack (IFA) is a Distributed Denial Of Service (DDoS) which capitalizes on the state created by regular Interests in routers' tables. Because of the easiness of mounting disruptive IFAs, the existence of a stateful component in the NDN's forwarding plane has also started to be reconsidered [108, 143, 142].

This dissertation work strengthens the understanding about the severity of the IFA DDoS and the potential of the countermeasures against it. As a first contribution, we disprove that the state-of-the-art IFA defense mechanisms are robust enough to mitigate any potential IFA. Indeed, our work shows how the reactive nature of the most efficient existing countermeasures makes those vulnerable to a certain type of stealthy IFAs. In fact, reactive countermeasures require to collect and analyze traffic statistics before triggering any mitigation strategy. Therefore, reactive countermeasures introduce monitoring overhead and delayed reactions. We show that both factors can be exploited by potential attackers in IFAs. Attackers can pollute metrics collected by routers by dosing different Interest types and circumvent observation time windows in routers by changing target prefix name. The IFAs proposed by our work adopt both strategies to successfully counteract existing reactive countermeasures.

As second contribution, we define proactive countermeasures against the IFA. This novel class of countermeasures includes defense mechanisms which do not perform any traffic monitoring to detect an attack. At first, we conjecture proactive defense eliminates the monitoring burden required and overcomes the vulnerabilities experienced by its reactive counterpart. Then, by designing Charon, a proactive IFA countermeasure where NDN routers rely only on the Interest name analysis to identify and drop fake Interests, we empirically prove proactive countermeasures counteract efficiently the latest stealthy IFAs. Further, the introduction of Charon opens a totally new perspective on the design of new and/or refinement of old IFA countermeasures.

As third contribution, we present NDN.p4, the first information-centric data-plane written in the novel high-level language for packet processors P4. Writing highly-portable and re-usable implementations of the NDN protocols is a necessary step to advance the experimentation on

the NDN architecture. The work done with NDN.p4 aims to show the foremost importance of exploring programmable network platforms in the design and test of NDN data-plane behaviors. NDN.p4 aims to leverage P4-targets in the long term as platforms to experiment with alternative forwarding mechanisms. Research on alternatives forwarding mechanisms for the NDN is required to design an NDN data-plane preserving its original properties yet avoiding the intrinsic IFA vulnerability. Furthermore, the work in NDN.p4 has highlighted limitations of the first P4 language version and contributed actively to the discussion about the set of features to be integrated in further language versions.

With regard to the future work, the work presented in this manuscript has delineated at least three different directions to pursue further research on. First, the thorough analysis of the state-of-the-art IFA countermeasures has revealed a few promising options to design more robust defense mechanisms against this NDN-specific security threat. Proactiveness, as empirically proven by this work, is a characteristic which can help improve existing defense mechanisms. As purely proactive countermeasures could not be widely applicable to all the attack scenarios, those could instead be embedded in hybrid reactive/proactive designs. This work shows how the IFA mitigation strategy in Charon never affects any legitimate traffic. So, Charon-like mitigation techniques could replace the rate-limiting techniques generally applied by the existing reactive countermeasures.

Second, among the so far proposed countermeasures against IFAs, the collaborative ones seem to be the most efficient according to the results reported in literature works [105, 116, 1]. Nevertheless, the coordination of routers to apply collaborative defense mechanisms is achieved by means of slowly converging message exchanges that may even require data-plane or packet format changes. At the time being, there exist no standard interfaces between control and data-planes in NDN. Yet a standard interface to the NDN data-plane could be used to design centralized at the control-plane applications to mitigate IFAs. The definition of such an interface would eliminate the need for ad-hoc protocol modifications required by the existing collaborative countermeasures. Further, the need for an NDN monitoring plane has been advocated by several other works [171, 172], so there could be the opportunity to partially leverage existing solutions.

Third, the P4 language has considerably evolved since the work in NDN.p4 was proposed. The NDN.p4 design was in some parts constrained by the limitations of the P4 early language specification. Similarly, a study of the performance of the NDN.p4 solution was not done because of the lack of hardware/software support for the P4 language at the time of its conception. Today, the original P4 language specification [2] has been improved and features a more flexible programming model in a new language version [173]. Several P4 hardware/software targets have also appeared, so offering the possibility to conduct a performance evaluation of NDN.p4. The NDN.p4 design can definitely benefit from the new P4 language's feature-set and the related hardware/software ecosystem. Indeed, NDN.p4 has already been leveraged for further work in [174] with the aim to improve its design with regard to certain components (e.g., the parser, the forwarding table) and measure achievable performance on real P4 hardware targets. It is to be ascertained yet whether or not improved versions of NDN.p4 will be able to achieve the same performance of platform-tailored NDN data-plane designs. Nevertheless, in our opinion, this open question should not limit further research on this subject since future NDN.p4 versions will offer immediate portability on multiple targets and a simplified open-source design, benefits which no other state-of-the-art design so far features.

Appendix A

Differences between CCN and NDN

This appendix reports some of the main differences between the CCN and NDN protocols at the time of writing this manuscript. The main criterion for the selection of the presented differences is their relevance to the topics covered in this manuscript.

Packet format:
The CCNx and NDN protocols follow a different encoding and packet format. CCNx uses a 16-bit Type and a 16-bit Length TLV structure. The length field contains the length of the Value field in octets. Instead, NDN uses a variable length Type and Length TLV structure, where the value contained in the first byte of the Type and Length field determines their bit length. The CCNx's packet format is made of a fixed header followed by optional TLV headers and then by the body of the message. Instead, the NDN's packet format does not contain any fixed header rather it contains a variable number of TLVs.

Matching:
Since version 1.0, CCNx only supports exact match on Interest names. Meaning that if a content with name */a/b/c.zip* is required, only a Data packet with the same identical name can consume that Interest. Differently in NDN, contents like */a/b/c.zip/v1/s100* or */a/b/c.zip/manifest.txt* are valid Data for the */a/b/c.zip* Interest. Exact matches achieve faster forwarding and are deterministic, that is, they always bring back the same content. Further, they avoid inspection of caches by means of generic queries.

Selectors:
NDN and CCNx contain additional, sometimes optional, fields in the Interest packet header which can be used to narrow the search to specific contents. NDN offers a wider range of such fields, called selectors. Their purpose ranges from limiting or excluding components beyond the Interest prefix to specifying a specific publisher via a reference to its public key or asking routers not to provide stale Contents. Conversely, CCNx has removed this feature in its 1.0 version to achieve faster and more deterministic forwarding with exact match on the Interest name. Nevertheless, CCNx has preserved two of those options, now called restrictions. CCNx Interests can carry a *keyIdRestriction* field to identify a specific publisher and/or a *contentObjectHash* to guarantee the network delivers the right packet.

Fragmentation:
Contents may be bigger than the Maximum Transmission Unit (MTU) on certain links, so a

mechanism may be needed to fragment and reassemble packets across the network. Fragmentation and reassembly of packets can be performed either in the network or at end-hosts. CCNx performs end-to-end fragmentation, while NDN performs hop-by-hop fragmentation through a link-layer protocol mechanism [175].

Negative Acknowledgments:
Special packets reporting feedback about packet transmission are widely used mechanisms to generate error notifications in network protocols. An analysis, with a strong focus on security, of pros and cons of Negative Acknowledgments (NACKs) in NDN/CCN networks is provided in [134]. Therein, the authors define two kind of NACK packets for ICNs: content NACK (cNACK) and forwarding NACK (fNACK). A cNACK is a negative acknowledgment generated by a producer to indicate that a requested content does not exist. A fNACK is generated by a router to notify downstream routers that it cannot forward an Interest packet because of congestion or absence of route for that name. The work in [134] presents in detail the security implications of NACKs concluding that network-layer NACKs are better avoided for security reasons. In fact, the CCNx protocol provides NACKs only as special type of Content Type and the NDN architecture delegates the NACK mechanism to a link-layer protocol (the NDNLP [175]).

Bibliography

[1] H. Salah, J. Wulfheide, and T. Strufe, "Coordination supports security: A new defence mechanism against interest flooding in NDN," in *40th IEEE Conference on Local Computer Networks*, 2015, pp. 73–81.

[2] The P4_14 Language Specification, version 1.0.3. [Online]. Available: https://p4.org/p4-spec/p4-14/v1.0.3/tex/p4.pdf

[3] D. Clark, "The design philosophy of the darpa internet protocols," vol. 18, no. 4. ACM, 1988, pp. 106–114.

[4] J. H. Saltzer, D. P. Reed, and D. D. Clark, "End-to-end arguments in system design," vol. 2, no. 4. New York, NY, USA: ACM, Nov. 1984, pp. 277–288. [Online]. Available: http://doi.acm.org/10.1145/357401.357402

[5] R. Braden, D. Clark, S. Shenker, and J. Wroclawski, "Developing a next-generation internet architecture," 2000.

[6] A. Feldmann, "Internet clean-slate design: what and why?" vol. 37, no. 3. ACM, 2007, pp. 59–64.

[7] J. Rexford and C. Dovrolis, "Future internet architecture: clean-slate versus evolutionary research," vol. 53, no. 9. ACM, 2010, pp. 36–40.

[8] J. Pan, S. Paul, and R. Jain, "A survey of the research on future internet architectures," vol. 49, no. 7. IEEE, 2011.

[9] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden, "Tussle in cyberspace: defining tomorrow's internet," in *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4. ACM, 2002, pp. 347–356.

[10] M. Waldvogel, G. Varghese, J. Turner, and B. Plattner, "Scalable high speed ip routing lookups," in *Proceedings of the ACM SIGCOMM '97 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '97. New York, NY, USA: ACM, 1997, pp. 25–36. [Online]. Available: http://doi.acm.org/10.1145/263105.263136

[11] "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021 White Paper," Tech. Rep., 02 2017.

[12] C. Perkins, "Ip mobility support for ipv4, revised," Internet Requests for Comments, RFC Editor, RFC 5944, November 2010, http://www.rfc-editor.org/rfc/rfc5944.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5944.txt

[13] A. Feldmann, "Internet clean-slate design: What and why?" vol. 37, no. 3. New York, NY, USA: ACM, Jul. 2007, pp. 59–64. [Online]. Available: http://doi.acm.org/10.1145/1273445.1273453

[14] Mobilityfirst future internet architecture project. [Online]. Available: mobility-first.winlab.rutgers.edu

[15] Xia, expressive internet architecture project. [Online]. Available: https://www.cs.cmu.edu/ xia/

[16] D. Grossman, "New terminology and clarifications for diffserv," Internet Requests for Comments, RFC Editor, RFC 3260, April 2002, http://www.rfc-editor.org/rfc/rfc3260.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3260.txt

[17] 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," vol. 16, no. 2. IEEE, pp. 1024–1049.

[18] The triad project. http://gregorio.stanford.edu/triad/. Accessed: 2017-09-10.

[19] M. Gritter and D. R. Cheriton, "An architecture for content routing support in the internet," in *Proceedings of the 3rd Conference on USENIX Symposium on Internet Technologies and Systems - Volume 3*, ser. USITS'01. Berkeley, CA, USA: USENIX Association, 2001, pp. 4–4. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251440.1251444

[20] The project ccnx at the palo alto research center. [Online]. Available: https://blogs.parc.com/ccnx/

[21] A new way to look at networking, van jacobson at googletechtalks. [Online]. Available: https://www.youtube.com/watch?v=oCZMoY3q2uM

[22] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 2009, pp. 1–12.

[23] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4. ACM, 2007, pp. 181–192.

[24] The content based networking project. [Online]. Available: http://www.inf.usi.ch/carzaniga/cbn/index.html

[25] A. Carzaniga and A. L. Wolf, "Content-based networking: A new communication infrastructure," in *Workshop on Infrastruture for Mobile and Wireless Systems*. Springer, 2001, pp. 59–68.

[26] A. Carzaniga, "Architectures for an event notification service scalable to wide-area networks," Ph.D. dissertation, Citeseer.

[27] A. Carzaniga and A. L. Wolf, "Forwarding in a content-based network," in *Proceedings of ACM SIGCOMM 2003*, Karlsruhe, Germany, Aug. 2003, pp. 163–174.

[28] A. Carzaniga, M. J. Rutherford, and A. L. Wolf, "A routing scheme for content-based networking," in *Proceedings of IEEE INFOCOM 2004*, Hong Kong, China, Mar. 2004.

[29] Publish-subscribe internet routing paradigm. [Online]. Available: http://www.psirp.org/

[30] Scalable and adaptive internet solutions. [Online]. Available: http://www.sail-project.eu/

[31] The named data networking (NDN) project. [Online]. Available: https://named-data.net/

[32] The convergence project. [Online]. Available: http://www.ict-convergence.eu/

[33] The comet project, content mediator architecture for content-aware networks. [Online]. Available: http://www.comet-project.org/

[34] The greenicn project - architecture and applications of green information-centric networking. http://www.greenicn.org/. Accessed: 2018-02-20.

[35] The point project - ip over icn - the better ip. https://www.point-h2020.eu/. Accessed: 2018-02-20.

[36] The bonvoyage project. http://bonvoyage2020.eu/. Accessed: 2018-02-20.

[37] Fia-np: Collaborative research: Named data networking next phase (NDN-np). [Online]. Available: https://www.nsf.gov/awardsearch/showAward?AWD_ID=1345318

[38] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox, "Information-centric networking: Seeing the forest for the trees," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, ser. HotNets-X. New York, NY, USA: ACM, 2011, pp. 1:1–1:6. [Online]. Available: http://doi.acm.org/10.1145/2070562.2070563

[39] V. Almeida, A. Bestavros, M. Crovella, and A. De Oliveira, "Characterizing reference locality in the www," in *Parallel and Distributed Information Systems, 1996., Fourth International Conference on.* IEEE, 1996, pp. 92–103.

[40] M. Demmer, K. Fall, T. Koponen, and S. Shenker, "Towards a modern communications api," in *In HotNets-VI.* Citeseer, 2007.

[41] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," vol. 50, no. 7. IEEE, 2012.

[42] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less pain, most of the gain: Incrementally deployable icn," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: ACM, 2013, pp. 147–158. [Online]. Available: http://doi.acm.org/10.1145/2486001.2486023

[43] C. Ghali, G. Tsudik, C. A. Wood, and E. Yeh, "Practical accounting in content-centric networking," in *Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP.* IEEE, 2016, pp. 436–444.

[44] G. Tyson, N. Sastry, I. Rimac, R. Cuevas, and A. 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.* ACM, 2012, pp. 1–6.

[45] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," vol. 35, no. 2. ACM, 2003, pp. 114–131.

[46] W. Shang, Y. Yu, R. Droms, and L. Zhang, "Challenges in iot networking via tcp/ip architecture."

[47] M. Amadeo, C. Campolo, J. Quevedo, D. Corujo, A. Molinaro, A. Iera, R. L. Aguiar, and A. V. Vasilakos, "Information-centric networking for the internet of things: challenges and opportunities," vol. 30, no. 2. IEEE, 2016, pp. 92–100.

[48] C. Perkins, "Ip mobility support for ipv4, revised," Internet Requests for Comments, RFC Editor, RFC 5944, November 2010, http://www.rfc-editor.org/rfc/rfc5944.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5944.txt

[49] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, "A case for stateful forwarding plane," vol. 36, no. 7. Elsevier, 2013, pp. 779–791.

[50] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," vol. 42, no. 3. New York, NY, USA: ACM, Jun. 2012, pp. 62–67. [Online]. Available: http://doi.acm.org/10.1145/2317307.2317319

[51] A. Chaabane, E. De Cristofaro, M. A. Kaafar, and E. Uzun, "Privacy in content-oriented networking: Threats and countermeasures," vol. 43, no. 3. ACM, 2013, pp. 25–33.

[52] R. Tourani, S. Misra, T. Mick, and G. Panwar, "Security, privacy, and access control in information-centric networking: A survey." IEEE, 2017.

[53] L. Zhang *et al.*, "Named Data Networking (NDN) Project," the NDN project team, Technical Report NDN-0001, October 2010.

[54] Ccnx 1.0 protocol specifications roadmap. [Online]. Available: https://www.ietf.org/mail-archive/web/icnrg/current/pdfZyEQRE5tFS.pdf

[55] Design choices and differences for NDN and ccnx 1.0 implementations of information-centric networking. [Online]. Available: https://icnrg.github.io/draft-icnrg-harmonization/draft-icnrg-harmonization-00.html

[56] marc.mosko@parc.com, "Ccnx semantics," Working Draft, IETF Secretariat, Internet-Draft draft-irtf-icnrg-ccnxsemantics-00, June 2015, http://www.ietf.org/internet-drafts/draft-irtf-icnrg-ccnxsemantics-00.txt. [Online]. Available: http://www.ietf.org/internet-drafts/draft-irtf-icnrg-ccnxsemantics-00.txt

[57] ——, "Ccnx messages in tlv format," Working Draft, IETF Secretariat, Internet-Draft draft-irtf-icnrg-ccnxmessages-00, June 2015, http://www.ietf.org/internet-drafts/draft-irtf-icnrg-ccnxmessages-00.txt. [Online]. Available: http://www.ietf.org/internet-drafts/draft-irtf-icnrg-ccnxmessages-00.txt

[58] Cisco announces important steps toward adoption of information-centric networking. [Online]. Available: https://blogs.cisco.com/sp/cisco-announces-important-steps-toward-adoption-of-information-centric-networking

[59] The community icn (cicn) project. https://wiki.fd.io/view/Cicn. Accessed: 2018-04-20.

[60] A. Afanasyev, R. Ravindran, G. Wang, L. Wang, B. Zhang, and L. Zhang, "Icn packet format design requirements," Working Draft, IETF Secretariat, Internet-Draft

draft-icn-packet-format-requirements-01, March 2015, https://tools.ietf.org/id/draft-icn-packet-format-requirements-01.txt. [Online]. Available: https://tools.ietf.org/html/draft-icn-packet-format-requirements-01

[61] NDN packet format specification 0.2-2 documentation. [Online]. Available: https://named-data.net/doc/ndn-tlv/

[62] C. Yi, J. Abraham, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "On the role of routing in named data networking," in *Proceedings of the 1st ACM Conference on Information-Centric Networking*, ser. ACM-ICN '14. New York, NY, USA: ACM, 2014, pp. 27–36. [Online]. Available: http://doi.acm.org/10.1145/2660129.2660140

[63] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, "DoS and DDoS in named-data networking," vol. abs/1208.0952, 2012.

[64] C. Ghali, G. Tsudik, and E. Uzun, "Needle in a haystack: Mitigating content poisoning in named-data networking," 2014.

[65] A. S. Tanenbaum and D. J. Wetherall, *Computer Networks (5th Edition)*. Prentice Hall, 2011.

[66] Y. Ren, J. Li, S. Shi, L. Li, G. Wang, and B. Zhang, "Congestion control in named data networking–a survey," vol. 86. Elsevier, 2016, pp. 1–11.

[67] I. Moiseenko and L. Zhang, "Consumer-Producer API for Named Data Networking," the NDN project team, Technical Report NDN-0017, august 2014.

[68] ——, "Consumer-producer api for named data networking," in *Proceedings of the 1st international conference on Information-centric networking*. ACM, 2014, pp. 177–178.

[69] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. Thorton, D. K. Smetters, B. Zhang, G. Tsudik, K. Claffy, D. Krioukov, D. Massey, C. Papadopoulos, T. Abdelzaher, L. Wang, P. Crowley, and E. Yeh, "Named Data Networking Tech Report 001," the NDN project team, Technical Report NDN-0001, October 2010.

[70] W. Shang, Q. Ding, A. Marianantoni, J. Burke, and L. Zhang, "Securing building management systems using named data networking," vol. 28, no. 3. IEEE, 2014, pp. 50–56.

[71] Open mhealth, the first and only open standard for mobile health data. [Online]. Available: http://www.openmhealth.org/

[72] Y. Zhang, H. Zhang, and L. Zhang, "Kite: a mobility support scheme for NDN," in *Proceedings of the 1st ACM Conference on Information-Centric Networking*. ACM, 2014, pp. 179–180.

[73] Y. Yingdi, A. Afanasyev, and L. Zhang, "Name-Based Access Control," the NDN project team, Technical Report NDN-0034, January 2016.

[74] A. Afanasyev, X. Jiang, Y. Yu, J. Tan, Y. Xia, A. Mankin, and L. Zhang, "NDNs: A dns-like name service for NDN," in *Computer Communication and Networks (ICCCN), 2017 26th International Conference on*. IEEE, 2017, pp. 1–9.

[75] N. Feamster, J. Rexford, and E. Zegura, "The road to sdn: an intellectual history of programmable networks," vol. 44, no. 2. ACM, 2014, pp. 87–98.

[76] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," vol. 38, no. 2. ACM, 2008, pp. 69–74.

[77] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," vol. 103, no. 1. Ieee, 2015, pp. 14–76.

[78] L. Veltri, G. Morabito, S. Salsano, N. Blefari-Melazzi, and A. Detti, "Supporting information-centric functionality in software defined networks," in *Communications (ICC), 2012 IEEE International Conference on*. IEEE, 2012, pp. 6645–6650.

[79] M. Vahlenkamp, F. Schneider, D. Kutscher, and J. Seedorf, "Enabling information centric networking in ip networks using sdn," in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. IEEE, 2013, pp. 1–6.

[80] N. B. Melazzi, A. Detti, G. Mazza, G. Morabito, S. Salsano, and L. Veltri, "An openflow-based testbed for information centric networking," in *Future Network & Mobile Summit (FutureNetw), 2012*. IEEE, 2012, pp. 1–9.

[81] P. Bosshart, G. Gibb, H.-S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, and M. Horowitz, "Forwarding metamorphosis: Fast programmable match-action processing in hardware for sdn," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: ACM, 2013, pp. 99–110.

[82] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming protocol-independent packet processors," vol. 44, no. 3. New York, NY, USA: ACM, Jul. 2014, pp. 87–95.

[83] S. Signorello, J. François, O. Festor, and R. State, "NDN.p4: Programming information-centric data-planes," in *Proceedings of the IEEE International Workshop on Open-Source Software Networking at NetSoft2016*, 2016.

[84] J. Shi, T. Liang, H. Wu, B. Liu, and B. Zhang, "NDN-nic: Name-based filtering on network interface card," in *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking*. ACM, 2016, pp. 40–49.

[85] H. Yuan, T. Song, and P. Crowley, "Scalable NDN forwarding: Concepts, issues and principles," in *Computer Communications and Networks (ICCCN), 2012 21st International Conference on*. IEEE, 2012, pp. 1–9.

[86] H. Yuan and P. Crowley, "Scalable pending interest table design: From principles to practice," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 2049–2057.

[87] G. Piro, S. Signorello, M. R. Palattella, L. A. Grieco, G. Boggia, and T. Engel, "Understanding the social impact of icn: between myth and reality," vol. 32, no. 3. Springer, 2017, pp. 401–419.

[88] H. Dai, B. Liu, Y. Chen, and Y. Wang, "On pending interest table in named data networking," in *Proceedings of the eighth ACM/IEEE symposium on Architectures for networking and communications systems*. ACM, 2012, pp. 211–222.

[89] Y. Wang, K. He, H. Dai, W. Meng, J. Jiang, B. Liu, and Y. Chen, "Scalable name lookup in NDN using effective name component encoding," in *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on.* IEEE, 2012, pp. 688–697.

[90] E. Fredkin, "Trie memory," vol. 3, no. 9. New York, NY, USA: ACM, Sep. 1960, pp. 490–499. [Online]. Available: http://doi.acm.org/10.1145/367390.367400

[91] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," vol. 13, no. 7. ACM, 1970, pp. 422–426.

[92] A. Kirsch, M. Mitzenmacher, and G. Varghese, "Hash-based techniques for high-speed packet processing," in *Algorithms for Next Generation Networks.* Springer, 2010, pp. 181–218.

[93] S. Dharmapurikar, P. Krishnamurthy, and D. E. Taylor, "Longest prefix matching using bloom filters," vol. 14, no. 2. IEEE, 2006, pp. 397–409.

[94] Z. Li, J. Bi, S. Wang, and X. Jiang, "Compression of pending interest table with bloom filter in content centric network," in *Proceedings of the 7th International Conference on Future Internet Technologies.* ACM, 2012, pp. 46–46.

[95] W. You, B. Mathieu, P. Truong, J.-F. Peltier, and G. Simon, "Dipit: A distributed bloom-filter based pit table for ccn nodes," in *Computer Communications and Networks (ICCCN), 2012 21st International Conference on.* IEEE, 2012, pp. 1–7.

[96] Z. Li, K. Liu, Y. Zhao, and Y. Ma, "Mapit: an enhanced pending interest table for NDN with mapping bloom filter," vol. 18, no. 11. IEEE, 2014, pp. 1915–1918.

[97] M. Varvello, D. Perino, and L. Linguaglossa, "On the design and implementation of a wire-speed pending interest table," in *Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on.* IEEE, 2013, pp. 369–374.

[98] W. So, A. Narayanan, and D. Oran, "Named data networking on a router: Fast and dos-resistant forwarding with hash tables," in *Proceedings of the Ninth ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, ser. ANCS '13. Piscataway, NJ, USA: IEEE Press, 2013, pp. 215–226. [Online]. Available: http://dl.acm.org/citation.cfm?id=2537857.2537892

[99] D. Perino and M. Varvello, "A reality check for content centric networking," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking.* ACM, 2011, pp. 44–49.

[100] D. Kirchner, R. Ferdous, R. L. Cigno, L. Maccari, M. Gallo, D. Perino, and L. Saino, "Augustus: a ccn router for programmable networks," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking.* ACM, 2016, pp. 31–39.

[101] J. Takemasa, Y. Koizumi, and T. Hasegawa, "Toward an ideal NDN router on a commercial off-the-shelf computer," in *Proceedings of the 4th ACM Conference on Information-Centric Networking.* ACM, 2017, pp. 43–53.

[102] F. Bonomi, M. Mitzenmacher, R. Panigrahy, S. Singh, and G. Varghese, "An improved construction for counting bloom filters," in *Proceedings of the 14th conference on Annual European Symposium-Volume 14.* Springer-Verlag, 2006, pp. 684–695.

[103] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino, "Pending interest table sizing in named data networking," in *Proceedings of the 2nd International Conference on Information-Centric Networking*. ACM, 2015, pp. 49–58.

[104] M. Wählisch, T. C. Schmidt, and M. Vahlenkamp, "Backscatter from the data plane–threats to stability and security in information-centric network infrastructure," vol. 57, no. 16. Elsevier, 2013, pp. 3192–3206.

[105] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang, "Interest flooding attack and countermeasures in named data networking," in *IFIP Networking Conference, 2013, Brooklyn, New York, USA, 22-24 May, 2013*, 2013, pp. 1–9.

[106] M. Wählisch, T. C. Schmidt, and M. Vahlenkamp, "Backscatter from the data plane — threats to stability and security in Information-Centric Networking," vol. abs/1205.4778, 2012.

[107] A. Compagno, M. Conti, P. Gasti, and G. Tsudik, "NDN interest flooding attacks and countermeasures," in *Annual Computer Security Applications Conference*, 2012.

[108] C. Ghali, G. Tsudik, E. Uzun, and C. A. Wood, "Living in a pit-less world: A case against stateful forwarding in content-centric networking," 2015.

[109] M. Mahdian, S. Arianfar, J. Gibson, and D. Oran, "Mircc: Multipath-aware icn rate-based congestion control," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*. ACM, 2016, pp. 1–10.

[110] Y. Wang, N. Rozhnova, A. Narayanan, D. Oran, and I. Rhee, "An improved hop-by-hop interest shaper for congestion control in named data networking," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 55–60.

[111] C. Labovitz, G. R. Malan, and F. Jahanian, "Internet routing instability," vol. 6, no. 5. Piscataway, NJ, USA: IEEE Press, Oct. 1998, pp. 515–528. [Online]. Available: http://dx.doi.org/10.1109/90.731185

[112] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, "DoS and DDoS in named data networking," in *Proceeding of the 22nd IEEE International Conference on Computer Communications and Networks, ICCCN*, 2013, pp. 1–7.

[113] S. Signorello, S. Marchal, J. François, O. Festor, and R. State, "Advanced interest flooding attacks in named-data networking," in *Proceedings of the 16th IEEE International Symposium on Network Computing and Applications*. IEEE, 2017.

[114] H. Salah and T. Strufe, "Evaluating and mitigating a collusive version of the interest flooding attack in NDN," in *Proceedings of the IEEE Symposium on Computers and Communication, ISCC-16*. Messina, Italy: IEEE Computer Society, 2016, pp. 938–945.

[115] A. Alston and T. Refaei, "Neutralizing interest flooding attacks in named data networks using cryptographic route tokens," in *2016 IEEE 15th International Symposium on Network Computing and Applications (NCA)*, Oct 2016, pp. 85–88.

[116] A. Compagno, M. Conti, P. Gasti, and G. Tsudik, "Poseidon: Mitigating interest flooding DDoS attacks in named data networking," in *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*. IEEE, 2013, pp. 630–638.

[117] X. Marchal, T. Cholez, and O. Festor, "Server-side performance evaluation of NDN," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, ser. ACM-ICN '16. New York, NY, USA: ACM, 2016, pp. 148–153.

[118] K. Wang, H. Zhou, H. Luo, J. Guan, Y. Qin, and H. Zhang, "Detecting and mitigating interest flooding attacks in content-centric network," vol. 7, no. 4, 2014, pp. 685–699.

[119] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, k. claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," vol. 44, no. 3. New York, NY, USA: ACM, Jul. 2014, pp. 66–73.

[120] J. Tang, Z. Zhang, Y. Liu, and H. Zhang, "Identifying interest flooding in named data networking," in *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*. IEEE, 2013, pp. 306–310.

[121] K. Wang, H. Zhou, Y. Qin, J. Chen, and H. Zhang, "Decoupling malicious interests from pending interest table to mitigate interest flooding attacks," in *Globecom Workshops (GC Wkshps), 2013 IEEE*. IEEE, 2013, pp. 963–968.

[122] H. Dai, Y. Wang, J. Fan, and B. Liu, "Mitigate DDoS attacks in NDN by interest traceback," in *IEEE INFOCOM NOMEN Workshop, 2013*, 2013.

[123] K. Wang, H. Zhou, Y. Qin, and H. Zhang, "Cooperative-filter: countering interest flooding attacks in named data networking," vol. 18, no. 9. Springer, 2014, pp. 1803–1813.

[124] T. N. Nguyen, R. Cogranne, and G. Doyen, "An optimal statistical test for robust detection against interest flooding attacks in ccn," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 2015, pp. 252–260.

[125] T. N. Nguyen, R. Cogranne, G. Doyen, and F. Retraint, "Detection of interest flooding attacks in named data networking using hypothesis testing," in *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*. IEEE, 2015, pp. 1–6.

[126] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," vol. 32, no. 3. ACM, 2002, pp. 62–73.

[127] J. Ioannidis and S. M. Bellovin, "Implementing pushback: Router-based defense against ddos attacks."

[128] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," the NDN project team, Technical Report NDN-0005, October 2012.

[129] N. Spring, R. Mahajan, and D. Wetherall, "Measuring isp topologies with rocketfuel," vol. 32, no. 4. ACM, 2002, pp. 133–145.

[130] Wikimedia downloads. https://dumps.wikimedia.org/. Accessed: 2017-03-04.

[131] S. Marchal, J. François, C. Wagner, and T. Engel, "Semantic exploration of dns," in *Proceeding of the 11th International IFIP TC 6 Networking Conference*. Springer Berlin Heidelberg, 2012, pp. 370–384.

[132] P. Kolb, "DISCO: A Multilingual Database of Distributionally Similar Words," in *KONVENS 2008 – Ergänzungsband: Textressourcen und lexikalisches Wissen*, A. Storrer, A. Geyken, A. Siebert, and K.-M. Würzner, Eds., 2008, pp. 37–44.

[133] "NDN-ifa signorello's github repository," https://github.com/signorello/NDN-IFA, accessed: 2017-08-31.

[134] A. Compagno, M. Conti, C. Ghali, and G. Tsudik, "To nack or not to nack? negative acknowledgments in information-centric networking," in *Computer Communication and Networks (ICCCN), 2015 24th International Conference on*. IEEE, 2015, pp. 1–10.

[135] Z. Li and J. Bi, "Interest cash: an application-based countermeasure against interest flooding for dynamic content in named data networking," in *Proceedings of The Ninth International Conference on Future Internet Technologies*. ACM, 2014, p. 2.

[136] S. Al-Sheikh, M. Wählisch, and T. C. Schmidt, "Revisiting countermeasures against ndn interest flooding," in *Proceedings of the 2nd ACM Conference on Information-Centric Networking*. ACM, 2015, pp. 195–196.

[137] "Bloom filter calculator," https://hur.st/bloomfilter, accessed: 2017-08-31.

[138] R. Zhang, J. Liu, T. Huang, T. Pan, and L. Wu, "Adaptive compression trie based bloom filter: Request filter for NDN content store," vol. 5, 2017, pp. 23 647–23 656.

[139] M. Mitzenmacher, "Compressed bloom filters," vol. 10, no. 5. IEEE, 2002, pp. 604–612.

[140] C. Tsilopoulos, G. Xylomenos, and Y. Thomas, "Reducing forwarding state in content-centric networks with semi-stateless forwarding," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 2067–2075.

[141] X. Wang, W. Wang, C. Zeng, R. Dai, S. Wang, and S. Xu, "Reducing the size of pending interest table for content-centric networks with hybrid forwarding," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[142] J. Garcia-Luna-Aceves and M. M. Barijough, "Content-centric networking using anonymous datagrams," in *IFIP Networking Conference (IFIP Networking) and Workshops, 2016*. IEEE, 2016, pp. 171–179.

[143] C. Ghali, G. Tsudik, E. Uzun, and C. A. Wood, "Closing the floodgate with stateless content-centric networking," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, July 2017, pp. 1–10.

[144] G. Carofiglio, M. Gallo, and L. Muscariello, "Optimal multipath congestion control and request forwarding in information-centric networks: Protocol design and experimentation," vol. 110. Elsevier, 2016, pp. 104–117.

[145] W. John, M. Dusi, and K. C. Claffy, "Estimating routing symmetry on single links by passive flow measurements," in *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*. ACM, 2010, pp. 473–478.

[146] A. Dabirmoghaddam, M. Dehghan, and J. Garcia-Luna-Aceves, "Characterizing interest aggregation in content-centric networks," in *IFIP Networking Conference (IFIP Networking) and Workshops, 2016*. IEEE, 2016, pp. 449–457.

[147] S. Shannigrahi, C. Fan, and C. Papadopoulos, "Request aggregation, caching, and forwarding strategies for improving large climate data distribution with NDN: a case study," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*. ACM, 2017, pp. 54–65.

[148] A. Detti, N. Blefari Melazzi, S. Salsano, and M. Pomposini, "Conet: a content centric inter-networking architecture," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*. ACM, 2011, pp. 50–55.

[149] P. Jokela, A. Zahemszky, C. Esteve Rothenberg, S. Arianfar, and P. Nikander, "Lipsin: line speed publish/subscribe inter-networking," vol. 39, no. 4. ACM, 2009, pp. 195–206.

[150] L. Subramanian, V. N. Padmanabhan, and R. H. Katz, "Geographic properties of internet routing," in *Proceedings of the General Track of the Annual Conference on USENIX Annual Technical Conference*, ser. ATEC '02. Berkeley, CA, USA: USENIX Association, 2002, pp. 243–259. [Online]. Available: http://dl.acm.org/citation.cfm?id=647057.713857

[151] J. Garcia-Luna-Aceves, "A fault-tolerant forwarding strategy for interest-based information centric networks," in *IFIP Networking Conference (IFIP Networking), 2015*. IEEE, 2015, pp. 1–9.

[152] ——, "A more scalable approach to content centric networking," in *Computer Communication and Networks (ICCCN), 2015 24th International Conference on*. IEEE, 2015, pp. 1–8.

[153] J. Aweya, "On the design of ip routers part 1: Router architectures," vol. 46, no. 6. Elsevier, 2000, pp. 483–511.

[154] G. Brebner and W. Jiang, "High-speed packet processing using reconfigurable computing," vol. 34, no. 1. Los Alamitos, CA, USA: IEEE Computer Society, 2014, pp. 8–18.

[155] S. Chole, A. Fingerhut, S. Ma, A. Sivaraman, S. Vargaftik, A. Berger, G. Mendelson, M. Alizadeh, S.-T. Chuang, I. Keslassy, A. Orda, and T. Edsall, "drmt: Disaggregated programmable switching," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '17. New York, NY, USA: ACM, 2017, pp. 1–14. [Online]. Available: http://doi.acm.org/10.1145/3098822.3098823

[156] H. Yuan and P. Crowley, "Reliably scalable name prefix lookup," in *Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for networking and communications systems, ANCS 2015, Oakland, CA, USA, May 7-8, 2015*, 2015, pp. 111–121.

[157] R. B. Mansilha, L. Saino, M. P. Barcellos, M. Gallo, E. Leonardi, D. Perino, and D. Rossi, "Hierarchical content stores in high-speed icn routers: Emulation and prototype implementation," in *Proceedings of the 2Nd International Conference on Information-Centric Networking*, ser. ICN '15. New York, NY, USA: ACM, 2015, pp. 59–68.

[158] M. Varvello, D. Perino, and J. Esteban, "Caesar: A content router for high speed forwarding," in *Proceedings of the Second Edition of the ICN Workshop on Information-centric Networking*, ser. ICN '12. New York, NY, USA: ACM, 2012, pp. 73–78.

[159] G. Varghese, "Life in the fast lane: Viewed from the confluence lens," vol. 45, no. 1. ACM, 2015, pp. 19–25.

[160] The P4 Language Consortium Website. [Online]. Available: http://www.p4.org/

[161] Intel Ethernet Switch Silicon FM6000. [Online]. Available: http://www.intel.com/content/www/us/en/switch-silicon/ethernet-switch-fm6000-series-brief.html

[162] The p4 software switch a.k.a. behavioral model. [Online]. Available: https://github.com/p4lang/behavioral-model

[163] Mini-NDN, a ligthweight networking evolution tool that enables testing, experimentation and research on the NDN platform. [Online]. Available: https://github.com/named-data/mini-ndn

[164] J. Postel, "Internet protocol," Internet Requests for Comments, RFC Editor, STD 5, September 1981, http://www.rfc-editor.org/rfc/rfc791.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc791.txt

[165] S. E. Deering and R. M. Hinden, "Internet protocol, version 6 (ipv6) specification," Internet Requests for Comments, RFC Editor, RFC 2460, December 1998, http://www.rfc-editor.org/rfc/rfc2460.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2460.txt

[166] Y. Rekhter, T. Li, and S. Hares, "A border gateway protocol 4 (bgp-4)," Internet Requests for Comments, RFC Editor, RFC 4271, January 2006, http://www.rfc-editor.org/rfc/rfc4271.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4271.txt

[167] P4-dev mailing list, "filling tables" thread. [Online]. Available: http://lists.p4.org/pipermail/p4-dev_lists.p4.org/2015-July/000056.html

[168] G. Bianchi, M. Bonola, A. Capone, and C. Cascone, "Openstate: programming platform-independent stateful openflow applications inside the switch," vol. 44, no. 2. ACM, 2014, pp. 44–51.

[169] P4-dev mailing list, "move parser pointer forward" thread. [Online]. Available: http://lists.p4.org/pipermail/p4-dev_lists.p4.org/2015-August/000085.html

[170] S. Kent and K. Seo, "Security architecture for the internet protocol," Internet Requests for Comments, RFC Editor, RFC 4301, December 2005, http://www.rfc-editor.org/rfc/rfc4301.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4301.txt

[171] D. Pesavento, O. I. E. Mimouni, E. Newberry, L. Benmohamed, and A. Battou, "A network measurement framework for named data networks," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ser. ICN '17. New York, NY, USA: ACM, 2017, pp. 200–201. [Online]. Available: http://doi.acm.org/10.1145/3125719.3132113

[172] H. L. e. a. Mai, "Towards a security monitoring plane for named data networking and its application against content poisoning attack," to appear in 2018 IFIP/IEEE Network Operation and Management Symposium (NOMS). IEEE, 2018.

[173] The P4_16 Language Specification, version 1.0.0. [Online]. Available: https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.pdf

[174] R. M. C. Ramos, "Named data networking with programmable switches," Master's thesis, The university of Lisbon, 2017.

[175] J. Shi and B. Zhang, "NDNLP: A Link Protocol for NDN," the NDN project team, Technical Report NDN-0006, July 2012.

Résumé

Ce travail illustre comment les tendances actuelles d'utilisation dominant sur Internet motivent la recherche sur des architectures futures de réseau plus orientées vers le contenu. Parmi les architectures émergentes pour l'Internet du futur, le paradigme du Information-Centric Networking (ICN) est présenté. ICN vise à redéfinir les protocoles de base d'Internet afin d'y introduire un changement sémantique des hôtes vers les contenus. Parmi les architectures ICN, le Named-Data Networking (NDN) prévoit que les demandes nommées de contenus des utilisateurs soient transmises par leur nom dans les routeurs le long du chemin d'un consommateur à une ou plusieurs sources de contenus. Ces demandes de contenus laissent des traces dans les routeurs traversés qui sont ensuite suivis par les paquets de contenus demandés. La table d'intérêt en attente (PIT) est le composant du plan de données de l'NDN qui enregistre temporaraiment les demandes de contenus acheminés dans les routeurs. D'une part, ce travail explique que le mécanisme d'acheminement à états de la PIT permet des propriétés comme l'agrégation de requêtes, le multicast de réponses et le contrôle natif de flux hop-by-hop. D'autre part, ce travail illustre comment l'acheminement à états de la PIT peut facilement être mal utilisé par des attaquants pour monter des attaques de déni de service distribué (DDoS) disruptives, appelées Interest Flooding Attacks (IFAs). Dans les IFAs, des botnets vaguement coordonnés peuvent inonder le réseau d'une grande quantité de demandes difficiles à satisfaire dans le but de surcharger soit l'infrastructure du réseau soit les producteurs de contenus. Ce travail de thèse prouve que bien que des contre-mesures contre les IFAs aient été proposées, il manque une compréhension complète de leur efficacité réelle puisque celles-ci ont été testées sous des hypothèses simplistes sur les scénarios d'évaluation. Dans l'ensemble, le travail présenté dans ce manuscript permet de mieux comprendre les implications des IFAs et les opportunités d'ameliorer les mécanismes de défense existants contre ces attaques. Les principales contributions de ce travail de thèse tournent autour d'une analyse de sécurité du plan d'acheminement dans l'architecture NDN. En particulier, ce travail définit un modèle d'attaquant plus robuste pour les IFAs à travers l'identification des failles dans les contre-mesures IFA existantes. Ce travail introduit un nouvel ensemble d'IFAs basé sur le modèle d'attaquant proposé. Les nouveaux IFAs sont utilisés pour réévaluer les plus efficaces contre-mesures IFA existantes. Les résultats de cette évaluation réfutent l'efficacité universelle des mécanismes de défense existants contre l'IFA et, par conséquent, appellent à différentes contre-mesures pour protéger le NDN contre cette menace de sécurité. Pour surmonter le problème révélé, ce travail définit également des contre-mesures proactives contre l'IFA, qui sont de nouveaux mécanismes de défense contre les IFA inspirés par les problèmes rencontrés dans l'état de l'art. Ce travail présente Charon, une nouvelle contre-mesure proactive contre l'IFA, et là teste contre les nouvelles IFAs. Ce travail montre que Charon est plus efficace que les contre-mesures IFA réactives existantes. Enfin, ce travail illustre la conception NDN.p4, c'est-à-dire la première implémentation d'un protocole ICN écrit dans le langage de haut niveau pour les processeurs de paquets P4. Le travail NDN.p4 est la première tentative dans la littérature visant à tirer parti des nouveaux techniques de réseaux programmables pour tester et évaluer différentes conceptions de plan de données NDN. Cette dernière contribution classe également les mécanismes alternatifs d'acheminement par rapport à un ensemble de propriétés cardinales de la PIT. Le travail souligne qu'il vaut la peine d'explorer d'autres mécanismes d'acheminement visant à concevoir un plan de données NDN moins vulnerable à la menace IFA.

Mots-clés: Information-Centric Networking, Named-Data Networking, securité, déni de service, Interest Flooding Attack

Abstract

This work illustrates how today's Internet dominant usage trends motivate research on more content-oriented future network architectures. Among the emerging future Internet proposals, the promising Information-Centric Networking (ICN) research paradigm is presented. ICN aims to redesign Internet's core protocols to promote a shift in focus from hosts to contents. Among the ICN architectures, the Named-Data Networking (NDN) envisions users' named content requests to be forwarded by their names in routers along the path from one consumer to 1-or-many sources. NDN's requests leave trails in traversed routers which are then followed backwards by the requested contents. The Pending Interest Table (PIT) is the NDN's data-plane component which temporarily records forwarded content requests in routers. On one hand, this work explains that the PIT stateful mechanism enables properties like requests aggregation, multicast responses delivery and native hop-by-hop control flow. On the other hand, this work illustrates how the PIT stateful forwarding behavior can be easily abused by malicious users to mount disruptive distributed denial of service attacks (DDoS), named Interest Flooding Attacks (IFAs). In IFAs, loosely coordinated botnets can flood the network with a large amount of hard to satisfy requests with the aim to overload both the network infrastructure and the content producers. This work proves that although countermeasures against IFAs have been proposed, a fair understanding of their real efficacy is missing since those have been tested under simplistic assumptions about the evaluation scenarios. Overall, the work presented in this manuscript shapes a better understanding of both the implications of IFAs and the possibilities of improving the state-of-the-art defense mechanisms against these attacks. The main contributions of this work revolves around a security analysis of the NDN's forwarding plane. In particular, this work defines a more robust attacker model for IFAs by identifying flaws in the state-of-the-art IFA countermeasures. This work introduces a new set of IFAs built upon the proposed attacker model. The novel IFAs are used to re-assess the most effective existing IFA countermeasures. Results of this evaluation disproves the universal efficacy of the state-of-the-art IFA defense mechanisms and so, call for different countermeasures to protect the NDN against this threat. To overcome the revealed issue, this work also defines proactive IFA countermeasures, which are novel defense mechanisms against IFAs inspired by the issues with the state-of-the-art ones. This work introduces Charon, a novel proactive IFA countermeasure, and tests it against the novel IFA attacks. This work shows Charon counteracts latest stealthy IFAs better than the state-of-the-art reactive countermeasures. Finally, this work illustrates the NDN.p4 design, that is, the first implementation of an ICN protocol written in the high-level language for packet processors P4. The NDN.p4 work is the first attempt in the related literature to leverage novel programmable-networks technologies to test and evaluate different NDN forwarding plane designs. This last contribution also classifies existing alternative forwarding mechanisms with respect to a set of PIT cardinal properties. The work outlines that it is worth to explore alternative forwarding mechanisms aiming to design an NDN forwarding plane less vulnerable to the IFA threat.

Keywords: Information-Centric Networking, Named-Data Networking, security, Denial of Service, Interest Flooding Attack.