ICN Communication Optimization for Internet of Things

Boubakr Nour

ICN Communication Optimization for Internet of Things

Candidate Name:	Boubakr Nour
School or Department:	School of Computer Science
Faculty Mentor:	Prof. Fan Li
Chair, Thesis Committee:	Prof. Liehuang Zhu
Degree Applied:	Doctor of Engineering
Major:	Computer Science and Technology
Degree by:	Beijing Institute of Technology
The Date of Defence:	June 23rd, 2020

中图分类号： TQ028.1

UDC分类号： 540

物联网下信息中心网络（ICN）的通信优化

作 者 姓 名 Boubakr Nour

学 院 名 称 计算机学院

指 导 教 师 李凡教授

答辩委员会主席 祝烈煌教授

申 请 学 位 工学博士

学 科 专 业 计算机科学与技术

学 位 授 予 单 位 北京理工大学

论 文 答 辩 日 期 2020 年 06 月

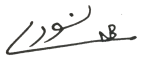
物联网下信息中心网络的通信优化

北京理工大学

研究成果声明

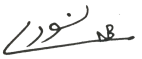
本人郑重声明：所提交的学位论文是我本人在指导教师的指导下进行的研究工作获得的研究成果。尽我所知，文中除特别标注和致谢的地方外，学位论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京理工大学或其它教育机构的学位或证书所使用过的材料。与我一同工作的合作者对此研究工作所做的任何贡献均已在学位论文中作了明确的说明并表示了谢意。

特此申明。

作者签名：  签字日期：24-06-2020

关于学位论文使用权的说明

本人完全了解北京理工大学有关保管、使用学位论文的规定，其中包括：① 学校有权保管、并向有关部门送交学位论文的原件与复印件；② 学校可以采用影印、缩印或其它复制手段复制并保存学位论文；③ 学校可允许学位论文被查阅或借阅；④ 学校可以学术交流为目的, 复制赠送和交换学位论文；⑤ 学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

作者签名：  导师签名： 

签字日期：24-06-2020 签字日期：6/24/2020

摘要

在过去的十几年中，物联网（IoT）受到了业界和学术界的广泛关注。万物与互联网的连通性开辟了大量新的研究方向与发展空间。但是，以主机为中心的通信仍然是目前互联网的最主要通信方式。随着网络相关产业的发展，基于主机的通信的性能已经到了瓶颈期，同时新的通信模型也正在逐步研究和发展。信息中心网络（ICN）作为未来 Internet 体系结构的主流模型，其特点是以数据内容为基本单位，并解耦了内容与位置的关系。ICN 在其架构中提供了基于内容的命名系统、网络内数据缓存、内置的移动性支持、多播支持和基于内容的安全性，而不需要额外的扩展组件。另一方面，大多数基于物联网的应用程序也遵循以内容为中心的通信范式。物联网用户往往请求或者产生特定的数据内容，而不是为了与特定的主机或设备进行通信，例如某个传感器探索到的数值，某一病人在家的观测数据。因此，ICN 可通过改善网络的性能、可扩展性、安全性、移动性以及优化设备的能耗来实现大规模的物联网网络部署。近年来，大量研究致力于将 ICN 集成到物联网中。

本文研究并探索了将 ICN 用作物联网应用通信支持者的可行性。首先，本文调研了当前主要的基于物联网的 ICN 研究成果，并提供了详细且系统的介绍。此外，本文分析了将 ICN 集成到物联网领域的一系列挑战，并将这些挑战分为两大类：网络通信和网络管理。例如，网络通信模块包括 ICN 架构的主要元素——内容命名、物联网应用中的主导通信模型——发布-订阅通信以及 ICN 网络中的架构基石——内容缓存。对应的，网络控制协议则可归为网络管理这个类别之中。

本论文的主要贡献和创新之处如下：

（1）为了解决物联网应用程序中 ICN 内容命名问题，例如识别内容/ 服务/ 设备，生成较长和无限制的内容名称，并提供高效率的聚合规则，本文提出了一种集成混合式 ICN 命名方案。该方案采用前缀标记法来描述分层位置名称并使用属性值设计的方法来提供可变长的编码方法，以较短的名称长度来嵌套较多的语义值。此外，该方案支持使用本地物联网通信名称到编码的快速转换来解决较长名称前缀的问题，并支持使用较小的数据包开销来实现多源数据内容检索与获取的功能。实验结果表明，该方案优于当前的 ICN 命名解决方法，并显著降低了内存消耗，查找时间，路由和转发开销，提高整个物联网通信效率。

（2）为了支持 ICN 中的发布-订阅通信模型，本文提出了一种基于组的订阅方案。

该方案不仅实现了无缝的发布者-订阅者模型，而且还实现了身份验证、访问控制和组管理等功能。在不违反 ICN 请求-响应一一对应的原始设计架构基础之上，本文在中间节点中通过设置半持久化数据包和定制化的路由表来提供一个可扩展的订阅管理方案。同时，该方案采用分层的逻辑密钥机制小号来有效地分发授权用户之间的安全密钥和共享内容。相比传统的基于拉形式的订阅方法，该方案在使用较低的控制开销的同时，增加了安全性和并支持隐私保护功能。性能分析表明，半永久性地存储 Interest 数据包可以将核心节点的内存需求保持在最低水平。

（3）针对于物联网应用高效的内容缓存问题，本文提出了一个结合了集中式和分布式的网络缓存机制。集中式缓存方案旨在选择最佳合适的缓存位置，以最小化内容从内容生产者到缓存节点和内容从缓存节点到内容消费者的传输开销，同时需要最小化中间缓存节点的缓存开销。该方案基于启发式的思想，采用“高请求优先，远距离置后”的思想来选择缓存节点。该算法综合考虑了内容的请求需求，节点的缓存空间、用户距离、用户数量等因素。分布式方案则基于“下放流行，上升冷门”的思想，旨在将频繁访问的内容压入边缘网络，并将请求频率较低的内容置于核心网络中。实验结果表明，该算法在网络延迟、跳数减少和缓存利用率等方面相对于现有的缓存策略都展现了较大的提升。

（4）为了设计与 ICN 网络架构原理兼容的控制协议，本文提出了一种专门用于命名数据网络的控制协议，该协议可以传送不同的网络故障、信息、通知和服务消息。该协议使用内容名称，而不是主机地址作为标识，很好地兼容了 Interest-Data 的交换模型和 ICN 转发平面，并支持 Interest 数据包的聚合。该协议定义了一组全新的控制数据包类型，包含控制类和控制类型。此外，该协议定义了控制消息的三个主要类别：标准错误消息，通知消息和服务消息。每个类别都有一组基于该类别和用法的控制消息。实验结果表明，该协议可以很好地提高网络性能，尤其是在物联网应用领域，并且可以较容易地被扩展来支持不同的以信息为中心的网络平台。

综上所述，本论文研究了 ICN 作为物联网通信模型的可行性，并从 ICN 的多个维度提出了相应的解决方案，包括命名系统、网内缓存和控制协议等。上述方案可以很好地支持无缝高效的基于物联网的 ICN 网络。大量实验和性能量化分析表明，本文提出的方案在效率上优于现有的主要工作，并以更少的开销提供更全面的功能支持和更加出色的系统性能。

关键词：信息中心网络; 命名数据网络; 内容中心网络; 物联网

Abstract

The Internet of Things (IoT) has gained extensive attention from both industry and academia in the past decade. The connectivity of each and every piece of technology in the environment with the Internet has opened many avenues of research and development. However, one thing still remains the same: host-centric communication, since it is the predominant way of communication on the Internet. However, host-centric communication is still the predominant Internet communication method. With the evolution of everything else, host-based communication has been stretched to limits, and the exploration of new models has been underway for several years. Information-Centric Networking (ICN) is a major contender for future Internet architecture, where content is the basic key/central/main regardless of its location. ICN intends to offer content-based naming, in-network caching, inherent mobility support, multicast support, and content-based security as part of the design and not an add-on functionality. On the other hand, most IoT applications inherently follow a content-oriented paradigm. IoT users or things ask for the content and consume the generated data from the network instead of communicating with a specific host or device, such as the retrieval of sensed values from a sensor, or monitoring the status of a patient in their home. In fact, ICN can facilitate large-scale IoT deployment, by improving network performance & scalability, enhance security & mobility, and optimize the energy consumption of devices. In recent years, numerous efforts have been made to integrate ICN with IoT. Nevertheless, new issues, due to the nature of IoT applications and the working principle of ICN, have emerged slowing down the ambitions besides using the ICN paradigm in IoT environments.

In this thesis, we investigate and explore the possibilities to use ICN as a communication enabler for IoT applications. Inspired by the extensive research results in IoT-based ICN, we provide a detailed and systematic review of this merger. Furthermore, we identify a set of challenges and issues which we classify into two main classes: Network Communication and Network Management. For instance, we find that content naming – the pillar element in ICN architecture, publish-subscribe communication – a dominant communication model in IoT applications, and content caching scheme – a fundamental building block in ICN network, can be listed in the network communication class, whereas control protocol – an indispensable

protocol for large-scale networks, can be arranged in the network management class.

The core contributions and innovations of this thesis are as follows:

(1) In order to overcome ICN naming issues such as identifying content/service/devices, producing long and unbounded names, and performing efficient aggregation rules, we propose a unified hybrid ICN naming scheme for IoT applications. The proposed scheme incorporates a prefix-labeling method to describe hierarchical location names and applies a variable-length encoding method to produce short names with various embedded semantic functionalities using attribute-value design. Moreover, the scheme supports fast local IoT communication using the Name-to-Code translation concept that tends to tackle long name-prefix issues, as well as multi-source content retrieval through the in-network function concept that aims at fetching content from multiple producers with a smaller number of packets and overhead. The simulation results show that the proposed scheme is inherently better than the existing state-of-the-art solutions and drastically reduces the memory consumption, lookup time, routing and forwarding overhead, and enhances the overall IoT communication.

(2) In order to support publish-subscribe communication in ICN, we propose a group-based subscription scheme. The proposed scheme enables not only a seamless publisher-subscriber model, but also authentication, access control, and group management features. Without violating the single request-response ICN communication primitive, we design a scalable subscription management scheme using semi-persistent packets and specialized tables at intermediate nodes. The scheme also adopts the Logical Key Hierarchy mechanism to efficiently distribute security keys and secure content sharing among the authorized subscribers. Compared to traditional pull-based subscription, the proposed scheme is able to achieve lower control overhead, with added security and privacy features. The performance analysis shows that with semi-persistent interest, the memory requirements of the core nodes can be kept at minimal levels.

(3) In order to design an efficient content caching schemes for IoT applications, we propose centralized and distributed in-network content caching schemes. The centralized scheme aims at selecting the most suitable placement to cache content by minimizing the cost of moving content from the original producer to the intermediate cache-store and from the cache-store to consumers, as well as minimizing the caching cost at intermediate nodes. The

designed heuristic algorithm, Highest-First Farthest-Later, selects intermediate nodes with the highest demands, free cache memory as well as they are far away to satisfy a maximum number of demands. The second distributed scheme, Push Down Popular Push Up Less-Popular, aims at pushing down the popular content into the edge network and keeping the less-popular content in the core network. The obtained results show the efficiency and superiority of our scheme against similar strategies in terms of network delay, hop reduction, and cache utilization.

(4) In order to design a control protocol compatible with the design principle of ICN, we propose a control protocol for named data networks that can relay different network error, information, notification, and service messages. Our protocol uses content name instead of host address, compatible with the interest-data exchange model and ICN forwarding plane, and supports interest aggregations. The protocol defines a new type of control packet that contains ControlClass and ControlType fields to specify the class of control message and its type. We define three main classes of control messages: Standard Error Messages, Notification Messages, and Service Messages. Each class has a set of messages based on class and usage. The designed protocol improves the network performance especially in the IoT environment, and can easily be extended to support different information-centric platforms.

In conclusion, this dissertation studies the applicability of ICN as a communication enabler for IoT. It proposes a series of schemes for different ICN aspects (i.e., naming, communication model, in-network caching, and control protocol) that aim at providing a seamless and efficient IoT-based ICN network. Qualitative analyses and extensive experiments prove the efficiency of our schemes that outperform the existing solutions and exhibit better system performance with less overhead and added features.

Key Words: Information-Centric Networking; Named Data Networking; Content-Centric Networking; Internet of Things

Contents

List of Figures

List of Tables

List of Abbreviations

AAL	Ambient Assisted Living
AT	Access Thing
BAS	Building Automation System
CCN	Content-Centric Networking
CDN	Content Distribution Network
CDS	Content Discovery Service
CS	Cache Store
DONA	Data-Oriented Network
ET	Edge Thing
FIA	Future Internet Architectures
FIB	Forwarding Information Base
GbPS	Group-based Publish-Subscribe
HFFL	Highest-First, Farthest-Later
ICMP	Internet Control Message Protocol
ICN	Information-Centric Networking
IoT	Internet of Things
IP	Internet Protocol
LCE	Leave Copy Everywhere
LFU	Least Frequently Used
LKH	Logical Key Hierarchy
LRU	Least Recently Used
MAC	Media Access Control
M2M	Machine-to-Machine
NCP	Name-Code Prefix
NFD	NDN Forwarding Daemon
NDN	Named Data Networking
NetInf	Network of Information
NNCP	Named Network Control Protocol

NRS	Name Resolution System
P2P	Peer-to-Peer
PDPU	Push Down, Push Up
PIT	Pending Interest Table
PURSUIT	Publish-Subscribe Internet
QoS	Quality of Service
SIT	Subscription Interest Table
TLV	Type-Length-Value
URL	Uniform Resource Identifier
VANET	Vehicular Ad hoc Network

Chapter 1 Introduction

In the coming decade, there will be tens of billions of connected devices such as sensors, smartphones, cars, and data centers. Figure 1.1 shows the growth projection of devices in use until 2025 globally that will be exponential. These connected devices with sensing and decision-making capabilities are usually small in size and potentially mobile. This ecosystem makes up the *Internet of Things* (IoT) [1] that can be deployed with sensing and intelligence capabilities so that these things can not only communicate, but also collect data, negotiate, collaborate, and exchange the collected values.

On the other hand, the current Internet model was developed many decades ago to allow connectivity between specific end-points or devices. Using unique *Internet Protocol* (IP) addresses, most Internet communication happens between a client and a well-known server based on addresses. However, the user's needs have since changed, where they are more related to content sharing rather than the connectivity[2]. The Peer-to-peer model is one example which primarily aims at content sharing and not on connectivity to a specific server. Nonetheless, communication is still among specific devices sharing the content. Moreover, this device or host-centric model lacks mobility and security as part of its design. Hence, various add-ons and patches have been developed to support them. This makes the model and the system more complex and can impact communication performance.

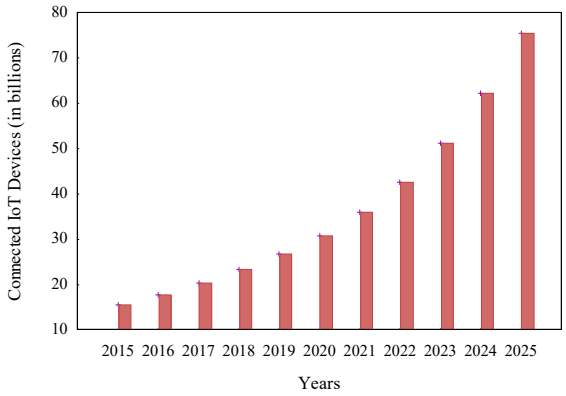


Figure 1.1 IoT device connectivity global statistics.

1.1 Motivation

IoT addresses a wide range of real-life business functions and applications, such as the following: *Smart Cities*: smart parking systems, traffic monitoring and control, bike-sharing, and smart bus, etc.; *Smart Homes*: automated home control devices such as Alexa and Google Home, etc.; *Smart Grids*: distributed renewable energy generation and advanced metering infrastructure, etc.; *Smart Transportation*: including vehicle safety, traffic efficiency, support for autonomous driving; *Smart Health*: for remote health monitoring and people with disabilities. All of these applications share many common characteristics such as energy efficiency, interactivity, real-time data connection and analyses, and non-intrusive monitoring systems. The main goal of IoT[3] by interconnecting devices, collecting, and processing data is to enable applications, machines, humans, and things to better understand their surrounding environments. This enables services and applications to make intelligent decisions and to respond to dynamic changes of the environment in order to improve our lives in different aspects such as reduced human efforts, efficient resource utilization, real-time marketing, and data analysis. However, various challenges need to be addressed in this context such as the heterogeneous nature of devices and sensors, efficient data retrieval, energy and memory limitations, power consumption, mobility, scalability, security, and dynamic network topology[4].

In light of these changing dynamics, new approaches[5] have been proposed for Internet communication. The objective is to address the challenges and limitations of existing systems by design. *Information-Centric Networking* (ICN)[6] has been proposed as a new network paradigm for the future Internet. The key element in ICN is the use of data/content name rather than the network address. A content name should be unique, persistent, and location-independent. A consumer requests the content by its name instead of the address of the provider. Hence, in-network caching can be applied during communication by storing the content more closer to consumers in order to improve data retrieval and reduce network traffic. As content is independent from the location, ICN natively handles the mobility, simply by re-issuing any unsatisfied requests. Furthermore, ICN provides easy data retrieval based on a request-reply exchange model, content-based security by attaching all security-related information within the content itself and native support of multicast.

On the other hand, most of the communication patterns in IoT applications inherently follow content-oriented paradigm[7]. IoT devices and users are interested in consuming the requested content from the network irrespective of knowing who is offering/hosting it. Use cases such as retrieval of sensed values from a sensor, update mobile application with recently published content (e.g., weather notification), or monitoring the status of patient in their home are mainly focusing on the content as the main elements instead of building a communication session with the content provider. In fact, ICN design and features can facilitate large-scale IoT deployment[8]. Specifically, ICN name abstraction enables seamless integration and interaction with IoT applications and devices, in-network content caching improves the content availability in the network and enhance the quality of service, content-based security enforces security and privacy by focusing on the content and not the communication channel, finally the clean ICN design and communication model improves network performance and scalability and optimizes the energy consumption of devices. Thus, we believe that ICN is a suitable communication paradigm that can be used in IoT networks. Although ICN attracts various researchers, it is still in its earlier stage before being widely deployed.

1.2 Problem Statement

Regardless of the ICN efforts that have been made to merge it with IoT, it still has have various aspects that need to be studied. In fact, the effectiveness and the performance of such a merger strictly depend on the underlying ICN architecture. To date, several ICN research projects have been proposed in the literature[9] such as *Data-Oriented Network* (DONA)[10], *Publish-Subscribe Internet* (PURSUIT)[11], *Network of Information* (NetInf)[12], *Content-Centric Networking* (CCN)[13], and *Named Data Networking* (NDN)[14]. Although these projects have different architectures, they share the same concept which is addressing the content by its name, rather than the network address of hosting device.

ICN uses the content name as the main network element to drive communication. The name must be persistent, identify the content and service, and ensure global uniqueness. However, it requires to be as short as possible, supports aggregation, and has a fast lookup process. In the IoT context, many applications are designed using the publisher-subscriber communication model. A set of subscribers register to receive content upon its creation from the

publisher. As ICN is built on interest-data exchange primitive, such type of communication is not supported by design in ICN. Moreover, enforcing security and access-control rules is indispensable, especially in dynamic environments where subscribers may join and leave the group. Besides, ICN aims at decoupling the content from its original location and implementing content-based security. Hence, the network layer becomes aware of content delivery by applying in-network caching. This feature enhances the content distribution and availability, improves the bandwidth and resource utilization, and overcomes a single-point of failure issue. However, it is essential to ensure that the caching scheme moves popular content near to consumers with less cost, and does not cache redundant copies in the network. On the other hand, IoT network including devices and sensors needs to be monitored and managed through a control protocol. Designing a control protocol for named network is extremely desirable.

1.3 Contributions

The main goal of this thesis is to study the applicability of ICN as a communication enabler for IoT networks. We aim at designing an efficient and scalable architecture that addresses IoT challenges, based on ICN features. We start by identifying the existing challenges and issues[15,16] and classify them into two main classes: *Network Communication* and *Network Management*. Then, we propose four contributions, as shown in Figure 1.2 and detailed as follows.

The first contribution addresses content naming. We propose a unified hybrid ICN naming scheme for IoT applications[17,18]. The proposed naming scheme provides built-in scalability, efficient routing, and security features. We incorporate a variable-length encoding

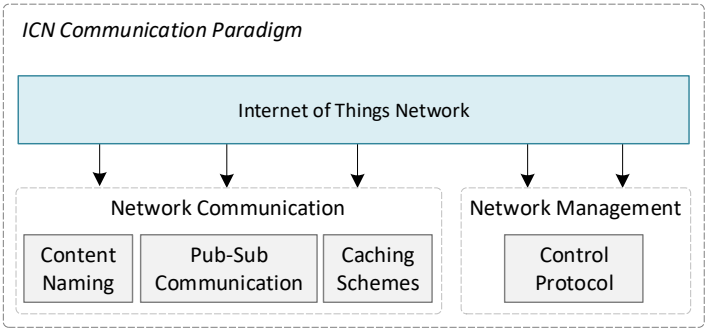


Figure 1.2 Summary of the research contributions.

method, with a prefix-labeling scheme in order to describe hierarchical location names with various embedded semantic functionalities. Moreover, the scheme supports fast local IoT communication using the Name-to-Code translation concept, as well as multi-source content retrieval through the in-network function concept. The simulation results show that our proposed hybrid scheme is inherently better than the existing state-of-the-art solutions, and drastically reduces the memory consumption, lookup time, routing and forwarding overhead, and enhances the overall IoT communication.

The second contribution addresses the Publish-Subscribe communication model[19,20]. We propose a group-based subscription architecture, which enables not only a seamless publisher-subscriber model, but also authentication, access control, and group management features, without modifying ICN principles. Compared to traditional pull-based subscription, we are able to achieve lower control overhead, with added security and privacy features. The performance analysis shows that with semi-persistent interest, the memory requirements of the core nodes can be kept at minimal levels.

The third contribution addresses the content caching in large-scale ICN networks[21]. we propose centralized and distributed in-network content caching schemes. The former scheme aims at selecting the most suitable placement to cache the content and serve all demands. The latter aims at pushing down the popular content into the edge network and keeping the less-popular content in the core network. The obtained results show the efficiency and superiority of our scheme against similar strategies in terms of network delay, hop reduction, and cache utilization.

The fourth contribution addresses the design and implementation of named control protocol[22]. We design a control protocol for named data networks that can relay different network error, information, notification, and service messages. The designed protocol improves the network performance especially in the IoT environment, and can easily be extended to support different information-centric platforms.

1.4 Thesis Outline

In this chapter, we have stated the research problem, highlighted the motivations, and discussed our major contributions. The remainder of this thesis is organized into seven chap-

ters as follows.

Chapter 2 provides the background of this thesis. We first start by providing an overview of the IoT technology and then presenting the ICN paradigm and its features. We also present the working principle of NDN, discuss the feasibility of using ICN in IoT, and state different IETF efforts. This chapter also presents state of the art of ICN solutions in two main parts. The first part deals with ICN solutions from IoT domain-specific perspectives (e.g., smart cities, smart homes, smart healthcare, etc.). The second part deals with the existing in ICN efforts for IoT applications (e.g., content naming, forwarding, caching, and publish-subscribe).

Chapter 3 introduces the first contribution, which is related to the naming scheme. It starts by highlighting the naming requirements and challenges; presenting in detail the proposed scheme including the reference network, multilayer multi-complement design, Name-to-Code concept, topological location labeling and encoding, and multi-source data retrieval; and then describing the implementation and discussing the obtained results.

Chapter 4 presents the group-based publish-subscribe architecture. It starts by highlighting the communication requirements and existing challenges and then presents in detail the proposed solutions including the subscription architecture, type of packets, and communication process. In the end, it presents the implementation and discusses the obtained results.

Chapter 5 provides a distributed content caching scheme. It presents the requirements of an efficient caching scheme and highlights the challenges. Then, it presents in detail the proposed scheme, the definition of content popularity, and content pushing toward the consumers. Then, it discusses the implementation and evaluation against other schemes.

Chapter 6 provides a named control protocol. It starts by highlighting the challenges and issues when designing such a protocol in named networks. Then, it presents the specification, packet processing, and proposed messages in detail. Later, it discusses the implementation and evaluation.

The last chapter concludes the thesis and all the work carried. It recapitulates the challenges, contributions, and summaries our findings. It also presents our future perspectives that can be followed-up by this thesis.

Chapter 2 Information-Centric Networks for Internet of Things: State of the Art

2.1 Introduction

The current Internet ecosystem has been designed to achieve end-to-end communication between two known devices. Accordingly, each device is assigned a unique IP address to allow communication and resource sharing. The current Internet follows the host-centric model, where the communication is based on, among other factors, the address of the destination node. This model has been adopted as the main communication paradigm over the years where resources sharing between two devices is required. However, today's Internet is witnessing tremendous growth in connected devices and huge changes in the application design, which affect the end-user requirements. To address these changes, the IP stack has been refined with add-ons and protocols to support various features. However, it became a complex protocol by adding extra patches to support security, mobility, and management.

To meet the user demands and address the changes in the nature of applications, various solutions have been proposed to realize *Future Internet Architectures* (FIA)[5]. The current Internet model is shifting from the host-based communication toward the content-oriented model[2]. End-users are more interested in what they request and consume regardless of who is offering the content or service. This vision of communication is known as Information-Centric Network[23].

In this thesis, we explore and study the use of ICN as a communication enabler for IoT applications. This issue has attracted some researchers in the last few years. In this chapter, we present the state of the art related to this topic. Towards this, we firstly introduce both IoT and ICN, and highlight the feasibility of using ICN as a communication model for IoT. Then, we divide the state of the art solutions into two main parts. In the first part, we detail ICN solutions from IoT domain-specific perspectives (e.g., smart cities, smart homes, smart healthcare, smart grids, and smart transportation). The second part deals with the existing efforts ICN paradigm for IoT applications (e.g., data/content naming, forwarding, in-network caching, publish-subscribe communication).

2.2 Internet of Things (IoT)

The basic idea of IoT[24] is to connect every object around us with the Internet, and enable intelligence feature on it. Thus, various technologies are combined, in order to allow sensors and actuators to sense and collect desirable data, interact and collaborate, provide smart data analytics, and take decisions without human intervention. Figure 2.1 represents the main IoT benefits.

2.2.1 IoT Characteristics

IoT is a complex network that represents the convergence of many real-world domains, where each domain has its own characteristics. Here we list the major characteristics:

• *Sensing:* The sensing characteristics can be utilized in a vast number of IoT use cases, such as smart mobile devices, healthcare, industrial control, climate monitoring, etc. Sensors allow measurement, in a context-aware manner, of environmental parameters, and enable the device to communicate with the physical world and people around.

• *Connectivity:* Different technologies are used to build connectivity between IoT devices or the Internet, enable service accessibility, global information exchange, and communication among different infrastructures.

• *Intelligence:* IoT devices facilitate data sensing and collecting. They may also incorporate different algorithms that can enable smart data analysis and take decisions accordingly.

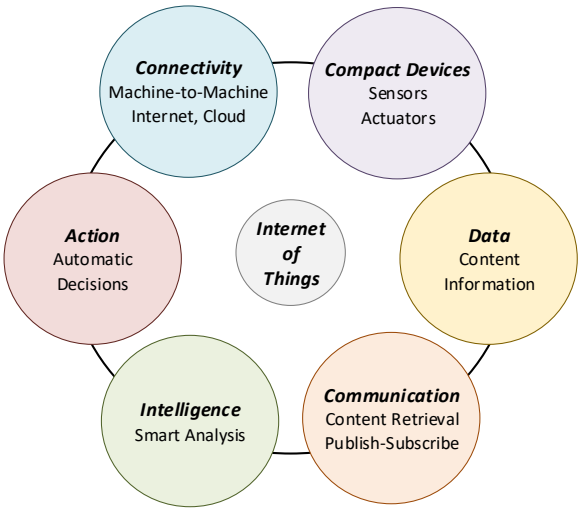


Figure 2.1 IoT characteristics & benefits.

• *Heterogeneity:* Various hardware platforms and operating systems are involved in enabling IoT. This complex ecosystem must allow an interconnection among heterogeneous devices and services in a way to provide seamless data exchange.

• *Dynamic changes:* IoT networks are characterized by the dynamic changes in topology since they can connect or disconnect according to their battery level or mobility. Moreover, the growing number of IoT devices and their usage makes the network topology more dynamic.

• *Scale:* A tremendous amount of data is generated by a huge number of IoT devices. This makes network management and data analysis more challenging and requires scalable IoT schemes and solutions.

2.2.2 IoT Communication Models

IoT devices tend to cooperate, exchange, and process data between each other. They are able to communicate with their domain gateway, or with the Internet. Hence, we can broadly classify IoT communication models into the following classes:

• *Device-to-Device Communication Model:* Two or more IoT devices may directly connect and communicate with each other instead of going through an intermediate service (e.g., smartwatch communicates directly with a mobile phone). Different protocols can be used for device-to-device communication such as Bluetooth and ZigBee.

• *Device-to-Gateway Communication Model:* This model is also known as Device-to-Application-Layer Model, where an IoT device connects with an associated service or gateway that acts as a go-between IoT network and Internet, to access the Internet or cloud services. For example, a local home gateway connects with various smart home sensors and allows user's access to these sensors through a smartphone application via Internet/Cloud.

• *Device-to-Internet Communication Model:* In this model, IoT devices can directly connect to an Internet cloud service (e.g., Application Service Provider) to exchange data and receive control messages. Users through the cloud service, are able to obtain remote access to sensor devices using smartphone applications. For example, they may monitor their Smart TV for shows, and take control over certain channels. Users may also export their desirable data from cloud services, merge data sources for aggregation purposes, and deep analysis, for

example analyzing the home energy consumption.

2.2.3 IoT Application Requirements

Although IoT has been integrated into most of the environments, various challenges need to be addressed. Below, we outline the most critical challenges in IoT-based applications.

• *Addressing:* Billions of IoT devices can interconnect and collaborate for different tasks, create, and exchange content between each other. Each IoT device should have a unique and persistent address. Furthermore, users and sensors are more concerned about the created content, and less about the address of sensors. Hence, addressing the content is as important as addressing the devices. A large space IoT addressing scheme is required with less processing, a simple header, and persistent abilities during the whole content lifetime and mobility across different networks.

• *Heterogeneity:* The heterogeneous nature of IoT devices and the deployed architectures and technologies, require various middlewares to collaborate and share the content between devices and across IoT networks. The existence of these middlewares makes the communication complex and prone to performance issues, where establishing secure data sharing becomes a complicated task. Therefore, designing a clean Internet architecture that processes the content regardless of the nature of the hosting device or the requesting application may overcome the heterogeneity issue.

• *Security & Privacy:* Most of the data is generated and collected by IoT devices and analyzed for monitoring purposes and decisions. Thus, the network layer should treat the content by its name rather than its network address or the availability of the original producer. Also, the content may face different attacks and privacy issues during the generation (application layer perspective) and transmission (communication layer perspective). Therefore, IoT-based smart applications must adopt serious security mechanisms to ensure data security, privacy, access control, reliability, and confidentiality, regardless of the type of channel used. Hence, keeping this data safe and reducing the risk, requires developing different trust models among content providers and consumers.

• *Mobility:* IoT-based applications may cause some reliability issues, due to the patient mobility in healthcare systems, or vehicle mobility in smart transportation systems. Provid-

ing anytime anywhere connectivity, and content availability in IoT requires serious design choices especially in a highly dynamic mobile environment. Hence, seamless mobility support in IoT is needed to provide fast data retrieval in a reliable and secure manner.

• *Network Scalability:* Millions of distributed IoT devices generate a large amount of data, analyze, and transfer them with each other or with the Internet service. Also, new devices may connect to the network, disconnect, or move from one network to another, e.g., a dynamic connection/disconnection of sensors in healthcare environment, or dynamic mobility for vehicles in smart transportation systems. This has a major impact on the communication reliability and quality of services. Hence, ensuring a scalable IoT network is an important aspect of the IoT ecosystem that may handle the computation of data and communication of device.

• *Resource Constraints:* Most of IoT devices are considered as resource-constrained with limitations in power, memory, computing capabilities, and bandwidth. Providing IoT applications that deal with these constraints is required to satisfy the user experiences and provide continuous service availability during high mobility and network outage.

2.3 Information-Centric Networks (ICN)

ICN[23] has been proposed as a new architecture for the future Internet, addressing many issues in the current IP-based networks, such as routing process, scalability issue, and content sharing performance[25]. ICN integrates all network functionalities around the name of content rather than the address of the network, in a way to ensure efficient data dissemination and access.

Formerly, different concepts such as *Peer-to-Peer* (P2P) and *Content Distribution Network* (CDN) have been developed to improve the content sharing and distribution in the Internet[26]. However, ICN in contrast to P2P and CDN, is a standardized protocol that works at the network layer. P2P is an application-specific protocol, whereas CDN is a proprietary solution working at the application layer. Moreover, P2P content is delivered from end-users, while in CDN proprietary infrastructure is used. However, ICN content can be delivered from the network infrastructure itself.

This redesign from "*where* the content is located" to "*what* is the content" will improve

the network performance, facilitate the content retrieval and replication using in-network content caching, and support native multicast delivery & mobility. Figure 2.2 illustrates the difference between IP and ICN communication. Content retrieval is shown in red and blue color respectively. Assuming that all consumers are requesting the same content *D* provided by the producer, IP-based communication requires that each consumer (consumer 1 and 2) individually knows the address of the content producer, and then independently fetches the content through an IP routed path. However, in ICN-based communication, the consumers should specify only the requested content name (without knowing the host IP). The request is forwarded based on name-based routing rules until it reaches a device which has the content. In Figure 2.2 consumer 3's demand is satisfied by the producer (first demand), during which routers (e.g., R3) can cache the content. When consumer 4 requests the same content, the request is satisfied by the content-store at R3, thus eliminating the need to reach the original producer.

Many research works have focused on different ICN functionalities such as naming, in-network caching, mobility, and security. In the following sub-sections, we highlight the key features of ICN and provide an overview of the working principle of NDN.

2.3.1 ICN Features

Content Naming: The content name[27] is the key element in ICN, which uniquely identifies the content itself. It should be compact, persistent, and be able to validate the

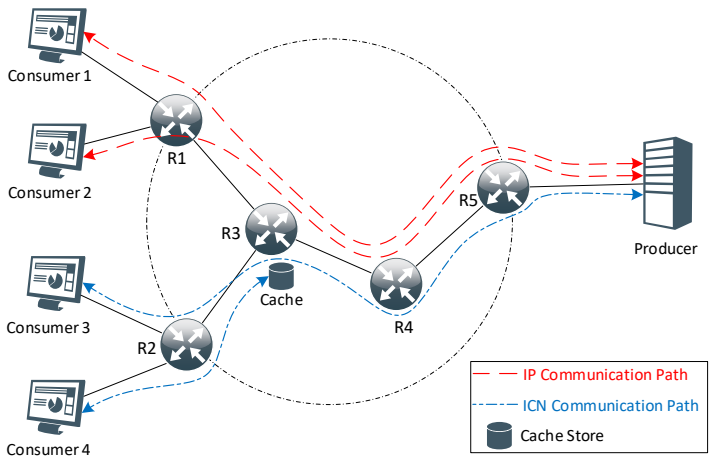


Figure 2.2 Content retrieval: IP-based versus ICN-based networks.

content. The used naming scheme must be scalable and should allow name aggregation. Four main types of naming schemes have been proposed in ICN. Table 2.1 provides an overview of their benefits and drawbacks.

- *Hierarchical Names:* They consist of multiple components[28] to identify the content and describe the application/services. It has a similar structure to current *Uniform Resource Identifiers* (URIs) and can create user-friendly and meaningful names for users. Hierarchical naming enhances the scalability since name prefixes can be aggregated, but may also be lengthy.

- *Flat Names:* They are typically obtained through hash algorithms applied to content[29]. There is no structure to the name. Hence it is not human-friendly and can hardly be assigned to dynamic content that is not published yet. Flat naming has scalability issues since it does not support routing aggregation.

- *Attribute-Value-based Names:* Attribute-value based naming scheme[30] has a collection of attributes, where each attribute has a name, a type, and a set of possible values (creation date/time, content type, location, version, etc.). Collectively, they represent a single content and its properties. This naming scheme supports an easy searching process by using known content keywords. However, it is hard to ensure naming unique-

Table 2.1 Types of ICN naming schemes.

Type	*Example*	*Benefits*	*Drawbacks*
Hierarchical Naming	`/bit.edu.cn/cs/research/2019/authors/title.pdf`	Enhanced network scalability. Semantic information included. Name aggregation possible.	Longer names. Required standardization.
Flat Naming	`ni://bit.edu.cn/sha-256(title)`	Fixed length names. Easy to generate.	No structure & semantic of the name. Difficult for dynamic content. No aggregation support.

Attribute-Value based Naming	`Title <str>: 'Title'`
	`Authors <list>: [Auth1, Auth2]`
	`Journal <str>: 'Journal'`
	`Year <int>: 2019`
		Easy searching process.
		Additional content information available.
			Naming uniqueness is difficult.
			Same attributes for different content.

Hybrid Naming	`/bit.edu.cn/cs/Year <int>: 2019,Authors <list>: [Auth1, Auth2]/sha-256(title)`	Combining different features.	Complex management.

ness, as many unique contents may have the same properties.

- *Hybrid Names:* A hybrid naming scheme combines at least two of the previously discussed schemes or all of them. Since base naming schemes have different drawbacks, a hybrid naming scheme attempts to use the best features provided by the base scheme to improve the network scalability & performance, and enhance the security & privacy. For example, taking advantage of name aggregation to enhance the lookup process, fixed length of flat names to save space, and attribute values to provide keyword searching and security/privacy.

Moreover, ICN can use *Name Resolution System* (NRS) instead of providing hop-by-hop name-based routing. The interest packet is forwarded to NRS server to provide a resolution of the requested name and forward the request to the content provider/producer.

Routing and Forwarding: The use of names to identify the content introduces name-based routing[27] to discover and deliver the content to the requester. Due to the receiver-driven design, a consumer triggers a request asking for content by specifying its name. The discovery process starts searching for the content based only on the name. The request is forwarded hop-by-hop using a forwarding/routing table until it reaches the original or a replica node that has the requested content, after which the content is delivered to the requester.

In-network Caching: As the content names are location-independent and each data packet is self-consistent, in-network caching[31] can be applied during ICN communication. Each ICN node can cache the content, and serve it for future requests, as shown in Figure 2.2. Caching improves network performance by reducing the delay and facilitating content retrieval. However, deciding what kind of content should be cached in ICN involves treating content based on various metrics including popularity and freshness. Hence, studying the network topology with analysis on consumer traffic demands is required, while taking into consideration device capabilities such as the cache memory and processing. In addition, removing old cached content and replacing it with new content introduces the need for replacement algorithms and strategies to keep the most used content with fewer changes in the cache store[32–34].

Content-based Security: ICN emphasis content-based security[35], where security mechanisms are applied to content itself rather than the communication process. Different trust

models have been developed based on network services. Also, each data packet is self-authenticating based on the original content security-related information (e.g., the publisher's public/secret keys and signature)[36–38].

Mobility: From ICN perspective[39], the content is independent from its original location, and only the desirable content name is used to discover and forward it back to the consumer. When a node moves from a network to another, it can re-issue any unsatisfied requests, and the producer replies with the requested data without any need to request a new address during the movement.

Due to the variety of features offered by the native ICN paradigm to solve the complex Internet design and functionalities, the research community has started developing and implementing different architectures using the ICN perspective. In the following, we present an overview of NDN architecture which is an active ICN implementation.

2.3.2 Named Data Networking (NDN)

To implement the vision of ICN, various architectures have been proposed including DONA[10], PURSUIT[11], NetInf[12], COMET[40], CONVERGENCE[41], MobilityFirst[42], CCN[13], and NDN[14]. In this thesis, we mostly focus on research relevant to CCN & NDN, as they have received more attention from the research community in the past, and continue to be favored as architectures of choice. NDN, in particular, is one of the most active architectures in ICN research that has been forked from CCN architecture in 2010. NDN uses two types of packets: interest and data. Both interest and data packets carry the name of the requested content. NDN is a named-based network where the routing is achieved by using names. NDN uses hierarchical, human-readable, and structured URL-like names. Moreover, NDN maintains three data structures: *Cache Store* (CS), *Pending Interest Table* (PIT), and *Forwarding Information Base* (FIB).

Figure 2.3 shows interest and data forwarding planes in NDN. When an NDN node receives an interest packet, it checks in its local CS. If the requested data already exists in the cache, it means this node is a replica node. In such a case, a data packet will be sent using the same face where the interest has been received. Otherwise, it will check in PIT if a match for a similar request has already been forwarded upstream. In this case, NDN will append the

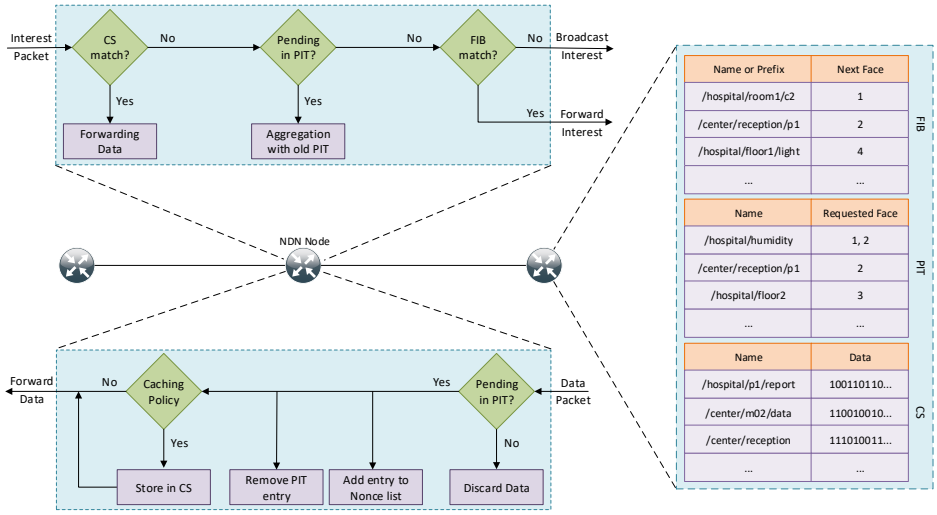


Figure 2.3 NDN operational logic and table structure.

face where the interest has been received to the PIT entry (interest aggregation), otherwise, a new PIT entry will be created that has the name of the requested content and the name of the face. Then it checks FIB to find the next hop toward the content provider.

In reverse path, when an NDN node receives a data packet, it checks its PIT. If no match is found, which means there is no request that has been sent before, the node discards the data packet by considering it as an unsolicited packet. Otherwise, it will forward the data packet to all the faces saved in the PIT table. At the same time, based on the cache policy, the nodes along the data path may cache content.

NDN designed ndnSIM[43], an NS3-based simulator, in order to allow researchers to implement and simulate the NDN architecture using various link-layer protocol such as point-to-point, Carrier Sense Multiple Access, wireless, as well as different network-layer protocols (IPv4, IPv6), and transport layer (TCP, UDP). ndnSIM combines NDN Forwarding Daemon (NFD) and ndncxx code-bases in order to provide a level of interoperability between simulation and prototyping. However, this integration causes many challenges, where ndnSIM cannot support the connection of the simulation network with NFD and ndn-cxx in case of running on an external host. Also, every change in NFD or ndn-cxx needs to be manually integrated with ndnSIM. Other challenges faced by ndnSIM include the limitation of memory consumption by simulating large-scale experiments on top of devices that have limited hardware resources.

2.4 Why ICN for IoT ?

The information produced by IoT smart devices can be regarded as content[44]. Consumers in the network request data in the IoT context without the need to know the location of the sensors or the actuators. There is no end-to-end session requirement for content retrieval, and ICN targets the content in the network by its name rather than its address. For example, asking for humidity value for a specific place, query some information, or data monitoring.

ICN nodes can act as replica-nodes by using content stores. Content can be cached and served for future requests regardless of the original producer's reachability. This caching improves data retrieval and reduces latency. In such scenarios, ICN is more suitable for IoT than IP[45], not only for the rapid content delivery but also for its receive-driven design and request aggregation. Moreover, multicast and mobility support of ICN is an additive point, where multicast can be done from the network layer, and any unsatisfied requests during mobility can be re-issued without the need for complex handover solutions like those of IP. Furthermore, by using global unique names, ICN provides content-based security and encryption, and ensures content integrity and authenticity, as part of its design.

IoT can be combined with different daily user routines, by enabling seamless integration and interaction with applications, sensors, and actuators. Daily personal monitoring, industrial processes control are some real examples of IoT applications. By using content-based naming and NRS, the addressing issue in IoT can be solved. This will also help in removing any restriction on the type of content and the nature of producer device[46]. Moreover, decoupling the content from its original location, and using the name as the main element to identify the content, the heterogeneous nature of IoT sensors becomes irrelevant[47].

From the security perspective, securing the communication channel in ICN is not required as the latter follows a content-based mechanism by encrypting the content itself and applying different trust models in the application layer. Furthermore, since IoT devices are resource-constrained, in-network caching, forwarding strategies, caching placement/replacement policies, interest aggregation, and session-less concepts improve energy efficiency and reduce power consumption.

Overall, all these features contribute to make ICN an efficient alternative solution for

IoT in terms of scalability and reliability[48].

2.4.1 IETF Efforts

Currently, various drafts from IRTF ICNRG research group are focusing on IoT-ICN. Work in[49] identifies the requirements to develop a unified IoT architecture that addresses IoT traffic, mobility, and management issues. Also, it elaborates the benefits of using ICN on top of IoT (e.g., naming, security, and caching) and lists major ICN challenges (e.g., standard naming, scalability, caching & storage, etc), by discussing the current IP-based IoT solutions and how to overcome the different disadvantages in order to build a suitable IoT-ICN framework. Work in[50] describes the potential modifications of ICN and more specifically CCN architecture, and different design choices that should be taken into consideration in order to integrate ICN in IoT. These include an appropriate naming scheme for IoT content, immutable data by leveraging meta-data files decoupling the consumers from the content publisher and combining Pull and Push-based traffic with the ICN communication model using name-based routing. The modification should consequently improve the efficiency and scalability of the overall IoT network and its applications by using a simple communication model with less processing overhead.

A recent draft[51] discusses the motivation and the needs of IoT-ICN architecture to connect heterogeneous IoT devices. The work proposes an IoT-ICN architecture combined with a middle-ware. It has various functionalities to ensure device onboarding and discovery, naming service, services discovery, and publish-subscribe management. Also, co-existence with heterogeneous networks such as IP-based has been discussed by using name-to-name primitives and introducing a named protocol bridge to forward the message through different core networks.

Broadly speaking, ICN opens new opportunities to implement a native content-oriented view of IoT[52]. However, most of the current proposed ICN solutions and architectures did not take IoT principles in their design[53]. Thus, enabling ICN in IoT environments needs careful design of different IoT network components, applications, and services[54]. In the next section, we present ICN solutions for IoT domain-specific applications.

2.5 Domain Specific IoT-ICN Solutions

IoT is involved in different applications. These domain-specific use cases have a common vision, but still, have different characteristics. In order to reap the benefits of merged IoT-ICN technology, it is important to address these characteristics of domain-specific applications. In the following, we present a review of different IoT applications using ICN as their communication model.

2.5.1 Smart City Systems

Smart Cities comprise of many interconnected ubiquitous services. Sensor and IoT networks are used to improve citizens' quality of life, and support other systems such as remote power usage monitoring, integrated parking and security services, etc. A huge amount of distributed devices are used in smart cities, including houses, vehicles, buildings, and public environments, as well as a massive number of services and applications. IoT-based smart cities aim at enhancing citizens' services. For example, finding an available parking spot gets harder due to the growing number of cars, and the limitation of parking spaces. Today's smart parking may use different IoT sensors to detect the availability of free space and facilitate parking asset management. Vehicles willing to get into the parking can subscribe for Parking Services, whenever a new space is available, all vehicles get notifications according to their Global Positioning System position. In such a scenario, a *content-oriented paradigm* is the perfect communication model, where subscribers are interested only in the data they need (e.g., available spot). Moreover, video surveillance solutions can be integrated to facilitate public parking management. Another scenario, where distributed sensors are implemented in different city locations to monitor the pollution levels. People may get notifications and warnings about pollution levels over time, and at different locations in real-time. These services can be integrated with deep learning modules to predict situations in users' workplaces, schools, and open public spaces.

Current ICN Efforts: Piro *et al.*[55] proposed an ICN-based platform for smart cities on top of NDN to take advantage of content-centric features. Secure communication model based on discovery and content delivery has been discussed. In the first phase, a consumer should find the content provider through service discovery, establish a secure communica-

tion channel by including both consumer name and provide name in the interest name field. Different use cases have been illustrated where hierarchical naming schemes have been used. The use of hierarchical naming is motivated by its efficiency to support various kinds of services used in smart cities, with simple routing operations and aggregation capabilities. The major limitation of this solution is the use of both consumer and provider names in the content name, which takes the ICN paradigm back to the host-centric model. Moreover, it uses long names which translate to larger overhead in the lookup processing. Furthermore, this method ignores the use of in-network caching. Ahmed *et al.*[56] discussed the feasibility of bringing NDN into smart cities. They have presented different concepts and scenarios that contribute to smart cities such as wireless sensor networks, smart grids, and vehicular networks where NDN can fit well. The authors also discussed different NDN aspects for future smart cities such as content naming, data exchange and forwarding process, and content discovery. Mochida *et al.*[57] presented the case of a weather monitoring system in a smart city, where a network of weather sensors and cameras generates a large volume of content. The authors used deep learning approaches to predict the components of the used naming scheme for such data. Although the scheme is shown to be very useful in generating names specific to the context of the content, it tends to generate lengthy names. This may create overhead for resource-constrained IoT devices. Naeem *et al.*[58] aimed at improving content delivery with reduced latency and producer load, by distributed content caching. The distributed nature of caches across the city may allow a higher number of connected devices and satisfy their demands with better service quality.

Due to the lack of complete solutions in this domain, integrating ICN in smart cities requires more investigation. Flexible trust models are required to address secure data dissemination and data privacy. Also, developing smart caching schemes to decide where content should be cached at a city-wide scale is needed.

2.5.2 Smart Home Systems

Smart Home services are used to enhance personal lifestyle, and make it easier and simpler. These include smart home security systems, fire detection, lighting control, and temperature monitoring, etc. Various home control gateways and solutions are available. These

solutions may provide complete control of the home automation, e.g., switching on the lights automatically when someone enters the room, or off when someone leaves it, intelligent face recognition at the home entrance, and managing heating systems by tracking data from outdoor and indoor temperature. All of these features and others enhance the quality of life and improves home energy efficiency. From both communication and application perspective, there is no need to know the address of the hosted sensors or the service. All what is required is information about the service. Furthermore, it is more feasible to apply access control mechanisms and authorization based on the content name regardless of the physical position. Hence, leveraging ICN as a communication model may help their fast deployment and management.

Current ICN Efforts: Ravindran *et al.*[59] compared the IPv6 IETF proposal for home networks with ICN-based approach from different perspectives such as service, control, and data plane as well as complexity. The authors discussed various challenges faced for such implementation and how to use ICN features to realize a unified IoT-ICN platform that is able to handle the heterogeneity of IoT devices, services, and user needs by using the ICN naming scheme and in-network caching for content distribution and native support of mobility. Only by using names, the authors proposed a service discovery protocol and content-based policy on top of ICN to enable user interaction with devices in infrastructure or non-infrastructure mode. Burke *et al.*[60] addressed the security in lighting control for *Building Automation System* (BAS). The authors identified the security requirements and built an NDN-based security architecture by combining a trust model with key attributes and access-control policies just by using the name of content. The application creates and signs the desired commands that are described using names. The sensor checks the command signature, executes it, and replies with a data packet as execution acknowledgment. Amadeo *et al.*[61] proposed a tailored ICN framework for smart homes, by using hierarchical naming scheme to identify various services and tasks. They have modeled different services for push and pull-based traffic by unsolicited data packets. However, this violates the primitives of the interest-response mechanism. Also, a multi-party forwarding strategy has been used to allow data retrieval from multiple producers in home environment based on the sharing of a common name prefix among consumers.

Although the previous works are focusing on the integration of ICN in smart homes,

more efforts are needed to provide a scalable publish-subscribe communication model in ICN. The use of unsolicited data violates the interest-data exchange model, while persistent interest is not sufficient to handle the growing number of subscribers. Also, providing secure data sharing on top of such models is indispensable to allow only legitimate subscribers to consume the data. Moreover, as smart homes are directly connected to people, the privacy of users needs to be addressed as a primary objective.

2.5.3 Smart Healthcare Systems

Smart Healthcare represents one of the most attractive IoT application, due to its potential for remote diagnostics and examinations, patient tracking and monitoring, and treatment and surgery. Deploying different health sensors on patient's body and surrounding is one promising solution to facilitate healthcare services, where patients are not required to stay in the hospital. Doctors and nurses may keep monitoring the patients' situation based on the collected data from the attached on-body and environmental sensors just by specifying the patient name and/or the target sensor/service. Also, when a patient comes to a hospital, it will recognize him automatically, hence all its reports and available data will be loaded and synchronized with medical professionals. Furthermore, third party members such as family members and assurance companies may have the ability to keep monitoring the patients' health based on their permission levels. These reports and information can be cached at any network level (home's access point, hospital database, clinical access point, etc.) to facilitate its distribution and increase its availability. Also, data and report sharing among different entities can be done using the name of patient or information that will enhance its access and distribution. Finally, all security and access control rules may be deployed based on the name of users, which is much easier and efficient than using the address of the host node/sensor.

Current ICN Efforts: Hail *et al.*[62] proposed an IoT architecture for *Ambient Assisted Living* (AAL) over NDN, that may improve the overall service by using caching and simple NDN design. They proposed a specific naming structure for AAL to solve the heterogeneous devices and integrate different services between them without the need for middleware. However, the work is mostly an abstraction and does not address various AAL scenarios, has a highly expressive naming scheme, and does not address patient mobility or secure publish-

subscribe communication. Zhang *et al.*[63] designed and implemented NDNFit which is a distributed mobile health platform running on top of NDN. NDNFit collects and shares mobile health data using cloud-enabled mobile architecture, and allows users to take control directly to their data deciding which application and user are allowed to use and manipulate such data. Also, data is named using hierarchical names and all security mechanisms are applied to the data itself that make it self-secure and independent from the producer. However, the proposed solution has different components to allow IoT data storage, process, and visualization. Saxena *et al.*[64] proposed NHealthIoT which is an NDN-based smart health IoT. A pure NDN *Machine-to-Machine* (M2M) communication is used to capture and transmit data from sensors to the home server, and detect emergency events using Hidden Markov Model. The proposed solution uses a context-aware adaptive forwarding strategy to send events to the cloud server and periodically pulls other information data using a publish/subscribe paradigm.

Due to different user access levels that can be integrated into one health-care application (e.g., patient, doctor, nurse, family members, etc), an adequate trust management scheme with data privacy must be developed with name-based access control, that defines access for each member. Also, providing quality of service support is necessary to allow emergency and event-based traffic. Moreover, the mobility factor is an important aspect (e.g., mobility of patients with attached on-body sensors) that should support reliable data delivery.

2.5.4 Smart Grid Systems

Smart Grids use IoT to provide smart management and control of energy distribution. Smart Grid systems tend to improve and enhance the energy consumption of homes and buildings. Smart grids aim to contribute to the economy and improve environmental health through moving the available energy in an optimized and reliable manner. The use of IoT devices in cities, homes, and grids may transmit the electricity in an efficient, secure, and scalable way, quickly run restoration and backup solutions after power disturbances, reduce the peak demand and enhance lower electricity rates. Considering the communication layer of smart grids, we find that it is independent from the host point of view, and is related only to the requested information. The use of caching will enhance the communication between smart grids entities without end-to-end device communication. Even if it is the case, the communi-

cation is still optimized and secure in nature by leveraging content-based security. Similarly, using the content name to allow data retrieval can enhance the multi-source content delivery in an optimized manner.

Current ICN Efforts: Zhang *et at.*[65] proposed an ICN-based architecture to secure the communication in home energy management systems. The idea consists of collecting the measurement information from household elements, performing data aggregation, and analysis for future intelligent decisions. To this aim, the authors adopted two different layers: publish-subscribe communication layer, and security layer which is responsible to share the encryption key along with the subscribers. Although the architecture provides secure data sharing, it cannot scale with the dynamic increase of subscribers, and only addresses the data collection. Katsaros *et al.*[66] used real grid topologies in Netherlands to study the feasibility of using ICN solutions to address different M2M communication challenges in smart grid, and compared the performance gained by ICN against host-centric solutions. The authors found that ICN may address various emerging challenges in smart grids including multi-rendezvous points selection and in-network processing. E. Oh[67] discussed network mechanics and requirements for the smart grid using an IoT system. The author highlighted the design choices and strategies for a secure smart grid using ICN, to enable reliable and efficient communications, and to decrease the system and computational complexity. However, the paper is limited to design only. Future work may include implementation and experimentation.

Machine-to-machine communication is a dominant model in smart grids. However, defining the M2M model on top of content names needs more investigation. Similarly, securing the publish-subscribe data flow among the active subscribes requires new and flexible key exchange protocols.

2.5.5 Smart Transportation Systems & Vehicular Networks

Smart Transportation couples computation and communication to monitor and control the transportation networks. IoT may play an important rule in such context, e.g., traffic control systems to monitor vehicular traffic in cities, deploy services to manage traffic routing and avoid congestion, or using sensors in smart parking systems to improve mobility in urban areas. Public transportation has a significant impact on the daily life of commuters, and ve-

hicles exchange large volumes of data with other vehicles like warning messages, or between infrastructure for navigation or downloading files. The exchanged data may cause network congestion and affect hundreds of vehicles in network. Thus, ICN caching can improve the data availability regardless of producer reachability (get the data from near content-store at the vehicle level). This may gain time for both citizens and public transport companies, and increase safety especially for children and elderly people. Another scenario, where passengers may subscribe to public transport services to better monitor bus routes and delays. These services may combine both real-time information with the personal needs to fetch the content, and fulfill its demands. Here, passengers are not interested in the bus address, the communication will be related to the name of the offered services. In such cases, the subscription may be satisfied by any subscriber in the same direction.

Current ICN Efforts: Bouk *et al.*[68] discussed the integration of NDN in intelligent transportation systems from the smart cities perspective. They elaborated on securing and enabling reliable communication among mobile devices, by benefiting from NDN features such as naming, in-network caching, and content-based security. The authors highlighted various research directions aiming to enhance the merger of NDN and transportation system, but excluded the mapping of NDN and *Vehicular Ad hoc Network* (VANET). Similarly, Amadeo *et al.*[69] discussed the applicability of ICN in vehicular environments, by elaborating different ICN functionalities from VANET perspectives. Although the authors debated the different opportunities used by ICN naming, caching, and support of multicast and mobility to overcome critical challenges in vehicular networks, they did not cover recent efforts in such context. Saxena *et al.*[70] studied the performance of NDN-based forwarding schemes in VANET in terms of content retrieving and disseminating data, by implementing different IP-based forwarding schemes such as Epidemic[71], Spray & Wait[72], and Adaptive Forwarding[73] on top of NDN in a real vehicular testbed. However, the proposed solution does not address duplicate requests and forwarding loops, nor it considers vehicle's mobility. Chowdhury *et al.*[74,75] proposed an anonymous authentication and pseudonym-renewal scheme for VANET on top of NDN. The authors divided the network elements into three entities: Vehicle, Manufacturer, and a Root Organization. By benefiting from NDN naming and security, they presented a trust model and Certificate Issuing Proxy for authentication, where vehicles use pseudonyms rather than real names to prevent tracking. They used Raspberry Pi-based

miniature cars for implementation and evaluation. Signorello *et al.*[76] addressed the security challenges in NDN by focusing on vehicular environments, and different vulnerabilities that may be created when applying NDN on top of VANETs. They focus on interest flooding attacks, and cache poisoning attacks. This work did not cover security in other ICN components such as naming, caching, mobility, etc. Similarly, Bouk *et al.*[77] discussed various security vulnerabilities and attacks in vehicular cyber-physical systems, and highlighting various issues and challenges. Based on their discussion, the authors proposed NDN-based cyber-resilient architecture, which contains an NDN forwarding daemon with threat aversion, detection, and resilience capabilities. However, the authors did not discuss in detail how their architecture will be implemented, nor any results have been presented.

The high mobility of vehicles is a critical issue in smart transportation that affects data dissemination and quality of service. Smart collaborative caching schemes must be developed with the integration of edge computing to overcome such issues. Also, there is a need to develop adequate forwarding schemes that can deal with the broadcast nature of traffic and take benefit of in-network caching.

2.6 General IoT-ICN Issues

Continuing from the earlier discussion, IoT communication follows data-oriented concept very closely[78]. Hence, ICN connectivity and principles must be adapted to IoT paradigm. In the following sub-sections, we review the existing ICN solutions tailored for IoT network.

2.6.1 Data/Content Naming

The content name is the main component to build an ICN network, request, and deliver data. The performance and the working principles of other aspects such as security, mobility, and caching are based on the efficient working of naming. Most of the existing naming solutions have been designed from a general network standpoint without taking the IoT characteristic into consideration. Hence, designing a flexible and custom-based naming scheme with a broad name-space in ICN is an important task, especially for scalability of networks.

ICN Naming Solutions: Bari *et al.*[27] discussed various naming solutions in ICN architectures such as DONA, NetInf, PURSUIT, and NDN; and presented a qualitative comparison

of their naming/routing schemes. Then, different requirements and design alternatives based on the study have been proposed to achieve an efficient content naming/routing model. Adhatarao *et al.*[79] presented a qualitative and quantitative comparison of hierarchical and flat names. Different metrics have been taken into accounts such as lookup efficiency, aggregateability, semantics, and manageability. The study shows that a higher lookup complexity can be achieved using hierarchical names due to the parse and lookup for each name component to determine the next interface. However, hierarchical names can reduce the FIB table size by using name aggregation compared to flat name that has non-structure/semantic features but no aggregation can be applied. Mochida *et al.*[57] earlier discussed from its use case perspective, also presented an interesting approach which may be applicable in other IoT use cases in general. In this scheme, names are assigned automatically at the monitoring camera using deep learning algorithm to generate a model of the predicted name sequence. With context awareness of other use cases, the same can be extended to other IoT devices, however, the length of the name generated can become a bottleneck.

Hierarchical Naming Schemes: Burke *et al.*[60] used NDN hierarchical names to secure the control in BAS. Their goal is to benefit from having a hierarchical structure to name all system components including authentication using the following */namespace/command/randomizer/auth-tag*. Although they added security aspects to naming, long hierarchical name is still an issue, that effects the lookup process. Also, the used algorithm (e.g., Rivest–Shamir–Adleman, and Hash-based Message Authentication Code) requires more processing and energy. Amadeo *et al.*[61] designed an NDN framework for smart homes, by proposing a namespace for the home with two classes: *config* and *task*. For each class, they defined the *task type* and its associated *sub-types*, such as */action/light/on* and */sensing/movement*, the last component in the name is the location of the sensors such as *kitchen, bedroom, etc.* This naming scheme can support rule aggregation to reduce the number of sent requests. However, it is designed for a specific use case and requires more effort from the application layer to understand the naming semantic. Similarly, the used hierarchical unbounded name may affect the lookup and forwarding performance. Bracciale *et al.*[80] introduced an abstract Lightweight Named Object scheme for IoT devices. The motivation behind such naming is to provide programming, simplicity, and extended functionalities to physical devices. The proposed solution uses NDN hierarchical names in order to represent physical IoT objects in a derived name-

space. However, the used names are very long as they include the device name, function, and a list of parameters. Although the functionality is lightweight, the storage and matching of long names may be a limiting factor in some IoT scenarios. Moreover, such topological naming is not recommended in high mobility IoT networks where device connectivity often changes.

Hybrid Naming Schemes: Ascigil *et al.*[30] proposed a keyword-based naming scheme in ICN for IoT applications that allows data retrieval from multiple sources without breaking the one-interest one-data rule. The proposed naming is a hybrid scheme and mainly has three parts: *(i) Hierarchical part* that follows the native NDN hierarchical names and usually describes the IoT domain, *(ii) Function part* that consists of a single tag and defines the used function to allow multiple data retrievals and node-local processing, and *(iii) Hashtags* is the last component which is, in fact, hashtag-like keywords separated by ' / ' and describes the needed IoT data to be retrieved. The main drawback of the proposed scheme is the length of name and the number of allowed keywords that may have an impact on the content reliability. Also, processing in-network aggregation without providing any trust model can directly affect data privacy. Arshad *et al.*[81] proposed a hybrid naming scheme for IoT that addresses smart campus use cases. The authors used four parts in the name: *(a) Primary Root Prefix*: to indicate the application type e.g., smart cities, Smart Transport, Smart Home. *(b) Hierarchical Components*: to combine campus information in a hierarchical NDN format, information such as campus location, content originator ID, and content type are listed at this level. *(c) Attribute Components*: to describe detailed information about the content itself such as content properties and task type. *(d) Flat Components*: to provide secure and signed names by using hash function. Although this scheme has a lot of information about the content and its properties, reaching a scalable and fast lookup process is a challenging issue, due to its length.

Local IoT Communication Support: Yang *et al.*[82] proposed a Local Naming Service to perform local communication in a resource-constrained IoT environment. The idea consists of making an inter-conversion between a sensor and a network sink, and convert the content-prefix to prefix-code. The sink node maintains a *Prefix Code Table* to map the content-prefix with the prefix-code. If the communication is local, only the code prefix is used in name, and

to ensure global communication, the sink converts the prefix-code with the original content-prefix. The major issue with this mechanism is that the node has to update all nodes with the new prefix-code whenever new data has been created. This will significantly decrease the network performance, and affect the communication.

2.6.2 Forwarding & Content Discovery

From the ICN perspective, the routing plane is used to set and update the network topology, handle their long-term changes, and update the intermediate node's forwarding tables. NDN and CCN architectures also use the forwarding plane to rank and probe faces used in the routing/forwarding process. The substantial difference between the routing and forwarding is that the former is used to decide about route/path availability, while the latter is used to decide about the preferences to use the best path based on different metrics. The existing forwarding schemes have been designed with the goal to forward the interest to the near provider. However, this is not always feasible in highly dynamic networks. An efficient forwarding scheme should select the best forwarding interface in a dynamic manner taking multiple metrics into consideration (e.g., link utilization, in-networking caching, mobility, etc.)

Forwarding Solutions: The forwarding plane is responsible to detect failures, perform recovery, and act as a control plane. NDN Forwarding plane may have different forwarding strategies that perform all decisions needed for each interest and data packet and use multiple forwarding options to choose the best face toward the content provider in an efficient manner. Also, network developers can develop different strategies based on the used network environments and context. Ascigil *et al.*[30] used a hybrid naming scheme by combining both NDN hierarchical names with hashtag-like keywords. A hybrid routing process has been introduced in the same work. The authors proposed to use generic NDN forwarding scheme to route the interests between Internet domains, and a modified version of TagNeT[83] to achieve local IoT routing. They used tags included in the name and exclude the hierarchical part in local routing, which is used only outside the local network. Melvix *et al.*[84] focused on the forwarding strategy in NDN-based IoT applications, where context-based forwarding has been proposed. This strategy determines the context of the requested content, the corresponding application's tolerance, and uses of the cached data to make an intelligent forwarding deci-

sion. By enabling all these features, the forwarding process will reduce the node's power consumption.

As NDN packets exchange follows a pull-based model, pushing the data from the consumer without interest is not possible, and the data packets will be considered unsolicited and dropped immediately. Persistent interest[85] has been proposed to keep interest for a longer time in PIT table, and establish a long-lived path. Hence, data packets can flow from the producer without continuous interest requests. Moll *et al.*[86] proposed an adaptive forwarding strategy for persistent interest, by using information from the FIB table such as reachable status, corresponding face, costs, and probing results. The idea is that the consumer sends a probing interest to calculate the delay and loss, in order to rate and evaluate the performance of the different paths through the network.

The forwarding plane has been proposed to overcome routing scalability and stability issues in IP-based networking. The NDN forwarding concept aims to forward the data without continuous updates in FIB table. More investigation is required to provide scalable and efficient forwarding schemes that support both IoT push and pull traffic.

Content Discovery Mechanisms: The structure of the generated and shared data between IoT devices is still a challenge due to the heterogeneous nature of devices and naming schemes. To overcome this issue, Quevedo *et al.*[87] proposed an ICN-based discovery mechanism that uses semantic matching[88] to provide a flexible and less-complex discovery process. In addition to *Clients* and *Service Providers*, the authors defined *Discovery Brokers*, which are responsible to manage information regarding the available services and incoming queries. Semantic Matching Engine is used to keep track of registered services by Service Providers and matches incoming queries with available services. Similarly, Ascigil *et al.*[89] addressed the content discovery in cache-enabled nodes, by using an opportunistic coordination approach that does not require any further signaling or update protocols to locate the cached content. It uses a new data structure on each router namely, *Ephemeral Forwarding Information Base* to keep track of the recent direction of the data chunks. Moreover, a budget-based multicast forwarding strategy has been used that consists of giving a forwarding budget to every interest packet to forward the interest toward on-path replica-node and/or the original content provider.

2.6.3 In-network Caching

In-network caching is one of the fundamental features to support content-centric data-delivery model in ICN. Using in-network caching either on or off-path will improve the data availability in the network, without requiring the content producer and consumer to be connected. Hence, any node in the network can serve the content requested[90]. The main advantages of using in-network caching in ICN are: a) dissociate the content from its original provider, b) by making multiple copies of the content in the network, requests can be served by any replica node in the network, and may reduce the overhead at the provider side and avoid a single point of failure, c) facilitate multicasting and retransmission due to packet loss, and d) improve content retrieval and reduce the network delay and latency. The existing in-network caching schemes focus on improving a single objective (e.g., end-to-end delay, cache hit, etc.), and they are centralized and not suitable for IoT traffic. Customized schemes that support IoT traffic, work on large-scale networks and in a fully distributed manner is extremely desirable.

Cache Content & Placement Strategies: ICN nodes need to decide whether the processed content (data packet) should be cached in the intermediate node or not. Hence, various cache placement schemes have been proposed such as: *Leave Copy Everywhere* (LCE)[14], *Copy with Probability* (LCE-Prob)[91]. LCE is an in-built NDN scheme that consists of caching and keeping a copy of the content in all the routers in the path from provider to consumer. The main drawback of LCE is that the same content is cached many times, which by consequence reduces the cached content diversity. To overcome these issues, LCE-Prob caches the content with a given probability $p = 1/(hopcount)$. Keeping IoT content characteristics and device limitations into consideration. Vural *et al.*[92] also discussed the in-network caching from IoT perspective. Because of the nature of IoT data, caching cannot be applied in similar ways as that of Internet traffic. Hence, the authors considered different metrics (data popularity) to decide if IoT data content should be cached or not. Furthermore, they discussed trade-off between retrieving freshly generated content from the original provider that is at multiple hops from the consumer or fetching the cached (not as fresh) content from replica-node with fewer hops. Different metrics are used in the study such as content lifetime, time range of incoming requests, and hop distance to the content source and

requesters. However, this work focuses on electrically powered and static devices ignoring the resource-constrained characteristics. Xu *et al.*[93] studied the content caching and updating in IoT networks. The authors formulated the content caching and cache store updating as a mixed 0-1 integer non-convex optimization problem and proposed a Harmony Search based content caching and updating algorithm. The proposed algorithm takes content freshness and device energy into consideration. The main limitation of this study is that the authors ignored other parameters of wireless devices, such as mobility and topological changes. Naeem *et al.*[58] designed a periodic caching scheme for smart cities where each node has a Distinctive Statistics Table. This table is used to find out the most frequently requested content based on a set of metrics (e.g., content name, frequency count, recently requested time, etc.). The use case assumed in this work is that of smart cities, however, within a smart city other use cases may be integrated, such as smart grid and transportation systems. It is not clear, if the solution can be utilized across the board or will suffer from limitations due to different use case characteristics. Zhang *et al.*[94] proposed a cooperative IoT caching scheme, based on data lifetime and content access rate to enhance content dissemination & reduce energy consumption. The authors introduced a slide time window to measure the request rate, and a local threshold to cache content. However, the selection of threshold value is an open question and may be different for different types of content/traffic. Moreover, this scheme is not applicable to content generated for single use. Kim *et al.*[95,96] discussed the cache poisoning attacks and suggested a verification scheme to minimize unnecessary verification. The authors proposed to verify only the served content and favors already-verified content in the content content-store. From the NDN perspective, the original content provider signs its data content, and consumers verify it. Hence, each and every data packet contains a digital signature. By using this mechanism, the intermediate router will discard poisoned content if they perform a signature verification, however, providing verification process in each router is expensive in terms of processing and decreases the overall performance.

Cache Replacement Schemes: Due to the space limitation of the content-store, some content should be removed to make room for new content. Hence, the cache replacement scheme consists of deciding on which content should be removed from the cache store[97]. *Least Recently Used* (LRU) scheme is well-known in NDN and most used. LRU consists of keeping the most recently used content used and removes the least recently accessed from

the CS. *Least Frequently Used* (LFU) consists of removing the less frequently used contents from the CS, hence only the most frequently used content is kept in the store.

From caching perspectives and increasing its benefits, only popular content should be cached. This may improve network performance, especially in highly dynamic networks such as IoT. Content popularity-based caching scheme has been proposed in[98], where authors created a new data structure that cooperates with *Content Popularity Table*, to store information about the cache hit, and content's previous & current popularity associated with its name. Their experiments show better results as compared to LRU and LFU schemes in terms of cache hit ratio, network capacity, and load. However, the scheme consumes more CPU that may not be suitable for large-scale network and resource-constrained devices. This issue has been addressed in[99], by proposing a line speed Bloom filter-based method to capture the content popularity in a continuous manner, and at the same time minimize the memory usage and overall resource consumption. However, the proposed method has a major issue with the storage cost. Meddeb *et al.*[100] designed Least Fresh First cache replacement scheme for IoT applications that integrates content freshness as the main element. The authors adopted Auto-regression Moving Average model[101] to predict the content freshness. However, using the freshness parameters is not enough in IoT context, other metrics (e.g., content popularity) must be taken into consideration.

2.6.4 Publish-Subscribe Communication

Publish-Subscribe model is widely used in IoT applications, where a group of users subscribes to a topic or content offered by a publisher. For each newly generated data element, some solutions recommend the publisher to push the data toward these subscribers. Moreover, in case of new events triggered at the publisher's side, an event report is sent towards the subscribers. These solution is not feasible in NDN since it violates the NDN's working principles and requires huge changes to the core NDN forwarding plane. It is important to note, that traditional IP networks implement a topic subscription system at the application level. The same can be used in ICN, however, each message under a topic is considered as independent content with a unique name. Hence, the ICN-IoT solutions available in literature usually focus on a content-based pub/sub model, where each piece of content when updated,

has to be delivered to consumers.

Persistent Interest Solutions: Deploying publish-subscribe communication in NDN is challenging. Various solutions have been proposed either by proposing long-lived persistence interest to allow flow of multiple data packets for one interest or adding an exception in the forwarding by allowing unsolicited data packet to be forwarded without any previous interest.

Amadeo *et al.*[102] proposed three different strategies to deliver push-based IoT traffic: a) the publisher sends an interest packet for periodical and event-triggered notification, that includes the data in interest name components, b) by sending unsolicited data without any interest, and intermediate nodes validate the content signature rather than checking the PIT table, and c) use Virtual Interest polling, where PIT entry has a long lifetime. Tagami *et al.*[103] proposed a Content-Oriented Pub-Sub system in fragmented networks by introducing subscribe/publish messages in the network, and a Rendezvous Point that allows subscribers to receive generated content even if they were off-line. Furthermore, intermediate nodes maintain a subscription state similar to IP Multicast to allow multicast forwarding. IoT can benefit from this solution, as many use cases of IoT are small networks. Tsilopoulos *et al.*[85] proposed the use of two customized types of interest packets to support different type of information, by introducing a channel concept which is a Transmission Control Protocol-like session created by the network to deliver data. Then, a reliable notification is initialized by the publisher upon content creation, and a semi-persistent interest packet that is stored in PIT table to satisfy multiple data packets. It cannot be removed until the subscriber removes it explicitly. The channel forwards each packet belonging to this session and follows Stop and Wait Automatic Repeat Request algorithm. Moll *et al.*[104] studied the use of persistent interests to support push-based communication in Interest-based ICN architectures, while using their earlier work[86] on forwarding. The authors proposed an adaptive persistent interest forwarding scheme to overcome the long-lived path, and extend the hierarchical naming scheme by adding the sequence number at the end of the name. A persistent entry is refreshed only when another persistent interest is received. As persistence in PIT is not the generic working of NDN or CCN, hence the adoption of such a mechanism requires large scale deployment and standardization.

Other Pub-Sub Solutions: Different solutions have been proposed in[105–107] to support

push-traffic in VANETs. The authors in[105] allowed vehicles to produce and process unsolicited packet to forward emergency messages to the network without any previous interest. While[106] proposed that a vehicle sends a one-hop beacon message that is a special interest packet. Neighbor vehicles create a temporary PIT entry when receiving the beacon and cache the incoming unsolicited data instead of discarding it. A dynamic PIT entry lifetime scheme has been proposed in[107] that calculates entry lifetime timer dynamically based on the hop count and interest satisfaction rate. Li *et al.*[108] proposed a distributed publisher-driven architecture targeting secure data sharing among IoT devices. The authors suggested creating for each content a set of attributes (i.e. attribute manifest and data manifest) that can be cached in the network and retrieved from a distributed set of nodes. The authors also proposed Automatic Attribute Self-update Mechanism to update the already published attributes. However, this scheme lacks a mechanism to handle the short-life duration of IoT content. Gundougan *et al.*[109] designed a robust and resilient publish-subscribe model for IoT applications. The authors suggested selecting the stable nodes as Content Proxies to act as a persistent cache point. Publisher nodes push the content to the Context Proxies where subscribers can fetch it with lesser delay. However, this model has various issues with the mobility of subscribers or even the content proxy. Also, content encryption and access control have not been addressed.

2.7 Conclusion

In this chapter, we have defined the building concepts used in this thesis, mainly, IoT applications and the ICN paradigm. We have also presented, in this chapter, the principle ICN features and an overview of NDN architecture. We have also detailed the existing ICN solutions from IoT domain-specific applications, such as smart cities, smart homes, smart grids, and smart transportation. In each domain, we have illustrated a use case scenario, the current ICN efforts along with a summary and insights. Furthermore, we have detailed the existing ICN issues from IoT including data/content naming, forwarding and content discovery, in-network caching, and publish-subscribe communication. In the following part, we will present our contributions to provide an ICN optimized communication for IoT. In the following part, we will present our contributions within the content naming, publish-subscribe, in-network content caching, and named control protocol in ICN-IoT networks.

Chapter 3 A Unified Hybrid Information-Centric Naming Scheme

3.1 Introduction

Current Internet applications and use cases are now shifting toward the content-centric paradigm, where the content is the key element in the infrastructure. In this chapter, we design a hybrid multilayer naming scheme with multi-component hierarchical and attribute-value components for a data-centric Internet of Things. The proposed scheme targets smart IoT applications, and compared with the existing works it provides a set of built-in features such as scalability, efficient routing, and security features. We incorporate a variable-length encoding method, with a prefix-labeling scheme in order to describe hierarchical location names with various embedded semantic functionalities. Moreover, the scheme supports fast local IoT communication using the Name-to-Code translation concept, as well as multi-source content retrieval through in-network function.

This chapter is organized as follows; we first discuss the requirements and existing challenges for a unified name scheme for IoT-ICN applications. Then, we describe in detail the proposed naming scheme and its working principle. Finally, we present the implementation and evaluation part.

3.2 Naming Requirements & Challenges

The naming scheme is the most critical component in ICN networks, not only because it identifies content but also it has a direct impact on other networking functionalities.

Content, Service, and Device Identification: Besides content naming, naming the service that offers the content is also necessary to distinguish between different services running on one or many devices. In addition, device/sensor naming in an IoT environment is also as important as content naming, especially for monitoring and management purposes.

Short Length Names: The use of long, variable, and unbounded names may consume more memory and affect the lookup process, thereby affecting the network scalability. Moreover, most IoT services generate data that is smaller than the naming itself.

Efficient Name Aggregation Rules: Unlike the IP address, ICN lookup process must match a prefix at the end of a component rather than any digit. The use of textual names affects not memory consumption but also the lookup process. Thus, it is important to design a careful naming scheme that takes name aggregation feature into consideration with a fast lookup process.

Multi-Source Data Retrieval Support: In an IoT environment, multiple services may provide the same content. Retrieval content from these services with one request is not feasible, consumers may end up by sending multiple interest packets. A unified naming scheme must support data retrieval from multiple sources in an efficient manner.

Despite the existing naming efforts, there is no clear answer about the appropriate naming scheme for IoT. Hierarchical names benefit from an easy name-aggregation and better scalability support; however, they require more space due to their length. Whereas flat names have a fixed length and may consume less-space, but they lack from scalability support and name-aggregation. A hybrid multilayer scheme with self-certifying flat names and a collection of keywords has been recommended in different studies. Such a scheme can be suitable for IoT applications, enhance content discovery & delivery, and improve the network performance. Also, the used scheme should support a fast lookup process, multi-source data retrieval, name persistence & security binding, authenticity, and global uniqueness.

It is worth nothing that most of ICN features such as routing, forwarding, caching, and security are based on the used naming scheme. Coupled with IoT applications, the naming should explicitly identify services, content, and devices in a highly expressive and customizable manner. Furthermore, the heterogeneous nature of IoT devices and applications (e.g., wearable, mobile, sensory, etc.) adds more challenges towards unified naming. Naming IoT devices is as important as naming the content, to manipulate them (e.g., switching a device ON/OFF, updating the firmware, etc.), overcome the heterogeneity challenge, and present an interoperability abstraction between different vendors and across domains. Moreover, a unified naming scheme should natively support efficient aggregation rules, fast lookup process, and dynamic content identification.

3.3 Unified Hybrid ICN-IoT Naming Scheme

In the following, we describe the design of the proposed hybrid ICN naming scheme (*H-ICNIoT*) for smart IoT applications, in which we aim to merge the features from flat and attribute-value names in a hierarchical structure. Furthermore, *H-ICNIoT* incorporates variable-length encoding with a prefix-based scheme for the topological-hierarchy location as well as a set of attributes, and fast and scalable local communication using *Name-to-Code* translation mechanism. The objective of this naming scheme is to provide a unified content, device, and service naming in an IoT environment within next-generation ICN architecture.

3.3.1 Network Reference Model

The main objective of IoT is to extend and allow connectivity between physical objects and the Internet, and ensure the interoperability among them. IoT devices can provide data sensing, collecting, and sharing among devices and Internet, that can be interconnected using wired or wireless technologies. This work considers an IoT network topology that involves different IoT applications including sensing and automation, as well as networking functionalities such as routing, mobility, quality of service, security & privacy, mobility, and heterogeneity of the environment.

We divide IoT network into a collection of sub-networks named *ICN-Net*, as shown in Figure 3.1. An *ICN-Net* is defined dynamically and is bounded by its gateway. It may comprise of different IoT devices, sensors, and actuators. Hence, we categories IoT devices

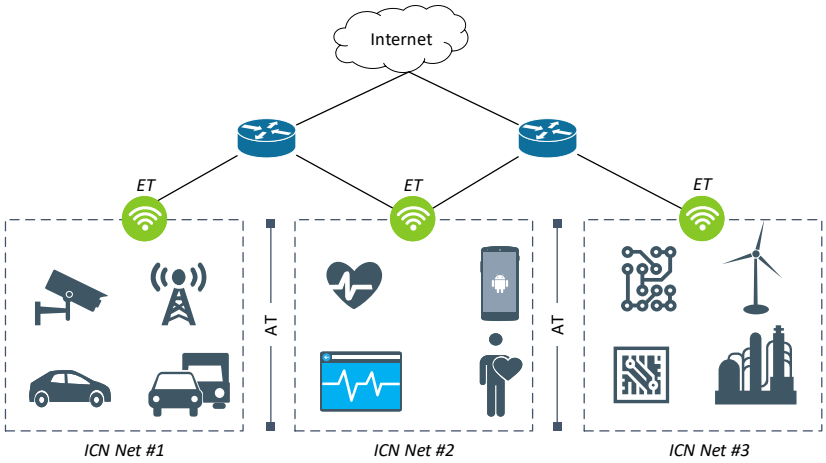


Figure 3.1 Virtual IoT topology.

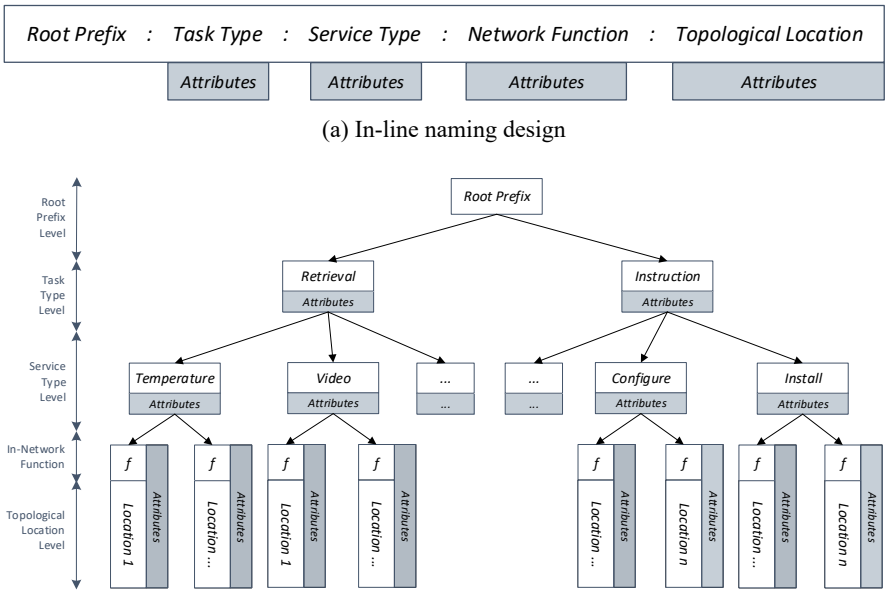
into *Access Thing* (AT) and *Edge Thing* (ET). An AT is a standard *thing* in the topology that can be a sensor or an actuator, while an ET acts as a gateway in-between ICN-Nets and/or Internet Service Provider. In highly dynamic environments, the ET can be mobile, and more than one attached to an *ICN-Net* for load balancing or in-network caching. For example, road side unit, home access point, and patient's mobile phone can act as ETs, while vehicles, home actuators/sensors, and in-body sensors can act as ATs.

3.3.2 Multilayer Multi-Components Design

Figure 3.2(a) illustrates an in-line *H-ICNIoT* naming design which is an IPv6-like naming with colon as a delimiter to separate each two components, while Figure 3.2(b) depicts the top-down multilayer naming scheme. We define four levels to identify different application semantics:

1. Root Prefix Level: aims to define the core domain or network prefix, for example, city name in a smart city, home name in a smart home, hospital name in a smart healthcare scenario.

2. Task Type Level: defines the IoT data namespace. Based on the requested *Task* to



(a) In-line naming design

(b) Multilayer naming design

Figure 3.2 Multilayer multi-components design.

perform. We propose two classes; *Retrieval* tasks such as on-demand sensing data, periodic monitoring, and content fetching; and *Instruction* tasks like action triggering, executing a configuration/installation, or event-triggered alarms.

3. Service Type Level: defines the *Service* to be performed such as temperature sensing, video frames fetching, turn *ON/OFF* light, or sending a configuration commands.

4. In-Network Function and Topological Location Level: identifies the used in-network function to allow data retrieval from multiple sources, as well as the physical location of the service regarding AT/ET level.

Level Attributes: In addition to the multi-level design, we used a collection of attribute-value pairs in each level, to store in an efficient way different content/service properties such as action type, security, owner, etc. These attributes play an important role to enforce content security, preserve user/communication privacy, etc., that can be saved as keyword or hash-value of attributes. Some of them are well-known and required, while others may be generated dynamically during the content creation or requesting.

Based on the attribute role, we classified them into three types of properties as presented in Table 3.1: the *Owner* properties provide more information about the service/content/thing owner, by sharing its signature, public key, etc. Whereas the *Thing* properties contain related information about the identification, location, security, and other physical thing's attributes. Finally, the *Action* properties combine the tasks and the associated actions with the eventual action type, etc.

It is noteworthy to highlight here that the properties and hierarchy levels are not the same. Properties gather different attributes together for better classification, while the hierarchy

Table 3.1 Attribute properties.

Attribute	*Example*
Owner Properties	`{Signature: …, Public Key: …, Seq Num: …, Hash: …}`
Thing Properties	`{Signature: …, Public Key: …, Seq Num: …, P-Level: IN/OUT}`
Action Properties	`{Type: Standard, Action: ON, S-Level: Auth, P-Level: IN}`

provides a complete naming view.

Finally, as IoT applications have not the same usage, we prefer to give the application implementer/designer the options to choose the number of used attributes for each service. The main advantage of using Attribute-Value pairs is to include as many different information required in the name without affecting the lookup performance. The used names and values could be used in plain text format, hashed/encoded, machine-readable, or encrypted. In this work, we only provide the classification of properties and guidelines for attributes. More attributes can be added and/or modified to the following list:

Well-Known Attributes: include *Action*: Action name to be performed on the thing, *Type*: Type of action (standard, system, reserved), *Security Level*: Open or Authenticated action, *Privacy Level*: Access from IN/OUT of network, *Location*: The location of the thing, and *Owner Sign*: The owner of the information.

Optional Attributes: include *Sequence No*: To ensure the freshness of data, *Cache Lifetime*: If/how long can the information be stored in repository, and *Meta-Data*: For both Owner and the Thing itself e.g. hash, certificates, keywords, content meta, etc.

3.3.3 Name-to-Code: Fast Local IoT Communication

Most of IoT sensors generate periodic data with long name-prefix and additional information such as the sensor location and timestamps. Furthermore, IoT traffic is locally consumed in the network rather than globally. Hence, using long names in local communication may require more processing, storage, energy, and bandwidth utilization. To achieve an efficient name lookup in a highly dynamic network such as IoT, we propose to translate the local used name-prefix from a long name to a small hash code value. IoT traffic generated for local usage only utilizes the code value instead of the long name-prefix.

At the ET level, we define a new data structure namely *Name-Code Prefix* (NCP) table. NCP table contains the *Original Name-Prefix*, the *Code Name-Prefix* generated by a hash function (e.g., Cyclic Redundancy Check – CRC, Secure Hash Algorithms – SHA, etc.), and a *Lifetime* field to keep the NCP table clean from un-used names. Figure 3.3 illustrates local communication using NCP table that involves communication between AT and ET in order to register a name and use it in the communication, and between ATs to update the local routing

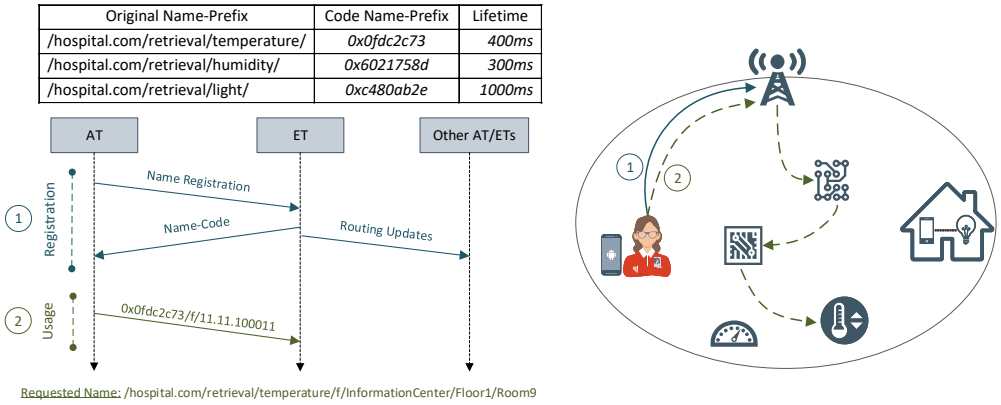


Figure 3.3 Local IoT communication support.

tables with the new Name-Code value. In the following, we describe the process in detail.

(a) Name-Code Registration: When a sensor generates/creates a new service/content, it should first register the name prefix in the NCP table running at the ET level (left part of Figure 3.3). Specialized interest and data packets are used for this task. The communication protocol used in the registration phase is based on a one-interest one-data exchange primitive between the AT and ET. A specialized interest packet is triggered from the sensor level using the predefined service name running at the ET level. After receiving the special interest at ET, the NCP service creates *Code-Name Prefix* value for that name-prefix using the hash function. This Code-Name value should be unique in local communication. At this stage, a special data packet is sent back to ET carrying the proposed Code-Name value, meanwhile, the ET node triggers a new routing update via the used routing protocol in order to inform all FIB tables about the new Name-Code value.

(b) Name-Code Usage: Any future interest packet used in local communication uses the Name-Code value rather than the long name-prefix. Intermediate nodes can forward packets using the Name-Code value. It is important to note that security and privacy-related information are bound with the content and its original name rather than the communication channel. Hence, they are preserved at the network level via the content name. The right part of Figure 3.3 illustrates a scenario of using NCP process. After registering the long name prefix in NCP, sensors use only the Code-Name value in interest and data packets. For example, the following name `/hospital.com/retrieval/temperature/` becomes `0x0fdc2c73`.

It is important to note here that the Name-Code concept is used only in case of local

communication, if the ET is required to forward the interest out of the local network, it should implicitly replace the Name-Code value with the original name prefix. Moreover, whenever the ET node forwards data packets for specific content, the associated NCP lifetime entry will be refreshed. This ensures that the content is still available. Finally, as we use a hash function to generate codes, the collision domain is negligible because they are local to ICN-Net and the probability of generating collisions is mainly based on the performance of the used hash function, which is efficient enough to make it collision-free. In this work, we use CRC-8 that generates a 9-bit hash-code.

3.3.4 Topological Location Labeling & Encoding

To retrieve content in ICN, consumers must be able to deterministically construct the name for a desired piece of data without having previously seen the name or data. This naming scheme is very challenging in dynamically generated content. To solve such a challenge, researchers are focusing on structured naming such as topological location-based naming. This naming scheme is very efficient for non-mobile devices and sensors. For example, sensors in a smart home or smart building are fixed at well-defined places. These places can be defined in terms of the building name, floor number, and room number (physical-based naming), e.g., `/university-domain/campus/school/building1/floor10/room109/`. While mobile IoT devices may use another type of topological location names based on the application context. For example, in smart health-care, the patient's body may play the role of the network, and sensors can be named based on that (logical-based naming), e.g., `/patient-ID/body/sensor1`. In such an application, the same concept is used but from another perspective (from the patient's point of view). The same naming concept can be expended to other smart IoT applications such as smart vehicular networks, smart parking, etc.

In the previous section, we discussed the location as an attribute embedded in the naming scheme. While in this section, we propose a device/location name labeling and encoding schemes in order to provide a fast, unique, and compressed variable length names. This method reduces the attributes/components redundancy, identifies in a simple way devices/services, and enhances the routing and forwarding. Thus, the location level, shown in Figure 3.2,

is used in the routing and forwarding process with the ability to apply name aggregation to reduce the FIB size and provide fast lookup process.

Location Labeling: As discussed in the network model, we consider an IoT network as a collection of *ICN-Net*s, for example, in a smart hospital IoT infrastructure, virtual *ICN-Net*s can be created for each room, office, and lab with at least one ET. Each *ICN-Net* can comprise a set of sensors and actuators as ATs. These *ICN-Net*s can be grouped at the floor, and the building level; according to the size of the hospital, we can also apply block-level regrouped, and ultimately hospital level. Hence, the following smart hospital logical hierarchy over the physical network can be defined as:

$$hospital \rightarrow block \rightarrow building \rightarrow floor \rightarrow room \rightarrow thing.$$

This location scheme can be modeled using a tree structure (Figure 3.4), which simplifies the forwarding process and identification. However, the major challenge of this scheme is the length of names, which can be a long name that consumes large memory and affects the lookup operation. In the following, we detail our proposed solution to overcome this issue, so the length of the name can be kept as minimal as possible, that by consequence will enhance the scalability and efficiency. It is important to note that we use the example of a hospital to elaborate the technique. However, the mechanism can be applied to other application scenarios also.

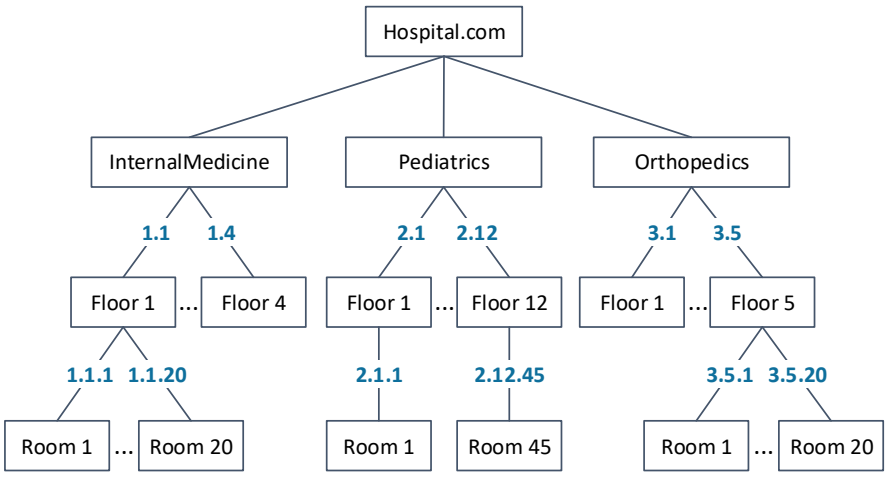


Figure 3.4 Hierarchical location names with prefix-based labeling.

Location Encoding: We have used variable-length location encoding along with a prefix-based scheme. The idea consists to assign each component (label) a variable-length number. Instead of encoding the entire location, each node locally encodes its own label into locally unique binary numbers. The size of each encoded label is primarily determined by a specific variable-length encoding method, that can be different from an *ICN-Net* to another.

Step 1 : Prefix-based Labeling Scheme: Decimal Classification is considered as one of the most simplest algorithms for prefix-based labeling, this scheme can easily be applied on a tree structure. Figure 3.4 provides an implementation example:

Let T a tree with root r.

Each node $n \in T$ is identified by $key(n).pos(n)$, where

$$key(n).pos(n) = \begin{cases} 1 & \text{if } n = r. \\ key(v).pos(v).i & \text{if } n \text{ is } i^{th} \text{ child of node } v. \end{cases} \tag{3.1}$$

Step 2 : Variable-length Encoding Scheme: The second step consists of applying a variable-length encoding method. Here, we prefer to use Fibonacci encoding[110], which is a universal code that may be used for dense codes for large word-based dataset. It is particularly a good choice for compressing a set of small integers, and fast decoding as well as compressed searches. Furthermore, Fibonacci codes are robust even against insertions and deletions, which means they are robust in terms of correcting errors. A Fibonacci sequence can be defined by: $F_i = F_{i-1} + F_{i-2}$, for $i \geq 1$, where $F_{-1} = F_0 = 1$.

The use of Fibonacci sequence (excluding the first 0 & 1) can generate a binary code word. Let n be the number to be encoded, if $d(0),\ d(1),\ \ldots,\ d(k-1),\ d(k)$ represent the digits of the code word associated to n, the Fibonacci binary encoding $V(f(n))$ can be generated by:

$$V(f(n)) = \sum_{i=0}^{k-1} d(i).F(i+2), and : d(k-1) = d(k) = 1. \tag{3.2}$$

$F(i)$ represents the i^{th} Fibonacci number, while $d(k)$ is always an appended bit of 1. In essence, the code represents 1 bits for Fibonacci sequence that can be summed to represent n, with a 1 appended at the end. Examples: $V(f(4)) = 1011, \quad V(f(7)) = 01011, \quad V(f(32)) =$

Table 3.2 Encoding process example.

Location	*Prefix Label*	*EncodingLocation*
/hospital.com/InternalMedicine/Floor1	1.1	11.11
/hospital.com/InternalMedicine/Floor1/Room2	1.1.2	11.11.011
/hospital.com/InternalMedicine/Floor2/Room29	2.2.29	011.011.00001011

$00101011, \quad V(F(143)) = 01010101011.$

In the following, we illustrate the used steps to encode a number n and generate its associated Fibonacci code string:

- If f is the largest Fibonacci number less than or equal to n, we prepend 1 to the final binary string. This implies the usage of f in representation for the number n. Then, we deduct f from n: $n = n - f$.
- Otherwise, if f is greater than n, we prepend 0 to the final binary string.
- We select the next Fibonacci number smaller than f.
- We repeat the previous steps until zero remainder ($n = 0$).
- At the end, we append an additional 1 to the binary string. The encoding word is obtained when two consecutive 1s appear. This also means the end of the current number and the start of the next one.

By the end of the encoding process, the lengthy name for a names is encoded to small binary values as shown in Table 3.2.

3.3.5 Multi-Source Content Retrieval Support

Various IoT scenarios (e.g., monitoring applications) may be interested in on-demand generated data (non-stored data) for a particular event or a specific geographic area regardless of who produces the data (in terms of the number of producers), where many providers may reply to the requested data. For example, asking about the average temperature of the whole building. In the following, we present the proposed multi-source data retrieval mechanism

Naming	
Long Name	/hospital.com/retrieval/temperature/avg/InternalMedicine/
Short Name	/0x0fdc2c73/avg/11/

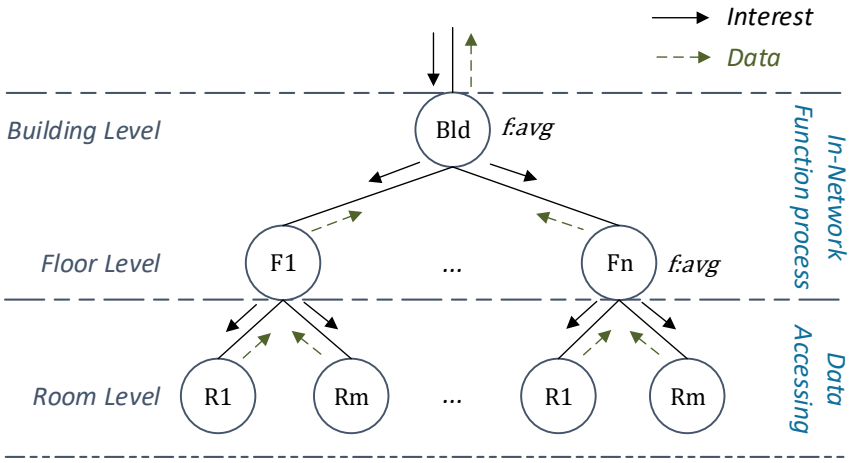


Figure 3.5 Multi-source data retrieval scenario.

on top of the naming scheme without violating the native one-interest one-data (one-to-one) NDN design primitive by using *in-network function* concept.

In-Network Function: Due to the one-interest one-data packet concept in NDN, enabling multi-source data retrieval is not an easy process as it may end up sending a storm of interest packets and overhead in the network. This also requires a significant change in architecture design. For this, we propose use a distributed *in-network function* concept to allow fetching the data from multiple sources without violating one-to-one primitive.

For example, localized sensing is one of the important use cases where an interest packet is sent out to request the average temperature in smart building/home, monitor the patients' blood pressure in smart hospitals, or query about the road details and vicinity to avoid collisions in autonomous vehicles.

Figure 3.5 depicts a simple smart building scenario in which ET devices are fixed at the floor and building level, while AT devices are in the room level. Assuming a consumer node requests the average temperature, the forwarding strategy multicasts the request to all interfaces that may have the requested data. To do so, we include a new component in the naming scheme, namely *in-network function*. For instance, assume the following requested long name `/hospital.com/retrieval/temperature/avg/InternalMedicine/`, which is shortened as `/0x0fdc2c73/avg/11/`. The in-network function

`avg` specifies the executed function, i.e., the *average* of temperature from individual IoT nodes toward the consumers. Hence, the in-network function is executed dynamically at ET depending on the location of sensors downstream.

The concept is similar to in-network aggregation. However, the function is executed only by ET nodes rather than any AT node, to provide the original requester one value. It is important to note that the function component in the name is not mandatory, instead, it is an application-specific. A name without that function means that the in-network function should not be executed. When the ET devices receive the data packet, they execute the function and generate a new data packet toward the consumer node.

3.3.6 Execution Strategy

By design, we execute the *in-network function* at each ET node that has more than one outgoing interface towards producers. However, customized strategies can be created based on the network design and topology to specify in which node the in-network function can be executed. Each ET node maintains in-network function module, as depicted in Figure 3.6. This module is responsible for analyzing the content name (i.e., query), split it into different sub-queries, and then generate the required interest packets to forward them downstream. Similarly, it is responsible to collect the data packets, aggregate the data, and generate one data packet to forward upstream.

The network administrator may select which ETs can execute the in-network function. This selection is done based on the virtual network topology or the physical topology (according to the use case). These ETs are allowed to aggregate the published data by ATs. Hence, after receiving the data and executing the in-network function, they sign the newly generated data packet before forwarding it to the requester in order to authenticate it.

3.3.7 PIT Table Management

Supporting multi-source content retrieval with a one-interest one-data packet requires a modification in the intermediate node's behavior especially for PIT Table. For the node which executes the in-network function, the PIT entry should not be deleted after receiving the first data packet, neither the data packet should be forwarded, unless receiving responses

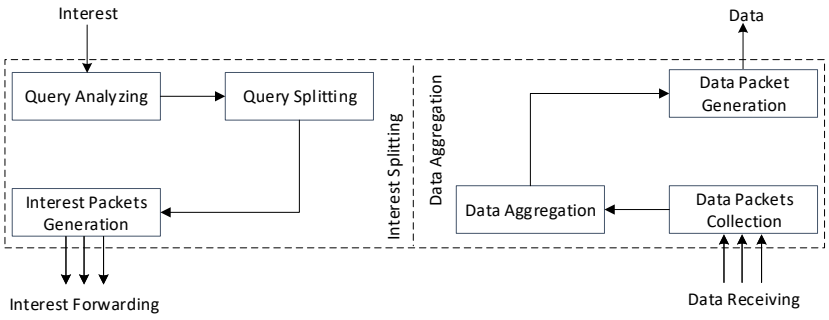


Figure 3.6 In-network function module.

for all the forwarded interests or timeout expired. In case of receiving all data downstream, the aggregation data is marked as full results, otherwise, it is marked partial result (e.g., packet losing, timeout expired, etc.). Then, the node executes the function, creates, and forwards the data packet with the new results and then deletes the PIT entry.

If the forwarded interest has been lost (e.g., packet lost) or the ET device did not receive a response downstream (e.g., timeout), the ET device executes the in-network functions using only the received information, adds a tag indicating that these data is for a partial result, and then sends it back to the requester. The requester application is responsible for re-issuing the same request or use the received data as it is. When the intermediate node receives all data packets, it tags the packet with full results informing the consumer of complete data aggregation.

Finally, as most of the IoT data are non-stored data and used for a short time, intermediate nodes can cache these aggregated data for a specific time. The content-store table already maintains the Time-to-Live tag to keep the content-store fresh. Moreover, the partial/full result tag is also used to maintain information freshness and correctness.

3.4 Implementation & Evaluation

To evaluate H-ICNIoT naming scheme, we have conducted several qualitative and quantitative evaluations. The qualitative study consists of comparing theoretically our scheme against exiting ICN architectures and naming solutions from different perspective such as human-readable feature, name aggregation, services/content identification, etc. The quantitative evaluation consists of implementing the proposed naming scheme with ndnSIM[43],

an NS-3 based module for NDN. The implementation includes: the naming labeling and encoding algorithms, customized forwarding schemes for multi-source content retrieval, and dedicated Name-to-Code process at the ET devices for local communication process. Toward this, we have defined customized applications using the `ndn::AppHelper` module, and forwarding strategy using the `nfd::Forwarder` class. Other classes have been also modified to ensure name encoding such as `ConsumerCbr` and `Producer` applications.

The objective is to compute the efficiency of the whole process and evaluate the efficiency, and scalability of the proposed naming scheme. For this purpose, we have evaluated the following parameters: name memory consumption, lookup time, the required number of interest for one round data retrieval from multiple sources, average times data packets are forwarded, as well as the required time to collect the data from multiple producers.

3.4.1 Qualitative Analysis

Table 3.3 summarizes the comparison of H-ICNIoT with other naming types. By combining hierarchy naming with multilayer attribute-value pairs, the proposed naming provides more flexibility in term of adding data integrity, readability aggregation, identification, naming, and security features to the IoT devices and the services/content they provide, that cannot be added by design to the basic hierarchical naming schemes unless adding extra functionality at the application layer.

The major difference is that H-ICNIoT by design support various features like device naming, multi-source data retrieval, local communication, security, in-cache/metadata support, while others have added more components to the architecture to provide some of these features.

3.4.2 Quantitative Evaluation

In the quantitative evaluation, we have implemented the naming scheme to carry out a sensing task scenario: fetching the average temperature value of the whole *internal medicine* in a hospital, including 4 *floors*, where the number of *rooms* (sensors/producers) per floor varies from 4 to 16. The application service flow chart is shown in Figure 3.7. The application is configured to have four different floors with ICN-Nets at the building level. The used names

Table 3.3 Comparison of H-ICNIoT with different types of names.

Properties	*Hierar.*	*Attribute-value*	*NDN-HNS*	*H-ICNIoT*
Human-readable feature	✓	✗	✓	Possible
Name aggregation feature	✓	✗	✓	✓
Services/content identification	✗	✗	✓	✓
Device naming feature	✗	✗	✗	✓
Short names	✗	✗	✗	✓
Name-Data binding	Sign.	Sign.	Sign.	Sign., Hash
Security/privacy support	✗	✗	✗	✓
In-caching support	✗	✗	✗	✓
Meta-information support	✗	✓	✗	✓
Multi-source retrieval support	✗	✗	✗	✓
Local communication support	✗	✗	✗	✓

are given to the wireless IoT things as proposed in our design.

As it is clear from Figure 3.4, H-ICNIoT performs one-time prefix labels tree construction, based on these labels, the location encoding is performed. Table 3.2 shows encoding examples, where dots are used to make encoded names readable.

Encoding Performance: To facilitate the user's usage, the application interface shows alphanumeric variable unbounded names, but their length combined with FIB entries creates naming challenges in terms of memory consumption and lookup processing. In our proposed scheme, names are encoded to very small bit values that can save the overall FIB size. In

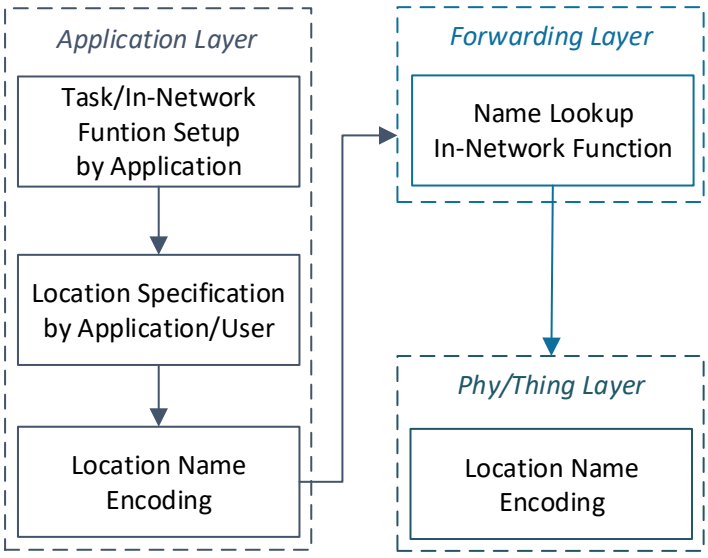


Figure 3.7 Application service design flow.

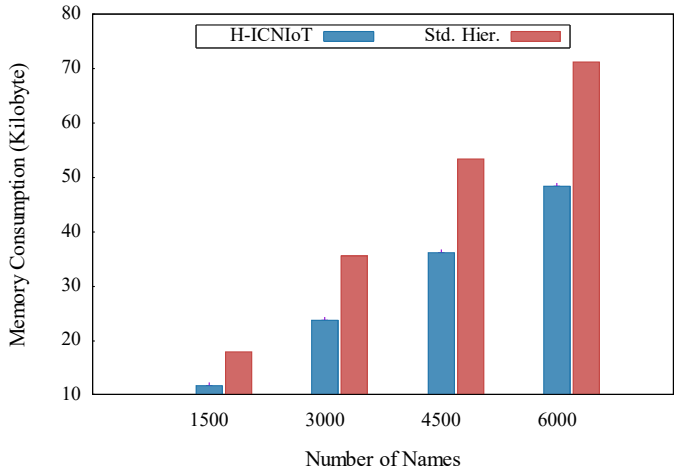


Figure 3.8 Hierarchical and H-ICNIoT names memory consumption.

the simulation scenario, we used 6480 names with different lengths. The overall size of FIB table using baseline hierarchical names is 414.41 KB, most of the names are between 25 and 35 bytes. On the contrary, after encoding names, the size of the FIB is reduced to 216.25 KB, which saves 47.82% in space, due to the use of Fast Fibonacci Encoding algorithms that takes negligible time for encoding.

Figure 3.8 shows the memory consumption in terms of occupied space by names (kilobyte) for hierarchical baseline and H-ICNIoT names. When the number of names increases, hierarchical names consume more memory which grows linearly as compared with H-ICNIoT that grows linearly and consumes less memory. This is due to location encoding based on small labels, and the use of Name-Code concept that reduces the size of names.

Lookup Performance: After populating the FIB tables with encoded names, nodes can perform lookup and routing operations on those encoded names. Finding an FIB match for a large-scale IoT network with thousands of wireless IoT devices and services can be time and energy-consuming, especially because of the limitation of resources of intermediate nodes. Although using 6480 names in our experiment, the lookup time is significantly reduced by using H-ICNIoT.

Figure 3.9 shows the lookup operation time for an exact match for ten randomly selected operations. Consistent performance is achieved across the board using H-ICNIoT, which is better than that of hierarchical names, and NDN-HNS[81].

Local Communication Performance: Figure 3.10 shows the lookup operation per-

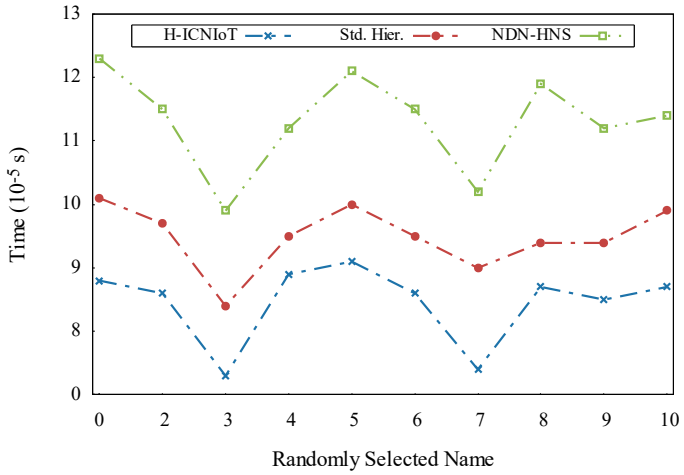


Figure 3.9 Lookup time in hierarchical and H-ICNIoT names.

formance when using the Name-Code concept to allow local IoT communication. In both scenarios, we use H-ICNIoT naming scheme with Name-Code translation and standard hierarchical names. From this figure, it can be seen that the use of Name-Code improves the lookup operation time by finding a match of small names (hash values) rather than looking for the long name-prefix. It is important to note that the effect of generating the name itself is insignificant and directly proportional to the number of producers.

Multi-Source Data Retrieval Performance: Figure 3.11 presents the evaluation performance result of our multi-source data retrieval solution using in-network function. Baseline refers to the scenario of sending one interest for each producer (retrieve one data packet). So, Figure 3.11(a) shows that the number of interest messages grows linearly when the num-

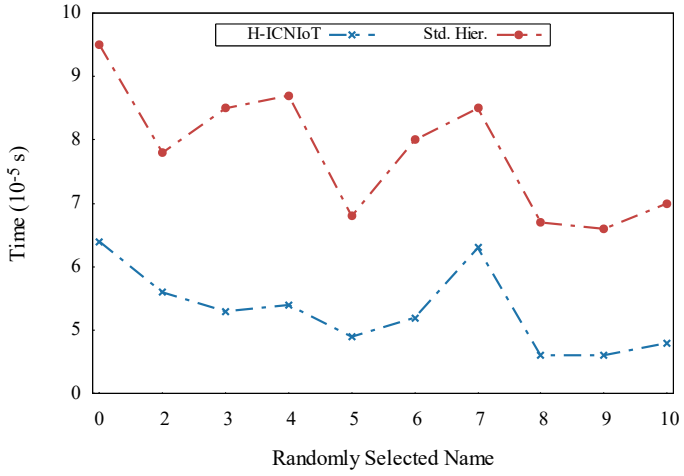
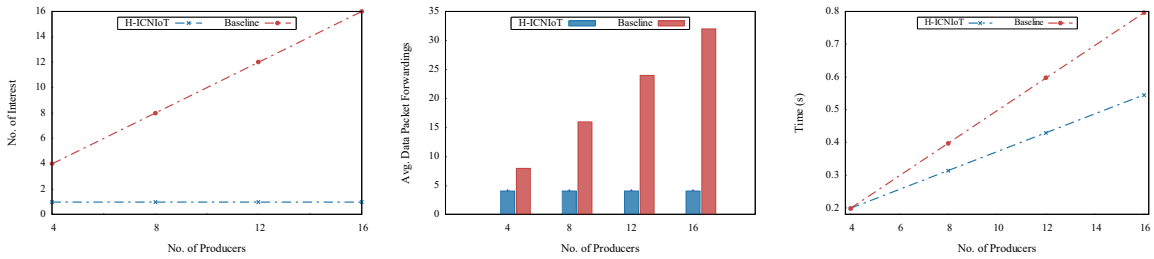


Figure 3.10 Lookup operation using Name-Code.



(a) Number of interest (b) Avg. data packet forwarding (c) Data collection time

Figure 3.11 Multi-source data retrieval results.

ber of producers increases, whereas H-ICNIoT in-network function allows only one interest to be sent and provide in-network data aggregation, therefore only one interest is used even when the number of producers increases. This results in improved bandwidth utilization and energy efficiency. Figure 3.11(b) shows the average number of times the data packets are forwarded to satisfy an interest, when in-network functions are being used by H-ICNIoT. Here, we have used the same topology illustrated in Figure 3.5 with 3 floors. For H-ICNIoT, we can notice that even if the number of producers increases, the average data packet forwarding is insignificantly effected. As the value is rounded to show hops, hence it has the same value. However, in the baseline scenario (without any in-network functions), the data packet forwarding increases as long as the density of producers increases. It is important to note that the result of the proposed scheme is insignificantly changing due to the structure of topology in the building. However, if the topological structure starts to change significantly, i.e. the tree-depth varies, then the resulting average of data packet forwarding will also have a significant effect. This result also indirectly indicates the network usage. More times the same data packet is forwarded, it will consume more network resources, and less average forwarding suggests better resource consumption of the scheme.

Finally, H-ICNIoT needs to receive all response data packets before performing in-network aggregation, whereas the baseline performs the same thing but in the application layer. We can see in Figure 3.11(c) that the data collection time for H-ICNIoT is rapid compared to the baseline scenario, especially when the number of producers increases. The error margin in case of unexpected delay is also considered in the results.

3.5 Conclusion

In this chapter, we designed a unified hybrid ICN naming scheme (*H-ICNIoT*) for smart Internet of Things applications. The proposed scheme has a multi-layer design based on the hierarchical structure. The hierarchical nature of name allows content, service, and device naming. H-ICNIoT also integrates attribute-value based names to help service identification and add more semantic fashion to names. Furthermore, we used both prefix-labeling and variable-length encoding to overcome the unbounded nature of ICN names and provide short names. Finally, we proposed a name-to-code concept to accelerate local IoT communication and use the in-network function to allow multi-source content retrieval. We validated the proposed naming scheme via a qualitative and quantitative analysis that proves, compared with existing works, the efficiency and outperformance of *H-ICNIoT* in terms of memory consumption and fast lookup process. In the next chapter, we will focus on the publish-subscribe communication model and we will present a secure architecture based on ICN network.

Chapter 4 Secure Publisher-Subscriber Group Communication Scheme

4.1 Introduction

Both NDN and CCN use pull-based communication for single interest response pairs. Content retrieval using this model is very efficient when the content already exists in the network. On the other hand, most IoT applications use subscription services for content that is being dynamically generated or will be generated in the future, which do not work well with pull-based systems. In this chapter, we propose a secure subscription scheme, which enables authentication, access control, and group management features, without modifying ICN principles. The scheme uses specialized interest and response packets to first authenticate users, and then delivers keys using *Logical Key Hierarchy* (LKH) mechanism. Compared to other pull-based subscription proposals, the proposed solution is able to achieve lesser control overhead, with added security and privacy features.

This chapter is organized as follows; we first highlight the requirements and challenges for ICN publish-subscribe communication. Then, we detail the proposed Group-based Publisher-Subscriber (GbPS) scheme including the design principle, subscription management, and key computation. Finally, we present the implementation and evaluation part.

4.2 Publish-Subscribe Requirements & Challenges

In order to design a Publish-Subscribe (Pub-Sub) communication model for ICN, different subscription mechanisms need to be explored. Varying requirements from these models may require changes to ICN primitives at the network layer. Different users may subscribe for a service offered by a publisher, where the information generated may be continuous (e.g. live video feed), periodic (e.g. weather updates), or when specific events occur (e.g. patient pulse rate change). Hence, we classify the traffic into the following categories based on its behavior.

Single-Request Single-Response: A requester node sends one request packet asking for some content by using its content name. The original content provider or a replica-node

replies and delivers the content to the requester. Any update to the same content can be obtained by sending another request. This model works perfectly when the content required already exists, or can be made available before the interest entry expires in the core network.

Single-Request Multiple-Responses: This model is used when a subscriber sends one request asking for data which may comprise multiple responses spread over time. The number of responses can vary depending on the application service.

- Periodic Delivery: A consumer sends one request packet asking for periodic data identified by name after a specific time interval, e.g., receiving sensor value every 10 minutes for the next 2 hours. The amount of data is limited and bounded by time.
- N Responses: A subscriber node sends one request packet for a specific number of responses, e.g., next n pieces of information generated for a certain topic (next 10 frames of a video). Here, the amount of data is limited but not bounded by time.
- Conditional Delivery: A subscribing user sends a request packet to receive data from the publisher only if certain conditions are met or events triggered, e.g., notify when mentioned in a tweet. The information flow is not bounded in time.

A vast majority of current Internet applications, especially mobile and IoT technologies, use these communication models for subscription services. Moreover, many of these applications also use ICN communication patterns[6]. The requirement to harness the benefits of rapid content delivery of ICN along with pub-sub communication, presents a unique and interesting challenge. NDN and CCN are primarily request-response (pull-based) systems. Although they are well designed and efficient in working, implementing a pub-sub model without modifying any of their primitives is non-trivial.

The limitations of exiting works are not based on their working principle, rather they are more related to security, scalability, and seamless integration into ICN implementations. Here, we dissect both pull-based and push-based mechanisms, to explore the technical requirements for an optimal solution.

Pull-based Model: This model can be used to realize periodic data subscriptions, where subscribers are required to send periodic interest packets asking for the data. The publisher replies to each request individually. The relation between content name and the amount of

content is very important. For example, *local_weather_update* is the content name where the content will change over time. Every time this information is required by a mobile device, a new request has to be sent. This solution fundamentally creates an overhead for publisher, as the requests arrive at disjoint times. Intermediate node interest aggregation also suffers due to the same reason. Content caching by intermediate nodes may provide some efficiency, but the overhead of request packet from every subscribing node still burdens the network. Once the content becomes stale, caches need to be expunged. Furthermore, this model has no authentication mechanism, as for who can access the information. Application layer authentication does not work, as ICN supports in-network caching and name aggregation. Existing pull-based models also lack security mechanisms. If an application layer public-private key mechanism is used, then the publisher has to obtain public key of each individual subscriber, and generate unique replies every time. This again results in caching and name aggregation failure in the core network. If a common key is used for encryption and decryption, then key safekeeping, updating, and distribution become a challenge. Lastly, event-triggered communication model cannot be optimally implemented.

Push-based Model: This model can be implemented in two ways. One method is that the publisher injects data into the network for subscribers, where the list of subscribers is known to the publisher. This method violates the interest-response mechanism. Data packets with no entry in PIT will be dropped in the core network. Even if the core architecture is changed, the publisher will generate as many responses as the subscribers and performs a unicast push for each. This increases the overhead and takes away the advantage of ICN. The second method is to create semi-permanent entries (which are not removed after a single response packet), in core routers using a specialized interest packet. This method is interesting but has the following challenges associated with it.

- Subscribers in the same domain may subscribe for the same topic but use different frequency/rate to receive data. Due to PIT aggregation feature, all subscribers within the same domain will receive data with no regard to subscription frequency rate.
- Keeping PIT compact and fresh requires persistent entries to expire after some time. Once the timer expires, a burst of interest packets will be injected into the network from all downstream subscribers. This burst may force intermediate nodes to overflow.

- No access control mechanism has been proposed for this model. Once the data is injected into the network, anyone can access it. Encryption mechanisms suffer from the same limitations as that of the earlier method.

- To achieve minimal levels of access control, each subscriber has to generate a unique interest packet, achievable only by a unique interest name (otherwise in-network aggregation will combine requests). This can be done by adding a subscriber ID to the content name. Content name standardization itself is an open research challenge. At the same time, such uniqueness takes away the caching and aggregation benefits from ICN.

In light of these challenges, it is better to use a hybrid $(1 : N)$ publisher-subscriber group model for dynamically generated content. It is important to note that, the content already published can be retrieved individually by subscribers using the native NDN interest-data model. Access control can be managed by group operations, and group-based encryption mechanisms can be used for privacy. This makes the overall scheme secure, scalable, and efficient.

4.3 Group-based Publisher-Subscriber scheme

The proposed Group-based Publisher-Subscriber (GbPS) scheme can be divided into two parts. The first part deals with subscription management, and the second handles key computation. It is an inherently secure design, which treats members as a $(1 : N)$ pub-sub group. Without violating ICN's working principles, we introduce new types of interest and data packets, with an additional table at the intermediate nodes. Figure 4.1 depicts the overall communication process and the message structure for each step of communication. Both publisher and subscriber modules are shown with corresponding components, which are explained in the sections below.

4.3.1 Design

The publisher-subscriber model can essentially be viewed as a group management scenario, with specific management operations. In order to keep it simple, we propose the use of

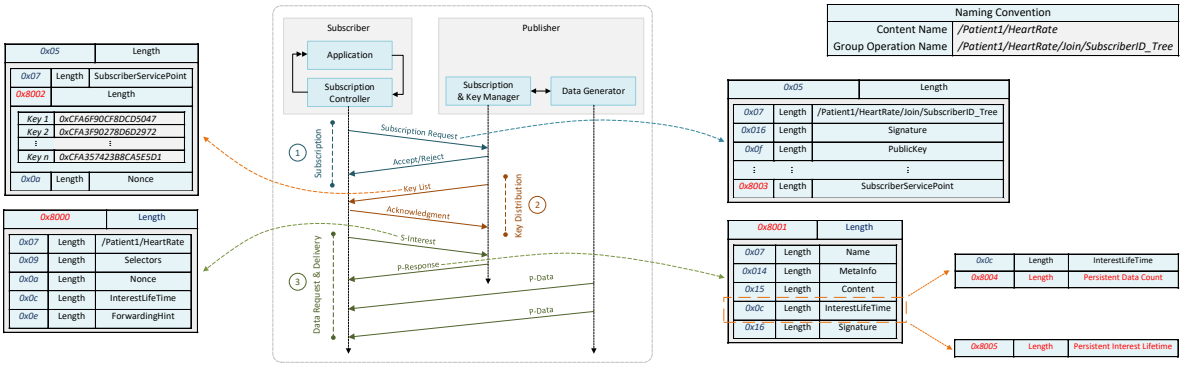


Figure 4.1 GbPS communication.

three operations, i.e. Join/Subscribe, Leave, and Evict. A subscriber sends a subscription request (interest packet) to join a topic (group) offered by a publisher. Similarly, it may request to leave, or the publisher can evict a subscriber for different reasons. The objective here is not to propose a new group management protocol but to add simple yet effective procedures that can enable secure group management communication.

Naming Convention: The complete process of subscription and data delivery is shown in Figure 4.1. The top part shows the naming convention used in this work. The name of subscribe-able content is different from the interest name used for group operations, where later is an extension of former with added semantics. The content/topic/group name is used to represent the actual content, which is used for content retrieval. On the other hand, the operation names contain added name-components to represent two elements: desired operation and requesting subscriber. This two-element name structure is used to enable unicast communication between subscribers and publisher. In its absence, the interest will be aggregated in intermediate network and no unique identification of subscribers will be possible.

Pub-Sub Modules: Subscriber and publisher systems contain specialized modules as shown in Figure 4.1. Subscriber implements a special *Subscription Controller*, which acts as a connecting point for different topics. Once the application requests subscription of a specific topic, the Subscription Controller creates and publishes a globally reachable service point. Signature can be used to verify the authenticity and authorization of a user. We assume that efficient and secure signature verification mechanisms are available which eliminate the risk of forging signatures[111]. This service point is represented by a name which is similar to content name. On the publisher side, there are two sub-modules. a) *Data Generator* module

is responsible for generating & publishing the actual content. b) *Subscription & Key Manager* module is responsible of subscriber authentication, as well as generation, management, and distribution of keys. It also provides the Data Generator with key that is used for encryption.

4.3.2 Subscription & Data Processing

Join Operation: The join operation is three steps process for subscribing to a specific topic, which are: Join request & authentication, Distribution of key set, and Data request & reply. The subscriber initiates a join request using the operation interest name as described earlier. ICN packets are a collection of *Type-Length-Value* (TLV) fields, hence signature, public keys, and subscriber service point information can be easily added to interest packets. The important point to note is that, the communication is unicast between *subscription controller* and *subscription manager*, as shown in Figure 4.1. The response to this request can contain additional TLVs to indicate approve or deny decision. It is assumed that existing mechanisms of signatures, password, biometric data, etc. are sufficient to authenticate a request. In case of an accepted subscription, the Subscription & Key Management module maintains the binding of subscriber to its keys and service point name. Following this, the group keys are updated and re-distributed in the group, which is a publisher initiated unicast process. The interest message for key distribution has piggy-backed data in form of encrypted key list, which can later be used by the subscriber to decrypt subscribed data.

Both of the above-mentioned steps have to be unicast, otherwise, authentication of individual subscribers and key distribution to specific group members is not possible. Additional TLVs do not change ICN request-response principle and has no effect on the intermediate node processing. It is important to note that, if unicast delivery of content is achieved, then the benefit of aggregation and caching is lost. Hence, we use group keys to secure communication, and the keys themselves are generated/changed and distributed within the group.

Data Delivery: Once the new subscriber has received the group keys, it can generate an interest for data (with actual content name). This interest (*S-Interest*) is not different in structure as compared to a generic NDN or CCNx interest packet. However, the Type value used is *0x8000*. This enables the distinction between the two types so that intermediate nodes can take appropriate action. Upon receiving S-Interest, the intermediate node creates an entry

in *Subscription Interest Table* (SIT). Hence, each node in our model has an additional table to keep track of interests related to subscriptions. The table, unlike PIT, retains entries for a longer period of time. Fundamentally, an intermediate node has two possible scenarios.

(a) S-Interest arrives with no related entry in SIT. In this case, an entry is created with interface ID, and the packet is processed using existing forwarding strategy. It is important to note that individual timers are kept for each requesting interface. In case, when there are no entries along the entire path, S-Interest will reach the provider. The provider then, generates a persistent response (P-Response), which does not contain data, but contains specialized TLV as show in Figure 4.1. This TLV is then processed at each return-path node to update the persistent interest lifetime value.

(b) S-Interest arrives at intermediate nodes and an existing entry is present in SIT. The node first forwards the signature of requesting subscriber to the publisher's subscription manager and verifies that the node has been previously authenticated. On positive response, the subject node adds the interface ID to list, sets timer to minimum timer value of other associated interfaces, and generates a persistent response packet.

The persistent response packet (either from publisher or intermediate node) specifies the time/count of responses before the entry is expunged from SIT. As seen from Figure 4.1, we propose the use of a counter in combination with a maximum time limit, where expiration of either will remove the entry. We also propose that a maximum limit must be set by the network for any timer value (10 minutes in this work). Every intermediate node receiving the persistent response, sets the appropriate value for an outgoing interface in SIT. Once it reaches the subscriber, it can record the specified time limit, and must rejoin the group after expiration.

Subscribe-able data is generated by the Data Generation process at publisher. It is published using the generic name structure. Once the publishing node has subscribers in SIT, the data is forwarded onto the interface as long as it has a valid timer. This is the only deviation from pull-based principle, where PIT entry is removed once the data is forwarded back towards consumer. This process ensures that multiple data packets are forwarded before the entry expires, or is renewed by the consumer. Renewal process can be identical to

join operation. The overall benefit of such a mechanism is that, multicast and in-network data replication still holds, while secure subscription management is also available.

Leave Operation: When the application no longer desires subscription, it initiates the leave operation through Subscription Controller. The publisher responds with an acknowledgment to the subscriber. Immediately after the request is received, the publisher removes the subscriber from its list, and updates the group keys. These keys are then unicasted to the remaining subscribers. It is not possible for the publisher to instruct intermediate nodes for SIT entry removal, as there may exist other active subscribers on the same downstream interface. Actually it is the key change that ensures that only active nodes are able to decrypt the information. Moreover, we assume that the trusted subscribers do not collude with non-subscribers by sharing the keys and data streams.

In case where publisher wants to revoke a subscriber access, it sends a unicast message to that subscriber's service point, with piggybacked revocation notice. Publisher immediately changes the keys, and performs redistribution operation.

4.3.3 Key Computation

In order to ensure that only authenticated subscribers can obtain data, it has to be encrypted. The use of the subscriber's public keys for encryption is not a scalable solution since the publisher is required to generate as many data packets for the same content as the number of subscribers and to encrypt them individually. A group key with a secure group key management system is recommended to handle this issue with more scalability. The objective of this work is not to develop a new group key generation solution for ICN. Rather we use Logical Key Hierarchy mechanism in our pub-sub communication scheme which uses a tree structure. LKH allows keying in an efficient and scalable way by minimizing the number of transmissions in re-keying as well as storage requirements. LKH is used to generate common encryption for all allowed subscribers. The key will be updated once a new subscriber joins or leave the group which will allow only authorized subscriber to receive and decrypt the content with less number of encryption/decryption operations and small number of packets. For each join or leave operation, all keys in the path from the subscriber location (leaf) to the root are changed. This change ensures both backward and forward secrecy requirements. The

publisher is responsible to maintain the list of subscribers and implements the centralized key management process. Since LKH is based on a balanced tree to manage group membership, we prefer to use the algorithm described in[112], which efficiently keeps the tree balanced in highly dynamic groups.

Join/Leave Operation: After receiving the join request, the node must be authenticated using the credentials provided in the interest packet. Authenticated subscribers are then provided with the necessary keys using their subscriber service point name. In this work, key changes due to join operations are delivered after a time delay, where the delay is dependent on the number of join requests per unit time. The maximum delay is set to 2 seconds. This time delay reduces the key distribution communication overhead. When a subscriber leaves the group or is forcefully evicted, the keys are changed & distributed immediately without delay.

4.4 Implementation & Evaluation

The performance of GbPS has been evaluated to determine relative control overhead created by group management requests, key exchanges, and entries made in PIT/SIT at intermediate nodes. It is important to note that native ICN design does not support secure group communication. Hence, we use Baseline Pub-Sub (PS) as a pull-based model, where every subscriber sends an interest for each required content packet. In-network name aggregation is also utilized in this mechanism. Moreover, Secure Baseline uses an encrypted version of Baseline PS, which creates unique encryption for each subscriber. In control overhead measurements, we also show unencrypted persistent interest performance, which uses a single interest to obtain multiple data packets.

4.4.1 Control Overhead

In this experiment, we measure the amount of non-data bits generated by subscribers or publishers, against increasing number of subscribers. All subscribers request the same published content. Figure 4.2 shows the overhead from two perspectives. The solid lines represent the overhead performance measured in bits (left vertical scale), and dotted lines represent the number of packets (right vertical scale). With an average path length of 6 hops

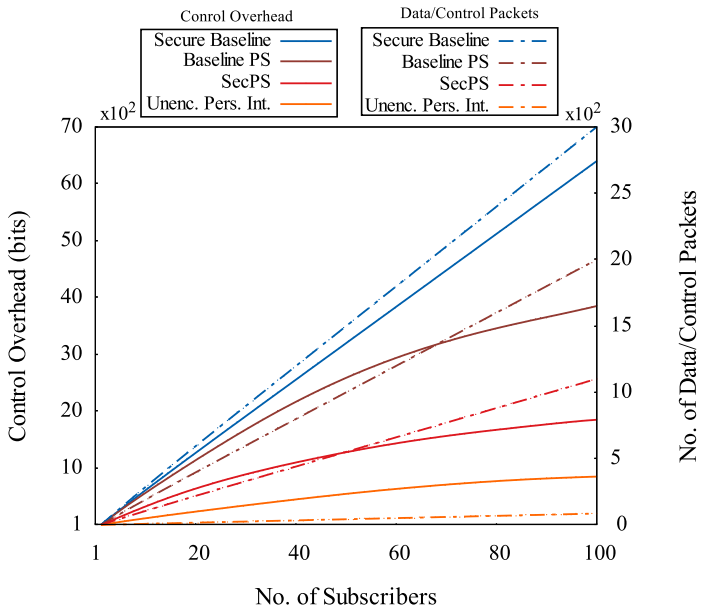


Figure 4.2 Control overhead performance.

and data rate of 5 packets per second, we observe that, as the number of subscribers increases, the chance of finding existing entries in intermediate nodes also increases. Hence all different algorithms see a tapering-off of control information at higher subscriber numbers. Encrypted baseline individually identifies the subscriber, so that the publisher can encrypt the content for it (unicast), which creates a continuous growth in overhead. Removing encryption gives less overhead, but still requires subscribers to generate as many interest packets as data. Unencrypted persistent interest algorithm gives the least overhead as a single interest delivers multiple data packets. The dotted lines show the number of data/control packet in different algorithms. Baseline PS has a 1 to 1 ratio between data packet generated against interest packets. GbPS generates more number of control packets as compared to unencrypted persistent interest algorithm, and hence has less data-control ratio. But at the same time it is able to provide a well defined secure group operation. The benefits achieved are far more in significance than the ratio.

4.4.2 Live Video Stream Analysis

To show the effectiveness and performance of GbPS in real-world scenario, we observed the number of subscribers and data exchanges in NASA Live Channel on YouTube for 10 minutes. The viewers/subscribers join or leave at will in real-time. Each response packet

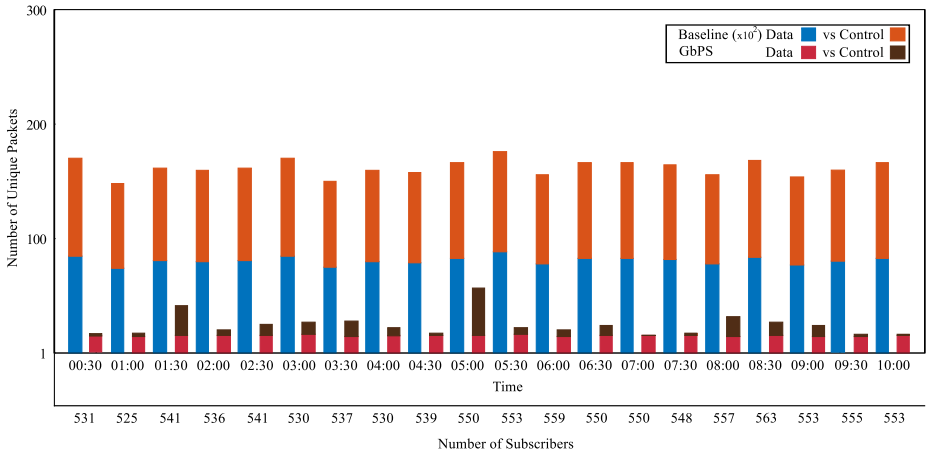


Figure 4.3 Live video stream control and data packets.

contains 60 frames from producer (Youtube stats). Figure 4.3 shows the number of both unique data and control packets exchanged. In case of secure baseline, the publisher needs to perform as many encryption operations as that of subscribers, and send these unique packets to individual subscribers, even though the packets contain the same frames. By using GbPS, only one encryption operation is used, hence the data packets are very less. The spikes at 1:30 and 5:00 are due to changes in subscribers which requires new keys to be distributed.

4.4.3 Memory Requirements

This experiment analyzes the average memory required at each node to store the interest entries. Baseline PS continuously generates interest requests, hence the memory requirement increases as the number of subscribers grow. On the other hand, keeping subscribed interests for longer period of time combined with aggregation has far less memory requirements. Figure 4.4 shows the effect for GbPS only, where we evaluate it in relation to different number of uniquely published content and number of subscribers. Memory requirement increase is impacted more by the number of unique content which can be subscribed, rather than the number of subscribers.

4.4.4 Compatibility with NDN & CCN

The fundamental designs of NDN and CCN are almost identical. Later NDN protocol modifications have added newer features independently from CCNx 1.x. To ensure compati-

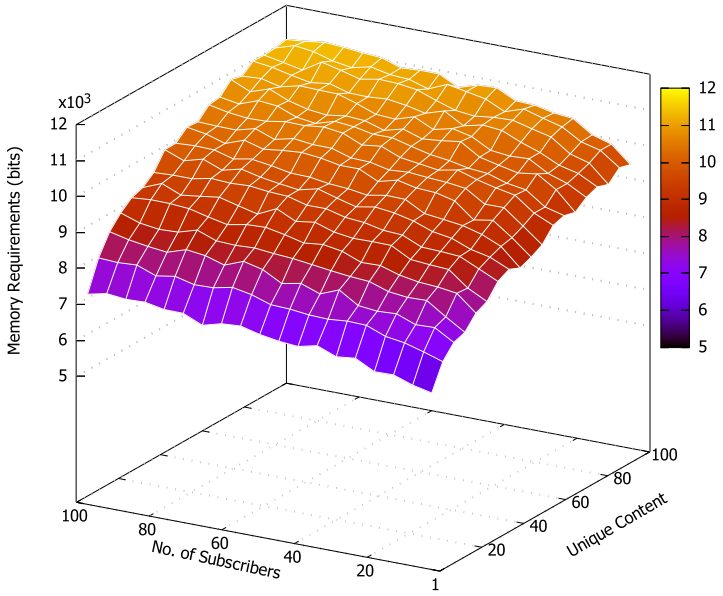


Figure 4.4 Avg. storage req. for GbPS at intermediate nodes.

bility with either of the architectures, we summarize the technical design choices in Table 4.1. GbPS can be implemented in both of them, and some of the design choices actually support the proposal put forward in this article.

4.5 Conclusion

Subscription-based content delivery in ICNs is a challenging issue, as the single-request single-response model creates massive control overhead for periodic, continuous, or event-

Table 4.1 NDN & CCN compatibility with GbPS.

	NDN	CCN	GbPS
Semantics	– Unbounded name components – Partial name matching in hierarchy	– Unbounded name components – Full name matching	– Requires full name matching. – Uses basic naming convention with name components.
Encoding	– TLV format (outer & inner TLVs)	– XML Encoded → TLV format. – Fixed header with TLVs within packets.	– Uses TLVs, and can be stored in either of packet structure.
Naming	– Proposal to have zero or multiple data packets for one Interest – Digest as last part of name	– Free to define name components – Implicit digest used as part of hash restriction for unique name.	– Uses *Subscribe_ID* to enable uniqueness in name when desired. – Digest can be used for same.
Interest Aggregation	– Available	– Available	– Capitalizes on aggregation for encrypted content. – Keeps individual timers for return interfaces.
Opportunistic Caching	– Available	– Available	– Does not require caching due to nature of data.
Forwarding	– Assumes loop freedom – Nonce for detection of duplicates – Hop limits	– Assumes loop freedom – Hop limits	– Additional SIT and semi-persistent Interest. – This does not interfere with the forwarding process.
Data-Centric Security	– TLV for signature	– Supports signature for Interests – CCNx Key Exchange Protocol	– Uses signatures for user authentication – Key Exchange Protocol can be used to encrypt, in addition to other key exchange methods.

triggered communication. Providing access control is an additional challenge. This chapter presented GbPS scheme that enables access control based secure publisher-subscriber communication, without violating ICN's fundamental principles. A combination of semi-persistent subscription interest and specialized authentication interest packets with bi-directional communication, enables authentication, key exchange, and group management operations. In the next chapter, we will address the content caching placement issue in large-scale ICN networks.

Chapter 5 Efficient Content Caching Schemes

5.1 Introduction

ICN decouples the content from its original location and consequently enables in-network caching at the network layer. The merger of ICN with the IoT is promising to overcome various challenges such as the addressing problem, heterogeneity, scalability, and resource constraints issues. In this chapter, we present ICN cache placement strategies for Edge-IoT applications. We design a centralized and a distributed cache placement schemes that aim to place the popular content neat to the consumer, typically at the edge network and keep the less-popular content in the core. Besides, we propose a collaborative mechanism to fetch the content from the near neighbor content-stores as well as a cache replacement policy based on popularity function.

This chapter is organized as follows; we first highlight the requirements and challenges for an efficient cache placement place in ICN. Then, we detail the proposed cache placement schemes. Finally, we present the implementation and evaluation part.

5.2 In-Network Caching Requirements & Challenges

In order to design an efficient cache placement plane in ICN, various issues and challenges need to be addressed. We classify these challenges into two main aspects:

- Cache Placement-related Challenges: ICN provides a transparent in-network cache feature in order to enhance the overall network performance. Hence, the content should be cached at the optimal place to satisfy as much as possible of demands. However, the huge amount of content and limited space of the content-store enforce ICN to select only popular content to be cache near to consumers. However, less-popular content should not be removed immediately from the network. An efficient cache placement scheme should trade-off between content popularity and cache resource optimization in the whole network.

- Cache Replacement-related Challenges: Removing content for the content-store to keep room for newly arrived content may lead to removing the popular content, and hence, all future demands must be satisfied from the original producer. An efficient cache replacement policy must remove only the non-popular content and keep the less-popular content at the core network with less cost.

5.3 Efficient Caching Placement Schemes

In the following, we define the content popularity metric as well as describe the proposed caching schemes.

5.3.1 Content Popularity

Instead of dividing the content into two major classes popular and non-popular content. We believe a content can be less-popular in compared to other popular content but still not non-popular. Thus, we define a weight function $\rho(d)$ that takes two metrics as an input, i.e., content access frequency (f_r) and content freshness (f_s) to decide the content popularity level.

$$\rho(d) = \alpha.f_r + (1-\alpha).f_s. \tag{5.1}$$

f_s is a Boolean variable indicates the content freshness, if the received data is fresh than the copy in the content-store then $f_s = 1$, otherwise $f_s = 0$. f_r indicates how many time the content has been requested from the content-store. We assume the popularity distribution of contents is characterized by the Zipf distribution.

5.3.2 HFFL: Highest-First, Farthest-Later

In the following, we discuss the first proposed cache placement scheme, namely *H*ighest-*F*irst, *F*arthest-*L*ater (HFFL). HFFL algorithm follows the incapacitated k-Media placement model[113], which consists of placing the cached data at medium place in a way the average delivery and caching cost is minimized. The use of a collaborative concept with HFFL will drastically enhance the cache hits and delivery large content from the content-store.

Assuming a large-scale IoT network with different subnetworks, all end-users request

different content. The native behavior of in-network caching consists of placing the data at the edge of the network. However, this leads to redundant data caching in different places and inefficient utilization of memory in different content-stores. The idea behind HFFL is to place the data not at the edge nodes but in the cost-median node (it can be the edge node at level 2 or level 3 in the topology). Although this can increase the delay, the content redundancy will be dramatically decreased and the cache utilization increased.

The HFFL scheme selects the more suitable cache placement nodes that are receiving the highest degree of demands for specific content, with the constraint that two content-stores for the same content must not being neighbors either close of each other. This constraint helps to optimize the cache resources utilization as well as decrease the cache redundancy. HFFL can be applied per content name or traffic class. Hence, the placement of content may not by the same, but still an optimal place. A pseudo-code of HFFL is presented in Algorithm 1.

Phase 1 - Highest-First: The first phase consists of sorting the list of all candidate intermediate nodes based on the received demands (degree) per content name or traffic class, the cost of data delivery $\mathcal{C}$, and the cost of data caching $\overline{\mathcal{C}}$.

Algorithm 1: HFFL algorithm.

Input: G: Graph, Degree, $\mathcal{C}$, $\overline{\mathcal{C}}$, $\mathcal{X}$
Output: SelectedCS: List of selected cache nodes.

```
1  IntmdNode = list();          ▷ List of intermediate nodes.
2  SortIntmdNode = list();      ▷ List of sorted intermediate nodes.
3  SelectedCS  = list();        ▷ List of selected cache nodes.
4  NotSelected = list();        ▷ List of not selected cache nodes.
   Phase 1: Highest cache nodes
5  SortIntmdNode = sorted(IntmdNode, key = lambda node: (node[Degree, C, C̄]));
     SelectedCS.append(SortIntmdNode[0]);
   Phase 2: Farthest cache nodes
6  for (node in SortIntmdNode) do
7      if (len(SelectedCS) == X) then break;
8      if (not isAdjacent(node, SelectedCS)) then
9          SelectedCS.append(node);
10     else
11         NotSelected.append(node);
12     end
13 end
14 return SelectedCS;
```

Phase 2 - Farthest-Later: The second step consists of selecting the optimal intermediate nodes to cache the content. Thus, it selects only the highest candidate nodes (Phase 1), with the constraint that two selected nodes must be as far as possible.

Based on the input parameter $\mathcal{X}$, we can decide the number of selected content-stores:

- if $\mathcal{X}$ equals 1: only the intermediate node that received the highest number of demands will be selected as content-store.
- if $\mathcal{X} > 1$: $\mathcal{X}$ intermediate nodes will be selected as potential content-stores. They are the nodes that received the highest demands and are farthest away from each other.

Despite the benefits of HFFL, it runs in a centralized mode. A node must have a vision of the network demands (i.e., suitable for IoT applications running on top of Software-Defined Networking concept[114]). Hence, we propose in the following a distributed caching placement scheme.

5.3.3 PDPU: Push Down Popular, Push Up Less Popular

The second cache placement scheme, namely *P*ush *D*own Popular, *P*ush *U*p Less-Popular (PDPU), aims to select the optimal and mostly near cache placement node to all requesters in a fully distributed manner. Toward this goal, we define two main caching operations:

1. *Push-Down (PD)*: whenever the cached content popularity reaches a predefined threshold based on $\rho(d)$, the requested content is push down only one-hop from the same interface receiving the interest packet, and removed from the current content-store. Only the direct intermediate neighbor node stores the said content. By pushing down the content, we end up by placing the popular (frequently asked and fresh) content closer to requesters.
2. *Push-Up (PU)*: taking the cache memory size into consideration, and due to the fact that popular content is traversed down the network because of the PD operations. The unpopular content must be removed from the content-store, while the less-popular can be cached in the core network. To ensure a balance between the frequently-used and less-used content, we propose that the content is pushed up one-hop before evicting it from the current content-store only if it has a worthy popularity value ($\rho(d) \geq \theta_m$).

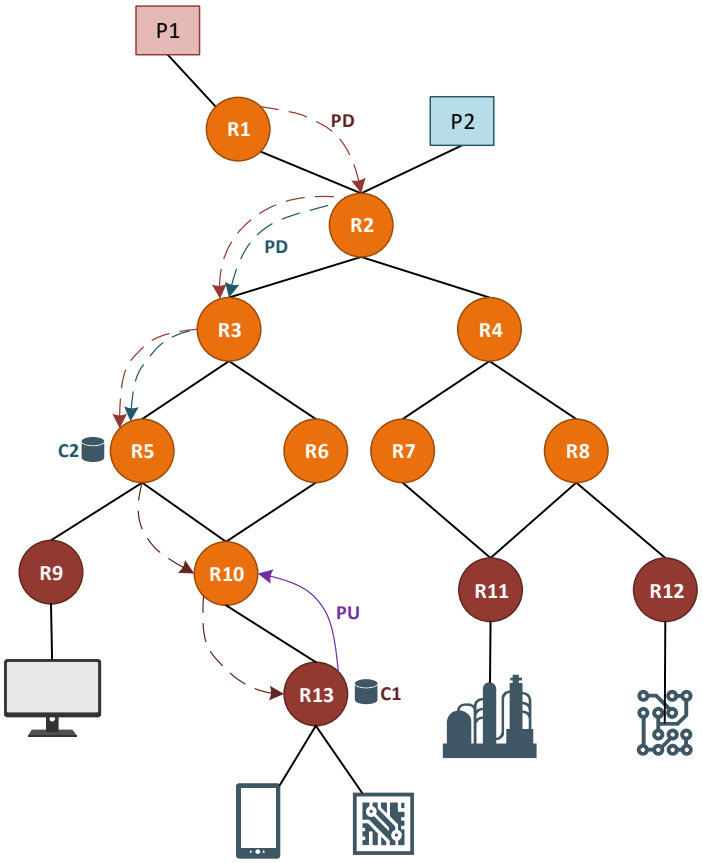


Figure 5.1 PDPU cache flow example.

The reason for Push-Down operation is to move the most popular content, in a fully distributed manner, closer to consumers (serve a maximum number of users in a very short time). While the main reason for Push-Up operation is to distribute the less-popular content in the core network more efficiency rather than remove them.

Figure 5.1 illustrates an example of using both PD and PU operations. In a trivial way, the content will be pushed down whenever a new cache hit occurs. However, this may affect the network overhead especially in scenarios where we have a large amount of popular content. To overcome this issue, we suggest content will be pushed down only if its popularity $\rho(d)$ reaches a predefined threshold (θ_c), until reaching an optimal place position that may serve all requesters. In case of a node has multiple interfaces and receives demands from all of them, the node pushes down the content from the interface that receives more demands (e.g. demands $((\theta_p)) > 80\%$). Otherwise, the content is cached at the said content to serve all interfaces.

Assuming an example (c.g. Figure 5.1) where nodes connected to $R13$ request the pop-

ular content $C1$, the content is kept pushed down till reaching $R13$ which is the optimal and near content-store (from $R13$'s nodes perspective). However, in case of a content requested from different sub-/networks, e.g., both consumers in $R9$ and $R13$ equally request the content $C2$. Pushing down $C2$ to both $R10$ and $R9$ is not feasible and may increase the duplicated cached content. Even thought, pushing it down to $R9$, demands from $R13$ need to be forwarded one again to the original producer which eliminated the effect of in-network cache and reduce the novelty of PDPU algorithm. Hence, PDPU selects $R5$ as the optimal place that can satisfy all demands from both branches.

It is important to highlight that when content is evicted and PU operation is triggered, the operation is executed repeatedly until reaching the producer. Then, the content will be removed from the content-store. Another important point that must be highlighted here is the threshold values are customized by the network administrator and can be changed from a router to another based on the network traffic and content-store capacities. The router keeps monitoring all received demands periodically in order to calculate the threshold values.

5.3.4 Collaborative ICN One-Hop Cache Notification

Knowing that the cached data at a local node cannot be advertised via the routing protocol. We design a new mechanism, namely Collaborative One-Hop Cache Notification, to maximize the benefits of in-network caching, and deliver the content from the near content-store instead to forwarding the request upstream. This notification message is based on the NNCP protocol[22]. The new message is namely *Cache Notification*, in which the message packet carries only the name prefix of the local cached data as well as their Time-to-Live values. The notification message is sent out to all neighbor nodes. A neighbor node, after receiving the notification message, adds the received content names into its FIB tables with the received interface name and the Time-to-Live value.

In PDPU, the content caching decision can be applied per traffic class. Thus, the content is cached in the optimal place from the overall network perspective. The collaboration of neighbor content-stores via the notification mechanism helps to build a partial view of different near cached content. Hence, it contributes to the efficiency of the used cache strategy by increasing the probability to serve content from near content-stores even if they are not in

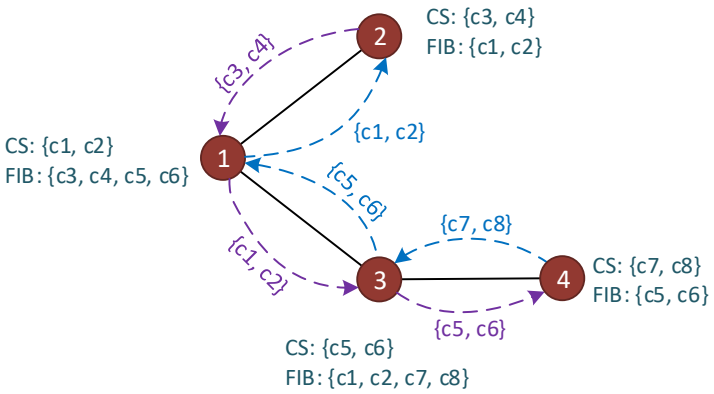


Figure 5.2 Cache notification example.

the communication path.

Figure 5.2 illustrates an example of using the *Cache Notification* mechanism. In the simple mode, whenever new data is added to the content-store, a notification message is set out. However, this mode will generate a storm of the packet may produce a huge overhead. Thus, we propose to use the content popularity function ($\rho(d)$) to advertise only the popular content in a periodic manner. Hence, we reduce the number of packets and consequently the network overhead. As shown in the figure, node N1 creates new notification message to inform both neighbors (N2 and N3) about contents (c1 and c2). Similarly, N2 replies with the content (c3 and c4), and N3 with (c5 and c6). We can also notice that N1 cannot advertise (c5 and c6) to N2 because they do not reside in its local CS. In addition, the Time-to-Live value is also sent with name prefix to remove the entry from the FIB tables and keep them fresh and clean.

5.4 Implementation & Evaluation

We have implemented the proposed caching schemes on top of ndnSIM[43]. The objective is to evaluate the efficiency and scalability of the schemes presented. The metrics which we have evaluated are: the network delay, the hop reduction ratio, the cache hit ratio, and the overall cache utilization. Moreover, we have compared the proposed schemes against the performance of other existing strategies including LCE, LCD, EC, and CC.

Network Delay: We denote the average network delay by the average time duration to satisfy all demands issued by all requesters.

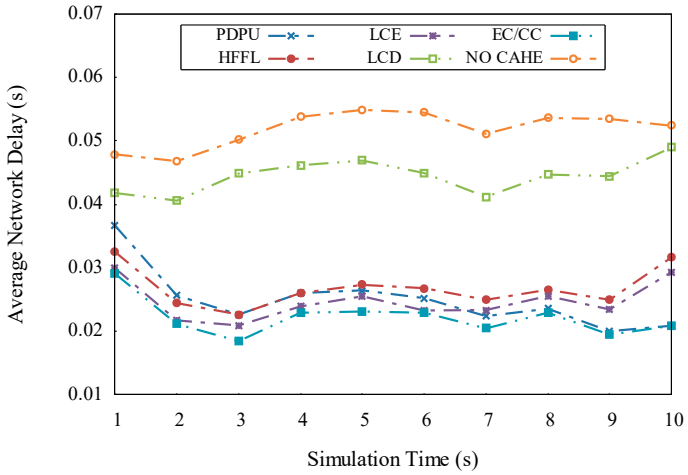


Figure 5.3 Average network delay evaluation.

In Figure 5.3, we observe that NO CACHE has the largest delay, because all demands must be satisfied by the original producer. While LCD is just below NO CACHE due to the fact that LCD selects one-hop after producer as a content-store. EC/CC has the smallest network delay as the content is cached near to consumers (only one hop). While the network delay in LCE varies, sometimes equals to EC/CC and sometimes larger a little bit. This can be justified as all nodes cache the content, including the edge node of the network, and due to changes in the content-stores, demands may be satisfied far away from the edge. HFFL scheme produces an average delay, close to LCE. This delay may satisfy users' quality of service and reduce the diversity of cached content. Finally, PDPU strategy starts with a delay near to LCD because the content is cached and pushed down hop-by-hop. However, we can notice that the delay is getting smaller and closer (sometimes similar) to EC/CC. This convergence is due to the fact that the popular content is kept pushed down to the optimal place.

Average Hop Reduction Ratio: The average hop reduction ratio represents the number of hops that should be traversed to fetch the data from the content-store(s) than the original content producer. The ratio is calculated as the average of all requesters over all demands.

Figure 5.4 represents the average hop reduction ratio results. From the figure, we can see that LCD has the smallest ratio as the content is placed one-hop far from the original producer. However, EC/CC has a better ratio due to the fact that the edge node(s) (one-hop from the consumer) is selected. In a similar manner, the ratio of LCE is close to EC/CC due to the

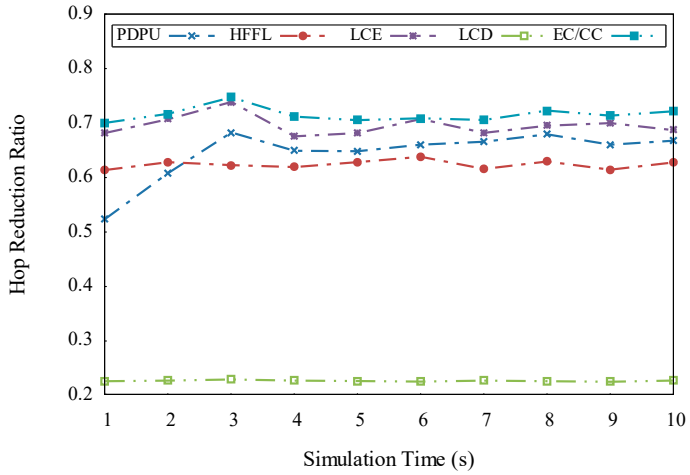


Figure 5.4 Hop reduction ratio evaluation.

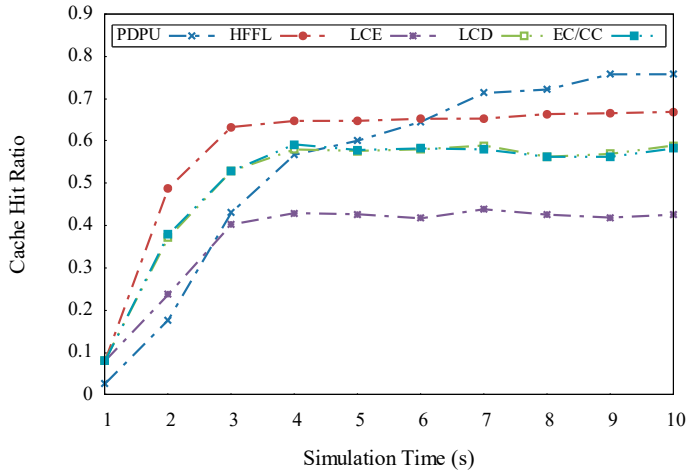


Figure 5.5 Cache hit ratio evaluation.

fact that the edge node is also selected as a content-store. However, we can notice that LCE may not overlap with EC/CC as the content is always evicted and demands may be satisfied by other content-stores. HFFL selects nodes that are farthest from each other. Hence, hop reduction is not as good as other strategies. Finally, PDPU starts by an average ratio as the content is pushed down based on its popularity in a hop-by-hop fashion. However, it reaches a good ratio compared to HFFL because the content is placed in an optimal place that may satisfy all demands from different sub-/networks.

Average Cache Hit Ratio: The cache hit ratio is calculated as the total number of cache hits on the network over the sum of cache hits and cache misses. In the simulation, we calculated the average cache hit ratio for all nodes over all demands.

In Figure 5.5, we show the cache hit ratio performance. Although LCE caches each traversed content, it has the lowest cache hit ratio compared to other strategies. This can be argued due to the non-cooperative nature of LCE that leads to cache any content regardless of its popularity. However, EC/CC and LCD involve only one intermediate content-store in the communication path. Thus, we notice that their ratio is overlapping. On the other side, HFFL selects the highest and farthest content-stores in a centralized manner. That is, its cache hit ratio is higher among all strategies. Finally, PDPU starts with a lower ratio and increases by time. This increase is due to the push-down operation. After pushing down the popular content, PDPU produces the highest ratio by serving the popular content from the optimal content-store(s).

Average Cache Utilization: The cache utilization denotes the number of cached data in the whole network for a demand. In the simulation, we calculate the average cache utilization for all requested content.

Figure 5.6 shows the average cache utilization in terms of simulation time and the number of unique names in the content-stores. We can notice that LCE has the larger cache utilization because all nodes are caching the same content (duplicated copies) which means the diversity of cached data is very small. However, in LCD and EC/CC only one node cache the content, hence the cache utilization is small even if the number of unique content increases. On the other side, HFFL has a larger cache utilization compared to LCD as it involves a

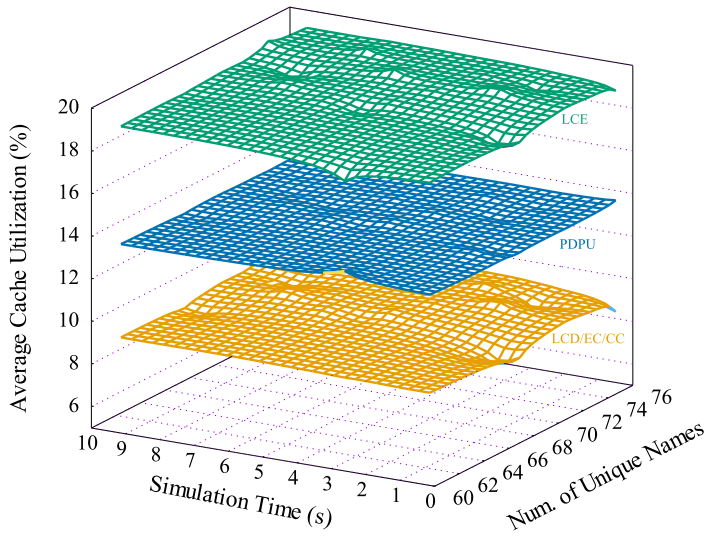


Figure 5.6 Average cache utilization evaluation.

small set of intermediate nodes (more than on content-store). While PDPU only one intermediate node in each push-down operation. Hence, it utilizes more cache memory compared to HFFL. Another way to improve the performance of in-network content caching is by merging both centralized and distributed schemes. For example, the distributed scheme can be used within a domain network (e.g., local IoT network), while the centralized scheme can be used between multiple networks.

5.5 Conclusion

The in-network caching is one of the fundamental ICN features. It aims to dissociate the content from its original provider, provide multiple copies of the content in the network, reduce the overhead at the provider side, avoid a single point of failure, improve content retrieval, and reduce network delay and latency. Although the integration of ICN on top of IoT promises to enhance IoT applications and data dissemination, the cache placement and replacement strategies remain open challenges. In this chapter, we proposed a hybrid cache placement schemes. Indeed, we designed a hybrid cache model with a centralized and distributed scheme. The proposed schemes aim at selecting the optimal placement to cache the content by pushing down the popular content toward the edge network and keep less-popular at the core network. In the next chapter, we will design a control protocol for named data networks.

Chapter 6 Named Networking Control Protocol

6.1 Introduction

The current NDN design and implementation only detail interest and data packets which ensures ubiquitous data dissemination. Currently, there is no support of control messages similar to Internet control messaging protocol of IP networks. The use of content names instead of host addresses, combined with interest-data exchange model, and interest aggregations makes the design of such a protocol a challenging task. In this chapter, we present a control protocol for named data networks, that can relay different network error, information, notification, and service messages.

This chapter is organized as follows; we first highlight the requirements and challenges to design a control protocol for named data networks. Then, we detail the designed protocol along with packet format, processing rules, and messages. Finally, we present the implementation and evaluation part.

6.2 Named Control Protocol: Requirements & Challenges

Most of these works highlight the applicability of NDN to IoT, but are limited to basic communication model. Hence, the need to properly define a control protocol is left out. Below, we list the fundamental challenges, which are present in designing such a protocol.

- Request-Response Model: From NDN communication perspective, data transfer is *always solicited*, where a consumer initiates communication using an Interest packet that carries the name of the requested content, using a receiver-driven hop-by-hop forwarding mechanism. The data packet follows the reverse path from any node which has the data. Moreover, without PIT entries, the packet flow is not possible. Hence, packet exchange outside of interest-data model is not defined. Unlike IP networks, where IP can identify a node and messages can be encapsulated in an IP-packet that can be sent to a layer in stack, NDN does not have any node identification.

- Interest Aggregation: NDN natively supports multi-path Interest forwarding towards multiple destinations using a *stateful forwarding plane*. Each NDN node is considered as a forwarder node. When receiving multiple Interest requesting the same content, Interest Aggregation is performed at the network level, and only one Interest is forwarded upstream. This improves network performance and reduces the overhead. A copy of the requested data may be forwarded downstream (multicast), taking benefit of the information recorded by PIT during Interest forwarding. However, the producer or intermediate node has no information as to individual requester. Hence, a traditional error message cannot be sent to a specific node at all.

- Layered Process: In IP-based network each node is identifiable and every protocol is running as a process, and encapsulation can be used to communicate with a single protocol on a specific node. In contrast, NDN system does not conform to this behavior. The only identity is the content name. Hence, it becomes a challenge to implement the same IP concept in NDN framework. The content requesting and data delivery in NDN is done using content name, without any explicit node addresses. Content consumers, producers, and intermediate nodes have no information about who requests the content, who served it (either original producer or content-store), or who forwarded the packet.

In essence, traditional Internet control message protocol cannot be directly used in NDN, unless there is major modification to the forwarding plane. It is important to note, that the performance of ICN has been demonstrated to be better than of IP networks, due its design. Hence, there is a need to capitalize on this improvement and have a control protocol specially designed for NDN. This forms the prime motivation behind this work. We present a control protocol, aiming to notify concerned nodes/service points with network information and errors, enabling an efficient state transfer between different network entities, without introducing new routing logic or violating NDN communication principles.

6.3 Named Network Control Protocol

The main objective of a robust network control protocol is to help realizing a reliable and efficient network. Thus, Named Network Control Protocol (NNCP) inherently uses the NDN

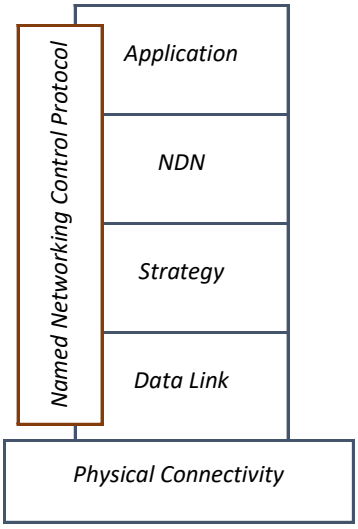


Figure 6.1 Vertical protocol implementation in NDN stack.

forwarding strategy to ensure an efficient control message dissemination between nodes. The protocol uses vertical implementation, as shown in Figure 6.1, and is able to deliver the message to appropriate layer of the NDN stack. Although there are fundamental differences between NNCP and *Internet Control Message Protocol* (ICMP), but some of the workings and messages are adopted from it. ICMP messages over the years have proven their need and benefits, which cannot be disregarded altogether. Below, we describe NNCP protocol general packet format and the processing rules.

6.3.1 General Packet Format

In addition to the Interest/Data packets of NDN, we define a new packet type named *Control* packet, depicted in Figure 6.2, which is based on TLV[115]. Each NNCP message is defined with *ControlClass* and *ControlType*. The *ControlType* is unique and defines the type of message along with the control message itself, while *ControlClass* identifies the class that the control message belongs to. Here, we define three main classes: Standard Error, Notification, and Service messages, with each class having a collection of control type messages. The corresponding numeric values are defined according to NDN project TLV-TYPE number assignment[116]. Under any Control Class, multiple control type TLV messages can be nested inside the packet. On the contrary, a control packet may only belong to a single class.

In most cases, the content of value field in Control-Type TLV is the name of content,

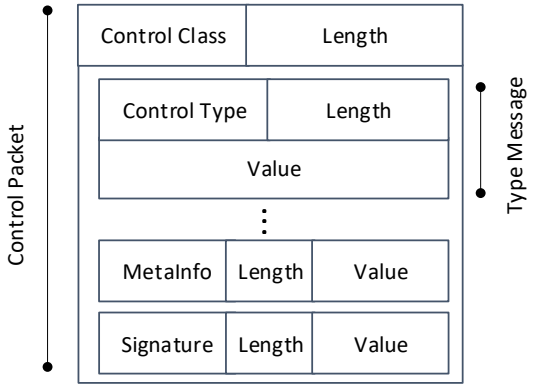


Figure 6.2 General control packet format.

which triggered the creation of this message. Additional information can be added in future works.

6.3.2 Packet Processing Rules

Fundamentally, an error message should not be generated in reply to another NNCP packet (unless it is a query, which has to be responded). Upon receiving any NNCP packet, the node extracts the *Control Class* of message and then the *Control Type* TLV messages. Each message is processed according to the value it contains by the appropriate layer of NDN stack. The forwarding actions are summarized in Table 6.1.

6.3.3 Control Messages

Based on the network usage behavior, we classify NNCP messages mainly into three classes, as listed in Table 6.1. It is important to note that this is not an exhaustive list of messages. Most of these have been adopted from ICMP, and discussed in the light of NDN. More messages can be easily added to the protocol by defining the appropriate Type value and processing rules.

Standard Error Messages: This class is used to represent standard NDN errors. They can be represented as shown in Figure 6.3, where `MetaInfo` contains information regarding freshness of this packet, and the `signature` is that of the creator of packet.

• *Content Unreachable:* This message is generated by content provider or a replica node, in response to the received Interest request for content that can not be reached. Either

!tt]

Table 6.1 List of NNCP messages.

Control Class	*Class Value*	*Control Type*	*Control Value*	*Processing Rules*				*Responding Layer/Module*
				Generated By	*Layer*	*Towards*	*Forward*	
Standard Error	*128*	Content Unreachable	*131*	Provider Replica	NDN	Consumer	Yes	Application: Depends on the application design.
		Prohibited Access	*132*	Provider Replica	Strategy	Consumer	Yes	
		Invalid Path	*133*	Intermediate	NDN	Received Face	No	NDN: Depends on the routing module
		Packet Too Big	*134*	Intermediate	Strategy	Received Face	No	Strategy: Fragment any future Data
		Unsolicited Data	*135*	Intermediate	NDN	Received Face	No	NDN: Expunge Persistent PIT entry
		Parameter Problem	*136*	Intermediate	NDN	Received Face	No	Strategy: Retransmit if possible
		Fragmentation Error	*137*	Intermediate	Link Layer	Received Face	No	Link Layer: Retransmit of fragments
Notification Message	*129*	Explicit Congestion Notification	*138*	Intermediate	Strategy	Neighbors	No	Application: Reduce Interest Sending Rate
		Explicit PIT Entry Remover	*139*	Intermediate	NDN	Upstream nodes	Depends	NDN: Update PIT table
		Undiscovered Content	*140*	CDS Node	Application	Consumer	No	Application: Re-query using different keywords
Service Message	*130*	Content Discovery & Meta Extraction	*141*	Consumer	Application	Provider CDS	Yes	Application: Send content name, meta file
		Link & Network Health	*142*	Any node	NDN	Any Node	Yes	NDN: Depends on the service point.
		Route Reservation	*143*	Consumer	NDN	Provider Intermediate	Yes	NDN: Reserve route/path.
		Mobility Handover	*144*	Intermediate	NDN	Intermediate	No	Strategy: PIT updating routine.
		Device Grouping	*145*	Provider	NDN	Consumers	No	Application: Group name assignement.

the content has been removed by the provider or has been moved to another place (due to mobility). The error message will be forwarded towards the consumer(s) using the reverse path created by the received Interest packet. The intermediate nodes remove PIT entries on forwarding this message in reverse direction.

• *Prohibited Access:* This message is generated by content provider or replica node, in response to Interest request for content that has been administratively prohibited using any access-control filter or the node has been configured to reject all the traffic for a specific prefix. The error message will be forwarded toward the requesters using the reverse path created by the received Interest packet. The intermediate nodes remove PIT entries on forwarding it. This error is specific to network level where all interests are treated similarly. For selective

```
Error ::= CLASS TLV-LENGTH
          ERROR-TYPE TLV List
          MetaInfo
          Signature
```

Figure 6.3 Error message TLV format.

interest filtering application level exchange should be used.

• *Invalid Path:* This message is generated by an intermediate node when processing an Interest packet that can not be forwarded as there is no matching entry toward the content name in FIB table in routing/forwarding strategy or link failure. The error message is sent only one hop toward the interface where the Interest packet has been received. The error message will be forwarded to the routing module, where a decision of further forwarding is done based on routing strategy.

• *Packet Too Big:* This message has been inherited from ICMP, as the MAC layer for pure NDN has not been defined or standardized yet. In a traditional MAC, size of frame is limited, depending on physical technology. Hence, we use the same mechanisms of ICMP to inform the next-hop only, that the data packet desired is too big to be forwarded on this link. The forwarding layer may find another path.

• *Unsolicited Data:* This message is generated by an intermediate node in response to received unsolicited Data packet (no PIT entry matches the name on that packet). The error should be sent back using the same interface from which the unsolicited packet has been received, and for only one hop. Technically, an NDN node should not receive an unsolicited Data packet under normal circumstances. However, this message can be used to support Persistent Interest or Long Live Interest for pull-based communication.

• *Parameter Problem:* The Parameter Problem Error Message could be generated by any NDN node, in response to finding a problem in packet parameters. The error message is sent only one hop to the neighbor, from which the erroneous packet is received.

• *Fragmentation Error:* Mapping MAC address to NDN faces is not well investigated in the literature, and all packets are broadcasted in the network and treated based on content name. The Fragmentation Error Message is associated with NDNLP protocol[117] that functions between NDN layer and physical/link layer[118], and will be generated if the reassembly timer exceeds. In such case we provide a mechanism to recover the missing fragment rather than dropping the message entirely.

Notification Messages: The Notification Messages are generated automatically by any NDN node based on the behavior of the networking process. The TLV format is shown in Figure 6.4. The generated notification will reach only its neighbors and can not be further

```
Notification ::= CLASS TLV-LENGTH
        NOTIFICATION-TYPE TLV List
        MetaInfo
        Signature
```

Figure 6.4 Notification message TLV format.

routed in the network. We define the associated packet to this class as follows:

• *Bottleneck Notification:* The notification message is generated by an intermediate node when a forward link utilization reaches some configurable threshold, and becomes a bottleneck for communication. Information is disseminated only to those neighbors which are using that link. Nodes that receive the notification can appropriately take counter measures, in collaboration with forwarding strategy.

• *Explicit PIT Entry Remover:* An Explicit PIT Entry Remover Notification Message is generated by an intermediate node where a PIT timeout entry has expired. The notification will be sent only one hop toward upstream nodes listed on the PIT face's entry. When a node receives this notification, it will check the PIT to find a match, if there is only one face associated to this entry then NNCP will remove the entry and forward the notification to the list of faces on the entry, otherwise (and because of the interest aggregation), it will remove only the face from the entry and discard the notification packet.

Service Messages: The service messages are similar to request/reply packets, and initialized by any NDN node toward another node in order to fetch information, apply policy, or actions. Figure 6.5 shown the TLV format. Packets under this class are structured as follows:

• *Content Discovery & Meta Extraction:* A consumer can send a Content Discovery

```
QueryMsg ::= CLASS TLV-LENGTH
        QUERY TLV
        Nonce
        Signature

RespMsg  ::= CLASS TLV-LENGTH
        RESPONSE TLV
        MetaInfo
        Signature
```

Figure 6.5 Service message TLV format.

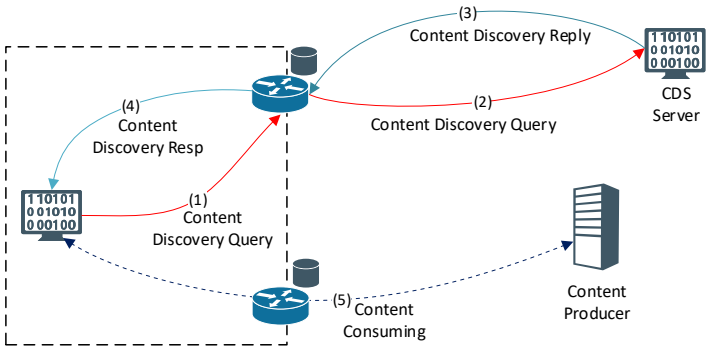


Figure 6.6 Content discovery & meta extraction scenario.

& Meta Extraction Service Message asking to discover the existence of content based on keywords. In case of existence, the consumer will receive meta-data file for the requested content. The meta file is a text file with information related to content, fragmentation, number of chunks, etc. As shown in Figure 6.6, a consumer sends a *Content Discovery & Meta Extraction* toward *Content Discovery Service* (CDS), a DNS-like service[119]. Based on this information, content can be better extracted from the producers. Similar to any content, the meta file can be cached by intermediate nodes, and served for future requests to reduce load on CDS.

• *Link & Network Health:* This message is used to query information about network health. We assume that each node has a service point which has a name and can respond accordingly. For example, it can be used to query information about an NDN component status (PIT, CS or FIB) using the name */NDN/Control/Query/NodeName/PIT* and as a response to this query, the status of the PIT table can be returned.

• *Route Reservation:* This message is used to reserve a route for a publish/subscribe communication system. A node before sending an Interest for periodic data delivery, sends Route Reservation Message. Each node in the path toward the content checks the FIB table to forward the Interest and records the requested name in the PIT. At the same time, it tags this entry with a lifetime (number of data packets or a time limit for communication). The publisher sends Data packet correspondingly, and forwarded based on the reserved path recorded by NNCP.

• *Mobility Handover:* The native NDN design has no explicit handover support. After a node moves and joins a new network, it resends an Interest packets for pending Interest on

its PIT. This control message can be used to indicate the node movement, and redirect any pending data. Also, it can be used to register the new node for pending data that may arrive after its mobility (subscription packets). We leave this control as future use in MAC layer mobility.

• *Device Grouping:* In various management cases, network administrators need to group different devices for application and configuration needs. This service message allows nodes and services to join a group under the same name. A unique group-name is assigned to a collection of services/devices. All generated information will be forwarded to this set of nodes similar to a multicast system.

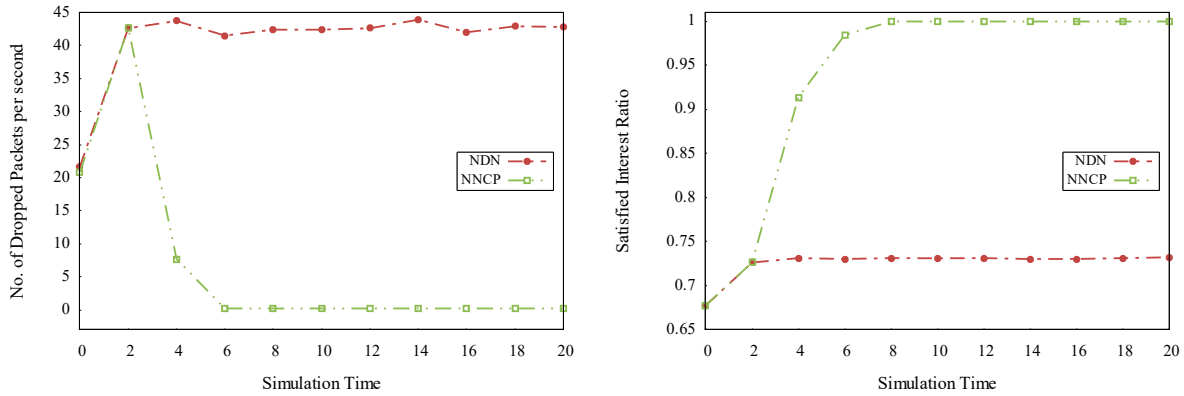
6.4 Implementation & Evaluation

The main objectives of NNCP is to enhance the network functionalities. In interest of page limit, we carry experiments only on congestion control to evaluate its performance and efficiency. Furthermore, we discuss the applicability of NNCP on top of other ICN implementations and co-existence with IP-based networks.

6.4.1 Implementation & Evaluation

We have implemented the proposed protocol on top of ndnSIM[43]. In order to validate the efficiency of NNCP, we used *Bottleneck Notification* message in order to avoid congestion. When link utilization reaches some threshold value, NNCP sends a notification messages to all consumers on the connected link asking them to reduce their *Interest Sending Rate* for an amount of time (t). Using this mechanism, we maximize the link bandwidth, ensure efficient data transmission, and minimize the packet drop rate. In the simulation, we used a bottleneck topology characterized with links capacity of 10Mbps, and 1Mbps for the bottleneck link. The network has four consumers, each consumer sends 100 interest/sec.

The result of drop packets on bottleneck link is shown in Figure 6.7a, and Figure 6.7b shows the satisfied interest ratio. As can be seen, that due to the use of this notification message, the number of drop packet is minimized and becomes approximately null, as compared to normal NDN behavior, where large number of packets are dropped. Similarly, the use of NNCP to avoid congestion maximizes interest satisfaction, by ensuring that maximum



(a) Dropped packets. (b) Satisfied interests ratio.

Figure 6.7 Bottleneck link simulation results.

requests are satisfied by producer.

6.4.2 Discussion on ICN Implementations

NDN deployment (experimental or real-world) has been done in two different ways. 1) Above L3: where NDN runs on top of IP, and 2) Pure NDN: where IP is replaced by NDN layer. In the first case, the performance of NDN is affected by the performance of underlaying network. However, this is a fast way to deploy NDN. The second type is underlay, which means that NDN replaces IP layer totally, and runs directly on top of Ethernet. In this case, NDN fully exploits the underlaying network and achieves better performance than overlay method. However, it is quite difficult to change the current stack on all the current routers and node.

In a hybrid environment, NNCP can easily be translated to ICMP, as some of the messages are intended for similar purposes. The actual mapping requires a translation mechanism that is left as future work of this paper.

Moreover, NNCP can also be easily extended to run on CCN architecture, with minor modifications due to the fact that both NDN & CCN have very similar design principles. Recent Cisco Hybrid ICN (hICN)[120] maps ICN names to IPv6 addresses. NNCP control messages can be easily extended to run with hICN, which means the translation from ICN-based networks to IP networks can also be done using mapping mechanics.

6.5 Conclusion

In this chapter, we have presented NNCP, a control protocol for named data networks. NNCP is designed to control the network behavior by disseminating information of different errors and network changes. Moreover, many services are offered by NNCP such as content discovery and meta file extraction that can be used to get a global view about the requested content. Also, network health monitoring and reservation of route can be done using NNCP. All these features combined with the adaptive forwarding of NDN can produce a reliable network and efficient data delivery.

Conclusions and Perspectives

The current host-centric model faces various challenges, such as scalability, mobility, addressing, security, etc., primarily due to the sharp increase in the number of devices and applications on the Internet. The other factor that has triggered the re-thinking of the host-centric model is the nature of applications and the demand for content by the users. The content-centric paradigm gives a fresh perspective to communication, by eliminating the need to find a specific destination machine in the network. IoT is a major driving force behind content-based applications, with billions of devices and large scale data generation.

Summary of Our Contributions

In this thesis, we presented and discussed the use of ICN as a communication enabler for IoT. We gave an overview of the IoT technology and the ICN paradigm and discussed the applicability of ICN in IoT from different perspectives. Subsequently, we presented state-of-the-art solutions in the area.

In the first contribution, we focused on content naming, we proposed a unified hybrid ICN naming scheme for IoT applications. The scheme integrates different mechanisms such as variable-length encoding, Name-to-Code concept, and in-network function to provide built-in scalability, efficient routing, and security features. Results proved that out scheme reduces the memory consumption, lookup time, and routing and forwarding overhead.

Our second contribution addressed the publish-subscribe communication model. We proposed a group-based subscription architecture to enable seamless publish-subscribe communication, access control, and group management features. Findings showed that our proposal reduces the memory requirements with added security and privacy features.

In the third contribution, we designed large-scale content caching schemes. We proposed a centralized and distributed schemes that considered IoT. The obtained results showed the proposed schemes out-performance existing solutions in terms of network delay, hop reduction, and cache utilization.

Our fourth contribution designed and implemented a named control protocol. The de-

signed protocol relies on different network error, information, and notification messages aiming at improving network performance.

Perspectives & Future Directions

Despite the existing ICN-based IoT solutions proposed in the literature and our proposed solutions in this work, there are still improvements in the area of IoT-based ICN applications and challenges to address. In the following, we list both short and long-term perspectives that can give potential enhancements to our study and future research directions in ICN-based IoT networks.

The short-term improvements of our work are related to the evaluation and deployment in real-world scenarios. Deployment in testbed running native NDN protocol is extremely required to evaluate the real performance of NDN. The use of a network simulator or NDN on top of IP limits the performance of the proposed solutions to the performance of the underlaying protocol (e.g., TCP or UDP). The use of Raspberry Pi network running native NDN protocol is a suitable fashion to achieve that. Similarly, the deployment cost is another metric to evaluate. For instance, the use of secure communication protocol or distributed caching schemes may have an impact on resource management and energy consumption.

Concerning the long-term perspectives, the proposed ICN caching scheme can be developed from a context-aware point-of-view. Only by using the content naming and some semantic attribute attached with the content, the ICN caching plane decides whether to cache the content or not. Advanced deep learning-based prediction mechanisms (e.g., reinforcement learning) can also be adapted to predict the content popularity, and hence decide the content cache placement and replacement. Finally, a comprehensive study is required to decide if a network needs to migrate to native ICN protocol or using ICN on top IP protocol. Both technical and economical factors need to be studied.

参考文献

[1] Al-Fuqaha A, Guizani M, Mohammadi M, et al. Internet of things: A survey on enabling technologies, protocols, and applications[J]. IEEE Communications Surveys & Tutorials, 2015, 17(4): 2347–2376.

[2] Choi J, Han J, Cho E, et al. A survey on content-oriented networking for efficient content delivery [J]. IEEE Communications Magazine, 2011, 49(3).

[3] Sheng Z, Yang S, Yu Y, et al. A survey on the IETF protocol suite for the internet of things: Standards, challenges, and opportunities[J]. IEEE Wireless Communications, 2013, 20(6): 91–98.

[4] Chen E T. The Internet of Things: Opportunities, Issues, and Challenges[M]. The Internet of Things in the Modern Business Environment. IGI Global, 2017: 167–187.

[5] Pan J, Paul S, Jain R. A survey of the research on future internet architectures[J]. IEEE Communications Magazine, 2011, 49(7).

[6] Amadeo M, Campolo C, Quevedo J, et al. Information-centric networking for the internet of things: challenges and opportunities[J]. IEEE Network, 2016, 30(2): 92–100.

[7] Waltari O, Kangasharju J. Content-centric networking in the internet of things[C]. IEEE Annual Consumer Communications & Networking Conference (CCNC). 2016: 73–78.

[8] Amadeo M, Campolo C, Iera A, et al. Named data networking for IoT: An architectural perspective [C]. European Conference on Networks and Communications (EuCNC). IEEE, 2014: 1–5.

[9] Ahlgren B, Dannewitz C, Imbrenda C, et al. A survey of information-centric networking[J]. IEEE Communications Magazine, 2012, 50(7).

[10] Koponen T, Chawla M, Chun B G, et al. A data-oriented (and beyond) network architecture[C]. ACM SIGCOMM Computer Communication Review: volume 37. 2007: 181–192.

[11] Fotiou N, Nikander P, Trossen D, et al. Developing Information Networking Further: From PSIRP to PURSUIT[C]. Conference on Broadband. Springer, 2010: 1–13.

[12] Dannewitz C, Kutscher D, Ohlman B, et al. Network of Information (NetInf) - An information-centric networking architecture[J]. Computer Communications, 2013, 36(7): 721–735.

[13] Oehlmann F. Content-Centric Networking[J]. Seminar FI & IITM: Network Architectures and Services, 2013, 43: 11–18.

[14] Zhang L, Estrin D, Burke J, et al. Named Data Networking (NDN) Project[J]. Technical Report NDN-0001, Xerox Palo Alto Research Center-PARC, 2010.

[15] Nour B, Sharif K, Li F, et al. A Survey of Internet of Things Communication using ICN: A Use Case Perspective[J]. Computer Communications, 2019, 142-143: 95–123.

[16] Nour B, Sharif K, Li F, et al. Security and Privacy Challenges in Information Centric Wireless IoT Networks[J]. IEEE Security & Privacy, 2020, 18(2): 35–45.

[17] Nour B, Sharif K, Li F, et al. M2HAV: A Standardized ICN Naming Scheme for Wireless Devices in Internet of Things[C]. International Conference Wireless Algorithms, Systems, and Applications (WASA). Guilin, China: Springer International Publishing, 2017: 289–301.

[18] Nour B, Sharif K, Li F, et al. A Unified Hybrid Information-Centric Naming Scheme for IoT Applications[J]. Computer Communications, 2020, 150: 103–114.

[19] Nour B, Sharif K, Li F, et al. A Distributed ICN-based IoT Network Architecture: An Ambient Assisted Living Application Case Study[C]. IEEE Global Communications Conference (GLOBECOM). Singapore: IEEE, 2017: 1–6.

[20] Nour B, Sharif K, Li F, et al. ICN Publisher-Subscriber Models: Challenges and Group-based Communication[J]. IEEE Network, 2019, 33(6): 156–163.

[21] Nour B, Sharif K, Li F, et al. NCP: A Near ICN Cache Placement Scheme for IoT-based Traffic Class[C]. IEEE Global Communications Conference (GLOBECOM). Abu Dhabi, United Arab Emirates: IEEE, 2018: 1–6.

[22] Nour B, Sharif K, Li F, et al. NNCP: A Named Data Network Control Protocol for IoT Applications [C]. IEEE Conference on Standards for Communications and Networking (CSCN). Paris, France: IEEE, 2018: 1–6.

[23] Xylomenos G, Ververidis C N, Siris V A, et al. A survey of information-centric networking research [J]. IEEE Communications Surveys & Tutorials, 2014, 16(2): 1024–1049.

[24] Lin J, Yu W, Zhang N, et al. A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications[J]. IEEE Internet of Things Journal, 2017, 4(5): 1125–1142.

[25] Arjunwadkar D P. Introduction of NDN with Comparison to Current Internet Architecture based on TCP/IP[J]. International Journal of Computer Applications, 2014, 105(5): 31–35.

[26] Passarella A. A survey on content-centric technologies for the current Internet: CDN and P2P solutions[J]. Computer Communications, 2012, 35(1): 1–32.

[27] Bari M F, Chowdhury S R, Ahmed R, et al. A survey of naming and routing in information-centric networks[J]. IEEE Communications Magazine, 2012, 50(12).

[28] Arshad S, Shahzaad B, Azam M A, et al. Hierarchical and Flat based Hybrid Naming Scheme in Content-Centric Networks of Things[J]. IEEE Internet of Things Journal, 2018, 5(2): 1070–1080.

[29] Baugher M, Davie B, Narayanan A, et al. Self-verifying names for read-only named data[C]. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). 2012: 274–279.

[30] Ascigil O, Ree S, Xylomenos G, et al. A keyword-based ICN-IoT platform[C]. ACM Conference on Information-Centric Networking (ICN). 2017: 22–28.

[31] Abdullahi I, Arif S, Hassan S. Survey on caching approaches in Information Centric Networking [J]. Journal of Network and Computer Applications, 2015, 56: 48–59.

[32] Din I U, Hassan S, Khan M K, et al. Caching in information-centric networking: Strategies, challenges, and future research directions[J]. IEEE Communications Surveys & Tutorials, 2018, 20(2): 1443–1474.

[33] Bernardini C, Silverston T, Vasilakos A. Caching Strategies for Information Centric Networking: Opportunities and Challenges[J]. arXiv preprint arXiv:1606.07630, 2016.

[34] Bernardini C, Silverston T, Festor O. A comparison of caching strategies for content centric networking[C]. IEEE Global Communications Conference (GLOBECOM). 2015: 1–6.

[35] Zhang Z, Yu Y, Zhang H, et al. An Overview of Security Support in Named Data Networking[J]. IEEE Communications Magazine, 2018, 56(11): 62–68.

[36] Yu Y, Afanasyev A, Clark D, et al. Schematizing Trust in Named Data Networking[C]. ACM International Conference on Information-Centric Networking. 2015: 177–186.

[37] He P, Wan Y, Xia Q, et al. LASA: Lightweight, Auditable and Secure Access Control in ICN with Limitation of Access Times[C]. IEEE International Conference on Communications (ICC). 2018: 1–6.

[38] Fotiou N, Alzahrani B A. Rendezvous-based access control for information-centric architectures [J]. International Journal of Network Management, 2018, 28(1): 1–11.

[39] Tyson G, Sastry N, Rimac I, et al. A survey of mobility in information-centric networks: Challenges and research directions[C]. ACM Workshop on Emerging Name-Oriented Mobile Networking Design-Architecture, Algorithms, and Applications. 2012: 1–6.

[40] Garcia G, Beben A, Ramon F J, et al. COMET: Content Mediator Architecture for Content-aware Networks[C]. Future Network Mobile Summit. [S.l.: s.n.], 2011.

[41] Melazzi N B, Salsano S, Detti A, et al. Publish/subscribe over information centric networks: A Standardized approach in CONVERGENCE[C]. Future Network Mobile Summit. [S.l.: s.n.], 2012.

[42] Seskar I, Nagaraja K, Nelson S, et al. Mobilityfirst: Future Internet Architecture Project[C]. Asian Internet Engineering Conference. ACM, 2011.

[43] Afanasyev A, Moiseenko I, Zhang L. ndnSIM: NDN simulator for NS-3[J]. NDN Technical Report NDN-0005, 2012: 1–7.

[44] Rayes A, Morrow M, Lake D. Internet of things implications on ICN[C]. International Conference on Collaboration Technologies and Systems (CTS). IEEE, 2012: 27–33.

[45] Shang W, Yu Y, Droms R, et al. Challenges in IoT networking via TCP/IP architecture[R]. [S.l.]: NDN Project, Tech. Rep. NDN-0038, 2016.

[46] Shang W, Bannis A, Liang T, et al. Named data networking of things[C]. IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI). 2016: 117–128.

[47] Baccelli E, Mehlis C, Hahm O, et al. Information centric networking in the IoT: Experiments with NDN in the wild[C]. International Conference on Information-centric Networking. ACM, 2014: 77–86.

[48] Lindgren A, Abdesslem F B, Ahlgren B, et al. Design choices for the IoT in information-centric networks[C]. IEEE Consumer Communications & Networking Conference (CCNC). 2016: 882–888.

[49] Zhang Y, Raychadhuri D, Grieco L A, et al. Design Considerations for Applying ICN to IoT[R/OL]. Internet Engineering Task Force, 2017. https://datatracker.ietf.org/doc/html/draft-zhang-icnrg-icniot-01.

[50] Lindgren A, Abdesslem F B, Ahlgren B, et al. Proposed Design Choices for IoT over Information Centric Networking[R/OL]. Internet Engineering Task Force, 2015. https://datatracker.ietf.org/doc/html/draft-lindgren-icnrg-designchoices-00.

[51] Zhang Y, Raychadhuri D, Grieco L A, et al. ICN based Architecture for IoT [R/OL]. Internet Engineering Task Force, 2017. https://datatracker.ietf.org/doc/html/draft-zhang-icnrg-icniot-architecture-01.

[52] Rao A, Schelén O, Lindgren A. Performance implications for IoT over information centric networks [C]. ACM Workshop on Challenged Networks. 2016: 57–62.

[53] Corujo D, Aguiar R L, Vidal I, et al. Research challenges towards a managed information-centric network of things[C]. European Conference on Networks and Communications (EuCNC). IEEE, 2014: 1–5.

[54] Quevedo J, Corujo D, Aguiar R. A case for ICN usage in IoT environments[C]. IEEE Global Communications Conference (GLOBECOM). 2014: 2770–2775.

[55] Piro G, Cianci I, Grieco L A, et al. Information centric services in smart cities[J]. Journal of Systems and Software, 2014, 88: 169–188.

[56] Ahmed S H, Bouk S H, Kim D, et al. Bringing Named Data Networks into Smart Cities[M]. John

Wiley & Sons, Inc. Hoboken, NJ, USA, 2017: 275–309.

[57] Mochida T, Nozaki D, Okamoto K, et al. Naming scheme using NLP machine learning method for network weather monitoring system based on ICN[C]. International Symposium on Wireless Personal Multimedia Communications (WPMC). 2017: 428–434.

[58] Naeem M, Ali R, Kim B S, et al. A periodic caching strategy solution for the smart city in information-centric Internet of Things[J]. Sustainability, 2018, 10(7): 2576.

[59] Ravindran R, Biswas T, Zhang X, et al. Information-centric networking based homenet[C]. IFIP/IEEE International Symposium on Integrated Network Management (IM). [S.l.]: IEEE, 2013: 1102–1108.

[60] Burke J, Gasti P, Nathan N, et al. Securing instrumented environments over content-centric networking: the case of lighting control and NDN[C]. IEEE Conference on Computer Communications Workshops (INFOCOM Wkshps). 2013: 394–398.

[61] Amadeo M, Campolo C, Iera A, et al. Information Centric Networking in IoT scenarios: The case of a smart home[C]. IEEE International Conference on Communications (ICC). 2015: 648–653.

[62] Hail M A, Fischer S. IoT for AAL: An architecture via information-centric networking[C]. IEEE Global Communications Conference Workshops (GLOBECOM Wkshps). 2015: 1–6.

[63] Zhang H, Wang Z, Scherb C, et al. Sharing mHealth Data via Named Data Networking[C]. ACM Conference on Information-Centric Networking (ICN). 2016: 142–147.

[64] Saxena D, Raychoudhury V. Design and Verification of an NDN-Based Safety-Critical Application: A Case Study With Smart Healthcare[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2017.

[65] Zhang J, Li Q, Schooler E M. iHEMS: An information-centric approach to secure home energy management[C]. IEEE International Conference on Smart Grid Communications (SmartGridComm). 2012: 217–222.

[66] Katsaros K, Chai W, Wang N, et al. Information-centric networking for machine-to-machine data delivery: a case study in smart grid applications[J]. IEEE Network, 2014, 28(3): 58–64.

[67] Oh E. Secure Information Network Design Strategies for Information-Centric Future Smart Grid [C]. Applied Science and Engineering for Better Human Life. 2016.

[68] Bouk S H, Ahmed S H, Kim D, et al. Named-data-networking-based ITS for smart cities[J]. IEEE Communications Magazine, 2017, 55(1): 105–111.

[69] Amadeo M, Campolo C, Molinaro A. Information-centric networking for connected vehicles: a survey and future perspectives[J]. IEEE Communications Magazine, 2016, 54(2): 98–104.

[70] Saxena D, Raychoudhury V, Becker C. Implementation and Performance Evaluation of Name-based Forwarding Schemes in V-NDN[C]. International Conference on Distributed Computing and Networking. ACM, 2017: 35.

[71] Vahdat A, Becker D. Epidemic routing for partially connected ad hoc networks[M]. [S.l.]: Technical Report CS-200006, Duke University, 2000.

[72] Spyropoulos T, Psounis K, Raghavendra C S. Spray and wait: an efficient routing scheme for intermittently connected mobile networks[C]. ACM SIGCOMM workshop on Delay-tolerant networking. 2005: 252–259.

[73] Hou F, Shen X. An adaptive forwarding scheme for message delivery over delay tolerant networks [C]. IEEE Global Telecommunications Conference. 2009: 1–5.

[74] Chowdhury M, Gawande A, Wang L. Secure Information Sharing among Autonomous Vehicles in NDN[C]. IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI). [S.l.]: IEEE, 2017: 15–26.

[75] Chowdhury M, Gawande A, Wang L. Anonymous authentication and pseudonym-renewal for VANET in NDN[C]. ACM Conference on Information-Centric Networking (ICN). 2017: 222–223.

[76] Signorello S, Palattella M R, Grieco L A. Security Challenges in Future NDN-Enabled VANETs [C]. IEEE Trustcom/BigDataSE/ISPA. 2016: 1771–1775.

[77] Bouk S H, Ahmed S H, Hussain R, et al. Named Data Networking's Intrinsic Cyber-Resilience for Vehicular CPS[J]. IEEE Access, 2018, 6: 60570–60585.

[78] Polyzos G C, Fotiou N. Building a reliable internet of things using information-centric networking [J]. Journal of Reliable Intelligent Environments, 2015, 1(1): 47–58.

[79] Adhatarao S S, Chen J, Arumaithurai M, et al. Comparison of naming schema in ICN[C]. IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN). 2016: 1–6.

[80] Bracciale L, Loreti P, Detti A, et al. Lightweight Named Object: an ICN-based Abstraction for IoT Device Programming and Management[J]. IEEE Internet of Things Journal, 2019: 1–1.

[81] Arshad S, Azam M A, Ahmed S H, et al. Towards Information-Centric Networking (ICN) naming for Internet of Things (IoT): the case of smart campus[C]. International Conference on Future Networks and Distributed Systems (ICFNDS). ACM, 2017: 30.

[82] Yang Y, Song T. Local naming service for named data networking of things[C]. ACM Conference on Information-Centric Networking (ICN). 2017: 190–191.

[83] Papalini M, Carzaniga A, Khazaei K, et al. Scalable routing for tag-based information-centric networking[C]. ACM Conference on Information-Centric Networking (ICN). 2014: 17–26.

[84] Melvix L, Lokesh V, Polyzos G C. Energy Efficient Context based Forwarding Strategy in Named Data Networking of Things[C]. ACM Conference on Information-Centric Networking (ICN). 2016: 249–254.

[85] Tsilopoulos C, Xylomenos G. Supporting diverse traffic types in information centric networks[C]. ACM Conference on Information-Centric Networking (ICN). 2011: 13–18.

[86] Moll P, Janda J, Hellwagner H. Adaptive Forwarding of Persistent Interests in Named Data Networking[C]. ACM Conference on Information-Centric Networking (ICN). 2017.

[87] Quevedo J, Antunes M, Corujo D, et al. On the application of contextual IoT service discovery in Information Centric Networks[J]. Computer Communications, 2016, 89: 117–127.

[88] Antunes M, Gomes D, Aguiar R. Semantic features for context organization[C]. International Conference on Future Internet of Things and Cloud (FiCloud). IEEE, 2015: 87–92.

[89] Ascigil O, Sourlas V, Psaras I, et al. A native content discovery mechanism for the information-centric networks[C]. ACM Conference on Information-Centric Networking (ICN). 2017: 145–155.

[90] Meddeb M, Dhraief A, Belghith A, et al. How to cache in ICN-based IoT environments?[C]. IEEE/ACS International Conference on Computer Systems and Applications (AICCSA). 2017: 1117–1124.

[91] Zhang G, Li Y, Lin T. Caching in information centric networking: A survey[J]. Computer Networks, 2013, 57(16): 3128–3141.

[92] Vural S, Navaratnam P, Wang N, et al. In-network caching of internet-of-things data[C]. IEEE International Conference on Communications (ICC). 2014: 3185–3190.

[93] Xu C, Wang X. Transient content caching and updating with modified harmony search for Internet of Things[J]. Digital Communications and Networks, 2018.

[94] Zhang Z, Lung C H, Lambadaris I, et al. IoT data lifetime-based cooperative caching scheme for ICN-IoT networks[C]. IEEE International Conference on Communications (ICC). IEEE, 2018: 1–7.

[95] Kim D, Nam S, Bi J, et al. Efficient content verification in named data networking[C]. ACM Conference on Information-Centric Networking (ICN). 2015: 109–116.

[96] Kim D, Bi J, Vasilakos A V, et al. Security of Cached Content in NDN[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(12): 2933–2944.

[97] Seetharam A. On Caching and Routing in Information-Centric Networks[J]. IEEE Communications Magazine, 2018, 56(3): 204–209.

[98] Ran J, Lv N, Zhang D, et al. On performance of cache policies in named data networking[C].

International Conference on Advanced Computer Science and Electronics Information. 2013: 668–671.

[99] Dai H, Wang Y, Wu H, et al. Towards line-speed and accurate on-line popularity monitoring on NDN routers[C]. International Symposium of Quality of Service (IWQoS). IEEE, 2014: 178–187.

[100] Meddeb M, Dhraief A, Belghith A, et al. Least fresh first cache replacement policy for NDN-based IoT networks[J]. Pervasive and Mobile Computing, 2019, 52: 60–70.

[101] Hyndman R J, Athanasopoulos G. Forecasting: Principles and Practice[M]. [S.l.]: OTexts, 2018.

[102] Amadeo M, Campolo C, Molinaro A. Internet of things via named data networking: The support of push traffic[C]. International Conference and Workshop on the Network of the Future (NOF). IEEE, 2014: 1–5.

[103] Tagami A, Yagyu T, Sugiyama K, et al. Name-based push/pull message dissemination for disaster message board[C]. IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN). IEEE, 2016: 1–6.

[104] Moll P, Theuermann S, Hellwagner H. Persistent Interests in Named Data Networking[C]. IEEE Vehicular Technology Conference (VTC Spring). 2018: 1–5.

[105] Arnould G, Khadraoui D, Habbas Z. A self-organizing content centric network model for hybrid vehicular ad-hoc networks[C]. ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet). 2011: 15–22.

[106] Majeed M F, Ahmed S H, Dailey M N. Enabling Push-Based Critical Data Forwarding in Vehicular Named Data Networks[J]. IEEE Communications Letters, 2017, 21(4): 873–876.

[107] Bouk S H, Ahmed S H, Yaqub M A, et al. DPEL: Dynamic PIT Entry Lifetime in Vehicular Named Data Networks[J]. IEEE Communications Letters, 2016, 20(2): 336–339.

[108] Li R, Asaeda H, Li J. A Distributed Publisher-Driven Secure Data Sharing Scheme for Information-Centric IoT[J]. IEEE Internet of Things Journal, 2017, 4(3): 791–803.

[109] Gündoğan C, Kietzmann P, Schmidt T C, et al. HoPP: Robust and Resilient Publish-Subscribe for an Information-Centric Internet of Things[J]. arXiv preprint arXiv:1801.03890, 2018.

[110] Thomas J H. Variations on the Fibonacci Universal Code[J]. CoRR, 2007.

[111] Yu Y, Li Y, Du X, et al. Content Protection in Named Data Networking: Challenges and Potential Solutions[J]. IEEE Communications Magazine, 2018, 56(11): 82–87.

[112] Kwak D W, Lee S J, Kim J W, et al. An efficient LKH tree balancing algorithm for group key management[J]. IEEE Communications Letters, 2006, 10(3): 222–224.

[113] Byrka J, Pensyl T, Rybicki B, et al. An Improved Approximation Algorithm for Knapsack Median

Using Sparsification[J]. Algorithmica, 2018, 80(4): 1093–1114.

[114] Bera S, Misra S, Vasilakos A V. Software-defined networking for internet of things: A survey[J]. IEEE Internet of Things Journal, 2017, 4(6): 1994–2008.

[115] Afanasyev A, Shi J, Wang L, et al. Packet Fragmentation in NDN: Why NDN Uses Hop-By-Hop Fragmentation[J]. NDN Technical Report NDN-0032, 2015.

[116] NDN TLV-TYPE number assignment[M]. [S.l.: s.n.].

[117] Shi J, Zhang B. NDNLP : A Link Protocol for NDN[J]. NDN Technical Report NDN-0006, 2012: 1–15.

[118] Kietzmann P, Gündoğan C, Schmidt T C, et al. The need for a name to MAC address mapping in NDN: towards quantifying the resource gain[C]. ACM ICN Conference. [S.l.: s.n.], 2017.

[119] Afanasyev A, Jiang X, Yu Y, et al. NDNS: A DNS-Like Name Service for NDN[C]. IEEE ICCCN Conference. [S.l.: s.n.], 2017: 1–9.

[120] Mobile Video Delivery with Hybrid ICN: IP-integrated ICN solution for 5G[J]. Cisco, 2017.

List of Publications

[1] Nour B, Sharif K, Li F, *et al.* "A Unified Hybrid Information-Centric Naming Scheme for IoT Applications" [J]. *Computer Communications* (SCI 二区, IF: 2.766). 2020, 150: 103–114. (DOI: 10.1016/j.comcom.2019.11.020).

[2] Nour B, Sharif K, Li F, *et al.* "Security and Privacy Challenges in Information Centric Wireless IoT Networks"[J]. *IEEE Security & Privacy* (SCI 三区, IF: 1.596). 2020, 18(2): 35-45. (DOI: 10.1109/MSEC.2019.2925337).

[3] Nour B, Sharif K, Li F, *et al.* "ICN Publisher-Subscriber Models: Challenges and Group-based Communication"[J]. *IEEE Network* (SCI 一区, IF: 7.503). 2019, 33(6): 156-163. (DOI: 10.1109/MNET.2019.1800551).

[4] Nour B, Sharif K, Li F, *et al.* "A Survey of Internet of Things Communication using ICN: A Use case Perspective"[J]. *Computer Communications* (SCI 二区, IF: 2.766). 2019, 142-143: 95–12. (DOI: 10.1016/j.comcom.2019.05.010).

[5] Nour B, Sharif K, Li F, *et al.* "NCP - A Near ICN Cache Placement Scheme for IoT-based Traffic Class"[C]. *IEEE Global Communications Conference* (GLOBECOM) (SCI, CCF 三区). Abu Dhabi, United Arab Emirates: IEEE, 2018: 1 – 6. (DOI: 10.1109/GLOCOM.2018.8647629).

[6] Nour B, Sharif K, Li F, *et al.* "NNCP: A Named Data Network Control Protocol for IoT Applications"[C]. *IEEE Conference on Standards for Communications and Networking* (CSCN) (EI). Paris, France: IEEE, 2018: 1–6. (DOI: 10.1109/CSCN.2018.8581844).

[7] Nour B, Sharif K, Li F, *et al.* "A Distributed ICN-based IoT Network Architecture: An Ambient Assisted Living Application Case Study"[C]. *IEEE Global Communications Conference* (GLOBECOM) (SCI, CCF 三区). Singapore: IEEE, 2017: 1 – 6. (DOI:10.1109/GLOCOM.2017.8255022).

[8] Nour B, Sharif K, Li F, *et al.* "M2HAV: A Standardized ICN Naming Scheme for Wireless Devices in Internet of Things"[C]. *International Conference on Wireless Algorithms, Systems, and Applications* (WASA) (EI, CCF 三区). Guilin, China: Springer International Publishing, 2017: 289–301. (DOI: 10.1007/978-3-319-60033-8_26).