sustainability

MDPI

Article

Hierarchical, Attribute and Hash-Based Naming and Forwarding Aided Smart Campus of Things

Sobia Arshad [1,*], Muhammad Awais Azam [2], Jonathan Loo [3], Muhammad Faran Majeed [4] and Ali Haider [5]

[1] Department of Computer Engineering, HITEC Uiversity, Taxila 47080, Pakistan
[2] Technology Innovation Research Group, School of Information Technology, Whitecliffe, Wellington 6145, New Zealand
[3] School of Computing and Engineering, University of West London, St Mary's Road, Ealing, London W5 5RF, UK
[4] Department of Computer Science, Kohsar University Murree, Murree 47150, Pakistan
[5] Computer Science and Information Technology Department, University of Sargodha, Sargodha 40100, Pakistan
* Correspondence: sobia.arshad@hitecuni.edu.pk

Abstract: In order to provide universal ability to access information and communication among Internet-connected devices, the Sustainable Internet of Things (IoT) is on a mission to bring all objects or devices under one roof. Future Internet architecture, especially Information-Centric Networking (ICN), can easily handle the connectivity offered and information created by the massive amount of devices to make it as sustainable IoT applications. Named Data Networking (NDN), one of the several future Internet designs that employ ICN as its foundation, shows promise. NDN integration with IoT-based applications gives solutions to numerous problems. However, this fusion makes accessing the IoT content easier, provided that an effective naming scheme is created to execute this operation. In this work, we build an innovative NDN-based naming scheme (NDN–NS) and put it into practise for consumer, producer, and content routers using our own secure forwarding schemes (NDN–NFS). Due to its scalability, heterogeneity, and security needs, IoT-based Smart Campus (IoT-SC) scenarios are taken into consideration for design and evaluation. We give a complete activity list based on NDN–NS that is split into two communication models (PusH Type Communication (PHTC) and PulL Type Communication (PLTC)) that can be applied to any IoT application. In terms of interest satisfaction rate (ISR), delay, and number of transmissions, we compare the NDN–NFS to legacy NDN. The outcomes demonstrate that NDN–NFS outperforms classic NDN in terms of performance and efficiency.

Keywords: content-centric Networks; named data networking; IoT traffic types; NDN-IoT smart campus; NDN naming and forwarding schemes; Information Centric Networking

Citation: Arshad, S.; Azam, M.A.; Loo, J.; Majeed, M.F.; Haider, A. Hierarchical, Attribute and Hash-Based Naming and Forwarding Aided Smart Campus of Things. *Sustainability* **2023**, *15*, 16361. https://doi.org/10.3390/su152316361

Academic Editors: Luca Carrubbo and Leonard Walletzký

Received: 15 November 2022
Revised: 24 May 2023
Accepted: 10 July 2023
Published: 28 November 2023

1. Introduction

IoT is a known concept through which it is aimed to achieve dynamic and global connectivity by combining both physical and virtual things with the help of inter-operable protocols [1]. Moreover, a plethora of sustainable smart devices and their connectivity have aided to transform this concept into reality. Sustainable smart devices can range from small constraint-oriented devices to super machines with larger specifications. When billions of these sustainable smart devices connect to the Internet, they produce much bigger data which appear as Big Data. Sustainable IoT application consumers can fetch the required data from any nearby device. Managing such Big Data along with billions of sustainable devices through legacy networking architecture (TCP/IP) is less feasible. This is why in the last seven years, people have started to think to design sustainable IoT network architecture on the basis of a well-known concept of Information-Centric Networking (ICN) [2–4]. Prominent features of ICN include focus on the following: named and secured contents [5] rather than devices, a receiver-driven pull-based communication model, data availability

through in-network caching [6–8], freshness and popularity of content [9], detection of cache pollution [10], probabilistic forwarding [11] and mobility support through a publish–subscribe mechanism [12,13]. There are many projects which explore ICN as a basis for Future Internet Architecture (FIA) instead of TCP/IP [14]. DONA (Data Oriented Network Architecture), CCN (Content Centric Networking), PURSUIT (Publish SUbscribe Internet Technology), COMET (COntent Mediator architecture for content aware nETworks), MobilityFirst, C-DAX (Cyber-secure Data And Control Cloud for Power Grids), SAIL (Scalable and Adaptive Internet soLutions), Green ICN and CONVERGENCE come under the umbrella of ICN [15]. However, in the context of IoT, Named Data Networking (NDN) is playing an important role which was formerly known as CCN. A list of important acronyms used in this paper is defined in Table 1.

Furthermore, the IETF (Internet Engineering Task Force)-based Information-Centric Networking Research Group (ICNRG) has begun to investigate ICN in the context of IoT [16]. Although, as stated in [17,18], there are still numerous IoT-related issues that need to be investigated, and content naming is one of them. We suggest that naming is the key problem that needs to be solved first in order to construct a sustainable IoT on the foundation of ICN. This is because extensive naming schemes can be used to provide solutions to other problems such as security and in-network caching. Regarding this point of view, ICN offers three naming mechanisms: (i) attribute-based naming, (ii) flat-based naming, and (iii) hierarchical naming [19].

Table 1. List of Acronyms Used.

Acronyms	Definitions
CCN	Content Centric Networking
ICN	Information Centric Networking
IoT-SC	IoT-based Smart Campus
ISR	Interest Satisfaction Rate
MTC	Machine Type Communication
NDN–NS	NDN-based Naming Scheme
NDN–NFS	NDN-based Naming based Forwarding Schemes
PHTC	PusH Type Communication
PLTC	PulL Type communication

CCN and NDN employ hierarchical naming, whereas DONA, COMET, SAIL, PURSUIT, CONVERGENCE, and MobilityFirst adopt self-certifying flat-based naming. Additionally, the CBCB (Combined Broadcast and Content-Based) utilises an attribute-based naming method. When The attribute-based method is combined with the other two schemes, a hybrid integrated content naming scheme can be designed. Due to the advantages offered by this hybrid naming scheme [19], we investigate the hybrid integrated naming scheme while integrating the hierarchical and flat naming systems in [20] since it can play a significant role.

1.1. Motivation

The naming of IoT information (content) is the true power of NDN for IoT. IoT content has some distinctive features, such as being ephemeral and frequently requiring location data, various freshness levels, and probability. We combine hierarchical and flat-based self-certifying to create the NDN-based naming scheme (NDN–NS) in order to provide attribute-based naming, which can be very helpful in lookup with attributes. Then, we divide IoT traffic types into seven categories using NDN–NS. We used these seven identified IoT traffic types to design ICN–IoT with enabled Machine Type Communication (MTC). As MTC is the main pillar of IoT, so we designed NDN–IoT in a way to support both push

and pull communication models. We achieved MTC functionality through proposed NDN–NS-based forwarding schemes, which we call NDN–NFS. We call proposed forwarding schemes NDN–NFS since these forwarding schemes for the NDN producer, consumer, and content routers (CRs) are designed on the basis of NDN–NS. We also proposed a lighter version of GuardICNg (which is an authentication and authorization scheme) to enable security in these forwarding schemes.

1.2. Our Contributions

The summary of our main contributions is as follows:

1. We describe the designing details of the proposed NDN–NS for IoT-based smart campus (IoT-SC). NDN–NS incorporates content and device properties, and names through three components: the hierarchical component (HNC), the attribute component (ANC) and the flat component (FNC). NDN–NS's main aim is to design a comprehensive naming scheme for IoT. Therefore, every sub-component or attribute is included in the name after very careful analysis so that it is applicable for general IoT (Refer Figures 1 and 2 and Algorithm 1).
2. We analyse the proposed NDN–NS to propose forwarding schemes NDN–NFS for the consumer, producer, and content routers. These NDN–NFS ensure that the message is received/sent by a tentative node (Refer Algorithms 3 and 5).
3. To achieve MTC through NDN–NFS, we incorporate *task and sub-task* as content name parts to decide message type and thus communication type as pull or push. NDN–IoT consumer can both push and pull the data towards/from IoT network (push when the consumer wants to perform some action and pull when the consumer wants to fetch some data) while NDN–IoT producer is enabled to send data (either critical or subscribed update) to the consumer without a request from the consumer. This is how we enable PusH Type Communication (PHTC) (Refer Figure 2 and Algorithm 2).
4. To maintain content security in terms of integrity, we include the hashes of content name, sub-name and IDs of devices at the end of the message. We also use this hashed information in GuardICNg to authenticate and authorize the message sender to validate its right to perform the action (Refer Figure 2 and Algorithm 4).
5. We simulate scenario NDN–IoT–SC (which can be visualized in Figure 3) with both static and mobile nodes in ndnSIM [21] and evaluate forwarding schemes. We find that the proposed schemes outperform legacy NDN (i.e., legacy NDN is the one which is provided by ndnSIM) in terms of interest satisfaction rate (ISR), number of transmissions, and delay to perform pull- and push-based activities of NDN–IoT–SC.

1.3. Organization of the Work

The rest of the paper is organized as follows. Section 2 discusses the basics of NDN, ICN naming mechanisms, related NDN naming research efforts for IoT and challenges for NDN based IoT. In Section 3, we describe the process of naming NDN–NS and proposed NDN–NFS forwarding schemes along with their corresponding algorithms with the help of IoT traffic types and communication models. In Section 4, application of proposed methodology is presented for a scenario of IoT smart campus. Simulation and implementation details along with results and evaluations are discussed in Section 5. Finally, we conclude the paper in Section 6.

2. Related Literature: Basics and ICN (NDN)-IoT Related Naming Research Efforts

In this section, the basics of NDN and ICN naming mechanisms are discussed, followed by related NDN- (ICN)-based naming schemes for IoT and challenges for NDN-based naming for IoT.

2.1. NDN Basics

NDN is one of the most promising projects of ICN. One of the reasons for NDN being such a successful project is that almost all active research is using it [12]. In NDN [22], there are three data structures that include content store (CS), pending interest table (PIT) or,

more specifically, interest table and forwarding information base (FIB). Each of these data structures have different roles and priorities. Whenever an interest message [23] against a content object is received, it is first searched in the PIT. If a PIT entry is found, then this new request is also aggregated with the old PIT entry. If the interest is not in PIT, then it is checked in CS and if corresponding content is found, it is replied to. In case the content object is not found in CS, then it is further checked in FIB. If a matching entry exists in FIB, then Interest is forwarded to that interface. Otherwise, upon an unsuccessful search, the interest is broadcast. Furthermore, the legacy NDN supports only pull-based mechanism [24], in which a consumer requests a content object by issuing an interest to the producer of the content object. However, there are some applications that require the pull-based mechanism to be overridden with a push-based mechanism and IoT-based applications are among them [24]. NDN follows hierarchical naming [25]; however, there are other ICN naming mechanisms which are discussed in the following subsection, along with their advantages and disadvantages.

2.2. ICN Naming Mechanisms

There are three basic naming mechanisms that ICN offers (hierarchical, flat, attribute). Each mechansism is briefly described below. However, a fourth one (hybrid) can be designed by using the basic mechanisms. Table 2 provides a summary of their merits and demerits.

In the ICN hierarchical naming scheme [26], names are human-readable and well-structured by using the website URL to form a hierarchy to name any content. Name components are separated by a delimiter '/'; for example, a hierarchical name can be */vip-lab/action/on/ac1*. Hierarchical names may have variable length and each component of a hierarchical name defines the characteristics of the content. The main advantage of hierarchical naming is that it provides good aggregation to reduce the number of routing information and has a faster lookup mechanism, which in return enhances scalability [27]. Further, it has good compatibility with the current applications and with the systems which are URL based. However, hierarchical names require some external trust mechanism for security and flexibility [28]. In addition, routing and forwarding table sizes can be huge as hierarchical names consist of a variable number of components and the length of each component can also be variable. And this huge size of these tables leads to decreased lookup efficiency. Moreover, users need to provide a full name (path) of the content for searching, which is difficult to memorize; for example, if a user wants to access the content '*abc.mp4*', then the user has to provide an official name (full path) to access the content which can look like '*fb/video/new/abc.mp4*'.

In the ICN flat naming scheme [29], flat names are also called self-certifying names. Usually, flat names are produced by applying a hash algorithm on the whole content name or on any part of it. Therefore, flat names are of fixed length and mostly are unreadable, but these can also be readable. The main advantage of flat names is that these are short in length (which leads a more efficient use of memory in constraint-oriented devices and offers easy searching) and more secure than hierarchical names. Moreover, flat naming offers flexibility in the naming structure because it has no fixed structure. However, flat names are not scalable and thus exhibit a low degree of aggregation, which increases the number of entries in the routing table. Usually, flat names are not human-readable, making it difficult to memorize the official names to access the content. Therefore, it may require additional mapping of the names.

Table 2. Merits and Demerits Summary of ICN Naming Mechanisms.

ICN Naming Mechanisms	Hierarchical Naming	Flat Naming	Attribute-Based Naming
Merits	Human readability Flexible name length–>scalability Name-aggregation–>less routing table entries	Fixed Length Memory Efficient Standardized and Unique Security and Privacy Easy generation	No need to memorize, Efficient searching
Demerits	Difficult to memorize full name due to variable length, No security and privacy, No Standardization	No name-aggregation–>more routing table entries Not scalable, Not possible to memorize	Extra information, No name uniqueness
Content Name Example	/uettaxila.edu.pk/cped/ contentTitle.pdf	/uettaxila.edu.pk/cped/ sha256(contentTitle)	contentTitle<str>: 'contentTitle', department<str>: 'Department', location<str>: 'Location', date<int>: 'Date', version<float>: 'Version'

In the ICN attribute naming scheme [30], names are specified using some known attributes of the content, including the following: category, level, format, date/time, version, or features such as content freshness and content popularity. The main advantage of attribute naming is that any content can be found based on the properties of the content. It supports searching based on keywords and fuzzy matching; thus, the user does not need to remember the official name of content. However, attribute names are non-unique and to make them unique, a large number of attributes would have to be stored in the routing system.

In the hybrid integrated naming scheme, any two or all of the above-mentioned naming schemes (hierarchical, flat, attribute) are integrated to get the benefits of these basic naming schemes and to overcome their disadvantages. Therefore, a hybrid integrated name can be an integrated version of two or three naming schemes. In the hybrid or integrated naming scheme, a hierarchical name can be used to improve the compatibility, a flat name can be used to improve the uniqueness or for high-security purposes, and an attribute name can be used to support fuzzy match searching in an architecture. In the hybrid integrated naming scheme, the hierarchical names can be generated by following a standard structure, the flat name can be generated by using a hashing technique while the attribute name can be generated by extracting the information from data such as file type, issue time, location, popularity, and file size.

In light of the benefits of hybrid integrated naming schemes identified above, we design NDN–IoT–SC naming using the hybrid integrated naming scheme.

2.3. Related NDN-Based Naming Schemes for IoT Applications

In both [31,32], NDN has been used for smart houses. The native NDN hierarchical naming method for smart homes is covered in both works. Additionally, ref. [33] found that NDN is more feasible when local traffic is present and this work has been implemented for lighting the home. For naming the data, they also use the NDN basic naming method. NDN is used for vehicular networks in [34,35]. Through NDN in [36], a smart campus lighting system is set into operation. Furthermore, they name the contents using the fundamental NDN hierarchical naming scheme. In order to handle underwater devices and their data and to create smart underwater environments, NDN is developed for underwater networks in [37]. They also created and used hierarchical naming conventions to name the contents connected to water. For IoT smart buildings without MTC support, a naming and encoding method based on ICN is developed in [38]. In [20], CCN-based hybrid naming incorporating only hierarchical and flat-based naming schemes were proposed for IoT-based smart campus, but without true secure realization of MTC. This work was CCN-based and security constraints such as authorization and authentication to perform particular

tasks were missing. Moreover, in [39], only the hybrid naming scheme is proposed, without any evaluations and analysis.

The majority of the aforementioned methods name the information exclusively using hierarchical naming algorithms, without push support, security, or content attributes. MTC-supported IoT, on the other hand, is crucial since it needs both push and pull-type communications as well as a strong support for security. The security of IoT-SC must be maintained in terms of authorization, authentication, and integrity.

2.4. Challenges for NDN–IoT Naming

NDN's hierarchical naming is truly universal and can incorporate various other naming schemes. However, major challenges of NDN–IoT naming mechanism are as follows:

- It should provide scalability and flexibility so that billions of contents can be named as IoT involves billions of devices which produce these contents.
- It should secure both IoT contents and devices. Moreover, the naming mechanism should provide authentication and authorization to IoT applications.
- It should provide MTC so that both push and pull operations can be performed.
- Regarding MTC, it should support all activities which can be performed in any IoT application.
- It should provide fast lookup using keywords or attributes.
- Finally, it should completely cover all the attributes of contents which can be used in any IoT application.

To support all of these essentials in a much better way, we selected the hybrid integrated naming scheme and propose NDN–NS to name NDN–IoT–SC contents in a meaningful way. We use the proposed NDN–NS to identify IoT traffic types to enable MTC and security through forwarding strategies, which we call NDN–NFS. NDN–NFS supports security in terms of content integrity, authentication, and authorization of message senders to validate tentative devices. Moreover, the forwarding strategies of NDN–NFS help to ensure that data are received only by intended devices.

3. Proposed Methodology: Proposed NDN-Based Naming Scheme (NDN–NS) and (NDN–NS)-Based Forwarding Schemes (NDN–NFS) for IoTs

In the following, we describe the design details of the NDN–NS for NDN–IoT–SC. We take IoT-SC as a scenario to describe NDN–NS, which is a true representative scenario of IoT due to its need for scalability, heterogeneity, security and MTC. In NDN–NS, features of hierarchical, flat, and attribute-based naming are combined. Moreover, we analyse NDN–NS to identify IoT traffic types. We use NDN–NS further to design the forwarding schemes NDN–NFS for the NDN–IoT producer and consumer/content router. Designing details of NDN–NFS are described through IoT traffic types and support for MTC.

3.1. Description of NDN–NS for IoT Application (IoT-SC)

The name components of NDN–NS can be seen in Figure 1. NDN–NS incorporates a hierarchical naming component as HNC, an attribute naming component as ANC and a flat naming component as FNC. In short, NDN–NS can be described as *HNC:ANC:FNC:/*. HNC is arranged to include general properties of application and content which can be used as a prefix. The proposed Algorithm 1, is designed to dissect the naming components. Any incoming message is processed through the Algorithm 1.

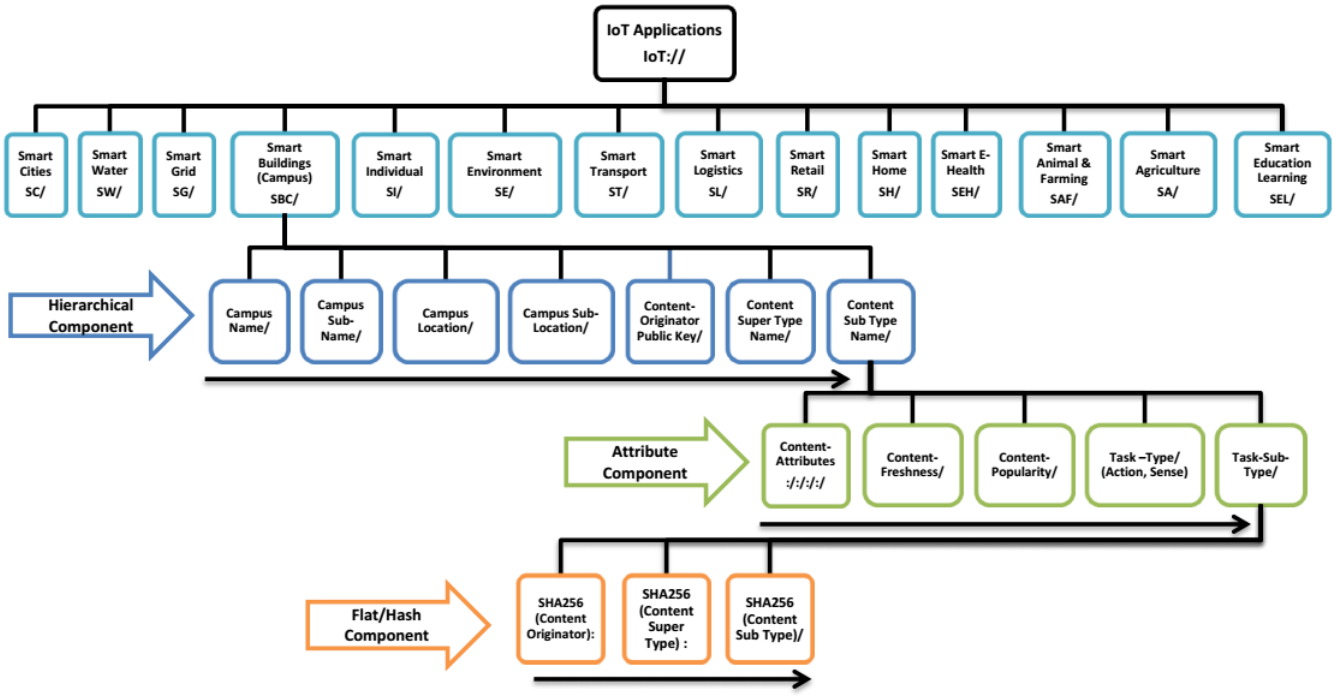

Figure 1. IoTs applications naming and resolution. Stage 1: Hierarchical Part and resolution. Stage 2: Attribute Part and resolution. Stage 3: Flat/Hash Part and resolution.

HNC includes information about the main IoT application (such as smart campus), location, device name or its public key, content name and the type (please refer to the upper portion of Figure 1 against Hierarchical Naming Component). These all-sub-parts of HNC are separated through '/' and ended by ':'.

Then, through ANC, we incorporate properties of content and MTC information. Content properties include general properties such as date, time, and version separated through ':/' and specific properties such as freshness, popularity and MTC information include task type and task subtype values separated through '/' (please refer to the middle portion of Figure 1 against attribute-based naming Component).

Lastly FNC, which is added with the purpose of providing security (integrity of content), includes hashed values of device information, and the content name and its type are appended and separated through ':/' (please refer to the lower portion of Figure 1 against the Flash hash-based naming component). NDN–NS include the name of components, which later helps in finding the content on the basis of attributes.

3.2. Description of NDN–IoT Message Types for NDN–IoT–SC

Through proposed Algorithm 1, naming components are extracted through scanning till last ':/'. When it comes to ANC, Algorithm 2 is called (line 11: Algorithm 1) to determine **message type** using the values of *Task-Type* and *Task-Sub-Type* as *action* and *subAction*. In Algorithm 1, Algorithm 2 is proposed and implemented to obtain the message type. In Algorithm 2, different values of *Task-Type* and *Task-Sub-Type* are compared and the message type is determined.

1. When the value of *Task-Type* and *Task-Sub-Type* is *action* and *retrieve* or *sense* and *retrieve*, then message type is *im* and it is for situation when some user is simply interested to get some data from any nearby devices or from nearby sensor.

Algorithm 1: Algorithm for NDN–NS

NDN–NS Message Processing for NDN–IoT–SC LS (Lab Sensors), CAS (CAmpus Server), Mobile Devices (MD) and LAS (LAb Servers, i.e., Wi-Fi)

Possible Input: Incoming message from any device
Possible Output: Incoming message is dissected and saved to call Algorithm 2 to identify *im*, *adv*, *dm*, *um*, *µm*, *bm*.

```
1  while last ':/' do
2      if '/' encounters then
3          save value before / in array[i] and update array[i+1]
4          continue scanning → goto while
5      else if '/:' encounters then
6          if first '/:' encounters then
7              save values of array[i-1] & array[i-2] & arr[i-3] into contentSuperName and
                 contentSubName and contentOriginatorID respectively
8              continue scannaing → goto while
9          else if second '/:' encounters then
10             save value of array[i-1] & array[i-2] into action and subAction respectively
11             Goto NDN–NS Message Type Determination Algorithm 2 and get
                 messageType
12             and then continue scanning → goto while
13     else if ':/' encounters then
14         if first ':/' encounters then
15             save value of array[i] into year
16             continue scanning → goto while
17         else if second ':/' encounters then
18             save value of array[i] into month and date
19             continue scanning → goto while
20         else if third ':/' encounters then
21             save value of array[i] into time
22             continue scanning → goto while
23         else if fourth ':/' encounters then
24             save value of array[i] into version
25             continue scanning → goto while
26         else if fifth ':/' encounters then
27             save value of array[i] into shaContentOriginatorID
28             continue scanning → goto while
29         else if sixth ':/' encounters then
30             save value of array[i] into variable shaContentSuperName
31             continue scanning → goto while
32         else if seventh ':/' encounters then
33             save value of array[i] into variable shaContentSubName
34             continue scanning → goto while
35         end
36     end
37 end
```

2 When the value of *Task-Type* and *Task-Sub-Type* is action and turn or *set-value* and *float-value*, then the message type is *um* and it is for situations when an NDN–IoT–SC user is interested in turning on or off some device or when the user wants to set the value (Temperature) of some device (AC).
3 When the value of *Task-Type* and *Task-Sub-Type* is *action* and *upload*, then the message type is *adv* and it is for situations when the user is interested in uploading some data.
4 When the value of *Task-Type* and *Task-Sub-Type* is *action* and *update*, then the message type is *µm* and it is for situations when the device (sensor) is interested in updating the subscribed periodic notification.

5 When the value of *Task-Type* and *Task-Sub-Type* is *action* and *trigger*, then the message type is *bm* and it is for emergency situations when the device (sensor) has to send an emergency notification without any subscription.

6 When the value of *Task-Type* and *Task-Sub-Type* is *action* and *receive*, then the message type is *dm* and it is for situations when the device (sensor) has to receive data.

We identify six message types, as follows: simple interest messages *im*, advertisement messages *adv*, data messages *dm*, update messages *um*, update messages about the values of subscribed periodic notification *µm* and emergency alert messages in case of disaster *bm*.

Through message type determination, the forwarding of messages towards the appropriate next node is decided.

Algorithm 2: NDN–NS Message Type Determination

NDN–NS Message Type Determination Processing for NDN–IoT–SC LS (Lab Sensors), CAS (CAmpus Server), Mobile Devices (MD) and LAS (LAb Servers, i.e., Wi-Fi)

Possible Inputs: *task and task sub-type* in the form of *action and subAction* from *Algorithm 1*
Possible output message: *im*, *adv*, *dm*, *um*, *µm*, *bm*.

```
Switch action & subAction
Case: action & retrieve || sense & retrieve
    return im
Case: action & turn
    return um
Case: set-value & float-value
    return um
Case: action & upload
    return adv
Case: action & update
    return µm
Case: action & trigger
    return bm
Case: action & receive
    return dm
    return messageType
```

3.3. Description of NDN–IoT Traffic Types for NDN–IoT–SC

The real power of IoT lies in handling MTC, where machines (devices) can talk to each other. Therefore, we addressed MTC by enabling PHTC along with PLTC. We identified seven different cases and divided them into two different categories of PLTC (Case 1 and case 2), PHTC (case 3 to case 7) & Data Message (case data message). Figure 2 discusses the specifics of various NDN–IoT–SC PLTC and PHTC cases.

3.3.1. Category PLTC

In NDN–IoT–SC, PLTC is dedicated to handle the situations when any consumer (subscriber), from the students, the teaching staff and the admin staff, is interested in fetching some content. PLTC cases can be seen in Figure 2 as case 1 and 2.

In case 1, we deal with a scenario when NDN–IoT–SC consumers (students/ teachers/ admin employees) are interested in retrieving course assignments, quizzes or meeting notifications from any CR (i.e., CAS or LAS). For example, student (or administrative employees) with their student (employee) number of 14F-UET-PhD-CP-43(EMP-ADM-778) as a device (Mobile phone) ID are interested in obtaining the document network assignment 2.xls from the nearby CR.

NDN-NS
CampusName/CampusSubName/CampusLocation/CampusSubLocation/ ContentOriginator-ID/ContentSuperTypeName/ContentSubTypeName/: ContentAttributes:/:/:/ContentFreshness/ContentPopularity/TaskType/TaskSubType/: SHA256(Content Originator):/SHA256(Content Super-Type):/ SHA256(Content Sub-Type):/
Category: Pull (PLTC)
Case 1 Interest Message: Student is interested to retrieve Network, Assignment 2, UET/CPED/Pakistan/Taxila/14F-UET-PhD-CP-43/14CPNetworkAssignemnt,2/.xls/: 17:/0125 :/1330:/01:/0/0/Action/Retrieve/: 968cbab1de...:/9f087fb8a1cc6...:/0ac8b624229a...:/ Case 2 Interest Message: Student is interested in temperature Value (may be of Network Lab) UET/CPED/Pakistan/Taxila/14F-UET-PhD-CP-43/Temperature/NetworkLab/: 17:/1202 :/1100:/01:/0/0/Sense/Retrieve/: 968cbab1de...:/b958ce8b87...:/94341d04c8...:/
Category: Push (PHTC)
Case 3 Interest Message: Admin Employee is interested to turn-off AC1 of Network Lab UET/CPED/Pakistan/Taxila/ EMP-ADM-778/NetworkLab/NL-AC1/: 17:/0327 :/1520:/01:/0/0/Action/TurnOff/: 80ce94778...:/94341d04c8...:/f6f7ae2e808...:/ Case 4 Interest Message: Admin Employee is interested to set temp. of AC1 of Network Lab UET/CPED/Pakistan/Taxila/ EMP-ADM-778/ Temperature /NL-AC1 /: 17:/0815 :/1100:/01:/0/0/setValue/26/: 968cbab1de...:/b958ce8b87...:/f6f7ae2e808...:/ Case 5 Interest Message: Teaching Employee is interested to Upload Network Assignment-2 UET/CPED/Pakistan/Taxila/ EMP-TECH-980/ 14CPNetworkAssignemnt 2/.xls /: 17:/1202 :/1100:/01:/0/0/Action/Transmit-upload/: 108b4fb6...:/9f087fb8a1cc6...:/0ac8b624229a...:/ Case 6 Interest Message: When a sensor NL-AC1 send a periodic notification (say updated Temperature Value) UET/CPED/Pakistan/Taxila/NL-AC1/Temperature /26 C/: 17:/0815 :/1100:/01:/0/0/Action/Update/: 968cbab1de...:/b958ce8b87...:/f6f7ae2e808...:/ Case 7 Interest Message: When a sensor have to send an event-triggered notification (say Temperature value exceeds Threshold) UET/CPED/Pakistan/Taxila/ NL-TEMP1/Fire!/Exit /: 17:/0624 :/1100:/01:/0/0/Action/Trigger/: 108b4fb6...:/9f087fb8a1cc6...:/0ac8b624229a...:/
Data Message
Case Data Message: Student received data for network assignment 2 UET/CPED/Pakistan/Taxila/14F-UET-PhD-CP-43/14CPNetworkAssignemnt-2/.xls/: 17:/0125 :/1330:/01:/0/0/Action/Receive/: 968cbab1de...:/9f087fb8a1cc6...:/0ac8b624229a...:/[Content]

Figure 2. Application of NDN–NFS: NDN-based Naming Scheme(NDN–NS) for IoT Traffic Types for NDN–IoT–SC.

In case 2, we deal with scenario when NDN–IoT–SC consumers (students/ teachers/ admin employees) need (are interested) to sense the temperature values from any nearby lab sensor to perform some other tasks. For example, student (or administrative employees) with their student (employee) number of 14F-UET-PhD-CP-43(EMP-ADM-778) as a device (Mobile phone) ID are interested in knowing the temperature value of Air conditioner 1 of NetworkLab (NL-AC1).

Although administrative employees primarily consume sensor information for various tasks, such as adjusting the air conditioner in a laboratory or classroom, students and teachers may also be interested in knowing the current temperature value. This information can be obtained from a nearby temperature sensor (i.e., LS).

In the category of PLTC, NDN–IoT–SC consumers (or subscribers) can issue Interest messages and content can be obtained through a legacy NDN. However, the naming scheme NDN–NS helps in fetching the content object with some more properties such as freshness, version, and popularity. CR search their CS with the above-mentioned properties to fetch content objects. In case of failure, the message is forwarded to the next neighbor to meet the content properties.

3.3.2. Category PHTC

In NDN–IoT–SC with MTC, PHTC is dedicated to handle scenarios when the producer only needs to push the data without sending any notification to the consumer (either a critical or subscribed update) and the IoT consumer can both push and pull the data towards (or from) the IoT network (push when the consumer wants to perform some action and pull when the consumer wants to fetch some data). PHTC is pushing data from both NDN–IoT–SC consumers and producers to enable MTC. Therefore, PHTC is discussed for a total of five cases, i.e., cases 3 to 5 for different NDN–IoT–SC consumers (teaching employees, administrative employees, students) with different queries, tasks, and interests, and two cases (cases 6 and 7) for NDN–IoT–SC sensors and producers.

We deal with following five scenarios: In case 3: When an administrative employee with his or her employee number of EMP-ADM-778 as ContentOriginatorID (mobile phone) is interested in turning off Air Conditioner 1 of NetworkLab (NL-AC1).

In Case 4: When an administrative employee with his or her employee number of EMP-ADM-778 as ContentOriginatorID (mobile phone) is interested in setting the temperature value of Air Conditioner 1 of NetworkLab (NL-AC1).

In Case 5: When a teaching employee with his or her employee number of EMP-TECH-980 as ContentOriginatorID wants to upload (transmit) a file with the name NetworkAssignment2.xls.

In Case 6: When a sensor with its sensor ID needs to send periodic updates to a subscribed consumer (admin employees) via general interest for further monitoring, and

In Case 7: When a sensor with its sensor ID needs to notify nearby consumers about any disaster event (for example, a fire or earthquake notification).

In the case of PHTC, NDN–IoT–SC, consumers are either interested in performing some action or NDN–IoT–SC producers (lab sensors) need to send a content object as a result of some emergency event like fire or earthquake.

These cases with the mentioned scenarios are listed in Figure 2.

3.3.3. Data Message

In the data message case, we deal with a scenario where any NDN–IoT–SC consumer can receive data against any request or without any request. For example, any student with his or her student-ID as ContentOriginatorID receives content for a network assignment number 2. This case with the mentioned scenario is listed in Figure 2.

3.4. Description of NDN–IoT Traffic Types for NDN–IoT–SC

NDN–NS-based Forwarding Schemes (NDN–NFS) are designed for both the NDN–IoT–SC producer, such as Lab Sensor (LS), and the consumer (CAS, LAS and Mobile Devices (MD)) with the aim of responding to messages in IoT-SC.

NDN–NFS for NDN–IoT–SC producers is described in Algorithm 3 and NDN–NFS for NDN–IoT–SC consumers is presented in Algorithm 5. Both NDN–NFS-enabled NDN–IoT–SC consumer and producer are designed to have the ability to send and receive at any moment, but some constraints are applied before the NDN–IoT–SC consumer can perform an action on the NDN–IoT–SC producer (sensor) or the NDN–IoT–SC consumer can receive data from the NDN–IoT–SC producer without any request. This involves GuardICNg, which authenticates the NDN–IoT node to perform a specific action. NDN (either interest or data) messages are digitally signed by the device itself using the NDN native mechanism.

Algorithm 3: Forwarding Algorithm for NDN–IoT Producer

NDN–NFS Processing for NDN–IoT–SC Producers such as LS (Lab Sensor)

Possible events: Initialize adv, arrival of interest *im* from consumer *C=CAS,LAS,MD* for chunk *c*, arrival of μm to perform action, periodic update ρu at time *t* to *C*, occurrence of critical event ξ near *LS*.
Possible output message: *adv*, *dm*, μm, *bm*

```
case: event
when initialize:
    broadcast adv each non-critical content object CO
when ξ; C:
    construct beacon bm for ξ
    broadcast beacon bm toward CAS,LAS,MDi...j
    create and send chunks c_1, ..., c_N toward CAS,LAS,MDi...j
when message is received from LAS, LSi or C
    check for message type im, dm, adv, μm or bm
    call NDN–NS Message Processing and get messageType
    switch messageType
when im; C; c:
    send c to C in dm
when um; C:
    perform GuardICNg() through content originator ID
    if (matches) then
        perform action and send status of action asked in dm
    else
        drop um
    end
when ρu:t;C: // for case of periodic update ρu at time t to consumer C
    construct μm w.r.t. ρu according to subscribed ID C
    create and send μm towards the subscribed ID C
```

Algorithm 4: NDN–NS GuardICNg

```
NDN–NS GuardICNg
NDN–NS GuardICNg for NDN–IoT–SC Producers such as LS (Lab Sensors) and CR
Possible inputs: contentOriginatorID, shaContentOriginatorID, contentSuperName,
  shaContentSuperName, contentSubName, shaContentSubName .
Possible output message: message authorized or not.
calculate sha256 of contentOriginatorID and compare with shaContentOriginatorID
if matches then
    calculate sha256 of contentSuperName and compare with shaContentSuperName
    calculate sha256 of contentSubName and compare with shaContentSubName
end
Check for contentOriginatorID in authorized list
if matches one then
    authorize message
end
```

Through **GuardICNg**, we maintain the integrity of the content by calculating SHA256 of ContentSuperTypeName and ContentSubTypeName (dissected and saved in the NDN–NS Algorithm 1 as contentSuperName and contentSubName), and then comparing with the attached SHA256s of ContentSuperTypeName and ContentSubTypeName (dissected and saved in NDN–NS Algorithm 1 as shaContentSuperName and shaContentSubName). To maintain authorization and authentication through GuardICNg, we calculate SHA256 of ContentOriginatorID (dissected and saved in the NDN–NS Algorithm 1 as contentOriginatorID) and compare it with the attached SHA256 of ContentOriginatorID (which is dissected and saved in the NDN–NS Algorithm 1 as shaContentOriginatorID). Then it is checked in the authorization list (which we stored in NDN–IoT–SC during the initialization phase). The authorization list is an access control list (ACL) which is stored in NDN–IoT–SC devices (producers, consumers, and content routers) beforehand. If the credentials are present in the stored list, then this message from this device is authorized for further processing. All these facts are mentioned in Algorithm 4.

When any **NDN–IoT–SC producer (e.g., LS)** produces a normal non-critical content object *CO*, then it would be advertised to its nearby consumers through *adv*.

And when NDN–IoT–SC sensors observe some emergency situation ξ, then a beacon message *bm* is constructed for that emergency ξ and this *bm* is broadcast towards CAS, LAS and all nearby MDs, and then chunks are created and sent from c1 to cN towards CAS, LAS and all nearby MDs. These are the two scenarios in which NDN–IoT–SC sensors generate messages on their own.

Below are descriptions of situations when an NDN–IoT–SC sensor receives some message from any consumer C such as LAS or LS. As a first step upon receipt of a message, NDN–NS is called to get the message type. Then, upon message type identification, if the identified message is a simple interest message *im*, then the corresponding content *c* is transmitted in *dm* as a response.

If the identified message is an update message *um*, then it is determined through GuardICNg (with contentOriginatorID) whether that user device is authorized to perform the action or not. If any *um* passes through it, then a mentioned action is performed by NDN–IoT–SC and its status of action is communicated through the data message *dm*; otherwise, this *um* gets dropped. Lastly, when the NDN–IoT–SC sensor needs to update its generated value ρu at time *t* towards its subscriber/consumer *C*, an update message μm is constructed with respect to ρu for the corresponding subscribed consumer *C*. Then, this μm is created and sent towards the corresponding subscribed consumer *C*. Mentioned facts are presented in Algorithm 3.

When any **NDN–IoT–SC consumers (e.g., CR, CAS, LAS)** receive any incoming message, NDN–NS is called to obtain the message type from *im*, *dm*, *adv*, μm or *bm*. If the message type is *adv* for any content *c*, then either *im* is issued for that *c* or a request for a subscribed update after every *t* about any sensor value ρu. This ρu can be any temperature

value from the temperature sensor to take further action such as turning on/off an air conditioner from that Lab or classroom.

Algorithm 5: Forwarding Algorithm for NDN–IoT Consumer and Content Routers

NDN–NFS Processing for NDN–IoT–SC Content Routers (CR) e.g., CAS (CAmpus Server) and LAS (LAb Severs i.e., Wi-Fi)

Possible events: Arrival of *adv*, arrival of interest *dm* from producer *P=LS* for chunk *c*, periodic update *μm* for event *ρu* at time *t* from producer *P=LS*, arrival of *bm*.
Possible output message: *im*, *dm*, *um*, *ρu*, *μm*, *bm*

```
case: event
when message is received from LAS, LSi or C
    check for message type im, dm, adv, μm or bm
    call NDN–NS Message Processing and get messageType
Switch messageType
Case: messageType advertisement adv is received from LSi
if need content then
    send interest message im or for ρu with t towards LSi in im
Case: messageType dm is received from LAS, CAS, LSi
    if entry exists for dm in PIT then
        perform GuardICNg() through content originator ID if matches then
            apply cache mechanism only if necessary
        else
            Apply cache mechanism
            Forward dm towards nearby next appropriate node
        end
    else
        Forward dm towards nearby next appropriate node and drop dm
    end
Case: messageType is im
    check CS if freshness == 0 & popularity ≥ 1
    if found chunk then
        Reply with chunk
    else
        create entry in PIT
        forward im to next neighbor if necessary
    end
Case: messageType is μm
perform GuardICNg() through content originator ID
if matches then
    create temporary Interest
    create entry in PIT
    apply cache mechanism if necessary
else
    Forward μm towards next appropriate node
end
Case: Message type is bm
    create temporary Interest
    create entry in PIT for c_1, ..., c_N
    apply cache mechanism and forward if necessary
```

When the message type is identified as *dm* and received from any *LAS, CAS or LSi*, it is checked in PIT to verify whether or not its entry exists. If any entry is present, then it is confirmed through GuardICNg that it is a response against the requested action and the data will be cached when necessary. If it fails in GuardICNg, then this is simple *dm* against any issued *im* and it is saved when necessary and forwarded to the next nearest node. If it is found that the PIT entry does not exist, then this *dm* is dropped and sent to an appropriate nearby next node.

When the identified message type is *im*, then CS is checked to determine if the more popular content is cached in CS. If the content in found in the CS, then the NDN–IoT–SC

consumer responds with this chunk. Otherwise, a PIT entry is created and *im* is forwarded to the next nearby node. This is a simple case of *im* that NDN handles natively in this manner.

When identified message is μm, a periodic update against subscribed content for ρu value according to some time t from any LS is received. For μm processing, the NDN–IoT–SC consumer first performs GuardICNg to verify that this is from the desired producer for subscribed content. If it is authorized through GuardICNg, then temporary interest is created to receive it. A PIT entry is then created and a cache mechanism is applied if it is required; otherwise, μm is sent towards the next appropriate device. When the identified message is *bm*, some emergency situation occurs such as a fire or an earthquake. The sensor identifies this emergency and communicates this situation through *bm*. NDN–IoT–SC create temporary interest and create the entry in PIT for content chunks *c1,. . . ,cN* and caches these received content data and also transmit them to the next appropriate device. Discussed scenarios are presented in Algorithm 5.

4. Application of Proposed Methodology NDN–NS Naming Instance for NDN–IoT–SC

At stage 1 of NDN–NS, HNC includes the following elements, which are organized in this way:

CampusName/CampusSubName/CampusLocation/CampusSubLocation/ ContentOriginator-ID/ContentSuperTypeName/ContentSubTypeName/:

CampusName is the main campus name (for example, UET). By CampusSubName, the department name is included like CPED. In CampusLocation/CampusSubLocation, the campus country name and campus city name are added (for example, Paksitan/Taxila). In ContentOriginator-ID, the public ID or name of the IoT device is included as EMP-ADM-778 (which means the Employee Device numbered 778 from administration). In ContentSuperTypeName/ContentSubTypeName/, the content name and its type are defined, such as 22CPNetworkAssignment.doc, which is a word document of a network assignment of 22 sessions, which any student need to access.

At stage 2 of NDN–NS, ANC includes following elements:

ContentAttributes:/:/:/ContentFreshness/ContentPopularity/TaskType/ TaskSubType/:

In ContentAttributes we mention general attributes of the content such as year, month, date and time; for instance, 22:/1126:/1750:/ can be referred to as 26 November 2022 at 5:50 pm. In ContentFreshness/ContentPopularity/, both freshness and popularity of the content is included, which can be used further while deciding about caching the content. Values of both attributes range between 0 and 1. A value of 0 means content is fresh while a value of 1 means that the content is old. For NDN–IoT–SC, a timer of 60 s is set. After this, a value of ContentFreshness is set to be 1.The ContentPopularity value is 0 when the content is fresh and it keeps increasing upon every request. TaskType/TaskSubType/: are included to enable MTC. For example, against TaskType/TaskSubType/ values, Action/TurnOf means some NDN–IoT–SC device has to turn off and Sense/Retrieve means some content has to be fetched from the environment. By using the values of this pair, the IoT traffic type is determined. More pair possibilities of values can be seen in Figure 2.

At stage 3 of NDN–NS, FNC includes the following elements:

SHA256(Content Originator):/SHA256(Content Super-Type):/SHA256(ContentSub-Type):/

In FNC, mainly hashes are included for maintaining the integrity of the content. Hashes of ContentOriginator-ID, ContentSuperType and ContentSubType are added. Later, these hashes are included for the authentication and authorization of the devices and relevant NDN–IoT–SC devices.

5. Implementation, Analysis, and Evaluation

5.1. Implementation Details

The naming and forwarding schemes are implemented using ndnSIM [21], which is the NS3-based NDN simulator. The implementation of both NDN–NS and NDN–NFS involve modified TLV files, both interest and data packet files, and modified consumer and producer files. Multiple new fields are added in interest and data packets according to

Figures 1 and 2. Two scenarios are designed to evaluate the proposed forwarding schemes. In the first scenario, native NDN is implemented by using ndnSIM functions. Then, in the second scenario, native NDN is extended by adding the proposed schemes. To evaluate the performance of the proposed schemes, three important parameters, including ISR, delay and number of exchanged messages, are calculated for both scenarios.

5.1.1. Simulation Environment

We installed ndnSIM [21,40] on a system with Linux Ubuntu 14.04 LTS (64-bit) OS. We assigned the processor of an Intel Core i5-2410M CPU 2.3 GHz and 4 GB memory to Ubuntu OS.

5.1.2. Scenario Description

A scenario of IoT-based Smart Campus (IoT-SC) is simulated in ndnSIM and presented in Figure 3. The considered Scenario is the smart campus of the University of Engineering & Technology (UET), Taxila, Pakistan, which consists of 14 departments (LB = Library, ECT = Electronics, EE = Electrical Engg., TRE = Transport Engg., CE = Civil Engg., ME = Mechanical Engg., AD = Admin, CPED = ComPuter Engg. Dept., SE = Software Engg., IE = Industrial Engg., HU = Humanities, TE = Telecom Engg., CS = Computer Sciences, TR = Transport). Every department is assumed to have one Wi-Fi (LAb Server consumer referred as LAS) and four Zigbee (Lab Sensor producer referred as LS) device(s). CAS is for CAmpus Server consumers, and MD is for Mobile Device consumers. We used ndn over layer-2 IEEE 802.11 (WIFI-PHY-STANDARD-80211a) nodes to simulate the scenario of NDN–IoT–SC. The number of nodes ranges from 80 to 130. Out of these total nodes, there are 70 static nodes, and mobile nodes range from 10 to 60. NDN consumer application is loaded on these mobile consumer nodes, and these mobile nodes are added to represent human mobile consumers. Additionally, of the static nodes, 56 are assigned as producers and have the NDN producer application installed on them, while the other 14 are assigned as content routers. For static and mobile nodes, *ConstantVelocityMobilityModel* and *RandomWalk2dMobilityModel* are used respectively. Any time a mobile consumer node makes a content request, the requested content can be obtained from producers with the assistance of neighboring CRs. For 500 s, we simulate this scenario employing both native NDN and NDN–NS. Table 3 presents the simulation's parameters.

Table 3. Simulation Parameters for NDN–NFS.

Parameter	Value
Area	250 m × 250 m
MAC Layer	Wi-Fi IEEE STANDARD 802.11a
Communication Stack	NDN
Number of Nodes	70 Static, 10 to 60 Mobile (Variable)
Number of producer nodes	56
Number of consumer nodes	10 to 60
Number of CR nodes	14
Interest frequency	variable
Number of Transmissions	500
Interest frequency	variable
Simulation Time	500 s
Mobility Model	ConstantVelocityMobilityModel RandomWalk2dMobilityModel

5.2. Qualitative Evaluation: NDN–NS

To evaluate the proposed NDN–NS, we have performed many qualitative comparisons with the relevant ICN–IoT naming schemes to prove the worth of the proposed

work. A summary of the most related works and relevant comparisons is presented in Table 4. NDN–NS supports MTC support, security in terms of content integrity and device authentication and authorization, device naming, attribute-based searching, while other presented schemes provide some of these features.

5.3. Quantitative Evaluation: Results and Discussions

The performance of the proposed scheme NDN–NFS is measured against the legacy NDN (the one which is provided by ndnSIM) in terms of ISR, delay and overhead through a number of transmissions (or a number of exchanged messages). These parameters are taken from [15], which are mentioned to measure ICN–IoT naming. The performance is measured and presented for both cases of PLTC and PHTC. We measure performance for every action mentioned in Figure 2.

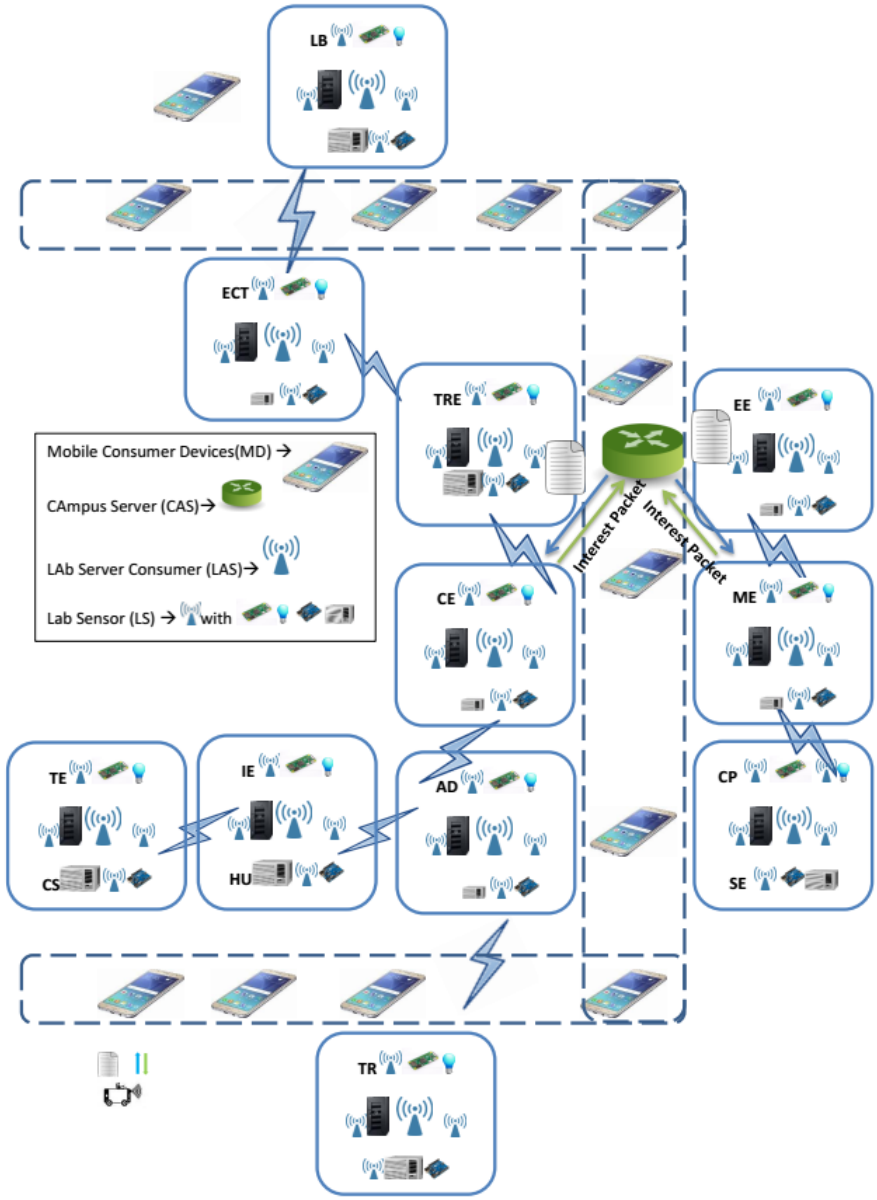

Figure 3. Simulated Smart Campus: Considered Scenario (UET = University of Engineering & Technology) consists of 14 departments, which is assumed further to have one Wi-Fi (LAb Server consumer as LAS) and four Zigbee (Lab Sensor(s) producer as LS) device(s). CAS is for the CAmpus Server consumer and MD is for Mobile Devices consumers.

Table 4. Comparison of NDN–NS with ICN–IoT Naming Schemes.

Application	Naming Scheme Format (Example)	Traffic Model Support		Security	Device Info	MTC Support	Attributes
		Push	Pull				
ICN-based naming for Smart Home [32]	/homeID/task/type/subtype/location/	✓	✓	✗	✗	✓	✗
CCN-based Naming for VANETS [34]	VHN://CTRY/STT/CTY/OID/VRP/TYPE/Sub-Type/SP/TEM/Attrs./hash(Base64)	✗	✓	✓Integrity only	✓	✗	✓
NDN-based Naming for smart building [41]	/ndn/ucla.edu/bms/building/melnitz/studio/1/data/panel/J/voltage/	✗	✓	✗	✗	✗	✗
NDN-based Naming for under water [37]	UNDN://spatial/(22.228,288.30,96)/temporal/2622487800 2682489000/type/Salinity/pref/all	✗	✓	✗	✗	✗	✓
ICN-based naming for IoT [38]	/Bit:edu/CentralBuilding/Floor10/Room33	✗	✓	✗	✗	✗	✗
CCN-based naming for IoT-SC [20]	/domain-name/location/task: device-name| data /uettaxila/CP/DC/action:on/0x9F091C54	✓	✓	✓Integrity only	✓	✓	✗
Proposed NDN–NS for IoT-SC	domain-name/location/deviceInfo/contentName/:contentAttributes/taskType/:hashedData:/	✓	✓	✓Authorization, Authentication and Integrity	✓	✓	✓

ISR is the percentage of ratio of the number of satisfied interests to the number of interests issued by a consumer to obtain content. We measure performance for every action mentioned in Figure 2.

In **case of PLTC**, ISR results exhibit that both legacy NDN and NDN–NFS produce almost similar results for both case 1 and case 2 and this can be seen in Figure 4.

The **reason** behind this that NDN natively supports such cases when a consumer is interested in fetching data, for which it can issue an interest message.

In **case of PHTC**, ISR results can be seen in Figure 5. For both case 3 and 4, the legacy NDN exhibited approximately 40% less ISR than that of NDN–NFS. The reason behind this is that under a fixed simulation time, NDN–NFS perform the actions of case 3 and 4 in less time as compared to the legacy NDN. For case 5, both NDN and NDN–NFS exhibit almost the same ISR because both implement case 5 in a similar fashion. For both cases 6 and 7, the legacy NDN gives zero output, as it does not support these types of cases such as pushing updates and critical information towards consumers. In contrast, NDN–NFS gives 75% to 90% ISR because it supports such cases.

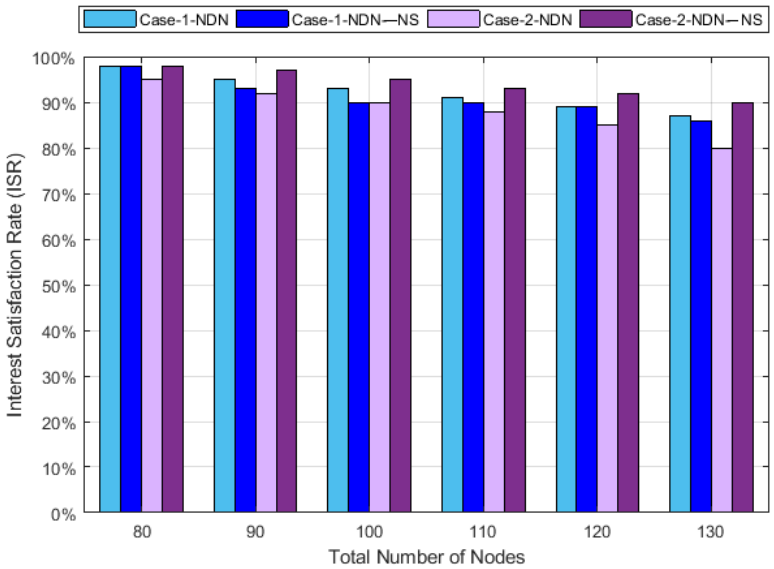

Figure 4. ISR for Case: NDN–NFS-IoT-SC-PLTC.

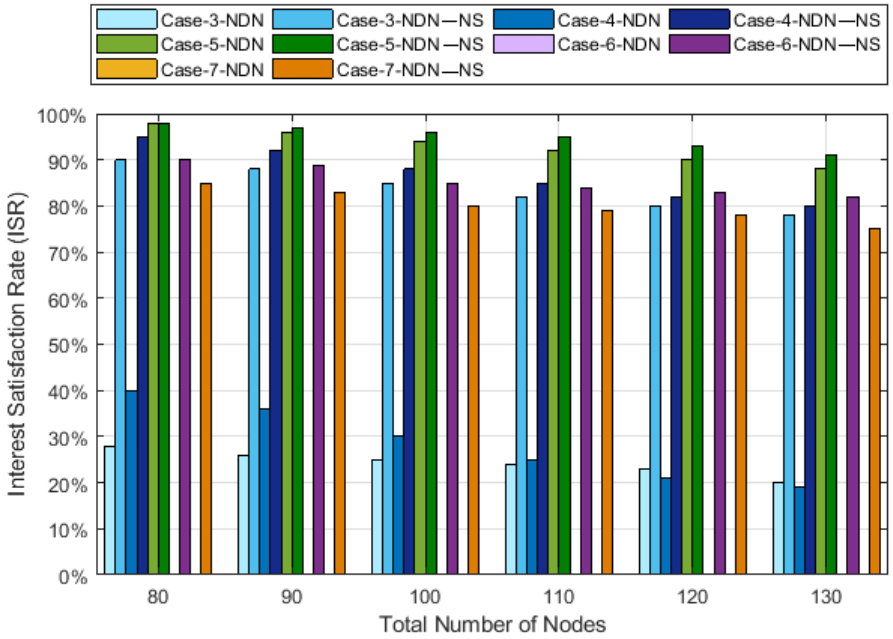

Figure 5. ISR for Case: NDN–NFS-IoT-SC-PHTC.

5.3.1. Interest Satisfaction Rate

In a nutshell, NDN–NFS outperforms the legacy NDN and provides support for every action, which any smart campus user requires. Moreover, it can be seen that ISR decreases (for both PLTC and PHTC cases) when the number of nodes increases. This is due to an increasing number of transmissions, with an increase in the number of nodes, but with a fixed simulation time.

5.3.2. Delay

Delay is the time difference between when a request is sent towards the producer (or consumer) and *response* is received by the consumer (or producer). We measure performance for every action mentioned in Figure 2.

The delay result **for case of PLTC** is visualized in Figure 6. It can be seen and inferred from Figure 6 that for the case of PLTC, both NDN and NDN–NFS exhibit almost the same behavior; however, NDN–NFS gives a little less delay as compared to the legacy NDN. Results **for case PHTC** are presented in Figure 7 for case 3 to case 7. It can be seen that the legacy NDN exhibits more delay than NDN–NFS for case 3, 4, 6 and 7. For both PLTC and PHTC (except case 5), NDN–NFS incurs less delay than NDN. This is due to the reduction in the time of very specific processing and responding. However, for case 5, NDN incurs less delay as compared to NDN–NFS. This is because the legacy NDN achieves case 5 by just sending advertisement messages.

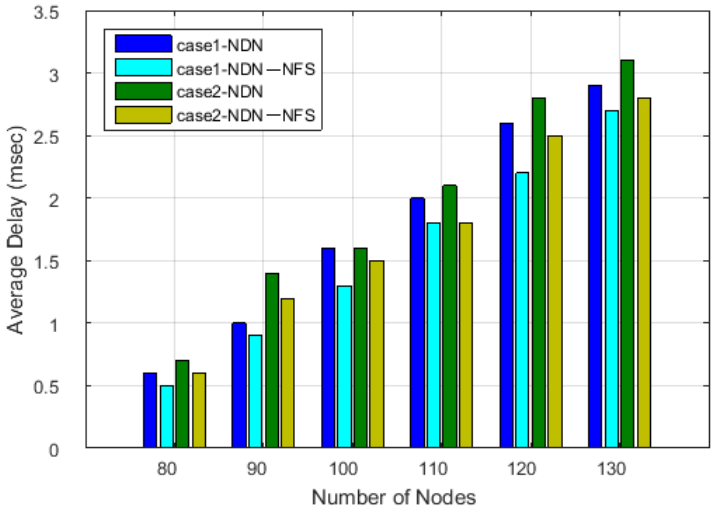

Figure 6. Delay for Case: NDN–NFS-IoT-SC-PLTC.

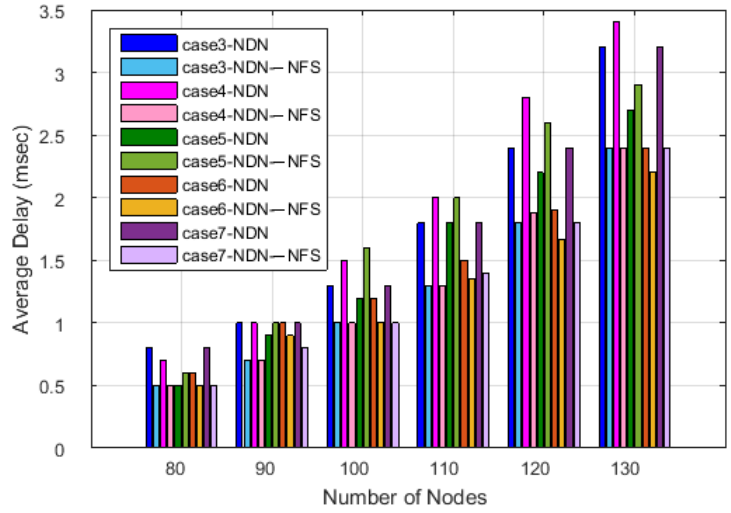

Figure 7. Delay for Case: NDN–NFS-IoT-SC-PHTC.

5.3.3. Number of Exchanged Messages

We measure performance for every action mentioned in the PHTC section in Figure 2. Regarding PHTC, it can be visualized from Figures 8 and 9 that the average number of transmissions of case 3, 4, 6 and 7 in NDN–NFS is less than the legacy NDN. This is because the legacy NDN does not support situations such as case 3, 4, 6 and 7 and thus incurs a higher number of transmissions. Moreover, for case 5, NDN performs better than NDN–NFS. This is due to the fact that NDN simply sends advertisement messages to perform case 5. In Figure 9, it can be seen that with an increasing number of nodes, the average number of transmission is also rising. This is due to the fact that with a higher number of nodes, the number of transmissions will also increase.

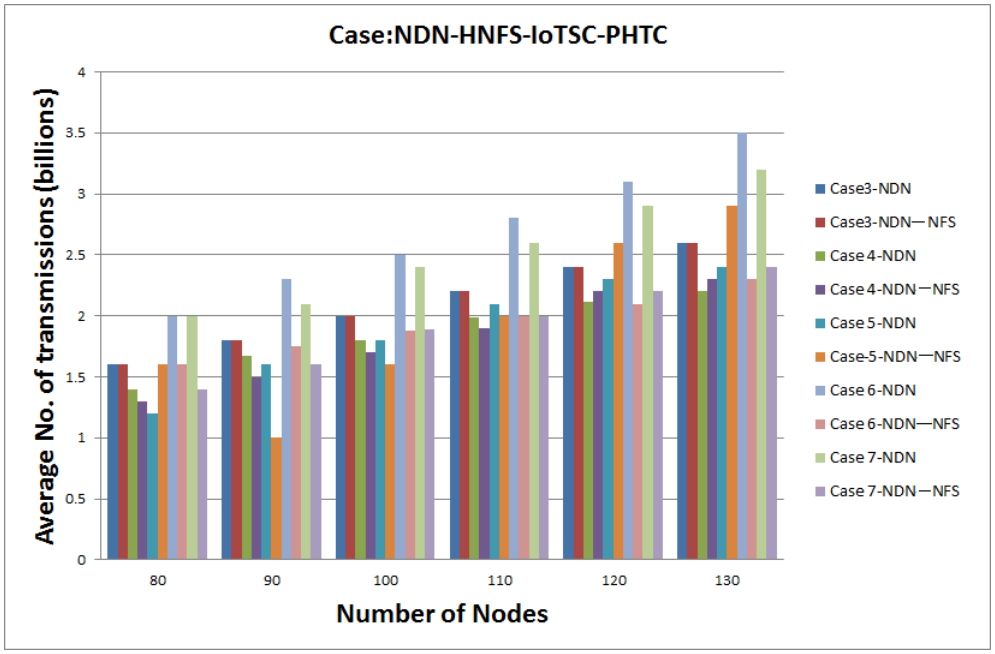

Figure 8. Overhead Transmissions Comparison for Case: NDN–NFS-IoT-SC-PHTC.

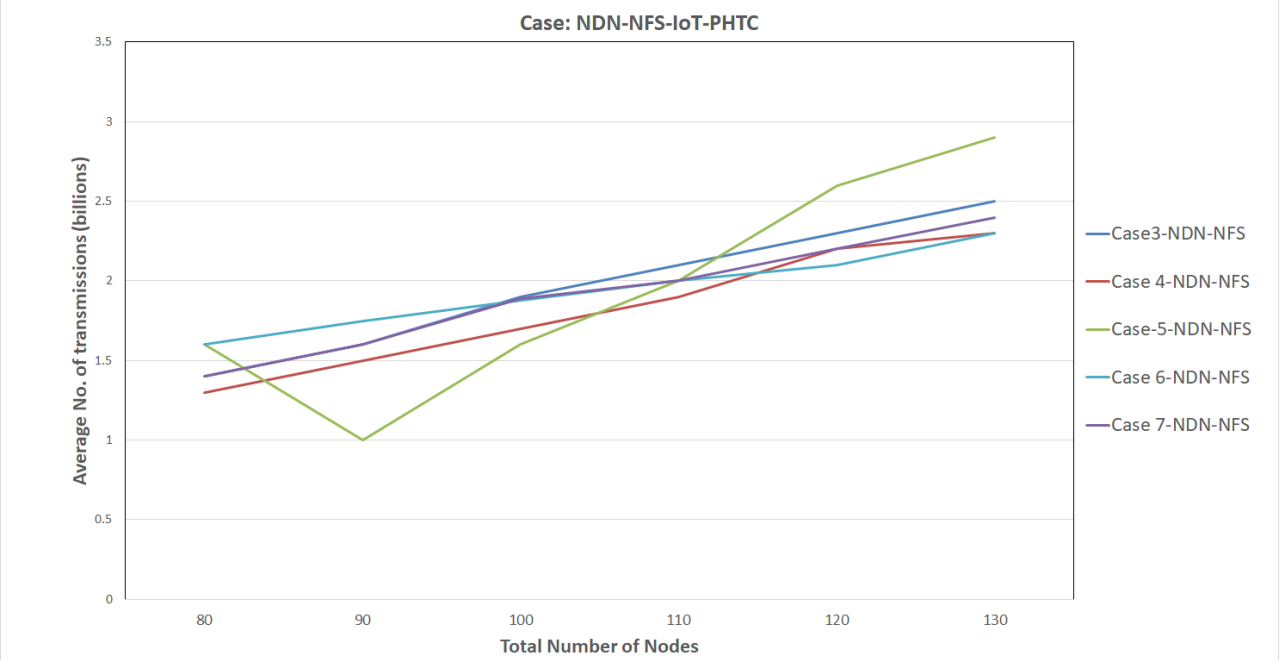

Figure 9. Overhead Transmissions for Case: NDN–NFS-IoT-SC-PHTC.

6. Conclusions and Future Work

For the NDN–IoT-based Smart Campus, NDN–IoT–SC, we have presented a novel NDN-based naming scheme called NDN–NS and NDN–NS-based forwarding method named NHN-NFS in this work. We categorized IoT traffic types based on NDN–NS into seven IoT case types and two communication models, PLTC and PHTC, to address pull- and push-type communication, respectively. Almost any action that may be performed in an IoT application is included in the seven scenarios for IoT traffic types, such as sending an update message, a critical message, performing action on a device, and changing the value of a parameter that may or may not be sensed by the device. Every NDN–IoT–SC node runs NDN–NS, which in turn invokes the message type determination algorithm. This method is designed to distinguish between the message types *µm*, *bm*, *im*, *dm*, and

adv. Then, for NDN–IoT–SC producers, consumers, and content router nodes, we design forwarding NDN–NFS algorithms based on NDN–NS. GuardICNg is used to forward messages in order to provide security in terms of authorization and authentication. We used ndnSIM to simulate the UET Taxila smart campus and compared NDN–NFS to the traditional NDN. For NDN–IoT–SC, we discovered that NDN–NFS performed better in terms of ISR, latency, and transmission count.

Limitations of the current work include the need to validate and verify the designed naming and forwarding algorithms for real NDN–IoT-based devices for an IoT-based Smart Campus. There are real NDN devices that have been developed for NDN experimentation and research. Here are some examples:

NDN Forwarding Daemon (NFD): NFD is an open-source software router that implements the NDN forwarding plane. It can be run on commodity hardware or virtual machines and is the most widely used NDN node implementation [42].

NDN Testbed Node (NTN): NTN is a hardware platform that was designed specifically for NDN experimentation. It is based on the Raspberry Pi and includes a custom NDN software stack that allows for easy deployment and experimentation [43].

As Future work,

- We look forward to designing a caching mechanism on the basis of the proposed novel naming scheme. Furthermore,
- We look forward to reducing the length of content names in NDN–NS because, so far, our major aim was to cover all the aspects of IoT to design complete and comprehensive naming and forwarding.
- We also look forward to verifying and validating current work against IoT attacks and proposing any NDN-based mitigation strategies.
- We aim to emulate the proposed schemes on hardware, including real NDN–IoT–SC devices such as mobile phones, campus servers, lab servers, and lab sensors.

Author Contributions: Conceptualization, S.A. and J.L.; Methodology, S.A.; Software, S.A.; Validation, S.A. and J.L.; Formal Analysis, J.L.; Investigation, S.A., J.L. and M.F.M.; Resources, M.A.A.; Data Curation, S.A., M.F.M. and A.H.; Writing—Original Draft Preparation, S.A.; Writing—Review & Editing, J.L., M.A.A., M.F.M. and A.H.; Visualization, J.L., M.F.M. and A.H.; Supervision, M.A.A.; Project Administration, M.A.A.; Funding Acquisition, M.A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research work was supported by UET Taxila, Pakistan and the publication was sponsored by Whitecliffe Enterprise Limited through the Contestable Research fund. This work was supported by the Computer Engineering Department of the University of Engineering and Technology (UET), Taxila, Pakistan under ASR&TD, a Full-time research scholarship.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: This work is extracted from my Ph.D. dissertation and it has been submitted to Higher Education Commission for the award of Ph.D. Degree. It can be accessed through the link https://tinyurl.com/4r5a4wcn (accessed on 1 October 2022) [44]. Moreover, the authors certify that this work has not been published anywhere; however, it has been made available on the Higher Education Commission website for the requirements of the Ph.D degree. Therefore, it is not published in any scientific journal or conference. The authors would like to thank the editor and reviewers for their valuable comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Commun. Surv. Tutorials* **2015**, *17*, 2347–2376. [CrossRef]
2. Lindgren, A.; Abdesslem, F.B.; Ahlgren, B.; Schelén, O.; Malik, A.M. Design choices for the IoT in information-centric networks. In Proceedings of the 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NJ, USA, 9–12 January 2016; IEEE: Piscataway, NV, USA, 2016; pp. 882–888.
3. Nour, B.; Sharif, K.; Li, F.; Moungla, H. A distributed ICN-based IoT network architecture: An ambient assisted living application case study. In Proceedings of the GLOBECOM 2017—2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6.
4. Meddeb, M.; Dhraief, A.; Belghith, A.; Monteil, T.; Drira, K.; Gannouni, S. AFIRM: Adaptive forwarding based link recovery for mobility support in NDN/IoT networks. *Future Gener. Comput. Syst.* **2018**, *87*, 351–363. [CrossRef]
5. Nour, B.; Sharif, K.; Li, F.; Wang, Y. Security and Privacy Challenges in Information-Centric Wireless Internet of Things Networks. *IEEE Secur. Priv.* **2020**, *18*, 35–45. [CrossRef]
6. Nour, B.; Sharif, K.; Li, F.; Moungla, H.; Kamal, A.E.; Afifi, H. NCP: A near ICN cache placement scheme for IoT-based traffic class. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6.
7. Amadeo, M.; Campolo, C.; Iera, A.; Molinaro, A.; Ruggeri, G. Client Discovery and Data Exchange in Edge-based Federated Learning via Named Data Networking. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 2990–2995. [CrossRef]
8. Amadeo, M.; Campolo, C.; Ruggeri, G.; Molinaro, A. Popularity-Aware Closeness Based Caching in NDN Edge Networks. *Sensors* **2022**, *22*, 3460. [CrossRef] [PubMed]
9. Amadeo, M.; Campolo, C.; Ruggeri, G.; Molinaro, A. Beyond edge caching: Freshness and popularity aware iot data caching via ndn at internet-scale. *IEEE Trans. Green Commun. Netw.* **2021**, *6*, 352–364. [CrossRef]
10. Yao, L.; Li, J.; Deng, J.; Wu, G. Detection of Cache Pollution Attack Based on Federated Learning in Ultra-Dense Network. *Comput. Secur.* **2023**, *124*, 102965. [CrossRef]
11. Ould Khaoua, A.S.; Boukra, A.; Bey, F. Probabilistic Forwarding in Named Data Networks for Internet of Things. In Proceedings of the International Symposium on Modelling and Implementation of Complex Systems, Mostaganem, Algeria, 30–31 October 2022; Springer: Berlin, Germany, 2023; pp. 17–30.
12. Xylomenos, G.; Ververidis, N.C.; Siris, A.V.; Fotiou, N.; Christos, T.; Xenofon, V.; Katsaros, V.K.; Polyzos, C.G. A Survey of Information-Centric Networking Research. *IEEE Commun. Surv. Tutorials* **2014**, *16*, 1024–1049. [CrossRef]
13. Majeed, M.F.; Ahmed, S.H.; Dailey, M.N. Enabling push-based critical data forwarding in vehicular named data networks. *IEEE Commun. Lett.* **2016**, *21*, 873–876. [CrossRef]
14. Betz, U.A.; Betz, F.; Kim, R.; Monks, B.; Phillips, F. Surveying the future of science, technology and business—A 35 year perspective. *Technol. Forecast. Soc. Change* **2019**, *144*, 137–147. [CrossRef]
15. Arshad, S.; Azam, M.A.; Rehmani, M.H.; Loo, J. Recent advances in information-centric networking-based Internet of Things (ICN–IoT). *IEEE Internet Things J.* **2018**, *6*, 2128–2158. [CrossRef]
16. Zhang, Y.; Raychadhuri, D.; Grieco, L.; Baccelli, E.; Burke, J.; Wang, G. *ICN Based Architecture for IoT—Requirements and Challenges Draft-Zhang-IoT-ICN-Challenges-02*; ICN Research Group: Newport Beach, CA, USA, 2016.
17. Information-Centric Networking Research Group. Information-Centric Networking (icnrg). 2012. Available online: https://irtf.org/icnrg (accessed on 15 January 2021) .
18. Amadeo, M.; Campolo, C.; Quevedo, J.; Corujo, D.; Molinaro, A.; Iera, A.; Aguiar, R.L.; Vasilakos, A.V. Information-centric networking for the internet of things: challenges and opportunities. *IEEE Netw.* **2016**, *30*, 92–100. [CrossRef]
19. Zhang, H.; Quan, W.; Guan, J.; Xu, C.; Song, F. *Uniform Information with a Hybrid Naming (hn) Scheme, draft-zhang-icnrg-hn-03.txt*; Technical Report, ICNRG Internet Draft; 2016. Available online: https://datatracker.ietf.org/doc/draft-zhang-icnrg-hn/ (accessed on 14 November 2022).
20. Arshad, S.; Shahzaad, B.; Azam, M.A.; Loo, J.; Ahmed, S.H.; Aslam, S. Hierarchical and flat-based hybrid naming scheme in content-centric networks of things. *IEEE Internet Things J.* **2018**, *5*, 1070–1080. [CrossRef]
21. Mastorakis, S.; Afanasyev, A.; Zhang, L. On the Evolution of NdnSIM: An Open-Source Simulator for NDN Experimentation. *SIGCOMM Comput. Commun. Rev.* **2017**, *47*, 19–33. [CrossRef]
22. Rezaeifar, Z.; Wang, J.; Oh, H.; Lee, S.B.; Hur, J. A reliable adaptive forwarding approach in named data networking. *Future Gener. Comput. Syst.* **2019**, *96*, 538–551. [CrossRef]
23. Qiao, X.; Wang, H.; Ren, P.; Tu, Y.; Nan, G.; Chen, J.; Blake, M.B. Interest packets scheduling and size-based flow control mechanism for content-centric networking web servers. *Future Gener. Comput. Syst.* **2020**, *107*, 564–577. [CrossRef]
24. Nour, B.; Sharif, K.; Li, F.; Yang, S.; Moungla, H.; Wang, Y. ICN Publisher-Subscriber Models: Challenges and Group-based Communication. *IEEE Netw.* **2019**, *33*, 156–163. [CrossRef]
25. Adhatarao, S.S.; Chen, J.; Arumaithurai, M.; Fu, X.; Ramakrishnan, K.K. Comparison of naming schema in ICN. In Proceedings of the 2016 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN), Rome, Italy, 13–15 June 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–6.

26. Khelifi, H.; Luo, S.; Nourz, B.; Mounglax, H. A Name-to-Hash Encoding scheme for vehicular named data networks. In Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 24–28 June 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 603–608.
27. Saxena, D.; Raychoudhury, V. N-FIB: Scalable, memory efficient name-based forwarding. *J. Netw. Comput. Appl.* **2016**, *76*, 101–109. [CrossRef]
28. Muralidharan, S.; Roy, A.; Saxena, N. MDP-IoT: MDP based interest forwarding for heterogeneous traffic in IoT-NDN environment. *Future Gener. Comput. Syst.* **2018**, *79*, 892–908. [CrossRef]
29. Baugher, M.; Davie, B.; Narayanan, A.; Oran, D. Self-verifying names for read-only named data. In Proceedings of the 2012 IEEE INFOCOM Workshops, Orlando, FL, USA, 25–30 March 2012; IEEE: Piscataway, NJ, USA, 2012, pp. 274–279.
30. Ascigil, O.; Reñé, S.; Xylomenos, G.; Psaras, I.; Pavlou, G. A Keyword-Based ICN–IoT Platform. In Proceedings of the 4th ACM Conference on Information-Centric Networking—ICN'17, New York, NY, USA, 26–28 September 2017; pp. 22–28. [CrossRef]
31. Ahmed, S.H.; Kim, D. Named data networking-based smart home. *ICT Express* **2016**, *2*, 130–134. [CrossRef]
32. Marica, A.; Claudia, C.; Antonio, I.; Antonella, M. Information Centric Networking in IoT scenarios: The case of a smart home. In Proceedings of the 2015 IEEE International Conference on Communications (ICC), London, UK, 8–12 June 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 648–653.
33. Upeka, D.S.; Adisorn, L.; Arjuna, S.; Kanchana, K. Named Data Networking Based Smart Home Lighting. In Proceedings of the 2016 ACM SIGCOMM Conference, Florianópolis, Brazil, 22–26 August 2016; ACM: New York, NY, USA, 2016; pp. 573–574.
34. Bouk, S.H.; Ahmed, S.H.; Kim, D. Hierarchical and hash based naming with Compact Trie name management scheme for Vehicular Content Centric Networks. *Comput. Commun.* **2015**, *71*, 73–83. [CrossRef]
35. Khelifi, H.; Luo, S.; Nour, B.; Moungla, H.; Faheem, Y.; Hussain, R.; Ksentini, A. Named Data Networking in Vehicular Ad Hoc Networks: State-of-the-Art and Challenges. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 320–351. [CrossRef]
36. Upeka, D.S.; Adisorn, L.; Arjuna, S.; Carlos, M.J.; Kanchana, K. Implementation and evaluation of an information centric-based smart lighting controller. In Proceedings of the 12th Asian Internet Engineering Conference, Bangkok, Thailand, 30 November–2 December 2016; ACM: New York, NY, USA, 2016; pp. 1–8.
37. Bouk, S.H.; Ahmed, S.H.; Kim, D. NDN goes deep: Foreseeing the underwater named data networks. In Proceedings of the Symposium on Applied Computing, Marrakesh, Morocco, 4–6 April 2017; ACM: New York, NY, USA, 2017; pp. 642–646.
38. Nour, B.; Sharif, K.; Li, F.; Moungla, H.; Liu, Y. M2HAV: A Standardized ICN Naming Scheme for Wireless Devices in Internet of Things. In Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications, Guilin, China, 19–21 June 2017; Springer: Berlin, Germany, 2017; pp. 289–301.
39. Arshad, S.; Azam, M.A.; Loo, J. Towards Information-Centric Networking (ICN) Naming for Internet of Things (IoT): The Case of Smart Campus. In Proceedings of the International Conference on Future Networks and Distributed Systems, ICFNDS'17, Cambridge, UK, 19–20 July 2017; pp. 1–6. [CrossRef]
40. Mastorakis, S.; Afanasyev, A.; Moiseenko, I.; Zhang, L. *ndnSIM 2: An Updated NDN Simulator for NS-3*; Technical Report NDN-0028, Revision 2; NDN: Rapid City, SD, USA, 2016.
41. Shang, W.; Ding, Q.; Marianantoni, A.; Burke, J.; Zhang, L. Securing building management systems using named data networking. *IEEE Netw.* **2014**, *28*, 50–56. [CrossRef]
42. NFD: Named Data Networking Forwarding Daemon. 2023. Available online: https://docs.named-data.net/NFD/current/ (accessed on 15 February 2021).
43. NDN Testbed. 2023. Available online: https://named-data.net/ndn-testbed/ (accessed on 12 January 2023).
44. Information-Centric Networking (ICN) for Internet of Things (IoT). Available online: http://prr.hec.gov.pk/jspui/bitstream/123456789/11755/1/Sobia/20Arshad_Computer20/Engg_2019_UET/28T/29_PRR.pdf (accessed on 15 January 2022).