





Article

Leveraging Software-Defined Networking for a QoS-Aware Mobility Architecture for Named Data Networking

Muhammad Adnan ¹, Jihad Ali ² , Manel Ayadi ^{3,*} , Hela Elmannai ⁴ , Latifa Almuqren ³ and Rashid Amin ^{5,6} 

¹ Department of Electronics Engineering, Kookmin University, Seoul 02707, Republic of Korea; adnan25408@gmail.com

² Department of AI Convergence Network, Ajou University, Suwon 16499, Republic of Korea; jehadali@ajou.ac.kr

³ Department of Information Systems, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia; laalmuqren@pnu.edu.sa

⁴ Department of Information Technology, College of Computer and Information Sciences Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia; hselmannai@pnu.edu.sa

⁵ Department of Computer Science, Taxila University, Taxila 47050, Pakistan; rashid4nw@gmail.com

⁶ Department of Computer Sciences, Faculty of Computing and Information Technology, University of Chakwal, Chakwal 48800, Pakistan

* Correspondence: mfayadi@pnu.edu.sa

Abstract: The internet's future architecture, known as Named Data Networking (NDN), is a creative way to offer content-based services. NDN is more appropriate for content distribution because of its special characteristics, such as naming conventions for packets and methods for in-network caching. Mobility is one of the main study areas for this innovative internet architecture. The software-defined networking (SDN) method, which is employed to provide mobility management in NDN, is one of the feasible strategies. Decoupling the network control plane from the data plane creates an improved programmable platform and makes it possible for outside applications to specify how a network behaves. The SDN is a straightforward and scalable network due to its key characteristics, including programmability, flexibility, and decentralized control. To address the problem of consumer mobility, we proposed an efficient SDPCACM (software-defined proactive caching architecture for consumer mobility) in NDN that extends the SDN model to allow mobility control for the NDN architecture (NDNA), through which the MC (mobile consumer) receives the data proactively after handover while the MC is moving. When an MC is watching a real-time video in a state of mobility and changing their position from one attachment point to another, the controllers in the SDN preserve the network layout and topology as well as link metrics to transfer updated routes with the occurrence of the handoff or handover scenario, and through the proactive caching mechanism, the previous access router proactively sends the desired packets to the new connected routers. Furthermore, the intra-domain and inter-domain handover processing situations in the SDPCACM for NDNA are described here in detail. Moreover, we conduct a simulation of the proposed SDPCACM for NDN that offers an illustrative methodology and parameter configuration for virtual machines (VMs), OpenFlow switches, and an ODL controller. The simulation result demonstrates that the proposed scheme has significant improvements in terms of CPU usage, reduced delay time, jitter, throughput, and packet loss ratio.

Keywords: software-defined networking; OpenFlow; Named Data Networking; mobility management; handover handling



Citation: Adnan, M.; Ali, J.; Ayadi, M.; Elmannai, H.; Almuqren, L.; Amin, R. Leveraging Software-Defined Networking for a QoS-Aware Mobility Architecture for Named Data Networking. *Electronics* **2023**, *12*, 1914. <https://doi.org/10.3390/electronics12081914>

Academic Editors: Lionel Nkenyereye and Christos J. Bouras

Received: 23 March 2023

Revised: 30 March 2023

Accepted: 14 April 2023

Published: 18 April 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The present network infrastructure is undergoing effective content delivery due to IP-dependent routing, as well as host-to-host exchanges of data in practice and location depending on IP. This innovative tool for the Internet of Things (IoT) addresses

these issues. The whole internet architecture has become more sophisticated as a result. Information-centric networking (ICN), a potential future internet design, has been developed to address these challenges. The fundamental principle of the ICN design is to use location-independent naming to separate data (service) from the physical devices storing it [1–3]. To improve content distribution for users, ICN offers the chance to switch from IP-based routing to name-based routing, which is independent of location and has a cache for storing material.

ICN seeks to transform the current Internet model from a challenging one to a simple and common one. The distinguished node is certainly not the crucial networking component (servers, switches, terminals). The network conducts all of its operations in accordance with ICN's defined content objectives. In an ICN design, the router would search for a particular piece of content when a user indicated an interest in it. The user would receive the content if it were discovered. As a result, ICN has a variety of characteristics, including self-secured content, local multicast, name-based routing, and location-independent naming [4].

Content-centric networking (CCN), data-oriented network architecture (DONA), along with network of information (NetInf), as well as the publish-subscribe internet routing paradigm (PSIRP), and NDN are just a few of the ICN architectures that have been presented [5–8]. In order to distribute content efficiently, for example, video streaming, audio streaming, and P2P sharing of data, NDN is considered a future design intended for the internet. It has come to be regarded as an alternative structure for the standard IP-related networking, employing name for routing as a substitute for IP. The data packet contains information like a digital signature, signed documents, and so on. The interest packet also contains information like the name of the seeking material. NDN promotes customer mobility as well. Consumer mobility is experiencing a number of problems, including connection initiation, data packet forwarding, and response delay.

The main goal of ICN is to address IP network restrictions through name-based routing and caching within the network in order to reduce bandwidth use and provide location-independent access to content through several paths as well as mobility management. The fundamental tenet of the present internet architecture is that information is transferred from one point to another and between clients and servers throughout communications. Via specialized routers with caches, which are made up of three tables called the pending interest table (PIT), forwarding information base (FIB), and content store (CS), across which data may be retrieved, NDN shifts the focus from locations (destinations and origins) to the information in itself. As a result, the network should not suffer as of connecting with a server through requesting information in itself [9].

The following are some mobility-related difficulties that the author mentioned in [10]: Mobile devices having mobility features may move between different points of attachment (PoA) with the least amount of handover latency and without affecting the transmission of content, much like in the ICN design. A PoA allows for mobile nodes in order to join a network. Moreover, mobility was therefore split into mobility regarding producers and mobility for consumers. The author claims in [11] that consumer mobility naturally complements the consumer-driven orientation of the NDN. As soon as the mobile user changes to a new PoA, then the I_ packets will be present to perform transmission to keep on typically or to restart after handoff. In contrast to IP designs, the creators of portable content face little difficulties [12].

In order to maintain the availability and location of their content for consumers and center routers, content producers must be able to move their material across locations as quickly as feasible. This is referred to as producer mobility. As a result, the author of [13] concluded that NDN does not aid producer mobility from a unique perspective on mobility assistance. NDN is impacted by a scaling issue with routing table size. Additionally, when providers relocate to a new location, the name system presents major scalability issues [14]. Significant handoff delay and excessive I_packet losses while broadcasting to a producer's prior location are other problems that need to be rectified [15,16]. The I_packets continue to

follow the prefix trail in the FIB whenever a producer switches locations, thus failing to get to the producer and being dropped. Due to the indirect points utilized to allow producer mobility, the data channel is likewise constrained and difficult. As a result, there is a long handoff interval as a result of conditions.

Problem Formulation and Our Proposed Solution

The authors in [17] suggested the mobility link service (MLS), which runs in the NDN interface and is in charge of managing a connection for a transaction. Due to the regularly shifting locations of users, the fundamental benefit of MLS is reusing the existing connection regarding a transaction as opposed to starting a new connection. The current connection is damaged if a consumer changes its position away from the adjacent node, and the data cannot be entirely received due to handover. Therefore, in this instance, the connection is reinstated rather than established anew, and fresh content requests are made to the original, but new, NAR. As a result, the data for the remaining material are delivered by the prior NAR to the NDN producer, who then forwards it toward the new NAR, which has been requested by the user already. The following, Figure 1, shows the existing [17] problem statement and provides our proposed solution, SDPCACM. In the existing work of [17], the NDN producer can send a video containing packets A, B, C, and D to NAR-1, as shown in Figure 1. The NAR-1 sends these video streaming packets to the MC. Let us consider that the MC is watching a real-time video in a case of mobility, and this real-time video is contained in packet A, which is accessed by the MC through AP-1. When an MC moves to the next location and is connected to AP-2, the remaining packet B is provided to the MC through AP-2. Now the MC has changed their position and connected to the next AP, thus moving to AP-4 from AP-3. In this instance, the connection is reinstated rather than established, and fresh content requests are made to the original new NAR-2. As a result, the data for the remaining material, i.e., packets C and D, was delivered by the prior NAR-1 to the NDN producer, who then forwarded it to the new NAR-2, which has been requested by the user already, as shown in Figure 1.

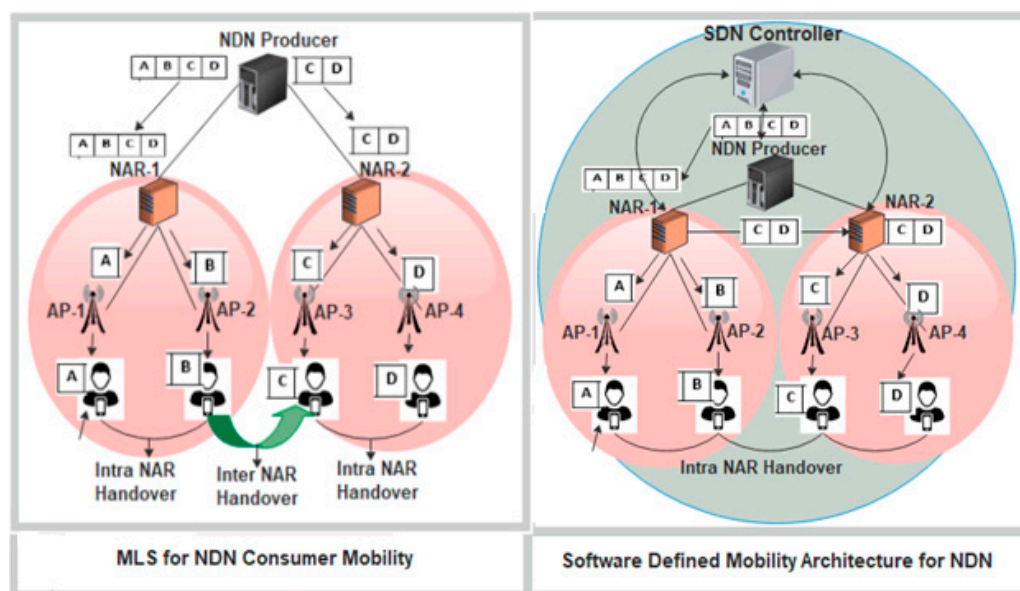


Figure 1. Problem statement and proposed solution.

The distinctive qualities of SDN, including its adaptability and programmability, along with dynamic resource allocation, can aid in meeting the performance and control needs of NDN. Through NDN, we imply that the data has several places, and the SDN chooses which site the data is transmitted through to fulfill the user request and preserve the flow of information as well as the network's stability. To put it another way, the SDN

can find the paths for data transfer if the data is already present in the router cache. In addition to enhancing productivity, scalability, flexibility, and security, SDN routing may help everyone make a smooth transition to a new network design [9]. To cope with the above-discussed issues faced in NDN consumer mobility and to tackle the advantages of SDN, we move toward the design of SDPCACM as shown in Figure 1, in which the SDN controller preserves the network layout and topology as well as link metrics to transfer updated routes with the occurrence of the handoff or handover scenario. Moreover, the proactive caching mechanism in SDPCACM is used, in which the consumer's data packet is proactively disseminated to the following nearby connected routers by the previous access router. The term "proactive caching" refers to a group of techniques used to act before a consumer issues a mobility-related interest packet or changes their current position. For SDN-based NDN consumer mobility concerning a real-time video, we employ a proactive caching approach that allows us to get the data straight from the newly connected attachment point (AP) after handover. In this way, the packets are delivered to the user without any handover-corresponding delay, which improves the packet delivery ratio by reducing handover delay time, jitter, improving CPU utilization, and throughput, as well as improving packet loss ratio.

The following are the paper's key contributions:

1. The design of SDPCACM (software-defined proactive caching architecture for consumer mobility) in NDN that extends the SDN model to allow mobility control for the NDN architecture (NDNA), through which the MC (mobile consumer) receives the data proactively after handover while the MC is moving. When an MC is watching a real-time video in a state of mobility and changing their position from one access point to another, the controllers in SDN preserve the network layout and topology as well as link metrics to transfer updated routes with the occurrence of the handoff or handover scenario, and through the proactive caching mechanism, the previous access router proactively sends the desired packets to the new connected routers.
2. We also illustrate the SDPCACM with respect to NDN regarding the intra- along with inter-domain handover management scenarios to sustain routing stability with the existence and presence of consumer mobility. Additionally, we present the SDPCACM's ability to efficiently control consumer mobility as well as request stuffiness issues, to decrease the loss of interest, and the hand-off delay for data packet.
3. This paper offers an illustrative methodology and parameter configuration for virtual machines (VMs), OpenFlow switches, and real open daylight (ODL) SDN controllers.
4. Finally, we conduct the simulation of the proposed SDPCACM for NDN using an emulated environment that is transmittable on the hardware directly. The experimental result shows that our scheme is significant in terms of delay time, CPU usage, packet loss ratio, jitter, and throughput, etc. Moreover, we make comparisons using experiments for NDN, SDN, and our proposed approach of leveraging SDN in NDN for mobility.

The rest of the paper is arranged as follows. Section 2 comprises of a background study and review of the literature which gives an overview of both SDN together with NDN architecture, as well as offers a review of the literature regarding NDN mobility designs. In Section 3, we describe the proposed architecture for NDN mobility that leverages SDN. Additionally, the intra-domain and inter-domain handover processing scenarios in the SDPCACM intended for NDN are described in this section. Section 4 consists of a discussion on the simulation environment and performance analysis of the suggested SDPCACM of NDN. Finally, Section 5 includes the conclusion and a summary of the results of the paper.

2. Background and Literature Review

Herein, we present a comprehensive literature assessment in the following subsections, including research about NDN, the methodology intended for its interest as well as a mechanism for forwarding, the SDN and its architecture, along with a review of mobility in NDN.

2.1. Named Data Networking (NDN) Overview

In order to efficiently distribute material such as streaming videos, music, and P2P services intended for data sharing, NDN is regarded as a future design with respect to the internet. In addition, it uses names to route packets rather than IP addresses and stores data in a cache [6–8]. Interest and data packets are the two categories of packets used in NDN. Moreover, the data packet includes information like a digital signature, signed documents, and so on. The interest packet also contains information like the name of the content being sought. The CS, PIT, and FIB are the three tables that make up the NDN node as shown in Figure 2. To store the data locally, CS is utilized, however, the PIT is applied to retain the interface between FIB and the consumer, it is also used as a forwarding strategy as illustrated in Figure 2. Each table has a distinct function. The router verifies the contents of data with the interested party's name in the CS when they show up at the router for placement of content. The PIT entails an interest list for future data, which is where the router searches if the data is not discovered in the CS. It is sent to the consumer if the operation accomplished on the look-up table has been successful; if not, it is verified with PIT before sending the packet of interest to the source. The router comprising of the whole interests that have conveyed but have not yet been satisfied are kept in the PIT. In this scenario, the interest packet is ignored since we know the router does not have the content and does not know how to deliver it from the source if no concerned entity is located. The forwarding strategy is maintained by the FIB, which also chooses how and when to convey interest [18].

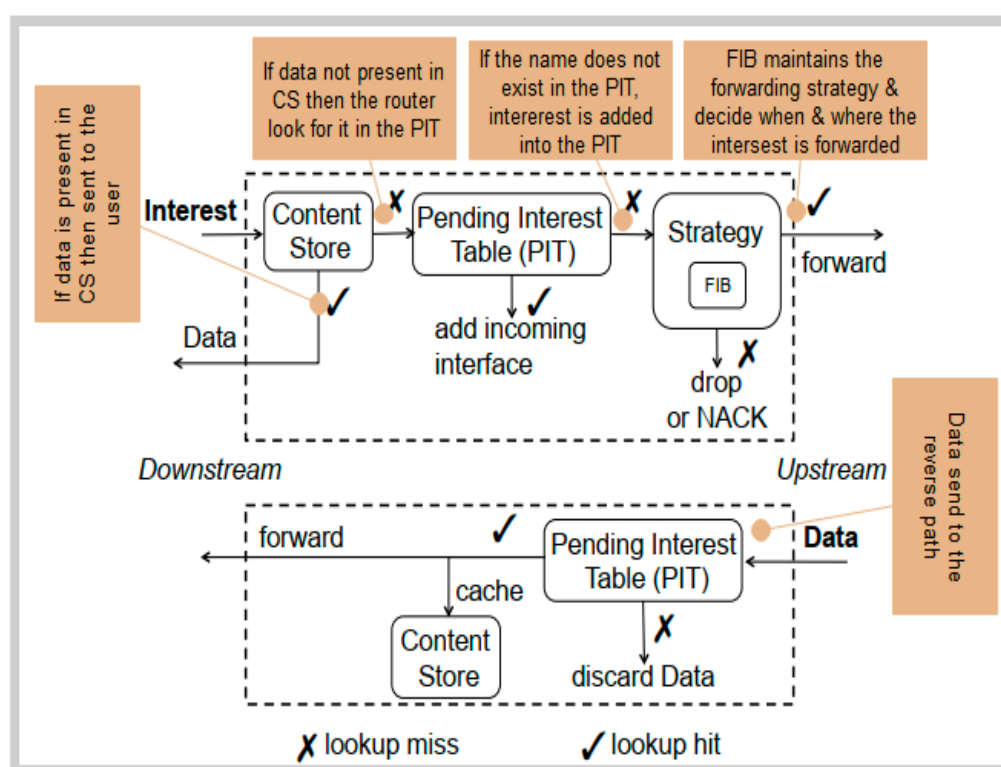


Figure 2. Interest vs. data processing approach in NDN [19].

2.2. SDN Model for NDN

The division of the network control from the data plane is the fundamental feature of SDN. As illustrated in Figure 3, SDN designs are composed of three layers: the infrastructure layer, the control, and the application layer. On conventional networks, the traffic will be distributed or transmitted, preventing them from having a comprehensive overview of the whole network. On the other hand, because they have logically centralized control, the SDN controller will retain the global perspective of the network.

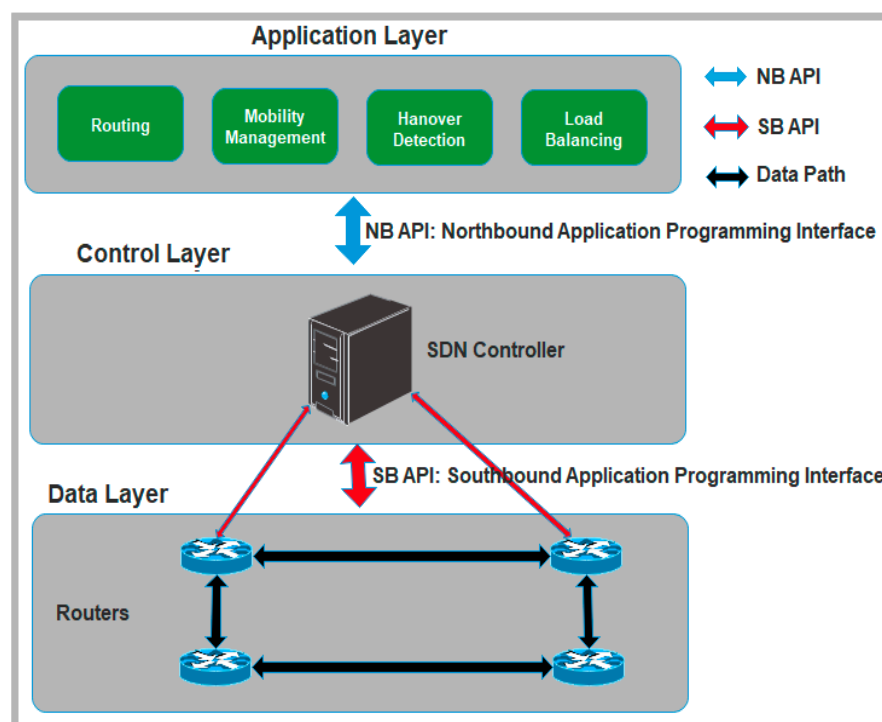


Figure 3. Software-defined networks model [20].

2.2.1. A. Infrastructure Layer

The infrastructure layer contains a number of network components that may exchange information and connect with one another, such as switches and routers. These devices carry out two tasks: they gather network status data, temporarily store it, and relay it to the controller. In addition, the second step entails processing the packet in accordance with the precise rules that the controller applies to the packets [20].

2.2.2. B. Control Layer

The management of network traffic falls within the purview of the control layer. It is made up of SDN controllers that allow for logical network control and give an overall picture of the underlying devices of the network. It is the SDN middle or control layer, through which the applications which execute on the controller control the behavior of the infrastructure layer [21].

2.2.3. C. Application Layer

Through various APIs, the application layer offers access points for various services. Northbound interfaces are used on devices to implement numerous APIs that are built for this layer's diverse functions [21].

2.3. Review of NDN Mobility

The author of the study [5] introduced a brand-new networking model called ICN. Through this review, they look at several crucial aspects, such as techniques for naming as well as routing, along with in-network caching options, and so on, as well as characteristics that distinguish the benefit of actually implementing ICN, unresolved research issues, and fresh appeal in this field. We provide a user mobility NDN communication model in the scheme [17] where interest and data packets are transferred with a single transaction unit. The link for transactions would be established through the mobility link service (MLS) suggested in this article, which operates in NDN.

Due to the regularly shifting locations of users, the fundamental help of MLS is in reusing the existing connection regarding a transaction as compared to starting a new

connection. The current connection is damaged if a consumer changes their position away from the adjacent node, and the data cannot be entirely received owing to handover. Therefore, in this instance, the connection is reinstated rather than being established anew, and fresh content requests are made to the original new NAR. As a result, the data for the remaining material was delivered by the prior NAR to the NDN producer, who then forwarded it toward the new NAR, which was requested by the user already [17]. The transaction is carried out using an IP-based architecture, as designated in the literature [22], which generates a simulation atmosphere for the NDN model to analyze performance in case of a mobile consumer changing their position between points to check the rate of generation for the interest packets leveraging and employing ndnSIM within the NS3 simulator.

An interest packet generated by a mobile user is initially wrapped with an IP header and sent to the NDN node through the associated AP. The packet is de-encapsulated and routed on the NDN layer when it gets an NDN node, which takes away the IP header from the packet. Additionally, the NDN node delivers IP header-encapsulated data packets to mobile users and returns them [22]. The technique will be explained in more depth for the SDN setting when utilizing OpenFlow, as explained in [23], where a method is provided that is based on hashing content names with the address field inside IP packets.

OpenFlow 1.0 and User Datagram Protocol (UDP) are used to implement this method. The UDP packet's payload contains the NDN packets that have been encapsulated. Interest and data packets are assigned to two separate port numbers. As a result, the switches may operate with both NDN and IP structures. This creates the routers prepared to detect these two NDN packets with further identification of IP packets from packets in NDN. There is no need in OpenFlow for directing packet payloads, hence NDN routers must be able to conduct longest prefix matching (LPM) on packet names. Consequently, the names are (encrypted), i.e., hashed and added to the packet header's IP field.

The authors in [24] demonstrate how the prediction of location can be utilized to perform the anchorless management of mobility and to guarantee a seamless handover for the producer with real-time communication for a multi-media environment. The results show that the methodology reduces round-trip time and handover latency.

In [25], authors enhanced and optimized a popular scheme for an anchor-based approach via an arrangement of multi-layer anchor nodes based on network topology in a hierarchical way, which is known as an anchor chain. Moreover, the interest packets are forwarded toward the producer, which also pass by it. As the producer changes its location, the newly produced anchor chain can reuse the prior forwarding route to guide further interests according to the modified location for the producer as well, as to reduce the drop rate of the interest packets. In addition, they introduce a mechanism for mobility preprocess that leverage connectionless as well as multi-path packet forwarding within the NDN to create the forwarding path for future interest in advance, that supports the producer to perform a seamless handover. The authors show, using numerical analysis, that their approach reduces latency during the handover and improves response ratio.

Similarly, the work in [26] suggests a mobility service managed in NDN which is accountable for supervision when a connection is performing a transaction. Hence, the new connection is determined in this manner that is intended for the transaction. Additionally, it hides the connection distortion from the new transaction. Through establishing a mobility service link to the NDN, the mobile service for the consumer is visible to the remaining NDN architecture. Consequently, the consumer's mobile service, as well as the architecture of NDN can be developed individually. The analytical evaluation and the results from simulation show that the author's proposed method reduces the transmission of data. Furthermore, the handover latency is compared with respect to the original solution for NDN mobility.

In [27], the authors proposed a mobility architecture for NDN that leverages SDN. The architecture was illustrated with a single controller and multi-controllers in SDN. However, there were no proof-of-concept experiments to evaluate the validity of the given strategy.

Moreover, the procedure was not effectively demonstrated with more details. Hence, our proposed scheme overcomes the limitations of the previous methods. In the following sections, we demonstrate our SDPCACM for NDN. Moreover, we provide the experimental setup and evaluation of our suggested approach.

3. Proposed Scheme

We introduced the idea of SDN technology and the design of an SDPCACM that improves QoS in mobility management.

3.1. Proposed SDPCACM Design for NDN

We will go on with an SDN-based NDN architecture once we have thoroughly reviewed the literature and comprehend these two networking innovations (NDN and SDN). These two emerging technologies (NDN, as well as SDN) are now being evaluated and developed by the global research community because of their characteristics and practical applications. Thus, developing an efficient mobility strategy for an NDN architecture based on SDN is essential. To combat this, we provide an effective and efficient SDPCACM suggested for NDN. The mobility and network model for the proposed software-defined routing are described below.

3.1.1. Network Model

Three planes—the administration plane, the control plane, and the data plane—make up the proposed SDN architecture. Though the control plane manages all actions of a router that are employed to gather and distribute traffic to cope with the performance of the network, the management plane is made up of various APIs created for various objectives and implemented upon devices by means of a northbound user interface. The data plane is made up of forwarding devices which forward packets, together with data and interest packets.

For the integration of these controllers to work together on the control plane, a mobility management program is needed. Northbound interfaces are used for communication between the SDN controller and the mobility management application. To facilitate the task of dissemination of mobility supervision applications, the communication between the SDN and the central controllers is carried out via an east–west contact interface. The controller, however, has a strong understanding of the status of the mobile consumers (MCs) as connected with the devices to send the packets to the MCs through a southbound interface, which is utilized for the collaboration of the data and control plane.

Three network elements—the NDN router, the user (MC), which includes an NDN access router (NAR) along with the NDN producer, as well as the controller as illustrated in Figure 4—are included in our planned architecture.

A. User

Data must be able to be pushed or pulled from the NDN by the user. If a user wants to add new material to the network, they may do so simply. If they want to retrieve data from the network, they submit a request to the NDN that contains their name.

B. NDN Router

The FIB, PIT, and content store (buffer memory) are the three main data structures found in every NDN router. The PIT records every interest and associates it with the network interfaces that receive pertinent requests. Data is then returned through the reverse request path utilizing this state. Since NDN supports on-path caching, a content object may be cached so that it may be sent in response to later requests for the same item (CS). The size, as well as popularity distribution for the object population, define the memory need for a specific traffic reduction [28].

C. Communication Procedure

NDN routers can communicate with the controller either directly (by sending report messages, for instance), or through a neighboring OF switch, which delivers reports delivered in a packet-in-packet (the OF switches send unspecified packets to controller). The OpenFlow channel, the OpenFlow protocol, and control messages (like traffic control messages) are all used in all of these techniques [28].

D. SDN Controller

The SDN controllers play a critical role in managing the mobility of network devices and users in a network. Here are some of the ways in which SDN controllers are important in mobility:

1. Centralized control: SDN controllers provide a centralized point of control for network devices, enabling network administrators to manage and control network traffic in real-time. This is particularly important in a mobile network where devices are constantly moving and changing their location.
2. Dynamic routing: In SDN, the network topology can be changed dynamically, allowing network administrators to route traffic to the most efficient path. This is essential in a mobile network where devices are moving in and out of coverage areas and need to be rerouted to ensure seamless connectivity.
3. Traffic engineering: SDN controllers allow for fine-grained traffic engineering, enabling network administrators to direct traffic to specific devices or areas of the network. This is important in a mobile network where devices are moving in and out of coverage areas and need to be directed to the most optimal path.
4. Network segmentation: SDN controllers allow for network segmentation, which enables network administrators to isolate network traffic from different devices and users. This is important in a mobile network where devices are constantly moving and may be carrying sensitive data.
5. QoS control: SDN controllers allow for the implementation of quality of service (QoS) policies, which enable network administrators to prioritize traffic based on its importance. This is important in a mobile network where devices are constantly competing for bandwidth and need to be prioritized based on their importance.

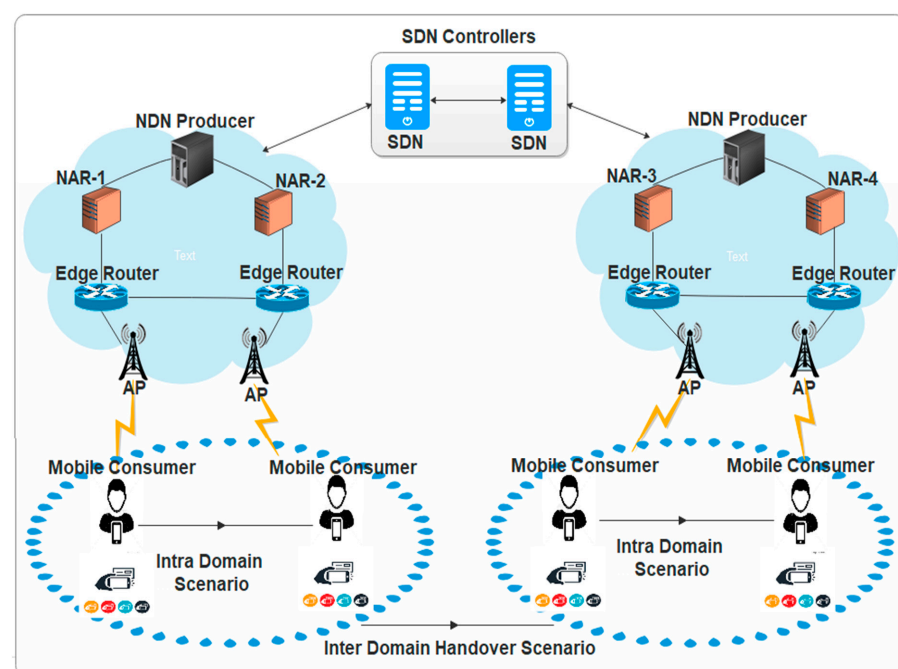


Figure 4. Software-defined mobility architecture for NDN.

In a nutshell, SDN controllers are essential in managing the mobility of network devices and users in a network. They provide a centralized point of control, dynamic routing, fine-grained traffic engineering, network segmentation, and QoS control, all of which are critical in a mobile network.

The SDN controller has a precise understanding of the MC's status. As a result, the SDN controller determines the routing parameter in support of user mobility in relation to the introduction of a certain protocol. The southbound API, also known as the OpenFlow protocol, is how the controller, who has a comprehensive view of the network, can understand the state of the underlying network. OpenFlow v3.0 has been utilized in this study.

3.2. Overview of Proposed SDPCACM Corresponding to NDN

In this part, the idea of SDN technology is employed to characterize the NDN's software-defined routing mobility.

3.2.1. Initialization Phase

Herein, we explain a scenario in the case where an MC is first connected with an edge router, which is linked to the NAR that the network controller has configured to enable NDN transactions. Moreover, the PIT, CS, and FIB make up the NDN node. When a mobile consumer contacts the router with an interest in content orientation, the router uses their name in the CS to examine the content data. The customer can then be provided with the desired info if it is discovered in the CS. The pending interest table (PIT), comprising an interest list awaiting data, is where the router searches if the data is not found in the CS. It is sent to the consumer if the operation inside the look-up table is effective; if not, it is checked with the PIT before sending the interest packet to the provider. In this scenario, the interest packet is ignored since we know the router does not have the content and does not know how to deliver it from the source if no concerned entity is located. The forwarding strategy is maintained by the FIB, which also chooses how and when to convey interest.

The edge router connects to the NAR via an IP router. The way it operates is as follows: primarily, once a mobile user creates an interest packet, it is further enclosed in an IP header, then sent to the node of an NDN over the connected AP. The packet is decapsulated when it gets to an NDN node by removing the IP header, and it is then forwarded in the NDN layer. The returning data packets are then once again wrapped with an IP header at the NDN node and sent to mobile users [18].

Inter-domain handovers and intra-domain handovers are the two forms of handovers that are illustrated below.

3.2.2. A Single SDN Controller Scenario

The inter-domain handover is supervised via a single controller. Therefore, herein, we demonstrate a scenario when an MC switches from AP-1 to AP-2, AP-3, and AP-4 accessing data packets from NAR-1 to a different NAR-2 surrounded by the reach of a single SDN network controller, or a scenario where the source of content and MC are participants or belong to the same controller. Next, the controller modifies the routing table, comprising the ID of MCs, interface, and content name, to keep itself aware of this movement via notifications. Furthermore, it sends updated new flow rules to the NDN access router (NAR). Once the NAR modifies its routing table according to updated flow entries, then the packets are destined to the MC, which is shown in Figure 5.

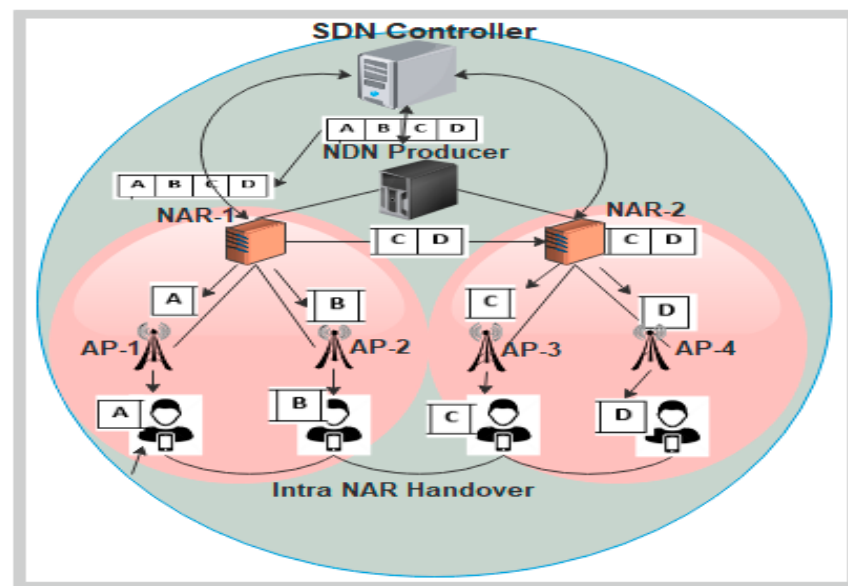


Figure 5. Single SDN controller or intra-domain handover operating scenario.

3.2.3. Intra-Domain Handover Operating Scenario

This section describes the handover handling scenario of SDPCACM to the NDN. In NDN, an MC only needs to know what packet they are requesting from the provider, not who provided it. Consider an MC watching a real-time video in a state of mobility at attachment point-1 (AP-1) provided by NAR-1, during this time their position is changing and moving from AP-1 to AP-2 and later from AP-3 to AP-4, as shown in Figure 5. The following steps will be required.

Before handover: in NDN architecture, a user MC want to watch a video, first must issue an interest packet and then the corresponding data packet is sent back to the MC. These data packets are stored in the CS of each intermediary router that delivers them [29]. **Step 1:** when the impending movement event is detected, the user, i.e., the MC, receives a control interest packet from NAR-1. The handover indication bit is then set to 1 by NAR-1, which also adds a handover field to the appropriate PIT entries. Following that, NAR-1 determines if a given PIT entry matches a data packet when it is received. The received data packet is cached by NAR-1 if the handover indication bit is set to 1 [29]. **Step 2:** in the meantime, NAR-1 has sent the control interest packet that contains all of the above information to the SDN controller. Thus, the SDN has a clear scope of the state of the entire movement.

During handover: as NAR-1 is aware of MC's movement and has particular PIT entries, the specific data packets that come into NAR-1 are not forwarded to the CS and are only cached. In this way, packets are prevented from being lost on the route to the MC's old location [28]. Meanwhile, NAR-1 sends a control interest packet (type: handoff initiation interest message) to the SDN consolidated controller. Furthermore, the SDN has a clear reach of the state of the fully updated movement toward the NAR-2.

Step 3: The controller within the SDN chooses the flow entry, hence, an updated routing table is shared with the NAR-2 to update the significant FIB entries of the routers providing this packet, triggering proactive caching actions on NAR-2. Therefore, NAR-2 actively initiates requesting and caching data packets previously stored in NAR-1 through the normal exchange of interest packet/data packet. In this way, NAR-2 proactively caches and receives the rest of NAR-1's contents.

Completing handover: **Step 4:** when the MC reaches within range of NAR-2, i.e., AP-3, it establishes a connection with NAR-2, transmits a handover complete message to NAR-2, and NAR-2 resends the remaining data packets to them. As a result of this command, the packet is sent to a new attachment point, i.e., AP-3 through NAR-2 from NAR-1 and is simply received by MC without any communication delay.

3.2.4. A Multi- (More Than One Controller) SDN Scenario

If an MC travels from the premises of one controller's coverage range to a new one, or if the content source and MC are from separate controllers, various controllers are used for system startup and handover management.

3.2.5. Inter-Domain Handover Use and Control Scenario

When a consumer moves to a new node, i.e., alters the domain area and enters another area of a different domain, then, inter-controller (between controllers) communication is mandatory to carry out the transfer of the MC. Figure 6 displays an inter-domain handover operating case, where the MC transitions from AP-4 to AP-5 accessing data packets (A-H) from NAR-2 and NAR-3, both of which are controlled by different SDN controllers.

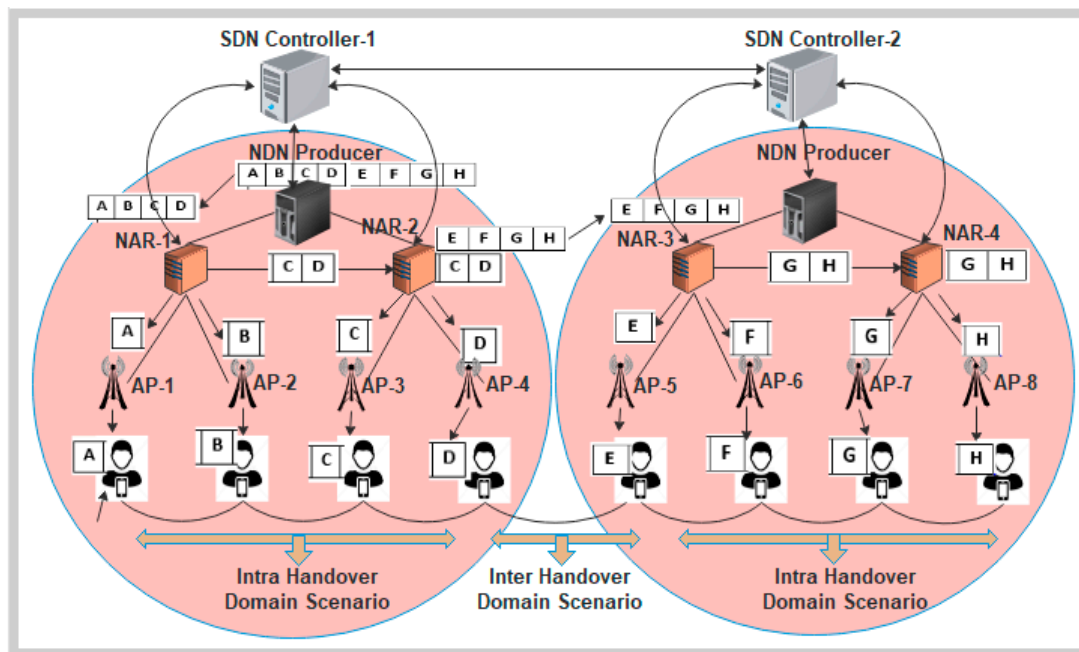


Figure 6. Inter-domain handover supervision scenario.

The before and after handover procedure is the same as in the intra-domain handover handling scenario, but the method is different during the handover, as illustrated in Figure 5.

Before handover: in an inter-domain handover handling scenario, such as when a user MC wants to watch a video, they first issue an interest packet and then the corresponding data packet is sent back to the MC and these data packets are stored in the CS of each intermediary router which delivers them [28]. **Step 1:** when the impending movement event is detected, the user, i.e., MC, receives a control interest packet from NAR-2 through AP-4. The handover indication bit is then set to one by NAR-2, which also adds a handover field to the appropriate PIT entries. Following that, NAR-2 determines if a given PIT entry matches a data packet when it is received. The received data packet is cached by NAR-2 if the handover indication bit is set to one [9]. **Step 2:** meanwhile NAR-2 has sent the control interest packet that contains all this above information to the SDN controller-1. Therefore, the SDN has a clear scope of the state of the entire movement.

During handover: as NAR-2 is aware of MC's movement and has particular PIT entries, the specific data packets that come into NAR-2 are not forwarded to the CS and instead are only cached. In this way, packets are prevented from being lost on the route to the MC's old location, which is AP-4 [30–34]. Meanwhile, NAR-2 sends a control interest packet (type: handoff initiation interest message) to SDN controller-1. Therefore, the SDN-1 has a clear scope of the state of the entire updated movement toward the NAR-3.

Step 3: in this scenario, the MC switches domains and enters another domain's territory, necessitating inter-SDN controller communication to manage the MC's travel. In this instance, it crosses over into another domain's territory and establishes a connection with NAR-3, which is situated near SDN controller-2.

For illustration, a technique is applied, i.e., in the first phase the controller-1 in SDN makes an inquiry for the subsequent controller which is controller-2 to report the movement of the MC, which is from AP-4 to AP-5, and also forward the information. This leverages an updated routing table with appropriate FIB entries of the routers delivering this packet, thus triggering proactive caching actions on NAR-3. Therefore, NAR-3 actively initiates requesting and caching data packets previously stored in NAR-2 through the normal exchange of interest packet/data packet. In this way, NAR-3 proactively caches and receives the rest of NAR-2's contents.

Completing handover: Step 4: when the MC is within range of NAR-3, i.e., the MC is within range of AP-5, it establishes a connection with NAR-3 and sends a handover complete message to NAR-3, who then resends the remaining data packets to the MC. As a result, the packet is transmitted in this manner from NAR-2 to NAR-3, where it is quickly received by the MC without any subsequent delay.

4. Simulation and Evaluation

This section presents the experimental setup and evaluation of the proposed SDP-CACM implemented for NDN. The suggested mobility model for software-defined NDN is simulated using Mini-net.

4.1. Experimental Setup

We use two virtual machines (VMs) running Ubuntu V20.04 LTS to demonstrate the reliability of our method. In one VM, the Mini-net V2.2.2 was loaded and used to simulate a software-defined network. At the edge, aggregation, and core levels of our architecture, there are eight, four, and two OpenFlow switches, respectively. There is a client linked to each access switch. The sub-flows of the module are supported by redundant links incorporated into the network at each level. Instead of employing a virtual machine on the same PC or machine to manage the internet, we increased the ODL controller operating in the same network. Because of the ODL controller's efficacy as described in [35], we employed it in our experimental analysis.

In the following paragraphs we explain the parameters we have evaluated for comparison of our proposed methodology with NDN mobility handover schemes proposed in [25] and [26]. We evaluate the delay, throughput, CPU utilization, packet loss ratio, and the jitter for comparison of our proposed strategy with the previous schemes suggested in the literature [25,26].

4.1.1. Parameters under the Evaluation to Validate the Effectiveness of Our Proposed Strategy

The following are the main parameters that were evaluated and measured in these experimental evaluations. First, we describe these parameters and then we discuss our results and evaluate the comparative results.

A. Delay Measurement

In our experiments, we measured the delay using Equation (1). Equation (1) shows that the delay was a combination of propagation, transmission, and querying delay. In Equation (1), the PD is the propagation delay, TD is the transmission delay, and QD is the querying delay. Hence, the total delay D is affected by these three factors contributing to delay. Further information to calculate the PD , TD , and QD is given in [36]. For sending the packets to the receiver we have used Mini-net and the Iperf tool [37]. Through Iperf we started a client and server on the sending and receiving hosts. Moreover, we recorded the time (ms) according to the Equation (1). Figure 7 shows the delay of handover scenarios in

single, multi-controller, inter- and intra-domain for SDN, NDN, and the proposed strategy (category 1 to 4). We can see that the delay with single controller category 1 is lower because of the involvement of a single controller. However, the delay increases with more controllers are mobile, i.e., from category 1 to category 4. The proposed architecture leverages the global view of the SDN and proactive measurements for transmission of packets resulting in a decrease in delay as compared to other schemes. We can also see that the delay in [25] is less than in the scheme of [26], because they present scheme for mobility with a preprocess that benefits from being connectionless. It also benefits from a multi-path mechanism for packet forwarding in NDN to produce a forwarding path with future interest in advance, that helps the producer in performing handover with less delay as compared to [26].

$$D = PD + TD + QD \quad (1)$$

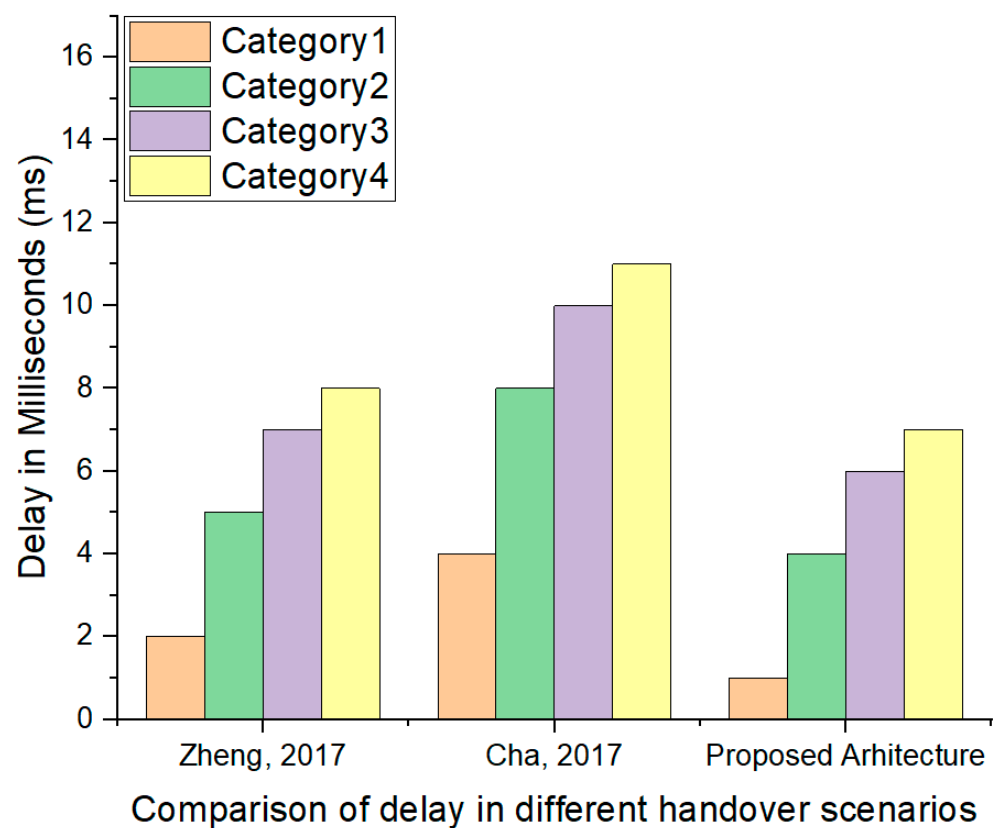


Figure 7. Comparison of the delay time with previous works [25] (Zheng, 2017) and [26] given by (Cha, 2017).

B. Throughput Measurement

The throughput is obtained by leveraging the Iperf tool. Throughput (T) is defined in Equation (2), i.e., the number of packets sent in unit time (t). In Equation (2), the Pi shows the packet, and the N is the total number of packets in unit time (t). We have recovered it over a period of 500 s. To execute the experiment, we have opened two hosts in the network and started the client server communication using Iperf. Iperf is mostly used to measure the throughput, i.e., the number of packets sent in a unit of time from the client to the server.

We measured the throughput of the SDN, NDN, and our proposed architecture leveraging SDN. Herein, the throughput is recorded for 500 s. Figure 8 shows the throughput comparison in various handover scenarios. For measuring the throughput, we used the Iperf tool and started a client and server sockets on the two hosts. Then, we started sending the packets from the client toward the server host on the listening socket. From Figure 8,

we can see that the proposed architecture has a global view provision for handover during mobility and, thus, the throughput is high as compared to [25] and [26]. The proactive caching as discussed in our scheme and the global perspective of SDN contribute to the improved throughput in comparison with other schemes. However, category 1, employing less movement, has increased throughput because of the low packet loss rate due to low mobility. In addition, the provisioning of services in [25] contributes to a higher throughput as compared to [26].

$$T = \frac{\sum_i^N P_i}{t} \quad (2)$$

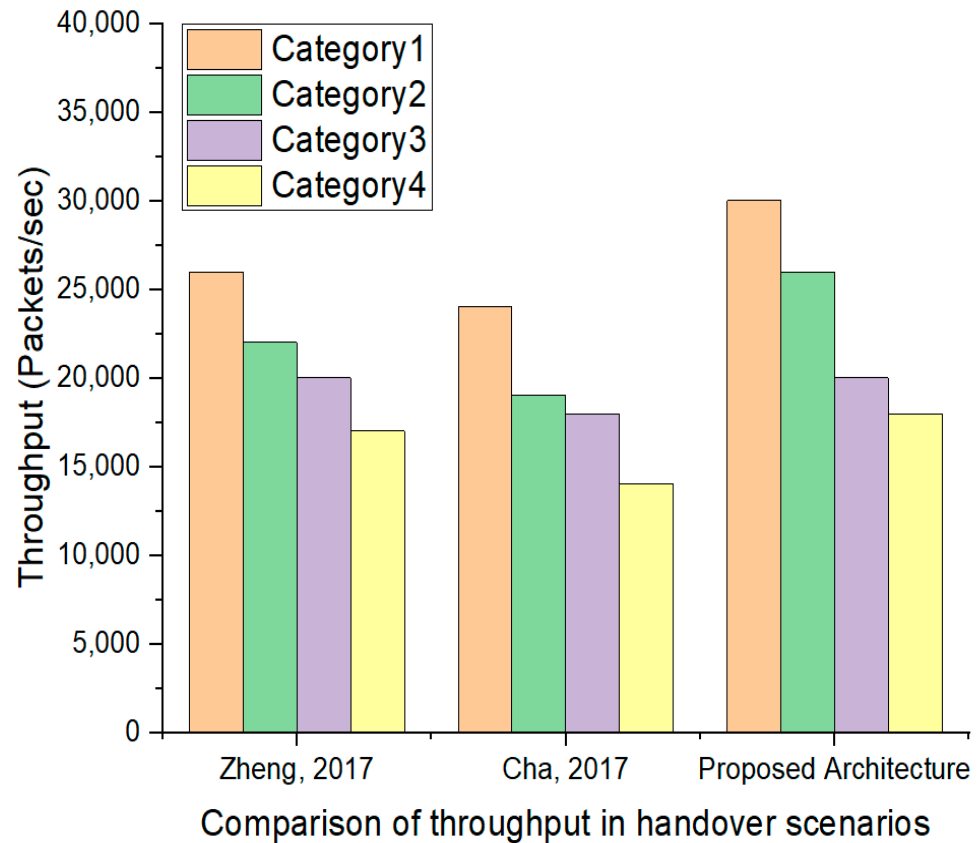


Figure 8. Throughput comparison of proposed scheme with [25] (Zheng, 2017) and [26] (Cha, 2017).

C. CPU Utilization

CPU usage refers to the workload that a CPU handles or the processing resources that a computer uses. The sysbench tool [38] is used to measure CPU utilization. The actual CPU usage is influenced by the amount and type of computing tasks managed by the system. The y-axis of the graph displays CPU usage as a percentage. CPU time (CT) is defined as the time amount that the process takes while utilizing the CPU. Moreover, the percentage conversion is performed via dividing it over the real time passed. Suppose if the CPU time calculated is 1 s out of total time for execution, i.e., 4 s, the CPU utilization will be $1/4 \times 100 = 25\%$. The time for CPU is computed as given in Equation (3).

$$CT = \text{Number of Instructions} \times \text{Number of Clock Cycles in the Instructions} \times \text{Time for a Clock Cycle} \quad (3)$$

The utilization of the CPU is a crucial aspect of cutting-edge technology, and higher CPU usage results in increased power consumption. To reduce CPU usage, we employed an efficient method that improved system speed and performance, as depicted in Figure 9. The results demonstrate the improvement of CPU usage measured with the sysbench tool. We can see a lower usage percentage as compared to the other two approaches presented

in [25] and [26]. This is due to the provision of resources in a proactive manner and a centralized view leveraging the SDN controller.

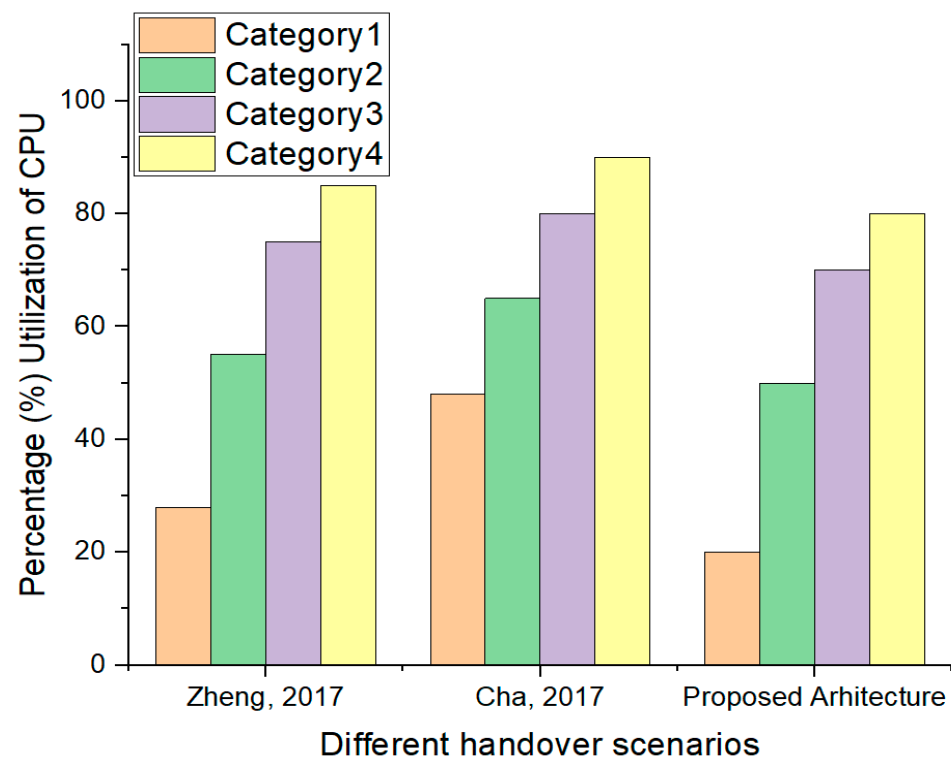


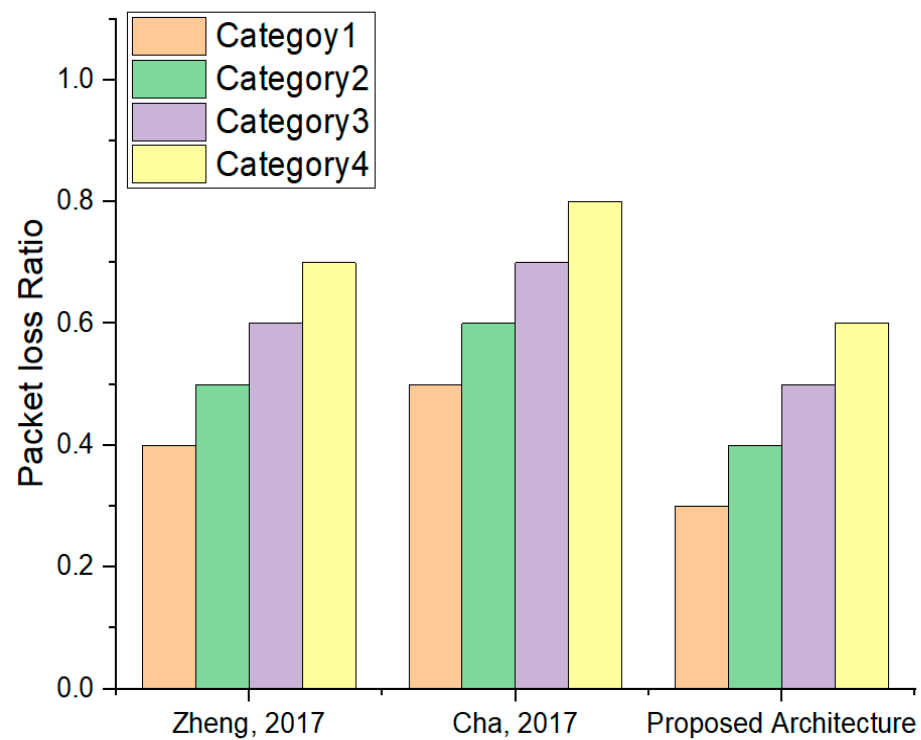
Figure 9. Comparison of CPU usages of proposed method with existing studies (Zheng, 2017), and (Cha, 2017) illustrated in [25,26].

D. Packet Loss Ratio

For packet loss, we have used the D-ITG (distributed internet traffic generator) tool [39] to record the packet loss ratio. To calculate the packet loss ratio, we have used Equation (4). The packets lost were calculated from the D-ITG log analysis on the receiver host. In Equation (4), the L shows the number of lost packets out of the total number of N packets. Herein, the k shows the total number of lost packets. The procedure to generate the packets and record the loss ratio is shown in Algorithm 1. On the target host, a listening socket is created for TCP communication from source nodes using ITGRecv (H2). The network used class C IP addresses, and on the source host (H1) ITGSend was utilized to send TCP traffic with a payload size of 500 bytes for a duration of 2000 s, at a rate of 1000 packets per second, to a destination node with the IP address of 192.168.1.50. This experiment was conducted 10 times, and the average packet loss results are presented in Figure 10.

$$PLR = \frac{\sum_i^k L}{N} \quad (4)$$

Figure 10 shows the packet loss ratio of our proposed scheme in comparison with [25] or [26]. We can see that the packet loss ratio of the proposed approach is low as compared to the other schemes. This is due to the proactive caching ability as explained in our proposed scheme and the overall network view of the underlying network infrastructure resulting in a lower loss of packets as compared to other schemes.



Comparison of Packet loss ratio in different handover scenarios

Figure 10. Comparison of the packet loss ratio for the proposed approach with (Zheng 2017) [25] as well as (Cha, 2017) [26].

Algorithm 1: Traffic production Algorithm

Step 1: Execute the graphical interface on H2.
 Step 2: At H2, modify the current directory and change to D-ITG/bin
 Step 3: Then give command: /ITGRecv at the H2 terminal.
 Step 4: In addition, open the GUI terminal of H1 host.
 Step 5: Then execute command on sending host H1: /ITGSend -T TCP -A 192.168.1.50 -c 500 -C 10,00 -t 2000 -l sender.log -x receiver.log
 Step 6: Then, perform log analysis for on H1, as well as H2
 Step 7: At H1: /ITGDec sender.log execute it
 Step 8: At H2: /ITGDec receiver.log execute this command

E. Jitter Evaluation

The jitter in our experiments is calculated using the following Equation (5). It is calculated according to the D-ITG definition of the jitter. The average of the direct jitter of network packets is computed as well as printed to a file known as “jitter.dat”. The results are displayed through opening the log analysis using D-ITG [39].

Figure 11 shows jitter results in milliseconds (ms) on the y-axis for the proposed method [25] and [26]. We can see from Figure 11 that the jitter in the proposed approach is small as compared to the other two approaches which do not use the proactive mechanism during handover. Moreover, the proposed approach takes advantage of SDN’s central management. Hence, the status of the network is visible to the controller resulting in a lower jitter during the handover scenarios for the proposed scheme. In addition, the jitter can be seen as higher in the other categories using more controllers for [25] and [26] during the handovers.

$$AvgJitter = \frac{\sum_i^n |D_i|}{n} \quad (5)$$

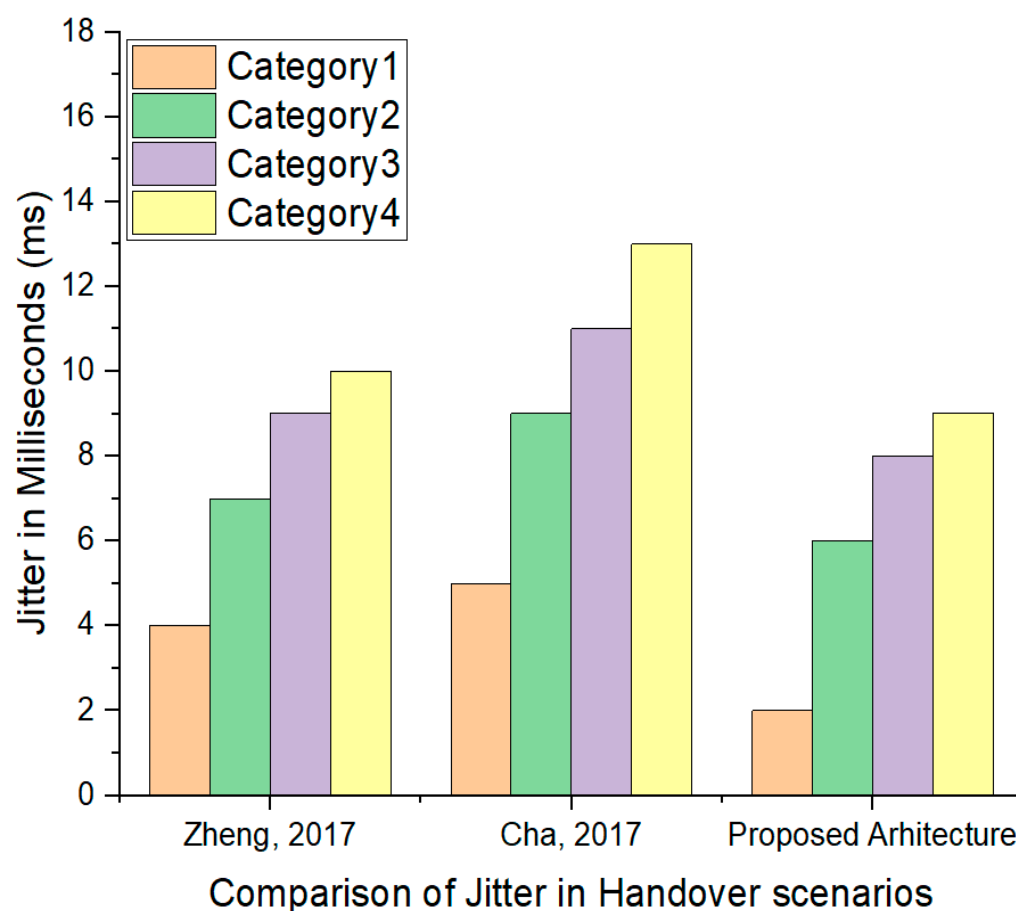


Figure 11. Comparison of jitter with [25] and [26] schemes proposed by (Zheng, 2017) and (Cha, 2017) in different scenarios.

5. Conclusions

The primary contribution of this study is the combination of SDN with NDN. It centered on three things. First, we designed the SDPCACM for NDN that addressed consumer mobility issues by extending new features to the NAR, such as pre-fetching and proactively caching contents on behalf of the consumer, in which the controller of the SDN maintains the network information for network topology to forward the updated routes in the handoff/handover situations. Watching a real-time video in a state of mobility that changes position from one access point to another, the controllers in the SDN preserve the network layout and topology. They also link metrics to transfer updated routes with the occurrence of the handoff or handover scenario, and through the proactive caching mechanism, the previous access router proactively sends the desired packets to the new connected routers which reduces the handover delay time. Secondly, this study described the intra- and inter-domain handover handling management scenarios for SDPCACM with respect to NDN to sustain routing stability with the existence and presence of consumer mobility, as well as to efficiently control consumer mobility and request stuffiness issues in order to decrease the loss of interest as well as the hand-off delay for a data packet. Thirdly, this paper provides examples of parameter settings for an ODL controller, OpenFlow switches, and virtual machines (VMs). This research offers exemplary techniques and parameter settings for OpenFlow switches, ODL controllers, and virtual machines (VMs). In addition to addressing problems like connection loss due to handoffs, incomplete data receipt, packet transmission delays, etc., this research also addresses benefits like flexibility, scalability, programmability, dynamic allocation of network resources, and centralized control. It also assists in meeting the NDN's performance and mobility management criteria. Additionally, it simulates the SDPCACM that is suggested for NDN. The experimental

and simulation result shows that our scheme has an improvement in results in terms of reducing handover delay time, increasing CPU usage, throughput, as well as reduction in jitter and packet loss ratio.

Author Contributions: Conceptualization, M.A. (Muhammad Adnan); data curation, M.A. (Manel Ayadi); formal analysis, H.E., L.A., M.A. (Muhammad Adnan) and J.A.; investigation, R.A. and M.A. (Manel Ayadi); methodology, M.A. (Muhammad Adnan); project administration, M.A. (Manel Ayadi) and J.A.; resources, R.A.; software, M.A. (Muhammad Adnan); supervision, M.A. (Manel Ayadi) and J.A.; visualization, R.A.; writing—original draft, M.A. (Muhammad Adnan), H.E. and R.A.; writing—review and editing, J.A., M.A. (Manel Ayadi) and M.A. (Muhammad Adnan); funding acquisition, M.A. (Manel Ayadi), H.E. and L.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Deputyship for Research and Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number RI-44-0229.

Data Availability Statement: Not applicable.

Acknowledgments: The authors extend their appreciation to the Deputyship for Research and Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number RI-44-0229.

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

References

1. Jiménez-Lázaro, M.; Herrera, J.L.; Berrocal, J.; Galán-Jiménez, J. Improving the Energy Efficiency of Software-Defined Networks through the Prediction of Network Configurations. *Electronics* **2022**, *11*, 2739. [\[CrossRef\]](#)
2. Ali, J.; Roh, B.-H. An Effective Approach for Controller Placement in Software-Defined Internet-of-Things (SD-IoT). *Sensors* **2022**, *22*, 2992. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Siris, V.A.; Ververidis, C.N.; Polyzos, G.C.; Liolis, K.P. Information-Centric Networking (ICN) Architectures for Integration of Satellites into the Future Internet. In Proceedings of the 2012 IEEE First AESS European Conference on Satellite Telecommunications (ESTEL), Rome, Italy, 2–5 October 2012; pp. 1–6.
4. Vasilakos, A.V.; Li, Z.; Simon, G.; You, W. Information centric network: Research challenges and opportunities. *J. Netw. Comput. Appl.* **2015**, *52*, 1–10. [\[CrossRef\]](#)
5. Ahlgren, B.; Dannewitz, C.; Imbrenda, C.; Kutscher, D.; Ohlman, B. A survey of information-centric networking. *IEEE Commun. Mag.* **2012**, *50*, 26–36. [\[CrossRef\]](#)
6. Gritter, M.; Cheriton, D.R. An Architecture for Content Routing Support in the Internet. In Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems (USITS 01), San Francisco, CA, USA, 26–28 March 2001.
7. Stanford University TRIAD Project, (Online). Available online: www.dsg.stanford.edu/triad/ (accessed on 20 January 2023).
8. Carzaniga, A.; Wolf, A.L. Content-Based Networking: A New Communication Infrastructure. In *Workshop on Infrastructure for Mobile and Wireless Systems*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 59–68.
9. Torres, J.; Ferraz, L.; Duarte, O.C.M.B. Controller-Based Routing Scheme for Named Data Network. *Electr. Eng. Program COPPE/UFRJ Tech. Rep.* **2012**. Available online: <https://www.gta.ufrj.br/ftp/gta/TechReports/TFD12.pdf> (accessed on 20 January 2023).
10. Azamuddin, W.M.H.; Aman, A.H.M.; Hassan, R.; Mansor, N. Comparison of Named Data Networking Mobility Methodology in a Merged Cloud Internet of Things and Artificial Intelligence Environment. *Sensors* **2022**, *22*, 6668. [\[CrossRef\]](#)
11. Saxena, D.; Raychoudhury, V.; Suri, N.; Becker, C.; Cao, J. Named Data Networking: A survey. *Comput. Sci. Rev.* **2016**, *19*, 15–55. [\[CrossRef\]](#)
12. Sinky, H.; Khalfi, B.; Hamdaoui, B.; Rayes, A. Responsive Content-Centric Delivery in Large Urban Communication Networks: A Link NYC Use-Case. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 1688–1699. [\[CrossRef\]](#)
13. Zhu, Z.; Afanasyev, A.; Zhang, L. A New Perspective on Mobility Support. *NDN Tech. Rep.* **2013**, *13*, 1–6. Available online: <http://new.named-data.net/wp-content/uploads/TRmobility.pdf> (accessed on 12 January 2022).
14. Detti, A.; Bracciale, L.; Loreti, P.; Rossi, G.; Blefari Melazzi, N. A cluster-based scalable router for information centric networks. *Comput. Netw.* **2018**, *142*, 24–32. [\[CrossRef\]](#)
15. Farahat, H. Proactive Caching to Support Mobility in Named Data Networks. Ph.D. Thesis, Queen’s University, Kingston, ON, Canada, June 2017.
16. Torres, J.V.; Alvarenga, I.D.; Boutaba, R.; Duarte, O.C.M.B. Evaluating CRoS-NDN: A comparative performance analysis of a controller-based routing scheme for named-data networking. *J. Internet Serv. Appl.* **2019**, *10*, 20. [\[CrossRef\]](#)
17. Cha, J.H.; Choi, J.H.; Kim, J.Y.; Han, Y.H.; Min, S.G. A mobility link service for ndn consumer mobility. *Wireless Commun. Mobile Comput.* **2018**, *2018*, 5149724. [\[CrossRef\]](#)

18. Fang, C.; Yu, F.R.; Huang, T.; Liu, J.; Liu, Y. A survey of green information-centric networking: Research issues and challenges. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 1455–1472. [CrossRef]
19. Yi, C.; Afanasyev, A.; Wang, L.; Zhang, B.; Zhang, L. Adaptive forwarding in named data networking. *ACM SIGCOMM Comput. Commun. Rev.* **2012**, *42*, 62–67. [CrossRef]
20. Kreutz, D.; Ramos, F.M.; Verissimo, P.E.; Rothenberg, C.E.; Azodolmolky, S.; Uhlig, S. Software-defined networking: A comprehensive survey. *Proc. IEEE* **2014**, *103*, 14–76. [CrossRef]
21. Todorova, M.S.; Todorova, S.T. DDoS attack detection in SDN-based VANET architectures. Master's Thesis, Aalborg University, Aalborg, Denmark, 2016; 175p.
22. Liu, Z.; Wu, Y.; Yuepeng, E.; Ge, J.; Li, T. Experimental Evaluation of Consumer Mobility on Named Data Networking. In Proceedings of the 2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN), Shanghai, China, 8–11 July 2014; pp. 176–472.
23. Ooka, A.; Ata, S.; Koide, T.; Shimonishi, H.; Murata, M. OpenFlow-Based Content-Centric Networking Architecture and Router Implementation. In Proceedings of the 2013 Future Network & Mobile Summit, Lisboa, Portugal, 3–5 July 2013; pp. 1–10.
24. Ali, I.; Lim, H. Anchor-Less Producer Mobility Management in Named Data Networking for Real-Time Multimedia. *Mob. Inf. Syst.* **2019**, *2019*, 3531567. [CrossRef]
25. Zheng, Y.; Piao, X.; Lei, K. Anchor-Chain: A Seamless Producer Mobility Support Scheme in NDN. In Proceedings of the 2017 IEEE 85th Vehicular Technology Conference (VTC Spring), Sydney, NSW, Australia, 4–7 June 2017; pp. 1–6. [CrossRef]
26. Cha, J.-H.; Choi, J.-H.; Kim, J.-Y.; Min, S.-G.; Han, Y.H. A mobility link service in NDN face to support consumer mobility service. In Proceedings of the 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), Milan, Italy, 4–7 July 2017; pp. 444–449. [CrossRef]
27. Ali, J.; Adnan, M.; Gadekallu, T.R.; Jhaveri, R.H.; Roh, B.H. A QoS-Aware Software Defined Mobility Architecture for Named Data Networking. In Proceedings of the 2022 IEEE Globecom Workshops (GC Wkshps), Rio de Janeiro, Brazil, 4–8 December 2022; pp. 444–449. [CrossRef]
28. Sun, Q.; Wendong, W.; Hu, Y.; Que, X.; Xiangyang, G. SDN-Based Autonomic CCN Traffic Management. In Proceedings of the 2014 IEEE Globecom Workshops (GC Wkshps), Austin, TX, USA, 8–12 December 2014; pp. 183–187.
29. Rao, Y.; Zhou, H.; Gao, D.; Luo, H.; Liu, Y. Proactive Caching for Enhancing User-Side Mobility Support in Named Data Networking. In Proceedings of the Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, Taichung, Taiwan, 3–5 July 2013; pp. 37–42.
30. Torres, J.; Duarte, O.C.M. *CRoS-NDN: Controller-Based Routing Strategy for Named Data Networking*; Universidade Federal do Rio de Janeiro GTA/COPPE/UFRJ: Rio de Janeiro, Brazil, 2014; pp. 1–6.
31. Aubry, E.; Silverston, T.; Chrisment, I. SRSC: SDN-Based Routing Scheme for CCN. In Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), London, UK, 13–17 April 2015; pp. 1–5.
32. Eum, S.; Jibiki, M.; Murata, M.; Asaeda, H.; Nishinaga, N. An ICN Architecture within the Framework of SDN. In Proceedings of the 2015 Seventh International Conference on Ubiquitous and Future Networks, Sapporo, Japan, 7–10 July 2015.
33. Rowshanrad, S.; Parsaei, M.R.; Keshtgari, M. Implementing NDN using SDN: A review on methods and applications. *IJUM Eng. J.* **2016**, *17*, 11–20. [CrossRef]
34. Rehman, M.A.U.; Ullah, R.; Kim, B.S. NINQ: Name-Integrated Query Framework for Named-Data Networking of Things. *Sensors* **2019**, *19*, 2906. [CrossRef]
35. Ali, J.; Roh, B.H. Quality of service improvement with optimal software-defined networking controller and control plane clustering. *Comput. Mater. Contin.* **2021**, *67*, 849–875. [CrossRef]
36. Wang, G.; Zhao, Y.; Huang, J.; Wu, Y. An Effective Approach to Controller Placement in Software Defined Wide Area Networks. *IEEE Trans. Netw. Serv. Manag.* **2018**, *15*, 344–355. [CrossRef]
37. Iperf. Available online: <https://iperf.fr/> (accessed on 19 March 2023).
38. Multi-Threaded Benchmark Tool. Available online: <https://github.com/akopytov/sysbench> (accessed on 20 January 2023).
39. Botta, A.; Dainotti, A.; Pescapé, A. A tool for the generation of realistic network workload for emerging networking scenarios. *Comput. Netw.* **2012**, *56*, 3531–3547. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.