D-NDNoT: Deterministic Named Data Networking for Time-Sensitive IoT Applications

Afia Anjum, Paul Agbaje, Sena Hounsinou, Nadra Guizani, Habeeb Olufowobi *Member, IEEE*

***Abstract*—Named Data Networking (NDN) revolutionized IP-based communication by introducing a content-centric model, based on name-based communication. This paradigm shift offers benefits, including optimized network traffic through in-network caching, improved data security, and resilient communication for Internet of Things (IoT) applications. While these benefits are significant, the deterministic data delivery necessary for time-sensitive IoT applications cannot be guaranteed using the NDN's best-effort routing mechanism. This paper addresses this challenge by proposing deterministic NDN of things (D-NDNoT), a protocol-level integration of a schedulability algorithm into NDN, making it deadline-aware and addressing the specific requirements of time-sensitive IoT applications. We present a time-sensitive NDN protocol incorporating a critical deadline-first scheduler to prioritize traffic. By integrating deadline awareness, quality of service metrics, and network characteristics, the algorithm ensures the delivery of time-sensitive data takes precedence over non-time-sensitive content. To validate the effectiveness of the proposed protocol, we evaluate using simulation experiments in OMNET++ and consider metrics such as end-to-end latency, delay, and deadline. The results demonstrate that the deadline-aware deterministic NDN protocol effectively meets the communication needs of time-sensitive IoT applications, ensuring the timely delivery of critical data.**

***Index Terms*—NDN, Scheduling algorithm, Time-sensitive applications, Deadline awareness, IoT**

I. Introduction

As the world becomes increasingly connected through the ever-growing web of interconnected embedded devices via the Internet, the Internet of Things (IoT) has revolutionized many aspects of our daily lives, from healthcare and transportation to agriculture and manufacturing. The prevalent use of these devices necessitates the growing need for communication protocols that can efficiently support the unique demands of IoT applications. This is particularly crucial for applications that are time-sensitive or require high levels of reliability, as failing to meet their stringent time requirements can have catastrophic consequences.

One protocol that has shown promise in meeting these requirements is the Named Data Networking (NDN) [1], which is based on the concept of named content rather than the traditional IP-based approach of addressing hosts. NDN allows for efficient and scalable content delivery with its unique features of content naming, security, and in-network caching, which is particularly well-suited for IoT applications. However, the inherent uncertainty and dynamic nature of wireless communication in IoT networks can lead to delays and packet losses, posing challenges for time-sensitive applications with strict response time requirements. Furthermore, since NDN provides best-effort data delivery [2], the in-network caching feature alone cannot meet the deterministic needs of time-critical IoT applications [3]. Therefore, the performance of IoT applications with varying resource and time requirements cannot be enhanced solely with the adoption of NDN.

To address these challenges, we propose a deterministic NDN for IoT applications called D-NDNoT. D-NDNoT ensures guaranteed data retrieval and transmission by incorporating a scheduling algorithm that facilitates reliable and deterministic communication. By employing D-NDNoT, IoT applications that require high levels of predictability and reliability, such as industrial automation, transportation, and healthcare, are better supported.

D-NDNoT achieves deterministic communication by enforcing strict rules on the behavior of network nodes using a critical deadline-first scheduler to minimize worst-case delays and ensure that packets are transmitted and received according to a pre-determined schedule. This scheduling policy guarantees delivery times and reduces the likelihood of packet loss due to interference or congestion. Additionally, D-NDNoT supports prioritization and synchronization of data delivery, ensuring higher priority real-time traffic meets their deadlines, which can be important for time-sensitive applications.

While NDN has been proposed for real-time communication in applications such as video conferencing [4] and video streaming [5], research initiatives exploring its use and integration in safety-critical systems are lacking due to the non-guaranteed data delivery. Despite this limitation, NDN offers desirable features which are beneficial for real-time communication in safety-critical systems. To address the lack of guaranteed data delivery, this paper introduces a novel approach by incorporating deadline awareness and scheduling to enable reliable and timely communication of time-critical data.

However, it's worth noting that integrating NDN into real-time IoT applications is still in its nascency and has yet to be widely adopted in real-world applications. Several challenges, including standardization, real-time performance, interoperability, and the availability of commercial off-the-shelf NDN devices and software, still need to be addressed. In this context, this paper analyzes the current state of time-critical traffic scheduling in NDN for time-sensitive IoT applications. Additionally, this paper proposes a deterministic deadline-aware

Corresponding author: Afia Anjum

Afia Anjum, Paul Agbaje, Nadra Guizani, and Habeeb Olufowobi are with the Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, TX, USA (E-mail: afia.anjum@uta.edu, pauloluwatowoju.agbaje@uta.edu, nadra.guizani@uta.edu, habeeb.olufowobi@uta.edu)

Sena Hounsinou is with the Department of Computer Science, Metro State University, Saint Paul, MN, USA. (E-mail: sena.houeto@metrostate.edu)

dynamic scheduler for traffic within the NDN router, aiming to provide reliable and predictable NDN communication for safety-critical systems.

Contribution. The contributions of this paper are:

- We provide a detailed assessment of the challenges faced in enabling the NDN protocol for time-sensitive IoT applications and address the limitations of best-effort NDN for real-time communication.
- We introduce a novel *critical deadline-first (CDF) scheduler* at the protocol level and propose D-NDNoT, which prioritizes network traffic based on the earliest critical deadline, effectively making the protocol deadline aware.
- We present a comprehensive analysis of the worst-case end-to-end delay of time-sensitive packets in the proposed D-NDNoT communication protocol.
- We demonstrate and evaluate the proposed approach through simulated evaluations in realistic automotive scenarios using OMNeT++, which indicates a significant reduction in time-sensitive packet delay compared to naive NDN and state-of-the-art approaches, meeting stringent deadline requirements in various scenarios.
- We present a detailed and comprehensive time complexity analysis of the proposed scheduler to demonstrate the impact of integrating the CDF scheduler into the NDN router.

The remainder of this paper is organized as follows. First, we discuss the necessary background information in Section II and Section III discusses the related works. In Section IV, we formulate the schedulability problem of time-sensitive traffic in NDN, and Section V introduces the proposed D-NDNIoT solution. Section VI analyzes the proposed approach's performance, and section VII concludes the paper.

II. Background

This section presents the core functionalities of the NDN of Things (NDNoT) network, emphasizing its key features over TCP/IP for IoT communications.

A. NDN of Things (NDNoT)

The named data networking is a proposed internet architecture that changes the communication model by shifting from the current host-centric model to a content-centric model. NDN names content objects hierarchically instead of identifying hosts using IP addresses. As a result, NDN offers several benefits over TCP/IP for IoT communications, such as content-centric security, content-based naming, in-network data caching, and improved routing and forwarding. These unique features of NDN offer potential solutions to the challenges encountered by IoT networks using TCP/IP and have the potential to meet the communication requirements of IoT [6]. This potential makes it possible to create an NDNoT network that utilizes NDN as its communication protocol for interconnecting IoT devices. Also, NDN provides several other advantages for IoT communications, such as scalability, flexibility, and improved quality of service by focusing on the data content rather than its location, allowing for more efficient use of network resources and providing a flexible data model that can support varying use cases. Following are the details of NDN elements in the context of safety-critical IoT systems.

1) ***Content-centric Security****:* NDN secures each content individually instead of protecting the communication channel by requiring each producer in the NDN network to sign the produced data cryptographically. Using a trust model comprised of a trust anchor, a trust chain, and a set of rules, consumers can verify the signature before utilizing the received data packet [7]. This content-centric security ensures the authenticity and integrity of data in NDN-enabled IoT applications, which can help protect against cyber attacks.

2) ***Content Naming****:* Traditional IoT applications that run over the TCP/IP stack require an additional server, such as a domain name system (DNS), to translate the application-level name to the IP address. The DNS overhead, coupled with the high mobility of IoT nodes, makes it challenging to keep track of billions of IP addresses. In contrast, with its distinctive content-naming feature, NDN eliminates the need for DNS's name-to-IP translation and continuous IP tracking by naming each data hierarchically rather than utilizing an IP address [8]. Consequently, the routing and forwarding are based on the content name. Moreover, the content naming approach reduces the overhead required for packet header processing, thereby reducing power consumption while increasing network efficiency.

3) ***In-network Data Caching****:* IoT data transmission over TCP/IP is protected through a secure communication channel between the producer and the consumer. Hence, a node can retrieve data only from the original producer or trusted cache servers. However, NDN's content naming and content-centric security allow data packets to be decoupled from their original producers in NDNoT applications. Therefore, the storage location of a data packet can be independent of its producer and held closer to its consumer, thereby reducing both the producer's load and the consumer's retrieval delays. NDN nodes implement this proximity to the user through an in-network caching functionality. The network's overall efficiency can be enhanced by utilizing in-network caching, which relies on well-designed caching policies.

4) ***Routing and Forwarding****:* In NDN, each node behaves as a router having three core components: a pending interest table (PIT), a content store (CS), and a forwarding information base (FIB). An NDN node issues an Interest packet (referred to as Interest herein), including the content name, while requesting data. The Interest first goes through PIT to see whether a pending Interest for the same data already exists. If a matching entry is found, the router interface ID of the incoming Interest is included in the existing PIT entry, enabling the received data packet to be sent to all the interfaces listed in the PIT entry. The Interest is added and forwarded to CS if no matching entry is found for the same content in the PIT. If the requested data packet is located in the CS, it is sent back to the requester. Otherwise, the Interest is forwarded to FIB which provides multiple forwarding paths. The paths route the Interest to the original producer or any node with the requested data stored in its CS. The router maintains an entry for a forwarded Interest in the PIT as long as the Interest's lifetime has not expired. After the expiration, some NDN strategies retransmit

unsatisfied Interests, and others leave the application with the decision of whether and when to retransmit [9].

III. Related Work

Incorporating NDN into time-sensitive applications requires reducing the delivery latency of time-sensitive traffic. Research efforts to improve traffic latency primarily focus on proposing different forwarding and caching strategies that do congestion control or improve in-network caching. In addition, several works have explored the integration of Time-sensitive Networking with other networking protocols to enable deadline awareness. We summarize these efforts here.

A. Routing and Forwarding

Prior works have proposed different solutions to address data transmission delay through routing and forwarding schemes [10]–[13]. Kalogeiton et al. [10] proposed an enhanced Geographical aware Routing Protocol (eGaRP). eGaRP enhances vehicle-to-vehicle communication through the use of directional antennas. This reduces the need for broadcasting messages during content retrieval and minimizing network resource usage. Rou et al. [11] addressed the issue of data transmission path-breaking due to node mobility. The authors proposed a vehicle tracking-based data packet forwarding (VTDF) scheme to reduce the latency caused by data delivery failure. VTFD introduces a Tabu node search algorithm and a quick handover method to track the directions of the motion of data-requesting vehicles and provide efficient handover to appropriate RSUs during data delivery, respectively. To mitigate data packet delivery delays resulting from path disruption caused by mobile nodes, Zhang et al. [12] propose a neighbor-aware forwarding approach that requires each node to maintain a neighbor table, which is periodically updated through a broadcast message and replies that include the geographical location, to assist in interest and data packet forwarding. However, due to high mobility, the neighbors of a specific node may keep changing, leading to redundancy and overhead in the network. Furthermore, Chowdhury et al. [13] proposed a new forwarding strategy for VANETs called content connectivity and location-aware forwarding (CCLF) to address the issue. CCLF uses location information to forward nodes and prioritizes well-connected nodes using connectivity information, thereby reducing redundancy and overhead in the network. Hashemi et al. [14] proposed a congestion control protocol by providing explicit feedback to the consumer. The proposed method incorporates additional formats in interest and data packets to convey congestion information and estimated resources along the transmission path, enabling the adjustment of interest packet sending rates for each interface. Omitsu et al. [15] proposed a multi-path routing algorithm that avoids congested paths to reduce delivery latency. The proposed method integrates various tables, such as connection and link usage tables, to determine optimal packet detour points, minimizing congestion and unnecessary hop increments. The authors proposed an efficient one-interest multi-data forwarding approach that can transmit all fragments of content within a short time.

B. Caching

Prior work have proposed solutions to improve caching and name lookup performance of NDN. Since the delivery time of packets is significantly influenced by the name lookup time and effective caching. Yu et al. [16] propose a caching policy to improve the cache hit rate in NDN. The proposed approach caches the most requested contents on routers close to the requester instead of caching each data packet that passes by. This approach reduces data delivery latency by ensuring frequently accessed content is cached relatively close to the user. Zhang et al. [17] propose an efficient ternary content addressable memory (TCAM) to enable fast name lookup. Content names from incoming interest packets are loaded directly into the TCAM, which provides faster memory lookups. To speed up the NDN name lookup process, Wang et al. [18] proposed a Name Prefix Tree (NPT) that utilizes memory resources in parallel. When multiple interest prefix arrives at the NPT node at once, all memories are visited concurrently, resulting in high-speed lookup. Similarly, Wang et al. [19] propose a GPU-based lookup engine that uses a trie-based multiple-aligned transition array data structure to enable large-scale name lookup at wire speed. Qu et al. [20] address the congestion caused by the limited caching capabilities of intermediate nodes. Space constraints at the intermediate node limit caching all the fragments of content, requiring multiple interest packets to retrieve the requested data packet and eventually creating congestion.

These routing and caching strategies mentioned in Section III-A and Section III-B can improve the content delivery rate to a certain extent through efficient caching and forwarding; however, time-critical data may get delayed due to frequent non-critical data since no priority is given. As a result, these approaches do not guarantee deterministic data delivery for time-sensitive IoT applications.

C. TSN Integration with other Networking Protocols

Time-sensitive networking (TSN) was developed by the IEEE 802.1 time-sensitive networking task group [21] to enhance the real-time performance of the IEEE 802 network. The task group established a set of standards to provide deterministic, low-latency communication for time-critical applications. To support deadline-aware traffic, TSN uses the Time-aware Shaper (TAS) to regulate the transmission rate of packets. TAS is a TSN scheduling mechanism for ensuring low latency and deterministic delivery of the time-triggered (TT) traffic [22].

Previous research has investigated the integration of TSN into various networking approaches. To leverage the benefit of an integrated network of TSN and software-defined networking devices, Böhm et al. [23] propose a unified control plane called Time-Sensitive-Software-Defined Networking (TSSDN). TSSDN offers the ability to configure a mixed network of TSN and SDN devices to provide deterministic and non-deterministic communication. Wang et al. [24] discuss the deterministic transmission and low latency requirements of 5G supporting V2X communication and emphasize the need for TSN and 5G integration. To evaluate the performance of

5G-TSN for real-time industrial applications, Kehl et al. [25] developed a prototype integrating 5G in a TSN network where the 5G system is treated as a TSN bridge in the framework.

Although several works have explored integrating TSN with different networking protocols, our analysis reveals that integrating TSN into NDN brings interoperability and real-time performance challenges. For instance, integrating preemption-based TSN, where a lower priority transmission is halted due to the arrival of higher priority frames, can cause security attacks, such as denial-of-service attacks. Alternatively, integrating non-preemptive TSN necessitates prior knowledge of the maximum packet size to incorporate a guard band, representing the transit time of the largest packet. Additionally, offline calculation of GCL poses difficulties for dynamic networks. Finally, the differences between the TSN switch and NDN router queues, where NDN routers have a single queue for each flow type, make it infeasible to adopt TSN in NDN. Therefore, instead of integrating TSN into NDN, we propose a router-level deadline-awareness integration through the proposed scheduler.

D. Deadline-awareness in NDN

Nagaraj et al. [26] explore the possibilities of integrating the advantages of NDN, such as naming schemes, in-network caching, and the reduction in the overhead of name-to-IP mapping, for building next-generation Industrial Automation and Control Systems (IACS). Despite the benefits, the authors describe the infeasibility of NDN to provide traffic scheduling functionalities to meet the stringent time requirements of IACS. In subsequent work, Nagaraj et al. [27] demonstrate the incompatibility of integrating NDN within IACS due to the lack of a traffic control system in the NDN stack. However, no approach solves the latency requirements in NDN-enabled IACS. Therefore, this paper investigates the feasibility of integrating schedulability into NDN and introduces deadline-awareness and deterministic data transmission to NDN protocol for time-sensitive IoT applications. The proposed deterministic NDN uses a critical deadline-first scheduler to ensure that time-sensitive application data meets their required timing constraints.

IV. Problem Statement

Various IoT applications are categorized as latency-critical applications, which require ultra-reliable low-latency communication [3]. This category of IoT applications can leverage the efficient data retrieval, data integrity in highly mobile networks, content-driven data forwarding, and content-centric security features offered by NDN. However, the best-effort routing of NDN [28] does not guarantee deterministic data delivery in time-sensitive applications such as the Internet of Vehicles (IoV), smart health, IoT telemetry, and industrial automation [29]. For example, autonomous vehicles need to quickly and reliably share safety-critical data, including collision warnings, road hazards, and emergency vehicle alerts. These vehicles also require efficient management of time-critical tasks, such as scheduling vehicle-to-vehicle communication and processing sensor data with urgency and importance. The challenge lies in timely transmission of vehicle

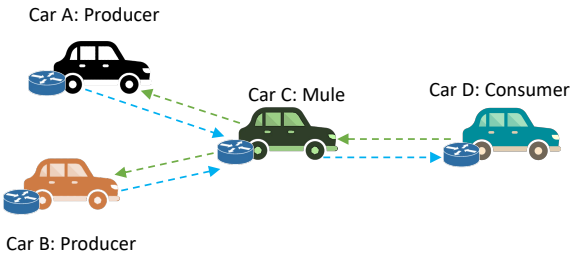

Fig. 1: Exemplary test scenario of NDN-enabled IoV nodes

data to neighboring vehicles, roadside units, and the cloud while accommodating other non-time-critical applications in the network. Similarly, in smart homes, delays may occur when requesting emergency medical assistance due to the network forwarding non-emergency data generated at a higher frequency. Likewise, in telemetry, even a millisecond delay in delivering crucial information, including mission termination, can have adverse effects on the robot and the environment, endangering nearby people. Moreover, factory automation has stringent latency and reliability requirements, ranging from 0.25 ms to 10 ms [3].

TABLE I: Summary of the Characteristics of Traffic Flow

Source	Destination	Transmission Interval	Traffic Type	Payload Size (bytes)
A	D	3 μs	BE	1000
B	D	75 μs	TT	50
Link		**Bandwidth (MBps)**		
$A \rightarrow C$		500		
$B \rightarrow C$		500		
$C \rightarrow D$		250		

To illustrate the challenge of deterministic data delivery of time-triggered traffic (TT) in NDNoT, let us consider an IoV topology with four nodes (A, B, C, and D), as shown in Figure 1. The characteristics of the traffic generated by the two producers are presented in Table I. Here, we focus on the interest-sending scenario to highlight the time sensitivity of NDN. As depicted in table I, node B initiates a 50-byte-sized interest packet at intervals of 75 μs and requests TT data. Node A transmits an interest of 1000-byte for best-effort data (BE)—data that can be delivered within the best possible time—with a transmission interval of 3 μs. This causes the transmission link between nodes C and D to be stressed by frequent BE traffic.

Several factors need to be considered to estimate the time it takes for an interest packet to be transmitted, including delays. When an interest packet is initiated, it is added to the buffer queue of the node's router. The packet then passes through various components, such as CS, PIT, and FIB, before being sent to the next node on the transmission path. Therefore, the packet transmission time between two adjacent nodes can be approximated using the following equation:

$$T_{n_1-n_2} = T_{Q(n_1)} + T_{p_{n_1}} + T_{tr(n_1-n_2)} \quad (1)$$

where $T_{n_1-n_2}$ is the total time to transmit a packet from $node_1$ to adjacent $node_2$, $T_{Q(n_1)}$ is the queuing delay at $node_1$'s router buffer, $T_{p_{n_1}}$ is the packet processing time at $node_1$'s

router, and $T_{tr(n_1-n_2)}$ is the time to transmit the processed packet using the link between $node_1$ and $node_2$. The queuing delay $T_{Q(n_1)}$ varies from packet to packet, depending on the order in the queue. For instance, the first packet that arrives in the queue will incur a negligible queuing delay. In contrast, the fifth packet must wait for the transmission of the four packets preceding it. $T_{tr(n_1-n_2)}$ can be computed using the payload size and transmission rate as:

$$T_{tr(n_1-n_2)} = \frac{Payload\ Size}{Link\ Transmission\ Rate} \tag{2}$$

As mentioned in Section II, when an interest packet arrives at a router, exact string matching for the content name of the interest is done in CS, PIT, and FIB. Therefore, the processing time of a packet is greatly influenced by the name lookup time. To decrease the name lookup time, researchers have proposed several techniques, such as parallel name lookup and fixed-length name codes [17]–[19], [30]. Here, we consider that the NDN routers utilize the fixed-length name code approach [30], which restricts the name lookup time to an upper bound to handle variable name length issues, ensuring a bounded processing time. The specific upper bound for this name lookup time can vary depending on the implementation. In our analysis, we assume the total name lookup time (and, consequently, the packet processing delay $T_{p_{n_1}}$) to be less than or equal to $1\mu s$ in the latency calculation. Using this value of $T_{p_{n_1}}$ and Eq. 1, we can derive the following equations to compute the end-to-end packet transmission delay for segments A-C, B-C, and C-D, respectively:

$$T_{A-C} \leq T_{Q(A)} + 1\mu s + T_{tr(A-C)} \tag{3}$$

$$T_{B-C} \leq T_{Q(B)} + 1\mu s + T_{tr(B-C)} \tag{4}$$

$$T_{C-D} \leq T_{Q(C)} + 1\mu s + T_{tr(C-D)} \tag{5}$$

The values necessary to compute the delays mentioned in Eq. 3, 4, and 5 are specific to each use case and may differ depending on the network considered. Hence, we derive these values from the example scenario depicted in Figure 1 and corresponding Table I for this section.

The transmission time for BE traffic from node A to node C and node C to node D are 2μs and 4μs, respectively. No queuing delay will arise at router A since BE traffic is generated at a 3μs interval and each BE packet requires at most 3μs to be transmitted, including the router processing time. Hence, $T_{Q(A)} = 0$. The first BE packet will be transmitted without any queuing delay as soon as it reaches the router buffer of node C. On the contrary, since packets are placed on the queue at 3μs intervals, and it takes at most 5μs to dispatch a packet from node C to node D, including the router processing time, the latter packets will experience a delay. Therefore, the upper-bound of the $T_{Q(C)}$ would be 2μs for the 2nd packet, $2 * 2\mu s$ for the 3rd packet, $3 * 2\mu s$ for the 4th packet. Consequently, for the N-th packet, it would be $(N-1) * 2\mu s$.

Similar to BE traffic, the transmission time for TT packet from node B to C and node C to D are 0.1μs and 0.2μs, respectively. No queuing delay will arise at router B since TT packet is generated at a 75μs interval, and each TT packet requires at most 1.1μs to be transmitted. Hence, $T_{Q(B)} = 0$. The first TT packet will be generated at 75μs, and at the same time, BE packet will be generated. Considering $T_{tr(A-C)} > T_{tr(B-C)}$, the 25th packet at the router C's buffer would be the first TT packet. Therefore, the upper bound of the $T_{Q(C)}$ would be $24 * 2\mu s$ and T_{B-D} would be less than or equal to 48.3μs. As the first time-sensitive packet is facing an additional queue delay of maximum 48μs, after a certain time, this delay will increase with the increase in packet number, eventually leading to a missed deadline.

TABLE II: Summary of Notations

Symbol	Description
Ω_l	Data rate of communication link l
$\mathcal{L}$	Set of available communication links
ϕ	Traffic type
R_i	Release time of packet i
D_i	Deadline of packet i
D_{R_i}	Remaining deadline of packet i
D_{T_c}	Total deadline for content c
P_i	Shortest paths for packet i
F_k	Packet flow, where $k \in \{interest, data\}$
δ_i	Priority value of packet i
d_i	Absolute deadline of packet i
clk	Current clock time
b_r	Buffer size of router r
e_i	Expected remaining time
γ	Interference delay
ρ	Blocking delay
D_i^+	Worst-case end-to-end delay of packet i
w_i	Worst-case queuing delay of packet i
Ψ_i	Arrival time of packet i
T_{tr_i}	Time to transmit packet i from current node to destination
T_{h_i}	Time to transmit packet i from current hop to next hop

The above discussion highlights NDN's inability to address the time sensitivity of the traffic; therefore, the end-to-end transmission time of TT flows increases linearly with time, leading to missed deadlines. Thus, in this paper, we introduce the time-sensitivity factor and refine the architecture of NDN to address deadline misses, minimize delays, and ensure reliable data transmission. The ultimate goal of this proposal is to ensure that data frames with tighter deadlines or higher priority levels are transmitted promptly, minimizing the risk of missing their timing requirements in time-sensitive applications.

V. Deterministic NDN of Things (D-NDNoT)

This section introduces the system model and the design of the proposed deterministic NDN of things (D-NDNoT). The approach exploits the idea of incorporating deadline-awareness and deterministic data delivery features to NDN-based routers. Table II summarizes the notations.

A. System Model

The system model depicts a network of NDN nodes, where each node serves as an NDN router. The communication between nodes occurs through transmission mediums with data rates Ω_l, here $l \in \mathcal{L}$. The set $\mathcal{L}$ represents all available communication links in the network. In this request-driven communication protocol, two types of packets are generated by each node: interest packets and data packets, as explained in Section II. We assume each packet i, in addition to the conventional packet structure [31], [32], contains the traffic

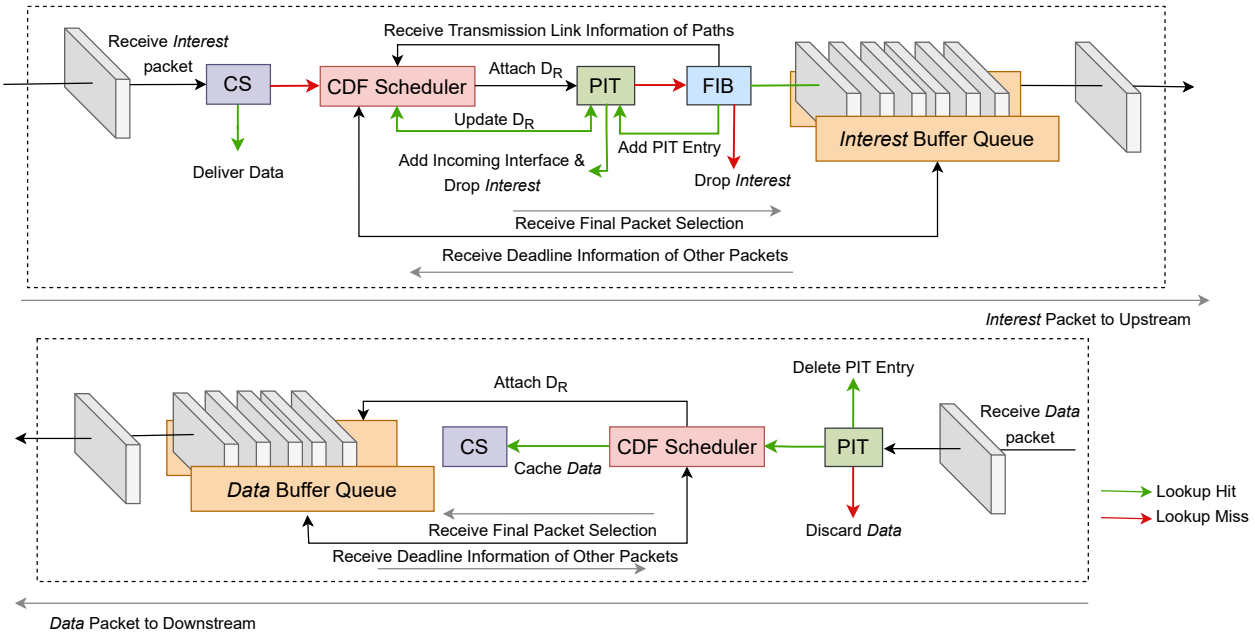

Fig. 2: Proposed D-NDNoT router framework for scheduling and forwarding process

type ϕ_i, which indicates whether the packet requests contain either TT or BE traffic, the packet release time R_i, the deadline D_i, the remaining deadline D_{R_i}, and the absolute deadline d_i. Here, the deadline is the maximum allowable time for a packet to reach its destination. The remaining deadline is the remaining time calculated at each hop, and the absolute deadline is the maximum allowable time for the corresponding data packet to reach the destination. Apart from these, the interest packet also contains the total deadline D_{T_i}, which is the maximum allowable time from interest packet release to data delivery for the requested content. Moreover, at the router level, each arriving packet gets multiple shortest paths P assigned to it to reach the destination. In a dynamic network, if any paths become unavailable, new shortest path are calculated using different routing approaches, such as [33].

We define two types of packets: interest packet i^{Φ} and data packet i^{Υ} with their respective attributes:

$$i^{\Phi} = (R_i^{\Phi}, D_{T_i}^{\Phi}, D_i^{\Phi}, D_{R_i}^{\Phi}, P_i^{\Phi}, \phi_i^{\Phi}, \delta_i^{\Phi}) \tag{6}$$

$$i^{\Upsilon} = (R_i^{\Upsilon}, D_i^{\Upsilon}, D_{R_i}^{\Upsilon}, P_i^{\Upsilon}, \phi_i^{\Upsilon}, \delta_i^{\Upsilon}) \tag{7}$$

where $D_i^{\Upsilon} = D_{T_i}^{\Phi} - D_i^{\Phi}$, is the deadline for the data packet based on the total deadline of the corresponding interest packet. δ_i^{Φ} and δ_i^{Υ} are the priority values assigned to the interest packet i^{Φ} and data packet i^{Υ}, respectively. In this paper, we use the term “BE packets" or “BE traffic" to refer to an interest or data packet that requests or contains BE data. Similarly, we use the term “TT packets" or “TT traffic" to refer to an interest or data packet that requests or contains TT data.

B. Design

In this design, each packet i is released at the beginning of an interval σ_i and must reach its destination before its absolute deadline, which is calculated as:

$$d_i = R_i + D_i \tag{8}$$

To facilitate the schedulability of packets in NDN routers, we introduce an additional component called the *Critical Deadline First (CDF) scheduler*. This scheduler is integrated with the router's core elements and is responsible for prioritizing and scheduling packets in the router buffer. The operation of the CDF scheduler is depicted in Figure 2.

Before forwarding an interest packet to the PIT and the FIB, the scheduler performs several important tasks. First, it calculates the remaining deadline and priority for each received packet. Additionally, it updates the remaining deadline and priority during interest aggregation. The scheduler also assists in distributing the received data to the appropriate requesting interfaces. Moreover, it plays a crucial role in selecting the most suitable packet for transmission, taking into account the approaching deadlines across the entire network.

Figure 3 shows a flow diagram of the D-NDNIoT router framework. This diagram illustrates the three major sections of the flow: interest packet preprocessing, data packet preprocessing, and packet selection for transmission.

1) Interest packet preprocessing: This phase starts with the reception of an interest packet, which is then stored in the interest buffer queue. At this stage, the NDN router checks if a matching data name is available in the CS. If there is a match, the corresponding data packet can be retrieved from the CS, and the subsequent data packet preprocessing phase (depicted in light blue in the flow diagram) is initiated. However, if the interest packet does not find a match in the CS, it is directed to the CDF scheduler. The CDF scheduler updates the remaining deadline of the interest packet based on the current clock time clk to account for the packet's deadline as follows properly:

$$D_{R_i} = d_i - clk \tag{9}$$

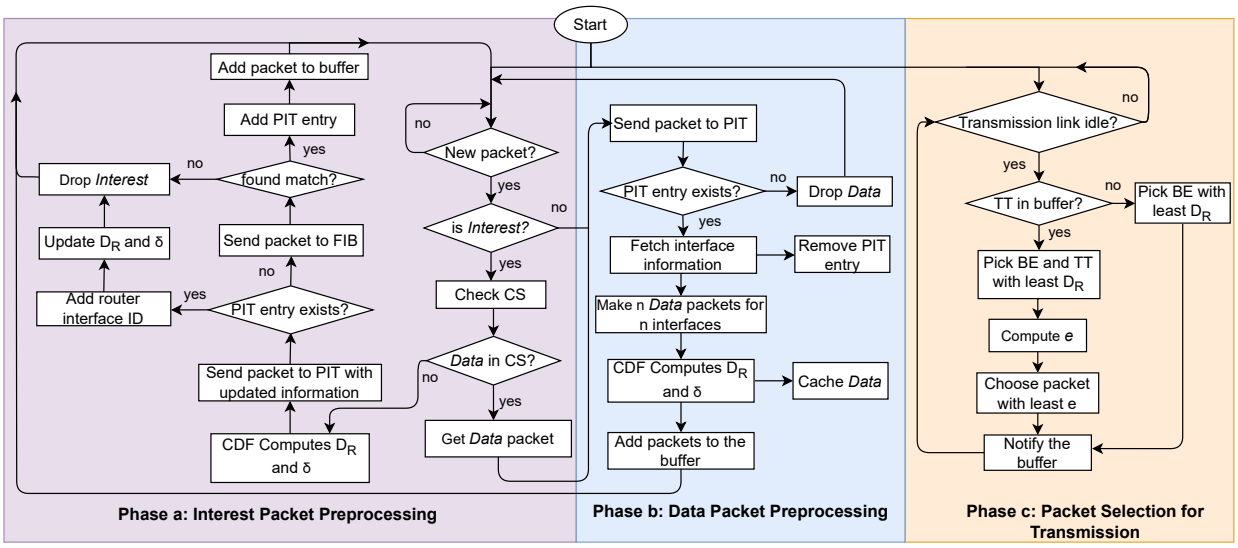

Fig. 3: Work-flow diagram of the modules in D-NDNoT router framework

Furthermore, the scheduler assigns a priority to the interest packet based on the remaining deadline. BE packets are assigned a priority value (δ) ranging from 2 to the buffer size (b) (i.e., $\delta \in [2, b]$), while real-time (TT) packets always have the highest priority value of 1.

The interest packet is then sent to the PIT to check if there is an existing entry requesting the same data. If an entry is found, the incoming interface is appended to the existing PIT entry. Additionally, if the interest packet (referred to as packet z) that created the entry still exists in the buffer, its remaining deadline is updated. This update prioritizes the packet with the closest deadline between packet z and the current interest packet i. The remaining deadline D_{R_z} is adjusted to be the minimum of D_{R_i} and D_{R_z}. Consequently, the updated remaining deadline affects the priority δ_z assigned to packet z. Finally, the interest packet i is discarded.

Conversely, if no matching entry is found in the PIT, the interest packet is forwarded to the FIB to determine the next hop for transmission. If a valid path is identified, indicating that information about the requested data node is available, an entry is created in the PIT, and the interest packet is stored in the buffer queue for subsequent transmission.

2) Data packet preprocessing: The process in this phase begins with the reception of a data packet by the NDN router. Upon receiving the data packet, the router forwards it to the PIT. If there is a pending interest for the corresponding data, the router attaches interface information to the data packet and transfers it to the CDF scheduler. In the CDF scheduler, n copies of the data packet are created, where n represents the total number of incoming interfaces. Each copy of the data packet undergoes the same computation procedure as in phase (a), which involves calculating the remaining deadline (D_R) and determining the priority (δ). This phase is denoted by the color purple in Figure 3. After the computation is performed for each copy, the packets are added to the data buffer queue, where they awaits transmission.

3) Packet selection for transmission: In this phase, the CDF scheduler selects a packet from the buffer for transmission when there is no traffic on the transmission channel. If the buffer does not contain any TT packets, the scheduler chooses a BE traffic based on the priority value. Otherwise, if there are TT packets in the buffer, the scheduler follows these steps for packet selection:

1) The scheduler picks a BE packet u_b with the highest priority and a TT packet u_t in First-In-First-Out order.
2) The final selection is based on a new factor called expected remaining time (e), which takes into account the propagation delay and the approaching deadline.

$$pkt = \begin{cases} u_b, & \longrightarrow (e_b \leq 0 \vee D_{R_b} < D_{R_t}) \wedge (e_t - T_{tr,u_b}) > 0 \\ u_t, & \text{otherwise} \end{cases} \tag{10}$$

where, pkt represents the selected packet for the transmission and T_{tr,u_b} represents the transmission time of the u_b packet.

Based on Eq. 10, the selected packet for transmission (pkt) is determined as follows:

If the subtraction of the T_{tr,u_b} from the e_t is greater than zero, indicating that the packet u_t will reach its final destination within the deadline even if u_b is transmitted before it, and one of the following conditions is met:

Condition 1. The expected remaining time of the BE packet is less than or equal to zero, indicating that the packet will exceed its deadline.

Condition 2. The remaining deadline of the BE packet is less than the TT packet's remaining deadline.

In such cases, u_b is selected for transmission. Otherwise, u_t is chosen.

The expected remaining time e_i is computed as:

$$e_i = D_{R_i} - T_{tr_i} \tag{11}$$

where, T_{tr_i} is the time required to transmit the packet i from the current node to the final destination.

C. Significance of the CDF Scheduler

The CDF scheduler acts as a schedulability protocol for D-NDNoT and is responsible for prioritizing packets, managing the PIT and FIB, and selecting packets for transmission based on their priority, remaining deadline, and expected remaining time. It plays a crucial role in ensuring timely and efficient packet delivery in the proposed approach.

To better understand the significance of the proposed CDF scheduler, we compare it with other scheduling algorithms commonly used for network traffic. Two types of scheduling algorithms are typically employed: fixed priority and dynamic priority scheduling algorithms [34]. Fixed priority algorithms, such as rate monotonic and deadline monotonic, prioritize traffic based on predetermined priorities and execute them accordingly. These algorithms work well when packet priorities are known in advance and remain constant. Dynamic priority algorithms, such as earliest deadline first (EDF), adjust priorities during runtime based on factors like deadlines and arrival time. In dynamic networks, dynamic algorithms perform better as they can adapt to unpredictable changes and prioritize critical traffic accordingly.

The EDF scheduler is commonly used in networks like TCP/IP and Ethernet, assigning priority based on absolute deadlines, where lower values indicate higher priority [35], [36]. However, the proposed scheduler differs from existing schedulers because it considers the entire network when assigning priorities. For example, if we have two packets, pkt_1 and pkt_2, with deadlines of 5 ms and 10 ms, respectively, an EDF scheduler would prioritize pkt_1 due to its lower absolute deadline. However, if pkt_1 only requires 2 ms to reach the final destination (e.g., next hop) while pkt_2 needs 9 ms (e.g., traveling further through the same next hop as pkt_1's destination), the EDF scheduler would cause pkt_2 to miss its deadline. In contrast, the CDF scheduler allows for more efficient and reliable network performance by considering the network's characteristics and calculating critical deadlines. In this case, the CDF scheduler prioritizes pkt_2 and ensures both pkt_2 and pkt_1 can be delivered within the deadline. The scheduler assumes knowledge of transmission times for communication channels along the route P_i, gathered through link state information in the NDN routers [33].

D. Worst-Case End-to-End Delay Analysis

We consider a TT packet as schedulable if its deadline falls within the worst-case end-to-end delay D^+. The worst-case transmission end-to-end delay of packet i, denoted as D_i^+, depends on the packet's transmission time and on the worst-case queuing delay w_i at the corresponding router buffer. We define $w_i(\Psi)$ as the time interval from when the packet i reaches the router buffer to when the packet is transmitted.

The queuing delay of a TT packet consists of several factors. In a highly mobile environment, consumer and producer mobility can influence packet loss and subsequent retransmissions, causing delays. However, we do not consider this delay while computing queuing delay since ongoing research efforts have addressed this specific issue comprehensively and provide seamless support for consumer or producer mobility without introducing a single point of failure issues, unnecessary interest packet loss, extra bandwidth consumption, or interest retransmission problems [37]–[39]. Nevertheless, a TT packet may encounter additional delays, such as *interference* delay and *blocking* delay.

1) Interference Delay: Upon arrival at the CDF scheduler, a TT packet must wait if there is ongoing transmission or processing of a BE or TT packet. This waiting time, referred to as the interference delay (γ) is measured as the total time that packet i had to wait due to the ongoing transmission. It can be calculated using the equation:

$$\gamma = (\beta + T_h) - \Psi_{tt} \tag{12}$$

where β represents the time when the ongoing transmission started, T_h is the time required to transmit the ongoing packet to the next hop, and Ψ_{tt} is the arrival time of the TT packet at the scheduler.

2) Blocking Delay: If a BE packet is chosen over a TT packet based on Equation 10, the TT traffic will experience a waiting period known as the blocking delay (ρ) until the selected BE packet finishes its transmission. Moreover, if multiple TT packets are in the buffer, the non-selected packets must wait until the chosen TT packet completes its transmission. The blocking delay can be defined as follows:

$$\rho = \sum_{x=1}^{n} T_{h,B_x} + \sum_{y=1}^{m} T_{h,T_y} \tag{13}$$

here, n and m represent the total numbers of selected BE and TT packets before transmitting the TT packet under analysis, respectively. T_{h,B_x} and T_{h,T_y} are the transmission times for the x-th BE packet and y-th TT packet, respectively.

By definition, the worst-case queuing delay of a TT packet i that arrived at Ψ_{tt} can be calculated as the sum of interference and blocking delays at each hop:

$$w_i(\Psi) = \sum_{z=1}^{N} (\gamma_z + \rho_z) \tag{14}$$

Here, N is the number of hops in the transmission path of packet i.

To determine the worst-case end-to-end delay D_i^+ for traffic i, we add the packet's transmission time along the route to the worst-case queuing delay:

$$D_i^+ = w_i(\Psi) + \sum_{z=1}^{N} T_{h,z} \tag{15}$$

The value of n significantly impacts the calculation of the blocking delay in the D^+. However, the proposed CDF scheduler adapts to the deadline of different TT packets by selecting the most appropriate value for n during scheduling, which can be either 0 or more, depending on how many BE packets can be sent without missing the TT deadline. This ensures consistent meeting of deadlines, thereby making the scheduler deterministic.

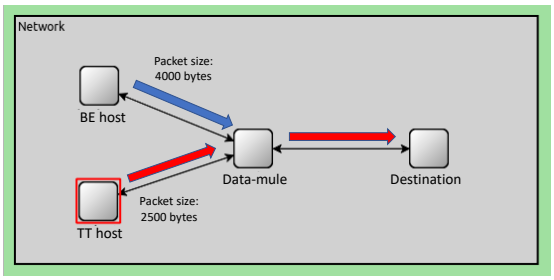

Fig. 4: Scenario 1: Exemplary test scenario of a converged network in OMNeT++ simulation environment

VI. Evaluation

This section presents a set of experiments to validate the effectiveness of the proposed approach and address two main questions: (i) Does introducing the CDF scheduler into NDN improve the delay incurred by TT traffic? and, more importantly, (ii) Does our approach guarantee that TT traffic will meet its deadline in the presence of overwhelming BE traffic?

To answer these questions, we have developed a network model that captures the essential characteristics of the system (see Section VI-A). In this network model, we consider factors such as network topology, traffic patterns, and the behavior of the CDF scheduler. Additionally, we define two performance metrics to evaluate the system's performance: a percentage increase in delay ($D^*\%$) and end-to-end time ($E2Et$). These metrics provide insights into the impact of the proposed D-NDNoT approach on TT traffic: using these metrics, we can visualize whether the TT traffic meet their deadlines. The findings of our experiments are discussed in Section VI-C, where we analyze the performance of the system under different scenarios and compare it against baseline approaches. We discuss the time complexity of our approach in Section VI-D, providing insights into the computational efficiency of the proposed CDF scheduler. Finally, we discuss the real-world feasibility of the proposed model in Section VI-E.

A. Experimental Setup and Network Model

We conducted a series of experiments using the OMNeT++ simulation environment, which allows for modeling and simulating the proposed approach. We present the simulation assessments of D-NDNoT in two distinct scenarios. These scenarios replicate IoV networks that handle communications between two domains: Automated Driver Assistance (ADAS) traffic, considered as TT flows, and multimedia or infotainment traffic, considered as BE flows. To streamline our analysis, we focus on simulating the behavior of interest packets, assuming that data packets exhibit similar performance characteristics.

In the first scenario, we consider a converged network scenario similar to the motivational example discussed in Section IV. As shown in Figure 4, the network consists of two traffic flows: TT and BE, generated by nodes A (BE host) and B (TT host), respectively. These flows share a bottleneck link between node C (Data-mule) and D (Destination). We intentionally set network parameters such that the bottleneck link is overwhelmed by the high frequency of BE packets. Specifically, the BE host sends 4000B-sized packets, while the TT host sends 2500B-sized packets at a fixed interval. The links between nodes A to C and B to C have a bandwidth of 3200Mbps, while the bottleneck link between nodes C and D has a bandwidth of 2500Mbps.

TABLE III: Summary of the Network Characteristics of Figure 4

Source	Destination	Transmission Interval	Traffic Type	Payload Size (bytes)
A	D	10 μs	BE	4000
B	D	[40, 50, 60] μs	TT	2500
Link		**Bandwidth (Mbps)**		
$A \rightarrow C$		3200		
$B \rightarrow C$		3200		
$C \rightarrow D$		2500		

We calculate the deadlines for each flow based on the specific network scenario used. For example, deadline for the TT traffic flow in this scenario is calculated to be 27 μs using Equation 4 (6.25 μs) and Equation 5 (8 μs). In both of the equations the processing delay is set to 0 μs since OMNet++ router does processing in real-time. In addition to that, we consider at most one interference or blocking delay along the path at data-mule router buffer (12.8 μs). The values for the equations are obtained using Table III. We explored this scenario by varying the TT traffic interval from 40 μs to 60 μs in steps of 10 μs, while keeping the interval of BE packets constant at 10 μs. Table III provides a summary of the network characteristics for this scenario.

To demonstrate the scalability of D-NDNoT, we extended the simulation scenario to a larger network with 30 NDN nodes. Figure 5 illustrates the extensive network, referred to as the complex scenario. Among these 30 nodes, four were randomly selected to release interest packets: node $N6$, node $N21$, node $N14$, and node $N26$. Node $N6$ and node $N21$ generate interest packets for TT data at intervals of 60 μs and 40 μs, respectively, while node $N14$ and node $N26$ generate interest packets for BE data at intervals of 10 μs each. These interest flows are labeled as $TTflow_1$, $TTflow_2$, $BEflow_1$, and $BEflow_2$ based on their requested packet types. Additionally, four nodes were randomly chosen as destinations for these flows: node $N11$, node $N10$, node $N18$, and node $N9$.

To evaluate the effectiveness of our proposed D-NDNoT approach, we deliberately selected the shortest paths between the requesters and destinations, along with specific network parameters to create a bottleneck for certain flows. The transmission link connecting node $N16$ and node $N17$ becomes the bottleneck for $TTflow_1$, $TTflow_2$, and $BEflow_1$. Similarly, the transmission link between node $N22$ and node $N16$ is overwhelmed by frequent BE packets from $BEflow_2$, aiming to interrupt $TTflow_2$. In this scenario, the estimated deadlines (D) for TT packets in $TTflow_1$ and $TTflow_2$ are 58.45 μs and 49.7 μs, respectively, considering at most one interference and one blocking delay. Table IV provides a summary of the network characteristics for this extended scenario.

B. Evaluation Metrics

We consider two performance metrics: the percentage increase in delay ($D^*\%$) and the end-to-end time ($E2Et$). These

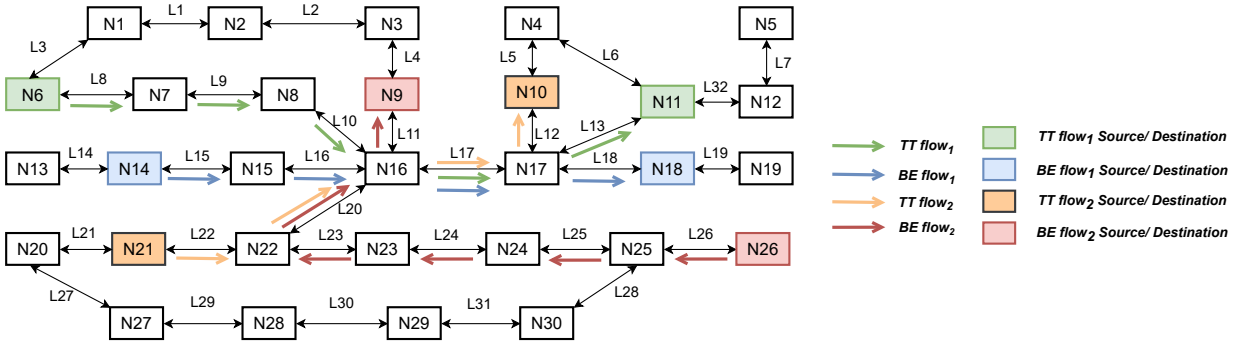

Fig. 5: Scenario 2: Illustration of a complex converged network of 30 nodes and the associated traffic flows

TABLE IV: Summary of the Network Characteristics of Figure 5

Source	Destination	Transmission Interval	Traffic Type	Payload Size (B)	Flow	Paths	Shortest Path
$N6$	$N11$	60 μs	TT	2500	$TTflow_1$	**P1:** $L8 \rightarrow L9 \rightarrow L10 \rightarrow L17 \rightarrow L13$ **(cost:** $33\mu s$**)** **P2:** $L3 \rightarrow L1 \rightarrow L2 \rightarrow L4 \rightarrow L11 \rightarrow L17 \rightarrow L13$ **(cost:** $60.5\mu s$**)** **P3:** $L3 \rightarrow L1 \rightarrow L2 \rightarrow L4 \rightarrow L11 \rightarrow L17 \rightarrow L12 \rightarrow L5 \rightarrow L6$ **(cost:** $80.5\mu s$**)**	P1
$N21$	$N10$	40 μs	TT	2000	$TTflow_2$	**P1:** $L22 \rightarrow L20 \rightarrow L17 \rightarrow L12$ **(cost:** $21.4\mu s$**)** **P2:** $L22 \rightarrow L20 \rightarrow L17 \rightarrow L13 \rightarrow L6 \rightarrow L5$ **(cost:** $37.4\mu s$**)**	P1
$N14$	$N18$	10 μs	BE	4000	$BEflow_1$	**P1:** $L15 \rightarrow L16 \rightarrow L17 \rightarrow L18$ **(cost:** $42.8\mu s$**)**	P1
$N26$	$N9$	10 μs	BE	3000	$BEflow_2$	**P1:** $L26 \rightarrow L25 \rightarrow L24 \rightarrow L23 \rightarrow L20 \rightarrow L11$ **(cost:** $45\mu s$**)** **P2:** $L26 \rightarrow L28 \rightarrow L31 \rightarrow L30 \rightarrow L29 \rightarrow L27 \rightarrow L21 \rightarrow L22 \rightarrow L20 \rightarrow L11$ **(cost:** $106.5\mu s$**)**	P1

Link	Bandwidth (Mbps)
$L8$, $L9$, $L10$, $L11$, $L12$, $L13$, $L15$, $L16$, $L18$, $L20$, $L22$, $L23$, $L24$, $L25$, $L26$	3200
$L1$, $L2$, $L3$, $L4$, $L5$, $L6$, $L7$, $L14$, $L19$, $L21$, $L27$, $L28$, $L29$, $L30$, $L31$, $L32$	2000
$L17$	2500

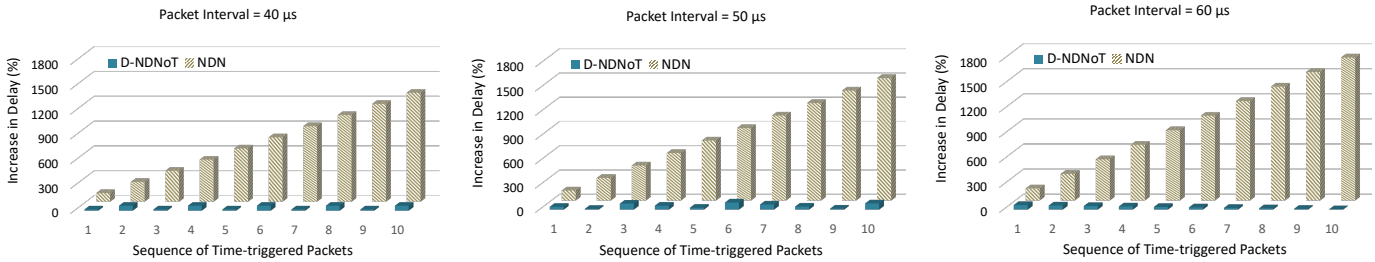

Fig. 6: Comparison of percentage increase in TT traffic delivery between NDN approaches with and without the CDF scheduler with varying TT intervals.

metrics allow us to assess the effectiveness of our approach and compare it with the estimated deadline D. $D^*\%$ measures the additional time required for delivering TT traffic compared to the theoretical delivery time that does not account for delays caused by propagation and queuing. It quantifies the improvement in TT traffic delay. The $E2Et$ represents the duration between the release time of a frame from the source node and its complete acknowledgment by the destination node. This metric provides an overall measure of the time taken for the entire transmission process.

C. Evaluation Results

The simulation results for the first scenario are presented in Figure 6 and Figure 7. The figures provide a visualization of the percentage increase in delay and $E2Et$ of TT packets for both approaches, where the x-axis represents the sequence of transmitted TT packets. Figure 6 compares the percentage increase in delay for TT packet source to destination delivery between the naive NDN approach (NDN without CDF scheduler), also referred to as the NDN approach, and the D-NDNoT approach. The delay in D-NDNoT is primarily caused by varying interference and blocking delay, while the naive approach encounters additional queuing delay caused by network congestion. Consequently, the figure shows that the delay increase in D-NDNoT is insignificant compared to that of the naive NDN approach. Figure 7 compares the $E2Et$ values of NDN approaches with and without the CDF scheduler for varying TT intervals. It demonstrates that D-NDNoT can still meet the TT traffic deadline (D) despite the slight increase in delay.

Figure 7a shows that the naive NDN approach exhibits a linear and significant increase in $E2Et$ for TT packets, exceeding the estimated deadline D. In contrast, the D-NDNoT approach ensures that the $E2Et$ of each TT packet remains within the deadline. However, since the proposed method does not

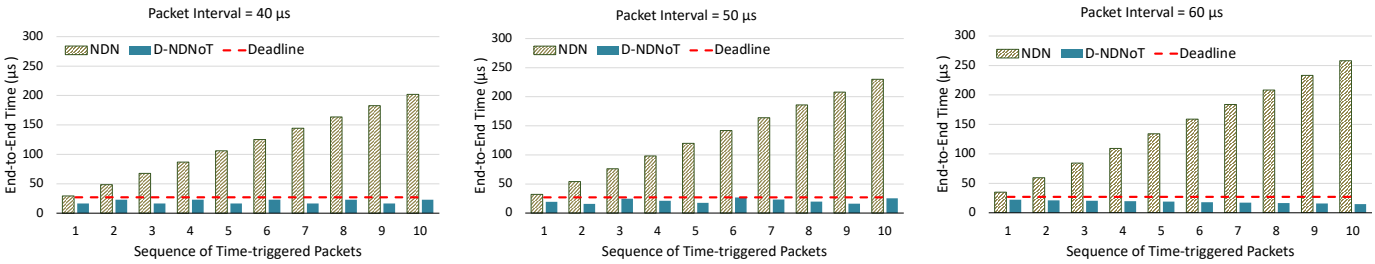

Fig. 7: Comparison between the end-to-end time of NDN approaches with and without the CDF scheduler with varying TT intervals.

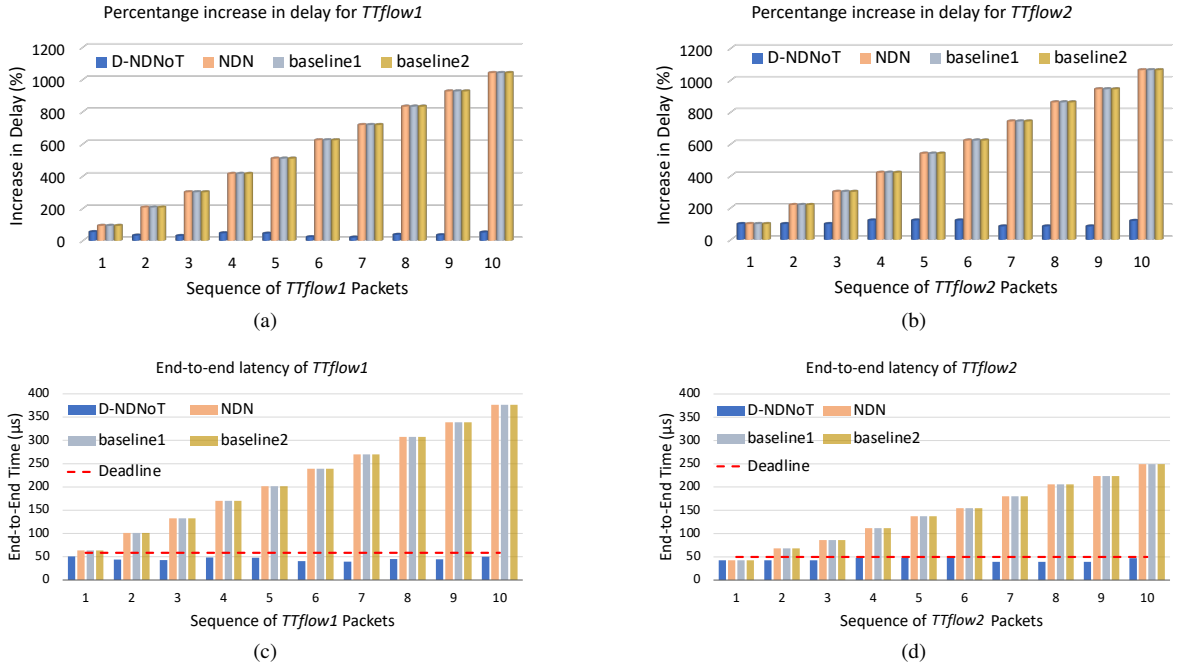

Fig. 8: Comparison of NDN approaches with and without the CDF scheduler based on: a) percentage increase in TT traffic delivery of $TTflow_1$, b) percentage increase in TT traffic delivery of $TTflow_2$, c) end-to-end time of TT packets of $TTflow_1$, and d) end-to-end time of TT packets of $TTflow_2$.

preempt the transmission of BE packets as soon as a TT packet arrives at the router, the TT packet may experience a delay due to varying interference and blocking delays. Nevertheless, it does not hinder the capacity of the proposed D-NDNoT to keep the experienced delay within the specific deadline. Also, when a new TT packet is given priority over a BE packet based on Eq. 10, the delay of the corresponding BE packet increases as it waits for the TT packet transmission. Despite the waiting time, D-NDNoT exhibits only around 15% increase in $E2Et$ for BE packets while achieving approximately 80% reduction for TT packets compared to naive NDN.

Figures 7b and 7c compare NDN and D-NDNoT approaches for TT packet intervals of 50 μs and 60 μs, respectively. As the interval increases, more BE packets overload the router buffer before the first TT packet arrives and between other TT packets. Consequently, the naive NDN approach experiences an increase in $E2Et$ for TT packets. However, the D-NDNoT approach remains unchanged by the increasing number of BE packets, ensuring that the $E2Et$ of TT packets stays within the deadline in every test scenario. Thus, the percentage decrease in $E2Et$ of TT packets continues to increase with longer intervals from 80% to 88%, while the increase in BE's $E2Et$ decreases from 15% to 10%.

Figure 8c and Figure 8d demonstrate a similar comparison of $E2Et$ between naive NDN and D-NDNoT approaches in the more complex scenario (Figure 5) with additional TT and BE traffic flows. In addition to comparing with the naive NDN, we benchmark against two most recent papers discussed in Sections III-A and III-B, referred to as $baseline_1$ [15] and $baseline_2$ [20]. $baseline_1$ opts for an alternative path during congestion, while $baseline_2$ attempts to retrieve all data packet fragments from intermediate nodes' content store using a single interest packet.

In the complex network scenario, the TT packets of $TTflow_1$ traverse a single bottleneck link ($L17$) connecting node $N16$ and node $N17$, while the TT packets of $TTflow_2$ pass through two bottleneck links, $L20$ and $L17$. Despite encountering frequent BE packets and a slight increase in the

percentage increase in delay ($D^*\%$), as depicted in Figure 8a and Figure 8b, both TT flows successfully reach their destinations within the specified deadlines (Figures 8c and 8d). Since the approach may have to prioritize TT packets over BE packets based on Eq. 10, the queuing delay of BE packets increases. Despite this queuing delay, the increase in $E2Et$ for BE packets is minor (around 20% for $BEflow_1$ and about 1% for $BEflow_2$) while it shows a significant decrease in $E2Et$ for TT packets (approximately 72% for $TTflow_1$ and around 60% for $TTflow_2$) for the proposed approach as opposed to the naive NDN. Furthermore, both the baseline approaches perform the same as the naive NDN approach in the considered network scenario.

In both scenarios (Figure 4 and Figure 5), we aimed to illustrate the worst-case caching and routing scenarios, where the interest packets travel toward the source due to cache misses in the intermediate nodes' content store, and there is no alternate path other than utilizing the congested bottleneck links. These are the reasons behind the performance degradation of the baseline approaches. However, despite these challenging situations in the considered network scenarios, the performance evaluation demonstrated that using the D-NDNoT approach guarantees that TT packets are delivered within the worst-case deadline without significantly affecting the BE flows and is adaptable to various scenarios. This emphasizes that leveraging the proposed scheduler alongside existing caching and congestion control mechanisms can further decrease packet delays.

D. Time Complexity Analysis

The functions performed by the D-NDNoT router buffer are insertion, selection, and deletion, and the time complexity for each operation is as follows:

The time to compute the remaining deadline as soon as a packet reaches the CDF scheduler is $O(1)$. The scheduler sends the packet to the router buffer to store according to the computed remaining deadline. The time complexity of storing a packet in the buffer depends on the specific data structure used. In this case, we consider the router buffer to be implemented as a priority queue. The time complexity for insertion in a priority queue can vary depending on the implementation. If a Fibonacci Heap is used as the underlying data structure, the worst-case time complexity for insertion is $O(1)$. When the transmission line is idle, the CDF scheduler accesses the router buffer and selects the packet with the least remaining deadline. The time complexity of selecting the packet also depends on the underlying data structure. If a Fibonacci Heap is used as the underlying data structure, the time complexity for selecting the minimum element is $O(1)$. After transmission, the selected packet needs to be deleted from the router buffer. If a Fibonacci Heap is used as the underlying data structure, the worst-case time complexity for deletion is $O(logn)$, where n is the number of packets in the buffer. However, since only the root node or a child of the root node needs to be deleted after transmission, we can consider the average-case time complexity for deletion, which is $O(1)$.

Therefore, the overall time complexity of the CDF scheduler, along with the buffer operations using a Fibonacci Heap is $O(1)$ for each packet in the router buffer.

E. Real-world Feasibility

In a real-world implementation, the latency of time-sensitive packets is significantly affected by cache-miss occurrences and congested networks. Therefore, this paper's simulation setup was meticulously designed to replicate the worst-case scenarios of real-world communication, such as the lack of in-network caching and congestion control, especially in the presence of a bottleneck link between the source and destination. The deliberate exclusion of in-network caching mirrors the constant cache-miss scenario in NDN networks, while the absence of congestion control reflects the conditions where there is a single path for packet forwarding without flow control. Despite these challenging conditions, our proposed approach effectively adapted and consistently met deadline requirements, demonstrating resilience despite the complexity introduced by scaling the network. This robustness highlights that in real-world deployment, where there will be efficient state-of-the-art (SOTA) in-network caching algorithms and effective SOTA congestion control mechanisms, the delivery latency will further reduce for the time-sensitive packets. However, our simulation setup does not account for packet retransmissions resulting from packet drops, usually caused by diverse channel conditions and poisoned contents. The network designers can leverage existing approaches that aim to address the packet drop issue while adopting the proposed approach in a real-world scenario [40]–[44].

VII. Conclusion

This work introduces D-NDNoT, a protocol-level integration of schedulability into NDN to support real-time scheduling of time-sensitive traffic, making it deadline aware. Through extensive simulations in the OMNeT++ environment, we have demonstrated the effectiveness of our approach in meeting deadlines by reducing the end-to-end delay of time-triggered packets. The evaluation results indicate that D-NDNoT shows promising performance in delivering time-sensitive packets within their specified deadlines. The proposed approach demonstrates the adaptability of the NDN architecture for communication in IoT applications with time constraints.

Future research directions include considering traffic patterns that do not strictly follow the time-triggered arrival pattern but still have real-time nature and deadlines. It is important to evaluate the performance of D-NDNoT under different network conditions and topologies, using various metrics to gain a comprehensive understanding of its capabilities. Additionally, incorporating real-world datasets and scenarios will further enhance the validity and practicality of the proposed approach. Furthermore, we envision expanding the D-NDNoT approach to enable dynamic resource allocation for transmitting multiple time-triggered packets simultaneously when conflicts occur to ensure optimal resource utilization and effective packet transmission in complex network environments and evaluation to include different NDN-enabled IoT application scenarios [45].

REFERENCES

[1] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, E. Uzun, B. Zhang, G. Tsudik, kc claffy, D. Krioukov, D. Massey, C. Papadopoulos, T. Abdelzaher, L. Wang, P. Crowley, and E. Yeh, "Named data networking (ndn) project, 2010 – 2011 progress summary," *Named Data Networking*, 2012.

[2] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," *ACM SIGCOMM computer communication review*, vol. 42, no. 3, pp. 62–67, 2012.

[3] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S. A. Ashraf, B. Almeroth, J. Voigt, I. Riedel *et al.*, "Latency critical iot applications in 5g: Perspective on the design of radio interface and network architecture," *IEEE Communications Magazine*, 2017.

[4] P. Gusev and J. Burke, "Ndn-rtc: Real-time videoconferencing over named data networking," in *Proceedings of the 2nd ACM Conference on Information-Centric Networking*, 2015, pp. 117–126.

[5] K. Matsuzono and H. Asaeda, "Nrts: Content name-based real-time streaming," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2016, pp. 537–543.

[6] Z. Zhang, E. Lu, Y. Li, L. Zhang, T. Yu, D. Pesavento, J. Shi, and L. Benmohamed, "Ndnot: a framework for named data network of things," in *Proceedings of the 5th ACM Conference on Information-Centric Networking*, 2018, pp. 200–201.

[7] Y. Yu, *Usable security for named data networking*. University of California, Los Angeles, 2016.

[8] M. A. Hail, "Iot-ndn: An iot architecture via named data netwoking (ndn)," in *2019 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*. IEEE, 2019.

[9] H. B. Abraham and P. Crowley, "In-network retransmissions in named data networking," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, 2016, pp. 209–210.

[10] E. Kalogeiton, D. Iapello, and T. Braun, "A geographical aware routing protocol using directional antennas for ndn-vanets," in *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. IEEE, 2019.

[11] R. Hou, S. Zhou, M. Cui, L. Zhou, D. Zeng, J. Luo, and M. Ma, "Data forwarding scheme for vehicle tracking in named data networking," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 7, 2021.

[12] X. Zhang, R. Li, and H. Zhao, "Neighbor-aware based forwarding strategy in ndn-manet," in *2017 11th IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID)*. IEEE, 2017.

[13] M. Chowdhury, J. A. Khan, and L. Wang, "Smart forwarding in ndn vanet," in *Proceedings of the 6th ACM Conference on Information-Centric Networking*, 2019, pp. 153–154.

[14] S. N. S. Hashemi and A. Bohlooli, "3cp: Coordinated congestion control protocol for named-data networking," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3918–3932, 2021.

[15] N. Omitsu and T. Shigeyasu, "A multipath routing algorithm avoiding congested links according to the link usage ratios on ndn," in *International Conference on Network-Based Information Systems*. Springer, 2023, pp. 1–11.

[16] M. Yu, R. Li, Y. Liu, and Y. Li, "A caching strategy based on content popularity and router level for ndn," in *2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC)*. IEEE, 2017, pp. 195–198.

[17] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos *et al.*, "Named data networking (ndn) project," *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, vol. 157, p. 158, 2010.

[18] Y. Wang, H. Dai, J. Jiang, K. He, W. Meng, and B. Liu, "Parallel name lookup for named data networking," in *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011*. IEEE, 2011.

[19] Y. Wang, Y. Zu, T. Zhang, K. Peng, Q. Dong, B. Liu, W. Meng, H. Dai, X. Tian, Z. Xu *et al.*, "Wire speed name lookup: A gpu-based approach," in *10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13)*, 2013.

[20] D. Qu, J. Wu, J. Zhang, C. Gao, H. Shen, and K. Li, "Efficient congestion control scheme based on caching strategy in ndn," *Journal of Network and Computer Applications*, vol. 216, p. 103651, 2023.

[21] I. . W. Group, "Time-sensitive networking task group," 2017. [Online]. Available: https://www.ieee802.org/1/pages/tsn.html

[22] J. Falk, D. Hellmanns, B. Carabelli, N. Nayak, F. Dürr, S. Kehrer, and K. Rothermel, "Nesting: Simulating ieee time-sensitive networking (tsn) in omnet++," in *2019 International Conference on Networked Systems (NetSys)*. IEEE, 2019, pp. 1–8.

[23] M. Böhm, J. Ohms, M. Kumar, O. Gebauer, and D. Wermser, "Time-sensitive software-defined networking: a unified control-plane for tsn and sdn," in *Mobile Communication-Technologies and Applications; 24. ITG-Symposium*. VDE, 2019, pp. 1–6.

[24] D. Wang and T. Sun, "Leveraging 5g tsn in v2x communication for cloud vehicle," in *2020 IEEE International Conference on Edge Computing (EDGE)*. IEEE, 2020, pp. 106–110.

[25] P. Kehl, J. Ansari, M. H. Jafari, P. Becker, J. Sachs, N. König, A. Göppert, and R. H. Schmitt, "Prototype of 5g integrated with tsn for edge-controlled mobile robotics," *Electronics*, 2022.

[26] A. H. Nagaraj, M. P. Tahiliani, D. Tandur, and H. Satheesh, "Leveraging named data networking for industrial automation: Opportunities and challenges," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2020, pp. 1–6.

[27] A. H. Nagaraj, B. Kataria, A. Sohoni, M. P. Tahiliani, D. Tandur, and H. Satheesh, "On the importance of traffic control subsystem in icn-based industrial networks," in *2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. IEEE.

[28] A. Abrar, A. S. C. M. Arif, and K. M. Zaini, "A systematic analysis and review on producer mobility management in named data networks: Research background and challenges," *Alexandria Engineering Journal*, vol. 69, pp. 785–808, 2023.

[29] S. Vusirikala, S. Mastorakis, A. Afanasyev, and L. Zhang, "Hop-by-hop best effort link layer reliability in named data networking," *NDN, Technical Report NDN-0041*, 2016.

[30] Y. Wang, K. He, H. Dai, W. Meng, J. Jiang, B. Liu, and Y. Chen, "Scalable name lookup in ndn using effective name component encoding," in *2012 IEEE 32nd International Conference on Distributed Computing Systems*. IEEE, 2012.

[31] "Ndn packet format specification v0.3- interest packet," *Named Data Networking*, 2023. [Online]. Available: https://docs.named-data.net/NDN-packet-spec/current/interest.html

[32] "Ndn packet format specification v0.3- data packet," *Named Data Networking*, 2023. [Online]. Available: https://docs.named-data.net/NDN-packet-spec/current/data.html

[33] L. Wang, A. Hoque, C. Yi, A. Alyyan, and B. Zhang, "Ospfn: An ospf based routing protocol for named data networking," Technical Report NDN-0003, Tech. Rep., 2012.

[34] R. Bikram, "Fixed – priority vs. dynamic – priority algorithms," *BenchPartner*, 2022. [Online]. Available: https://benchpartner.com/fixed-priority-vs-dynamic-priority-algorithms

[35] H. Hoang, M. Jonsson, U. Hagstrom, and A. Kallerdahl, "Switched real-time ethernet with earliest deadline first scheduling protocols and traffic handling," in *Proceedings 16th International Parallel and Distributed Processing Symposium*. IEEE, 2002, pp. 6–pp.

[36] G. Patti, L. L. Bello, and L. Leonardi, "Deadline-aware online scheduling of tsn flows for automotive applications," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 4, April 2022.

[37] A. Abrar, A. S. C. M. Arif, and K. M. Zaini, "A mobility mechanism to manage producer mobility in named data networking," in *2022 IEEE Region 10 Symposium (TENSYMP)*. IEEE, 2022, pp. 1–6.

[38] M. Gohar, N. Khan, A. Ahmad, M. Najam-Ul-Islam, S. Sarwar, S.-J. Koh *et al.*, "Cluster-based device mobility management in named data networking for vehicular networks," *Mobile Information Systems*, 2018.

[39] Z. Yan, Y.-J. Park, Y.-B. Leau, L. Ren-Ting, and R. Hassan, "Hybrid network mobility support in named data networking," in *2020 International Conference on Information Networking (ICOIN)*. IEEE, 2020.

[40] A. Anjum and H. Olufowobi, "Towards mitigating blackhole attack in ndn-enabled iot," in *2023 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 2023, pp. 1–6.

[41] T. Mick, R. Tourani, and S. Misra, "Laser: Lightweight authentication and secured routing for ndn iot in smart cities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 755–764, 2017.

[42] S. DiBenedetto and C. Papadopoulos, "Mitigating poisoned content with forwarding strategy," in *2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2016, pp. 164–169.

[43] K. Zhu, Z. Chen, W. Yan, and L. Zhang, "Security attacks in named data networking of things and a blockchain solution," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4733–4741, 2018.

[44] N. Yang, K. Chen, and M. Wang, "Smartdetour: Defending blackhole and content poisoning attacks in iot ndn networks," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12 119–12 136, 2021.

[45] A. Anjum, P. Agbaje, A. Mitra, E. Oseghale, E. Nwafor, and H. Olufowobi, "Towards named data networking technology: Emerging applications, use cases, and challenges for secure data communication," *Future Generation Computer Systems*, vol. 151, pp. 12–31, 2024.