# CIC-SIoT: Clean-Slate Information-Centric Software-Defined Content Discovery and Distribution for Internet-of-Things

Md Monjurul Karim, Kashif Sharif, Sujit Biswas, Zohaib Latif, Qiang Qu, and Fan Li

***Abstract*—The rapid expansion of the Internet of Things (IoT) introduces critical challenges in scalability, mobility, and security, particularly in large-scale deployments. While Information-Centric Networking (ICN) addresses these by enhancing content mobility, multipath support, and edge-embedded caching with inherent security features, it faces limitations in handling large heterogeneous environments due to its in-network caching and content-based forwarding strategies. Software-Defined Networking (SDN) complements ICN by employing a centralized controller to intelligently orchestrate content caching and forwarding, yet struggles with the efficient allocation and acquisition of content across expansive IoT systems. In response to these challenges, we propose CIC-SIoT, a novel Information-Centric Software-Defined Networking (IC-SDN) solution, designed to optimize the ICN-IoT framework. Our solution incorporates specialized algorithms for controllers, consumers, producers, and ICN nodes. These algorithms improve content forwarding decisions by moving beyond the traditional reliance on the Forwarding Information Base (FIB) and instead utilizing the Pending Interest Table (PIT) to efficiently manage and distribute content. Validated through ndnSIM and MATLAB simulations, CIC-SIoT achieves substantial performance enhancements, including an 80% increase in throughput, a 34% reduction in latency, and a 25% savings in bandwidth. Additionally, it reduces packet loss by 67% and communication overhead by 66%, compared to existing solutions. These results underscore the framework's ability to significantly improve the efficiency and scalability of content distribution in IoT environments, highlighting its robustness and adaptability in addressing the complex dynamics of modern networked systems.**

***Index Terms*—Software-Defined Networking, Information-Centric Networking, Internet-of-Things, TLV, Pending Interest Table.**

## I. Introduction

Emerging technologies such as the Internet of Things (IoT), Content Delivery Network (CDN), big data, and blockchain are heavily dependent on fifth-generation (5G) cellular networks [1]. This dependence necessitates the rapid distribution of vast amounts of content to multiple locations within milliseconds. Consequently, network operators face significant challenges in balancing available bandwidth, Quality-of-Service (QoS), and Quality-of-Experience (QoE) [2]. For instance, ensuring scalability, mobility, and security in distributed IoT networks becomes a complex task for operators, especially when these factors need to be addressed concurrently and on the same scale [3]. Software-Defined Networking (SDN) has emerged as a promising solution to address some of the aforementioned challenges, offering programmability and virtualization of data plane entities such as routers, switches, and WiFi access points. This is achieved by separating the data plane from the control logic [4]. However, while SDN presents numerous advantages, it is not without its drawbacks. Specifically, due to the stateless behavior of data plane devices and the single point of failure inherent in SDN controllers, it faces challenges related to scalability, resiliency, awareness, abstraction, and automation tradeoffs [5].

Information-Centric Networking (ICN) emerges as a new paradigm of communication model to tackle some of the issues faced by SDN in a heterogeneous IoT network by offering content mobility with access-agnostic multipath support, edge-embedded caching capabilities, and built-in security [6]. ICN adopts a communication principle that relies on data requests and retrieval through multiple nodes, routers, or forwarding instances capable of stateful forwarding. ICN routers are most influential in offering in-network caching functionalities and dictating named-based stateful forwarding among multiple consumers and producers. Content-Centric Networking (CCN) [7] and Named Data Networking (NDN) [8] are two of the real-world ICN implementations extensively used in academia and industry. Despite these advancements, the performance of ICN routers suffers significantly in large-scale networks [9]–[11]. For example, Drescher et al. [9] highlight the direct impact of PIT occupancy on network congestion by suggesting that the scalability of ICN routers is inherently tied to their ability to manage the growing volume of interest packets in large-scale deployments. On the other hand, Liu et al. [10] propose enhancements to content router architectures, focusing on distributed caching and modular design to better accommodate the scaling demands of ICN routers. Moreover, the increasing number of cache hits in ICN routers introduces overflow in FIB and PIT due to their limited table size, impacting the ICN routers' request and retrieval efficiency [11]. In

This work was supported in part by the Beijing Natural Science Foundation under Grant IS23056. The work of Fan Li is partially supported by the NSFC (No. 62372045). *(Corresponding author: K. Sharif.)*

M.M. Karim is with School of Computer Science and Technology, Beijing Institute of Technology, Beijing, and Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China (email: mkarim@bit.edu.cn, karim@siat.ac.cn). K. Sharif and F. Li are with School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China (e-mail: {kashif,fli}@bit.edu.cn). S. Biswas is with FinTech Department of Computer Science, School of Science and Technology, City, University of London, United Kingdom (email: sujit.biswas@city.ac.uk). Z. Latif is with Department of Computer Science, School of Engineering and Digital Sciences, Nazarbayev University, Astana, Kazakhstan (email: latif.zohaib@nu.edu.kz) Q. Qu is with Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China (email: qiang@siat.ac.cn).

Digital Object Identifier 10.1109/XXX.2023.XXXXXXX

this context, an alternative approach is necessary to overcome the inherent limitations of ICN, particularly the scalability challenges and the complexity of managing dynamic network conditions in large-scale IoT environments. The impact of PIT on scalability is also highlighted in [12]–[15].

The Information-Centric Software-Defined Networking (IC-SDN) model addresses some of the above-mentioned challenges by embedding an SDN control logic within ICN-capable routers. This integration facilitates stateful routing and enhances the efficiency of flow-based data forwarding and content caching [16]. Content distribution in IC-SDN is managed by a centralized controller that facilitates named-based intelligent content discovery and distribution within a large-scale IoT network [17]. The IC-SDN controller uses a content management module to extract content names from requested packets and to insert new entries into the ICN routers. Unlike traditional ICN/NDN scenarios, ICN routers in IC-SDN act as stateless devices, simply forwarding content requests from various consumers to the controller. This integration streamlines control over the content flow but also introduces challenges, particularly within the IoT paradigm. These challenges include managing the substantial volume of data from IoT devices, ensuring their real-time processing and responsiveness, and securing the data during transmission. The diverse range of IoT devices and their communication protocols requires IC-SDN to adopt adaptable and flexible routing strategies. Moreover, the stateless routers in IC-SDN do not make autonomous caching decisions or optimize data flow based on local conditions, leading to potential inefficiencies in network resource utilization. Such architecture has difficulty in balancing the network load, often leading to congestion and increased latency. The centralized controller, overloaded with content requests, faces the risk of becoming a bottleneck, thereby reducing the network's overall throughput. Addressing these issues requires innovative controller designs, improved algorithms for content discovery and caching, and robust security mechanisms for ICN-IoT paradigm.

The majority of ICN-IoT solutions face scalability challenges as the number of contents significantly exceeds the capacities of diverse IoT devices and applications [18]. For example, the hierarchical namespacing of content and the in-memory size of ICN routers necessitate efficient naming and optimization solutions. These are required to alleviate transmission delays and storage inefficiencies. Besides, standardizing northbound APIs is crucial for improving communication and stimulating innovation in software without hardware modifications. Specifically, developing RESTful APIs that meet precise constraints and robust authorization mechanisms is essential for enhancing service portability and interoperability [19]. Due to dynamic mobility between producers and consumers, applying machine learning (ML) techniques, such as deep learning, is essential to enable intelligent content forwarding and caching strategies [20], [21]. This introduces further complexities due to their heterogeneous composition and fluctuating channels, necessitating advanced and integrated approaches using ICN, SDN, and network function virtualization (NFV). Caching in ICN-IoT networks faces challenges in meeting low latency, resource utilization, and QoS requirements of heterogeneous applications [22]. Similarly, the repeated caching of popular content for user groups with shared interests requires intelligent and scalable solutions. Furthermore, network slicing via NFV facilitates parallel handling of tasks, necessitating integration with dynamic caching strategies and content-centric routing to adapt to dynamic IoT conditions [23]. Hence, existing solutions must leverage these technologies to advance the efficiency of ICN-IoT. In addition, more research is needed into consolidated frameworks that jointly optimize caching, routing, and network virtualization in accordance with application-specific demands. This integration is pivotal to the progression of IC-SDN efficiency and functionality within the context of IoT networks.

To address these issues, a communication module is proposed that incorporates an IC-SDN controller and ICN routers to enable efficient content management and scalable forwarding in the softwarized clean-slate ICN-IoT system. Specifically, the IC-SDN controller is designed to collect essential information from ICN routers during the bootstrapping phase. This information includes the network topology, the number of nodes (e.g., consumer, ICN router, producer), and path-related metrics. The controller is also configured to obtain available content names from each node, enabling effective distribution and management throughout the network. In response to new content requests, consumers are instructed on how to establish connections with the ICN router, typically providing a link to the producer. Subsequently, the controller calculates the optimal path and installs the necessary flows into the corresponding ICN routers to retrieve the requested content. In essence, our work addresses the challenge of processing multiple content requests simultaneously, which affect the controller's performance in terms of flow processing time, average delay, and response rate.

The main contributions are summarized as follows:

- A novel ICN packet structure is introduced, which features modified TLVs to facilitate IC-SDN communication within IoT systems. More specifically, custom-defined TLV components have been developed to create new types of request (*Req_Pkt*) and response (*Res_Pkt*) packets. This packet structure orchestrates scalable inter-layer communication between participating nodes to enhance content delivery efficiency and reduce communication overhead.
- Graph theory is applied to the proposed IC-SDN controller to achieve a global view of the network topology and optimize content distribution within the ICN system. This enables the controller to identify the most efficient routes for data flow and guide various nodes, such as consumers (e.g., IoT devices), producers (e.g., remote servers, data centers), and ICN routers.
- A PIT-centric forwarding mechanism is proposed to address the overhead limitations of existing IC-SDN architectures. This mechanism aims to improve content retrieval times as the number of requests and content volume grow. By dynamically assigning caching responsibilities to specific intermediate ICN routers based on the initial consumer request, the proposed approach promotes a scalable and adaptable content distribution and

management system.
- The performance of the proposed solution is evaluated through simulation-based studies and numerical analysis. These evaluations demonstrate that the novel PIT-centric forwarding mechanism significantly reduces forwarding overhead. Additionally, the logical decision-making capabilities of the controller enhance dissemination between the consumer and producer, regardless of their placement within the system.

The remainder of the paper is organized as follows. Section II discusses the related work highlighting the motivation behind this study while Section III presents the layered architecture of the proposed CIC-SIoT solution alongside its fundamental components. Besides, Section IV introduces the implementation and working principle of CIC-SIoT, including packet structure, workflow of the participating nodes and their communication principle. Furthermore, Section V discusses the proof-of-concept implementation and performance evaluation against the baseline solutions. Finally, Section VI concludes the paper recalling the primary contributions and hinting the future work.

## II. Related Work

The ongoing research shows that the inherent benefits of integrating ICN in a softwarized network make it a suitable networking model for various emerging technologies, such as the IoT, Internet of Vehicles (IoV), and 5G [17], [22], [24]. Previous works have focused on the integration of ICN and SDN, aiming to merge the solutions for multipath routing, cooperative caching, flow processing, and holistic network management. These innovations emphasize the strategic overlay of control and caching mechanisms to enhance data delivery and network management.

In terms of multimedia transmission optimization over 5G networks, Zhang and Zhu et al. [25] propose an integrated solution that merges ICN for optimal in-network caching, NFV for abstracting physical infrastructures, and SDN for dynamic resource allocation to ensure quality of service in wireless networks. Although, the approach improves statistical delay-bounded QoS for multimedia big data services, the reliance on extensive virtualization introduces complexities in deployment and scaling, particularly in environments with variable network conditions. Likewise, Amadeo et al. [26] focus on the orchestration of computing services in edge networks. The authors leverage SDN's centralized intelligence for service allocation and NDN's adaptive forwarding and caching to manage services by name. Their framework excels in reducing service provisioning delays by integrating network and computing resource management. Nevertheless, the intensive requirements for centralized control impact scalability and adaptability in highly distributed networks.

Regarding traffic engineering (TE) [27] optimization, Zhang et al. [28] apply deep learning (e.g., Deep Reinforcement Learning) into an IC-SDN scenario to enhance content-aware network management. While their demonstration shows significant improvements in network throughput and balance, the practical application of AI techniques in TE poses challenges due to the inherent complexity and dynamic nature of real-world networks. On the other hand, Benedetti et al. [29] present an integration of ICN, multi-access edge computing (MEC) [30]), and SDN to improve the management of mobile networks, particularly targeting the challenges of mobile consumer connectivity. The authors propose a protocol architecture to reduce communication overhead and increases network efficiency. However, the integration complexity and heavy reliance on advanced, coordinated functionalities across different network layers limit rapid deployment and flexibility.

As for adaptive caching in ICN-IoT, Sharif et al. [31] propose a sliding window-based model and a neural prediction module for dynamically adjusting to changes in data popularity. This approach significantly reduces energy consumption and response times. A limitation of their work is the assumption of consistent and accurate popularity predictions, heavily depends on the accuracy and timeliness of the popularity predictions. This presents a challenge in highly dynamic IoT environments. In addition, Khalid et al. [32] address the challenges of mobile adhoc networks (MANETs) in the context of NDN and SDN. The authors propose a routing protocol that considers node mobility and energy levels to improve communication reliability and efficiency. Although, their result show reduced retransmissions and enhanced data retrieval times, the reliance on a single controller introduces bottlenecks and limits scalability in larger network setups.

In the context of IC-SDN applicability in smart city environment, Demiroglou et al. [33] explore an adaptive multi-protocol system to maintain low latency and high reliability by intelligently switching between NDN, delay-tolerant network (DTN), and a combined scheme based on network conditions. This innovative approach ensures robust performance under varying urban network conditions. However, it faces challenges in standardization and interoperability across different network protocols. Furthermore, Anjum et al. [34] tackle the deterministic delivery challenges in NDN for IoT by integrating a deadline-aware schedulability algorithm. They significantly enhance the timeliness of critical data delivery. While the specific focus on deadline-aware scheduling is advantageous for time-sensitive applications, broader application requires more generalized network management strategies to handle diverse IoT traffic patterns effectively.

Several solutions have been proposed to enhance the integration of SDN and ICN, focusing on the modification or extension of existing packet structures, protocols, and architectures. For example, Sun et al. [36] introduce an SD-CCN solution with an OpenFlow switch, a CCN router, and an extended SDN controller on top of the existing NOX controller [37] to process CCNx [38] packets using custom TLV match fields. The approach enhances multipath routing, traffic management, and content-based forwarding but lacks evaluation of flow processing efficiency and content caching. Besides, Son et al. [39] propose a forwarding solution with distributed SDN controllers managing PIT and FIB functionalities from ICN, improving content awareness by modifying packets. However, the solution lacks a clear approach for content caching and distribution. Moreover, Gao et al. [40] focus on scalable forwarding and caching using multiple controllers in a multi-

Table I: Comparison between related works

| References | ICN/NDN Enhancements | | SDN Controller Enhancements | | Inter-domain Content Distribution | Source code Availability |
|---|---|---|---|---|---|---|
| | Clean-Slate Design | Packet Modification | Multi-path Routing | Caching Awareness | | |
| [25] | × | × | ✓ | ✓ | ✓ | × |
| [26] | × | ✓ | ✓ | × | × | × |
| [28] | ✓ | × | ✓ | ✓ | × | × |
| [29] | × | × | ✓ | × | ✓ | ✓ |
| [31] | × | × | × | ✓ | ✓ | × |
| [32] | ✓ | ✓ | ✓ | × | × | × |
| [33] | × | × | ✓ | × | ✓ | × |
| [35] | × | × | ✓ | ✓ | × | × |
| Our Work | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

domain environment. While handling interactions through essential services, this solution overlooks content caching prioritization and new content request allocation, potentially leading to slower response rates. Likewise, Torres et al. [41] implement a centralized controller for routing decisions and reducing ICN routers' storage consumption, but the scheme suffers from increased hand-off latency during high flow rates. Meanwhile, Jmal et al. [42] incorporate OpenFlow switches into the content-centric system using controller-based forwarding and caching with CCNx and BGP principles. Although performance evaluation is initiated, the integration principle for BGP-based routing and caching coordination is not fully explained. Additionally, Lv et al. [43] address scalability by segregating ICN into clusters based on content type similarity and assigning distributed controllers for content tasks. The solution aims to improve forwarding flexibility and content retrieval but lacks details on load balancing, QoS support, and controller interaction. Furthermore, Chen et al. [35] integrate SDN with ICN as an overlay to tackle cooperative caching, bandwidth, and resource allocation in large distributed environments. The approach, involving SDN controllers, OpenFlow switches, and ICN routers, shows promise with simulated results highlighting minimal energy consumption and efficient content aggregation. However, it remains theoretical with limited practical details.

Regarding ICN-IoT deployments in Wireless Sensor Networks (WSN) scenarios, existing solutions face challenges such as data dissemination efficiency, reliability, and scalability. For example, CCIC-WSN [44] faces limitations in scalability and flexibility due to its reliance on a static cluster hierarchy. This structure leads to bottlenecks at cluster heads, especially in dynamic environments characterized by frequent node mobility and fluctuating data demands. Similarly, CIDF-WSN's [45] reliance on the Neighbor Information Base (NFIB) for next-hop selection becomes complex and resource-intensive as the network density increases. This is because NFIB requires maintaining a comprehensive list of virtual faces with associated costs for all nodes within the transmission range. This task grows increasingly demanding regarding computational and storage resources in densely populated networks.

Our proposed solution addresses these limitations by adopting a multi-layered, distributed control mechanism that significantly enhances network scalability and flexibility, enabling dynamic adaptation to varying network conditions and demands. Unlike the existing works, our architecture is designed to efficiently manage single- and multi-interface environments, ensuring optimal performance across a broader range of IoT scenarios. While many studies have explored packet structures, our approach emphasizes refining these structures to enhance the packet identification and decision-making capabilities of the controller. Furthermore, the proposed controller operates using distributed logic, ensuring optimal data routing even while being physically centralized. We have introduced a new packet format aimed at improving communication scalability and efficiency, thereby addressing challenges that prior research has either neglected or only touched upon briefly. In addition, in accordance with the specifications outlined in RFC 9138 [46], this study systematically tackles the challenges associated with name resolution in the context of system scalability, manageability, deployment considerations, and fault tolerance capabilities. Table I shows a comparison between this work and the state-of-the-art solutions for the listed challenges.

## III. System Architecture

This section presents a technical overview of the proposed architecture alongside the necessary components and their workflow. First, we present a brief overview of the proposed architecture that focuses on the softwarization of the information-centric clean-slate packet transmission principle. Then, a detailed overview of the proposed packet modification is given, along with the impact of the packet structure on the overall system. Finally, we present a technical overview of each component of the proposed architecture.

### *A. Layered Overview of the Architecture*

In a traditional SDN system, the controller communicates with forwarding devices. However, in a typical ICN, there is no central or distributed control logic to manage participating ICN routers [12]. Instead, underlying ICN routers forward and cache desired content in a stateful manner based on the consumer's intention, unlike the stateless communication between data-plane devices in SDN [17]. The proposed solution incorporates both principles of SDN and ICN to forward and cache multiple content chunks based on specific instructions from the control layer. Specific flow instructions are determined by the service type, depending on the application or service that the consumer requests from the control layer. The layered

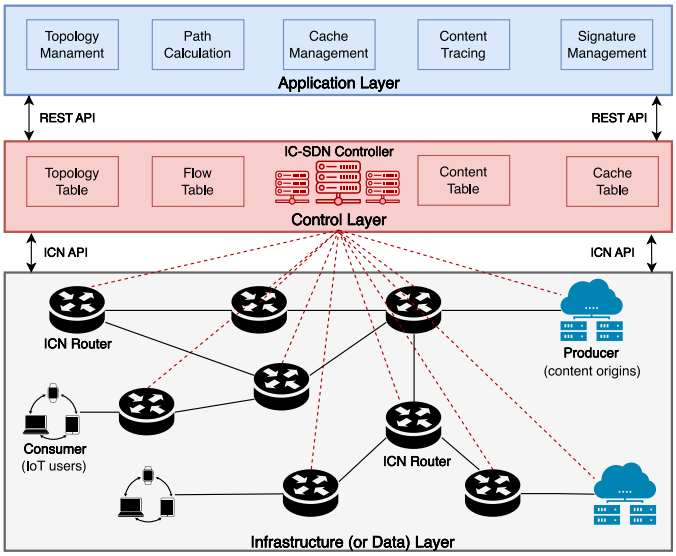



Figure 1: Proposed system architecture.

architecture of the proposed scheme is shown in Figure 1, which is divided into application, control, and infrastructure (or data) layer. The architecture employs two types of application programming interfaces (APIs) – ICN API [47] and REST [19]. The ICN API enables the controller to communicate with other ICN routers in the infrastructure layer, while the REST API provides high-level programmability abstraction between the application and control layer. The APIs enhance the proposed CIC-SIoT principal to be compatible with both distributed flat and hierarchical controllers. However, detailed implementation and working principles of distributed controllers are left for our future work. The control layer features a centralized IC-SDN controller that is distributed and has multiple instances. This allows hyper-scalability and removes the single-element bottleneck. The synchronization among the instances is not trivial. However, existing solutions for distributed server management can be applied. The control layer manages and orchestrates all nodes in the network. The infrastructure layer accommodates consumers, ICN routers, and producers (such as content origins or data centers).

On the other hand, the application layer in our proposed IC-SDN architecture serves as a critical intermediary between high-level application requirements and the network's control plane. This layer bridges the gap by harmonizing SDN's programmability with ICN's data-centric approach to foster seamless integration. It empowers applications to interact with the network based on data names rather than relying on traditional location-based methods. The RESTful or REST APIs fulfill a dual function – upholding SDN's core principles of abstraction and programmability while facilitating ICN's content-oriented communication. Applications, for instance, leverage these APIs to instruct the IC-SDN controller to compute optimal delivery paths for managing content caches across the network and tracing the flow of named content for diagnostics or optimization. Furthermore, the application layer ensures that the control logic provided by the IC-SDN controller aligns with ICN's objectives. These objectives include efficient content dissemination, enhanced security by binding security measures directly to the content itself and providing a diverse range of content-oriented network services. Through REST API interfaces, application layer modules like topology management, path calculation, cache management, content tracing, and signature management instruct the IC-SDN controller to align network resources with data-centric operations. This includes mapping data names to network paths, managing distributed content caches to minimize latency, tracing content for improved network visibility, and implementing named-based security measures for data integrity and authentication. The REST APIs abstract the complexities to allow applications to seamlessly request and control network services, ensuring an adaptive, efficient, and secure data flow across the IoT landscape.

### *B. Overview of Controller*

In traditional ICN, the decision making of some critical tasks such as caching and forwarding of the contents are assigned to ICN routers. However, we design our controller to take control of some of these tasks. Therefore, the controller is the most vital entity in our solution. Before making all the forwarding and caching decisions, the controller is responsible for maintaining two essential tables. The first one is the content table, where it stores the contents' names along with the producer id. Therefore, the controller is aware of the content along with their location. On the other hand, after receiving the necessary information from ICN routers from the data layer, such as their active faces and neighboring nodes, the controller builds a graph of the entire topology. In later stages, the controller is also required to calculate the available path's weight to find the most optimal path to allow the forwarding the most efficient way. We discuss the path calculation method in more detail in the next section.

### *C. Overview of ICN Router*

The ICN router in our framework has multi-folded functions in terms of forwarding and caching capabilities. Firstly, the ICN routers with the built-in caching capability preserve necessary contents and make them available based on the controller's instructions. Secondly, ICN routers with non-caching capabilities participate in a request-response message exchange with the controller. The exchange allows the content to move from the producer to the consumer and behaves more like an SDN-based switch. The primary task of these nodes is to find the interfaces (e.g., incoming and outgoing ) for each intermediate ICN router and to update the PIT entries based on the optimal path from the consumer to the producer.

### *D. Overview of Consumer*

In our framework, the consumer acts as a data requester, initiating the content retrieval by broadcasting an interest packet. Unlike IP-based systems, where data is retrieved from a specific source, conventional ICN architectures (e.g., NDN) handle the interest packet by the network of routers. These routers return the data from any location where it is

cached, enhancing robustness against individual link failures. To further improve reliability and efficiency in our design, the consumer leverages the proposed packet structure (shown in Figure 2), enabling it to maintain communication with multiple ICN routers and the controller. If an intermediate router fails or does not respond, our system's controller promptly identifies and redirects to an alternative path to mitigate any negative impact on content retrieval rates. Therefore, while our consumers operate within the ICN paradigm, they benefit from enhanced reliability through a design that supports dynamic rerouting by the controller.

### E. Overview of Producer

The producer in the solution acts as the primary source of the content. Therefore, it can either be a data center or a remote server with the original version of the content. The producer also validates the signature of each content it contains. The primary difference between the traditional ICN producer and our proposed producer remains on how it connects to the controller despite its location. In the initial phase, the producer prepares a *Req_Pkt* for the controller to share the content's necessary information, including name, location, signature, and path to the producer. Once it receives the *Req_Pkt* from the controller, it prepares the data with signature and forwards it to the ICN router.

## IV. Implementation and Working Principle

In this section, we provide necessary technical details of the proposed packet structure alongside the specific implementation, including the workflow and algorithms of individual nodes such as controller, intermediate nodes, consumer, and producer. We begin by providing an overview of the integrated components of the TLV structure of the proposed *Req_Pkt* and *Res_Pkt*. Then we present each participating node individually with their novel working principles.

### A. Proposed Packet Structure

In general, every ICN packet is encoded in a Type-Length-Value (TLV) format which helps the existing packets (e.g., interest or data) to be distinguished by their behavior and functionalities. Hence, the TLV structure of ICN-based implementations, i.e., CCN or NDN can be different. For example, CCN supports fixed-length TLVs for its packet structure and data-plane parsing capabilities [38], while NDN uses nested variable TLVs, composed of sub-TLVs. Each of those values, type, and length from sub-TLVs offer variable size [48]. This makes the modification and customization of NDN packets to be more challenging. Despite the complexities of TLV modification, it allows customized ICN name tag with necessary objects which can be embedded inside a packet. Previous IC-SDN solutions [49]–[51] propose TLV modifications to enable compatibility between ICN packet structure, OF switch, and the SDN controller. For example, Signorello et al. [50] separate nested-TLV sequence of NDN *Int_Pkt* and *Dt_Pkt* into separate header fields and payloads so that the P4 [52] data plane parser can process them individually. Although the implementation promises to tackle compatibility between SDN and ICN, having different field values leads to performance overhead and portability issues.

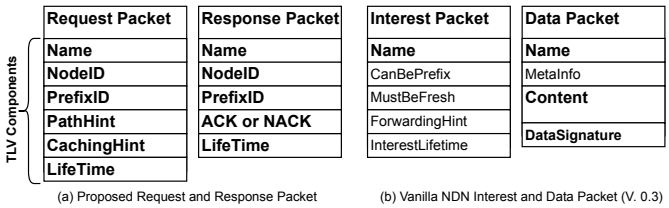



Figure 2: Proposed packet structure in CIC-SIoT.

To overcome these challenges, we propose a novel packet structure to allow the controller to communicate with other nodes by forwarding necessary instructions in the form of modified flows. As shown in Figure 2, the proposed packet structure consists of a request packet (*Req_Pkt*), and a response packet (*Res_Pkt*). To ensure the feasibility and effectiveness of our approach, we add necessary service-related objects wire-encoded as TLV into the *Req_Pkt* and *Res_Pkt*. The concept is inspired from the *flow_mod* messages – one of the variations of available TLV architecture that the traditional SDN controller regularly uses alongside *packet_in* and *packet_out* messages [53]. The specifications of our custom-defined TLV objects are illustrated as follows:

- **Name:** Serves as the unique identifier for the content. Consumers use this object to identify desired content, while producers utilize it to advertise available content. For intermediate ICN routers, the object's representation varies depending on caching capabilities. ICN routers with caching capabilities carry the cached content names; otherwise, they ignore this object.
- **NodeID:** Acts as a unique identifier for active nodes within the data layer. It significantly enhances network efficiency and reliability. NodeIDs are crucial for streamlining content retrieval and optimizing routing as the proposed IC-SDN controller maps the requested content names to the appropriate NodeID. This can be either the NodeID of the original producer or the nearest caching node. This mapping facilitates optimal delivery path decisions, considering factors such as network congestion, node availability, and geographic proximity to the consumer. NodeIDs are particularly advantageous in environments characterized by high mobility or frequent content updates as they enable the controller to rapidly adapt routing paths [54]–[56].
- **PrefixID:** The fundamental ICN feature that helps the controller track and record requests from consumers across different locations.
- **PathHint:** Deployed by the controller to direct ICN routers towards facilitating content retrieval along the most optimal paths. The controller embeds this hint in the *Req_Pkt* after calculating the optimal routes to assist ICN routers in setting up the necessary forwarding paths.
- **CachingHint:** Used by the controller to inform ICN routers about the availability of cached content, promoting efficient delivery within the network.

Table II: Notations used in the Paper.

| Symbol | Representation |
|---|---|
| $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ | Topology (as undirected graph), nodes, and links between nodes |
| $\mathcal{C} = \{1, \ldots, C\}$ | Set of Controllers where each controller $c \in \mathcal{C}$ |
| $\mathcal{N} = \{1, \ldots, N\}$ | Set of ICN routers where each router $n \in \mathcal{N}$ |
| $\mathcal{U} = \{1, \ldots, U\}$ | Set of IoT users where each user $u \in \mathcal{U}$ |
| $\mathcal{S} = \{1, \ldots, S\}$ | Set of producers where each producer $s \in \mathcal{S}$ |
| $x, y$ | TLVs, packet |
| $x_{\text{id}}, x_{\text{pre}}, x_{\text{node}}, x_{\text{path}}, x_{\text{tag}}, x_{\text{data}}, x_{\text{sig}}$ | ID, prefix, node, path, tag, payload, signature |
| $y_{\text{int}}, y_{\text{data}}, y_{\text{req}}, y_{\text{res}}$ | Interest, data, request, response packets |

**Algorithm 1:** Discovery and distribution.

**Function** COLLECT *Path from* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
  **forall** $y_{req}$ *from* $\mathcal{N}, \mathcal{S} \in \mathcal{V}$ **do**
    // here $y_{\text{req}}$ is incoming
    **Read** $y_{req} = (x_{name}, x_{pre}, x_{node}, x_{tag})$
    **while** $x_{node} = s$ *and* $s \in \mathcal{S}$ **do**
      Topology Table ← **Add or Update** $x_{\text{node}}$
      Content Table ← **Insert** $x_{\text{name}}, x_{\text{pre}}$
      $\mathcal{S}$ ← **Forward** $y_{\text{res}}(x_{\text{tag}})$ with ACK
    **while** $x_{node} = n$ *and* $n \in \mathcal{N}$ **do**
      Topology Table ← **Add or Update** $x_{\text{node}}$
      Cache Table ← **Insert** $x_{\text{name}}, x_{\text{pre}}$
      $\mathcal{N}$ ← **Forward** $y_{\text{res}}\,(x_{\text{tag}})$ with ACK
**Function** INSTALL *Path to* $n \in \mathcal{N}$
  **forall** $y_{req}$ *from* $\mathcal{N} \in \mathcal{V}$ **do**
    **Read** TLV $y_{\text{req}} = (x_{\text{name}}, x_{\text{pre}}, x_{\text{node}}, x_{\text{tag}})$
    **while** $x_{node} = n$ *and* $n \in \mathcal{N}$ **do**
      Content Table ← **Lookup** $x_{\text{name}}, x_{\text{pre}}$
      **if** *Lookup = Successful* **then**
        // map shortest path
        $\mathcal{E}_{\mathcal{CS}} \cup \mathcal{E}_{\mathcal{CN}} \cup \mathcal{E}_{\mathcal{NS}} \cup \mathcal{E}_{\mathcal{NU}}$ ← **Apply** *Optimal Path*
        // here $y_{\text{req}}$ is Outgoing
        $\mathcal{N}$ ← **Forward** $y_{req}\,(x_{path}, x_{tag})$
        $\mathcal{S}$ ← **Forward** $y_{\text{res}}\,(x_{\text{tag}})$ with ACK
      **else if** *Lookup* ≠ *Successful* **then**
        Pending Table ← **Insert** $x_{\text{name}}, x_{\text{pre}}, x_{\text{node}}$
        $\mathcal{N}$ ← **Forward** $y_{\text{res}}\,(x_{\text{tag}})$ with NACK

**Algorithm 2:** Caching and forwarding strategy.

**Function** UPDATE *and* RANK (*PIT entries*)
  **forall** $y_{req}$ *from* $\mathcal{C}, \mathcal{U}, \mathcal{S} \in \mathcal{G}$ **do**
    Read TLV $y_{\text{req}}(x_{\text{name}}, x_{\text{pre}}, x_{\text{tag}})$
    **while** $x_{node} \in \mathcal{C}$ **do**
      PIT ← **Insert** $(x_{name}, x_{pre}, x_{tag})$
      $\mathcal{C}$ ← **Forward** $y_{\text{res}}$ with ACK
    **while** $x_{node} \in \mathcal{S}$ **do**
      PIT ← **Insert** $(x_{name}, x_{pre}, x_{tag})$
      $x_{\text{data}}$ ← **Validate** *Signature*
      CS ← **Add or Update** $(x_{name}, x_{pre}, x_{tag})$
      $\mathcal{S}$ ← **Forward** $y_{\text{res}}$ with ACK
    **while** $x_{tag} \in \mathcal{U}$ **do**
      CS, PIT ← **Lookup** $y_{req}\,(x_{name}, x_{pre}, x_{tag})$
      **if** *Entry exists* **then**
        $\mathcal{U}$ ← **Forward** $y_{res}\,(x_{pre}, x_{pre}, x_{tag})$
      **else**
        $\mathcal{N}, \mathcal{S}$ ← **Forward** $y_{req}\,(x_{name}, x_{pre}, x_{tag})$

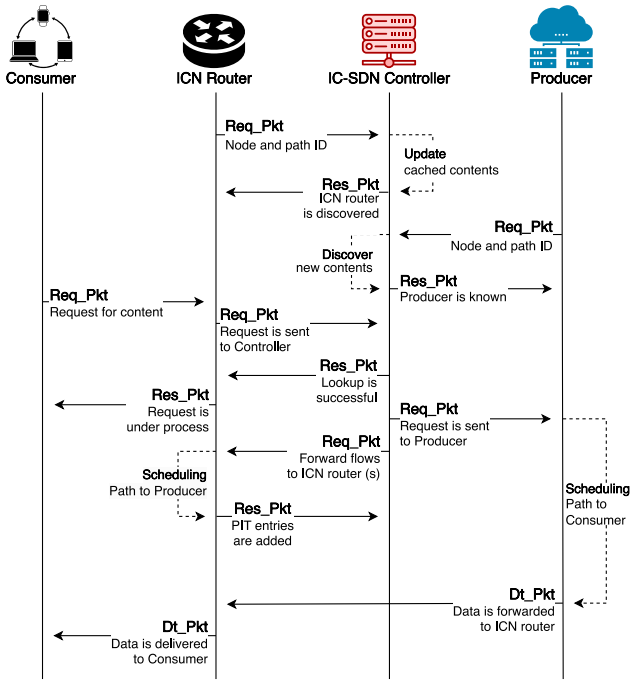



Figure 3: Discovery and distribution procedure in CIC-SIoT.

- **(N)ACK:** Stands for positive or negative acknowledgment and is a fundamental component of the *Res_Pkt*. It notifies the source node of the outcome of the requested content or service.
- **LifeTime:** Defines how long the *Req_Pkt* and *Res_Pkt* remain active or pending at the controller, ensuring timely processing and expiration of requests.

### B. Communication Principle

The communication method (shown in Figure 3) is divided into two phases – i) discovery (or bootstrapping) phase, and ii) allocation (or distribution) phase. Firstly, we discuss the discovery of available contents which allows the proposed controller to discover the contents in local and global regions. Later, we describe the content allocation model which discuss on the content dissemination process in CIC-SIoT system. The discovery and distribution algorithm is presented in Algorithm 1.

*1) Discovery Phase:* In the discovery phase, all available producers in the topology sent their prefix information to the controller. After receiving this information, the controller stores them into a table. The communication process between a controller and multiple producers initializes in the discovery phase. The controller acknowledges the available contents and their location immediately. The finite state machine (FSM) representation of all the participating nodes is shown in

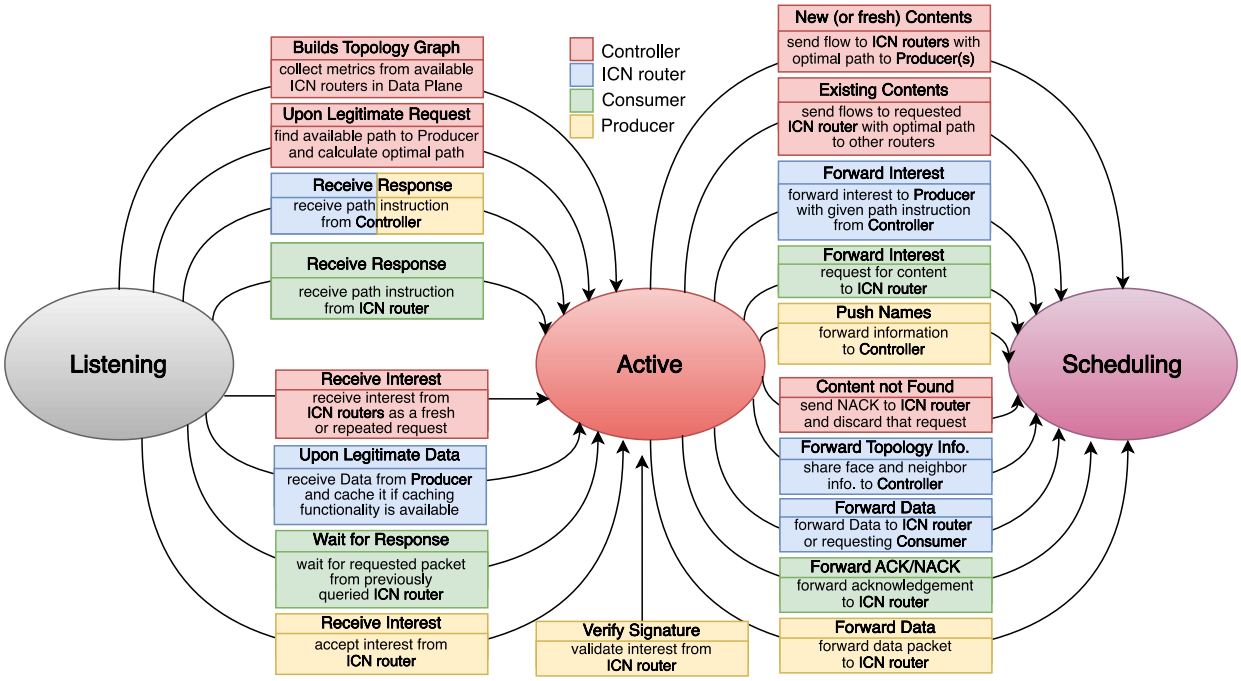



Figure 4: Task FSM of Controller, ICN router, Consumer, and Producer.

Figure 4.

In the network topology, as producers are integrated, they automatically transmit their content names to the controller. A push-based method is adopted over a pull-based approach to mitigate the potential overload on the controller from continuous content availability inquiries from producers. The TLV modifications enable the precise routing of *Req_Pkt* and *Res_Pkt* for specific operations, ensuring efficient content discovery and routing. The hierarchical namespace allows the specific prefix to build an automated service-oriented forwarding using distributed controllers. On the other hand, the application running under each producer allocates content using a specific namespace. For example, "*ndn://domain_id/producer_node/content_name/segment_number/version/*". This allows the controller to recognize the location of the content during the discovery period. After receiving contents from different producer, the controller stores those names in the *content_records*. To implement such data structure, we add a vector table to list contents' origin and to guide the controller discovering the producers with those available contents. During the discovery phase, the ICN routers send necessary information that includes bandwidth, metric, delay and queue to the controller. To allow transmitting such type of information, we have made changes on ICN router in accordance with the NDN paradigm. Note that, the controller is maintaining two different data structure. The first one represents the name of the contents from the producer and the second one represents the information of each ICN router.

*2) Distribution Phase:* The controller register the name of the content along with *node_tag*, content name and the timer into its *content_records*. If the content is not requested for 30 seconds then the entry from table is deleted but if content is requested then timer incremented. The critical decision making initiates while it receives a *Req_Pkt* from the consumer for a the desired content. The controller extracts the TLV components of that *Req_Pkt* and looks upon the *content_records* and *cache_entries*. If the name is found on the *content_records* instead of *cache_entries*, the controller looks for the producer associated with the name of that desired content. After searching content the controller calculate the most optimal path using Dijkstra to find the best optimal route from the requested Consumer to the Producer that has that content. Correspondingly, the controller sends *Req_Pkt* with flow instruction that would populate the PIT entries of those nodes that are in that path. Afterward, the controller make connection status table as two communicating nodes never interrupted. The controller guides the ICN nodes based on whether certain contents are cached or not. For example, if specific contents are available in different ICN nodes, the controller registers those contents' prefixes in its *cache_entries* to keep track of them. Upon receiving a request, the controller looks up the content's name on the *cache_entries* to identify the *node_tag* of the intermediate ICN router with the requested content. Afterward, it calculates the available paths from the consumer to that ICN router using Dijkstra to define the best optimal path. Suppose the intermediate ICN nodes do not cache the requested content. In that case, the controller discovers the *Producer_ID* and other intermediate *Node_IDs* to calculate the shortest route from the consumer with the requested content to the producer. Hence, the controller forwards *Req_Pkt* with flow instructions to all those intermediate ICN nodes by populating PIT entries. Finally, the controller forwards *Res_Pkt* to the initial ICN node on the path to guide other ICN nodes to shift the desired data to the requested origin. The caching and forwarding strategy of the proposed

CIC-SIoT is shown in Algorithm 2.

## V. Performance Evaluation

In this section, we evaluate the proposed CIC-SIoT implementation using simulation study to demonstrate the efficiency of the controller. Firstly, we categorize the performance evaluation section into two sub-categories, i.e simulation-based performance analysis and numerical analysis. In the simulation-based analysis, we use a physical testbed using a single workstation. The hardware specification is similar to the one used in [57], [58]. We use a heavily modified version of ndnSIM [59] to execute all the simulation-related experiments. The experimental parameters are mentioned in Table III. We first present the technical details of our proposed solution as a proof of concept implementation in the opening portion of this section. Then we give a brief technical overview of the testbed environment that has been used to conduct all the experiments to evaluate the performance of our proposed CIC-SIoT framework. Next, we present the simulation parameters of the simulation-based experiments. Finally, we discuss in details regarding the result analysis from the simulation-based analysis. We conclude the section by presenting the result from the numerical analysis. MATLAB [60] is used in the numerical analysis to show the feasibility of the solution while comparing against some of the recent works in IC-SDN implementation.

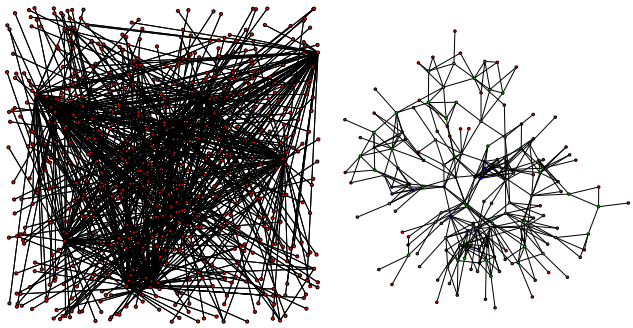
### A. Proof of concept Implementation

Here, we give a brief overview on the proof of concept implementation of our proposed solution, which includes packet modification, simulator library files, topology for simulation and custom-designed applications for the main modules such as consumer, producer, controller, and ICN router. Sources of the proof of concept implementation are provided in Github[1]. We utilize ndnSIM [59] to deploy the IC-SDN framework as it is widely used for simulating NDN based projects. It is developed as an additional module of ns-3 [61].

The ns-3 integration allows ndnSIM to take advantage of the available abstraction, function, and simulation module. The current version of ndnSIM improves the integration with real-world NDN codebase like ndn-cxx [62] and NFD [63] library that enable the abstraction for fundamental NDN and ICN tasks such as named content transmission, encoding or decoding of named packet and security-based implementation. To utilize the simulation environment to its maximum potential, we make changes on both packet level and application level. Packet level changes include addition and modification of TLV values through customizing ndn-cxx libraries. Besides, we also make some minor changes on *helper model* and *NFD forwarder*. Afterward, we designate individual application module for the controller, information-centric ICN router, custom consumer and producer.

### B. Overview of Simulation Testbed and Parameters

To evaluate the performance of our proposed system, we use a custom-designed topology alongside the widely used Rocketfuel topology [64]. The structure of both these topologies is shown in Figure 5. Our simulation testbed is structured into a three-layer custom topology, reflecting the hierarchical nature of large-scale networks. The control layer includes multiple instances of our IC-SDN controller, each embedded with algorithms for path discovery and content distribution. The core layer comprises 50 routers that serve as intermediaries, facilitating communication between the producers and the control layer. The access layer includes 20 routers proportionally distributed among the consumer nodes. Each access router provides connectivity for a cluster of 25 consumers, collectively supporting 500 consumers. It is important to note that ICN routers in the core layer have in-network caching capabilities, whereas the access nodes are non-caching forwarding nodes, similar to SDN-based OF switches [65]. We set the link capacity of core and access nodes randomly in the range of 500 Mbps to 1 Gbps. All consumers generate data (or content) at a rate of 100 Mbps. The link between the producer and core node is set to 10 Mbps, and the propagation delay of the available links is set to 1 ms. Due to the initialized path and caching hint on the *Req_Pkt*, the header size may slightly differ, including optional and static TLV headers. Therefore, the encoded data, packetized into a set of *Res_Pkt*, may vary in size than the maximum transmission unit (MTU). As a result, we defined the size of the *Req_Pkt* and *Res_Pkt* as 200 and 1028 bytes, respectively. Each experiment is executed 30 times to maintain a confidence level of 95% and avoid possible randomness and errors, obtaining an averaged calculation for the statistical analysis of the proposed and baseline solutions.



(a) Custom topology. (b) Rocketfuel topology.

Figure 5: Topologies used in the simulations.

Table III: Simulation parameters and their specifications.

| Parameter | Specifications |
|---|---|
| Topology | Custom, Rocketfuel [64] |
| Link delay | 10 ms |
| Link bandwidth | 1 Gbps (core and access) |
| PIT size | 10,000 entries |
| Number of contents | 100 to 300 |
| Number of caching nodes | 100 |
| Request rate | 20 to 100 interests/sec. |
| Payload size | 1024 bytes |
| Popularity rate | Zipf ($\alpha$ = 0.80) |
| Forwarding strategy | Best route and multicast |
| Caching strategy | LCD |
| Simulation duration | 60 sec. to 10 minutes |

[1]https://github.com/karimmd/ndnSIM-PIT

### C. ndnSIM-based Results Discussion

In this section, we provide a detailed analysis of the experimental results. We evaluate our proposed CIC-SIoT solution using four key metrics: (i) throughput, (ii) flow installation, (iii) caching efficiency, and (iv) packet loss. Firstly, we define throughput as the rate at which our system successfully delivers data to each consumer, serving as a primary indicator of network performance. Moreover, we measure the flow installation rate, which reflects the IC-SDN controller's efficiency in initiating network traffic flows. Besides, we assess caching efficiency by determining how effectively data can be retrieved from ICN router caches, which measures the network's capacity to reduce the burden on content producers. On the other hand, we calculate packet loss as the ratio of lost packets to total transmitted packets, thereby providing a critical measure of network reliability. To conduct the evaluation, we use separate simulation scripts for four different experiments, all using the topologies shown in Figure 5. However, extensive experimentation shows that the topological change has an insignificant impact on the results. Figure 6 shows that the throughput (further explained in sections below) is almost the same for both topologies; hence, we have presented the average results from both in the later sections.

*1) Throughput Analysis:* We measure throughput up to 500 consumers for both traditional NDN implementation and our proposal. We execute this experiment to showcase the level of successful contents delivery from consumer to producer. It is worth mentioning that, either packet loss or network congestion can have a detrimental impact on throughput. We obtain the number of total packet counts and divide them with the total number of consumers to obtain the average throughput rate into bits per second. Then we convert them into Mbps for better presentation. To measure the throughput rate (TR) for each consumer, we use $TR = TPC/NoC$ where number of consumer and total packet count are denoted as NoC and TPC, respectively. Figure 7a shows the throughput evaluation between standard NDN-based implementation and proposed CIC-SIoT prototype. The traditional NDN implementation achieves an average throughput from 1 Mbps to 10 Mbps for 10 to 500 consumers. On the contrary, the proposed CIC-SIoT solution shows constant performance increment as we increase the consumers from 10 to 500. As for 10 consumers, the framework achieves 1 Mbps, and later the performance rises above the traditional NDN by reaching up to 20 Mbps for 500 consumers. The IC-SDN framework achieves the maximum throughput from 1 Mbps to 19 Mbps as consumers' amount increases from 10 to 500. On the contrary, despite a promising start, the traditional NDN framework fails to cope with our framework as we increase the consumers to 500. As a result, the IC-SDN system maintains a well-balanced performance between high throughput and scalability as the consumers request multiple contents to the network simultaneously. The centralized controller acknowledges these requests from different consumers and installs necessary flows so that these consumers receive those contents properly.

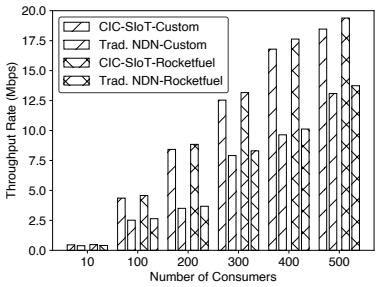
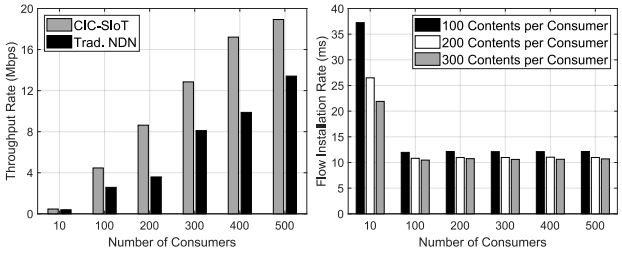
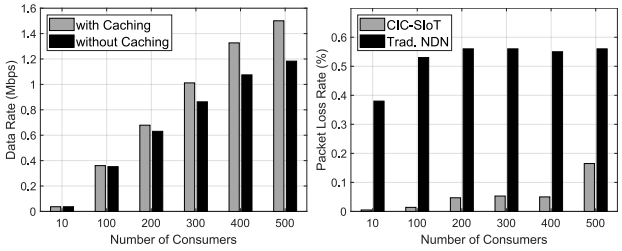



Figure 6: Throughput comparison for different topologies.




(a) (b)

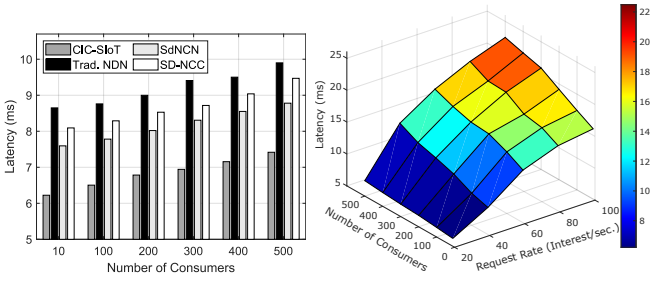
Figure 7: Performance analysis of (a) throughput and (b) flow installation rate




(a) (b)

Figure 8: Performance analysis of (a) caching efficiency, and (b) packet loss rate

*2) Flow Installation Analysis:* In the traditional SDN-based scenario, execution or installation efficiency of flow messages determines the adaptability and effectiveness of the controller in that scenario. To determine the flow processing rate of the proposed CIC-SIoT controller, we obtain the number of total packet counts and divide them with the multiplication of total number of consumers and number of unique contents. We acknowledge the flow installation time of the controller based the successful content retrieval. We conduct multiple instances of simulation execution on the basis of the number of unique contents requested by the consumers which is again from 10 to 500 consumers. In the previous experiment on throughput evaluation, we consider random amount of contents requested by the consumers. However, in this experiment, we conduct multiple experiments by setting the custom consumer app to request fixed number of unique contents to 100, 200 and 300 per consumer. As soon as the number of requested unique contents are satisfied, we take the output of the flow installation time and compare the generated data between them. Due to the fact that, the traditional NDN does not have the controller (or management) feature, we only consider the IC-SDN scenario to evaluate the available data among the number of unique




(a) (b)

Figure 9: Measurements of latency against varying consumers and request rate.

contents requested by each consumer. Figure 7b shows the flow installation rate for processing different amount of unique contents. In the beginning, the controller takes around 38 ms to install necessary flows that allows producer and the ICN routers to forward those 100 unique contents back to 10 consumer. In case of 200 contents, the controller takes less time under 28 ms and for 300 contents with 10 provides, it takes 10 ms to finish installation of the necessary flow instruction. As soon as, the consumers are increased to 100, the processing time decreases into 10 ms which is similar for 200 and 300 unique contents as well. Finally, we see that, the flow install time seems more or less similar for 200 to 500 consumers. To calculate flow installation rate (FIR) for the IC-SDN controller, we use $FIR = TPC/(NUC \times NoC)$, where number of unique contents is denoted as NUC. The experiment demonstrates that, the flow processing time is higher at the beginning due to the bootstrapping time and acknowledging the topology and discovering the path to the producer. However, as soon as the consumers are increased, the controller starts to process the necessary flows more efficiently which results low flow processing rate and it remains linear for the rest of the experimental time-frame. The increasing number of consumers does not make any negative impact on the flow installation procedure as the results become linear till the end. Additionally, the result also demonstrates that, due to less amount of packet loss, the controller is able to process the increasing amount of contents in a flexible manner while keeping a stable performance. As a result, having a centralized controller in a information-centric environment makes the content retrieval process smoother than the traditional NDN framework.

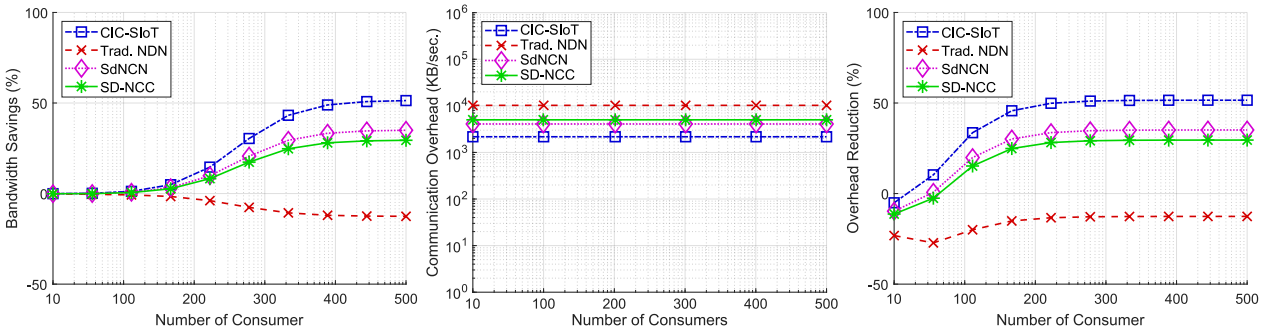
*3) Caching Efficiency Analysis:* As caching is an important functionality of information-centric network environment, we perform the caching efficiency of the proposed controller implementation my measuring the success or hit ratio. Prior to this experiment, we modify the ICN router in two different ways. Firstly, we enable the caching ability for the ICN routers and collect the number of packets. Afterward, we disable the caching function of the available nodes and execute the same scenario. Figure 8a depicts the average data rate between caching and non-caching prototypes of our IC-SDN framework. There are no significant changes in the overall data rate when the consumers remains at 200, while the framework achieves a data rate up to 700 Kbps in both scenarios. As we increase the consumers from 300 to 500, the ICN routers start to cache necessary contents whenever possible. The overall caching performance improves compared to the non-caching one. For non-caching scenario, the data request and retrieval processes between 200 consumers and 20 producers have similar performance compared to the caching one. However, as the consumers increases, the data retrieving process from those 20 producers starts to impact the overall performance. On the contrary, in the cache-enabled scenario, consumers directly get the contents from the ICN router without reaching the available producer. Therefore, the limited number of producers does not negatively impact the performance with the availability of cached contents. As a result, the framework achieves a data rate of around 1500 kbps while outperforming the non-caching one that manages up to 1200 kbps data rate.

*4) Packet Loss Analysis:* In this section we calculate rate of packet loss from consumer 10 to 500 with traditional ICN and our proposed CIC-SIoT solution. The technical specification of testbed and simulator configuration is same as throughput. However, in this experiment we calculate the number of lost packet between our IC-SDN implementation and the traditional NDN implementation which is shown in Figure 8b. We denote total transmitted packet as TT and total received packet as TR. Therefore, we use $PLR = (TT - TR)/TT$ to calculate the average packet loss. Compared to the above-mentioned experiments, here we normalize the results within 0% to 0.1%. The traditional NDN system showcases the packet loss rate between 0% to 0.15% regarding the consumers from 10 to 500. On the contrary, the average packet loss rate of our framework achieves between 0.39% to 0.55%. Note that, packet loss occurs due to the limited PIT size which is defined as 10,000 entries.

### D. Result Discussion of Numerical Analysis

In the numerical analysis, we compare the efficiency of the proposed algorithm with traditional NDN designed for MATLAB environment [60]. Besides, we also use SD-NCC [35] and SdNCN [66] to compare the feasibility of our solution in terms of content retrieval latency, communication overhead, reduced overhead rate and bandwidth savings. The topology from 5b is deployed in the numerical experiments. Each experiments are performed 10 times and then we calculate the average of these 10 executions for the final outcome.

*1) Content Retrieval Latency:* The content retrieval latency measures the time interval from when the controller receives a request from a consumer to when the consumer responds after successfully obtaining the desired content. As the number of consumers increases, the volume of requests grows exponentially. Figure 9a illustrates the content retrieval latency comparisons among SdNCN, SD-NCC, traditional NDN, and our proposed solution. As the number of consumers rises from 10 to 500, an estimated 200 to 10,000 requests are generated per second. Our proposed solution outperforms the others, attributed to its efficient shortest path algorithm and flow installation prowess. Consequently, the latency for successful content retrieval remains between 6.2 ms and 7.4 ms. In




(a) (b) (c)

Figure 10: Performance analysis of (a) bandwidth savings, (b) communication overhead, and (c) overhead reduction.

contrast, the latency for traditional NDN ranges from 8.65 ms to 9.9 ms. Furthermore, while SdNCN records a latency slightly better at 8.78 ms, SD-NCC lags at 9.5 ms when accounting for 500 consumers in the topology.

To further evaluate CIC-SIoT's performance under varying workloads, we conduct a comprehensive analysis of content retrieval latency under increasing request rates and consumer numbers. Figure 9b illustrates CIC-SIoT's capability to handle a growing number of consumer requests under different request rates. Starting with 10 consumers, we observe that latency increases from 6.224 ms for 20 requests per second to 16.02 ms for 100 requests per second. As the number of consumers increases to 100, latency rises from 6.504 ms at 20 requests per second to a manageable 17.45 ms at 100 requests per second. This trend remains consistent when we extend the analysis to 500 consumers; latency starts at 7.417 ms for 20 requests per second and increases to 22.48 ms for 100 requests per second. These gradual increases in latency correspond to the increases in request rates, from 20 to 100 requests per second, across consumer groups ranging from 10 to 500. Despite the simultaneous increase in consumer numbers and request rates, the relatively small increase in latency demonstrates CIC-SIoT's effectiveness in scaling and maintaining performance within acceptable limits. This ability to manage higher demands with minimal latency increase is essential for the demanding conditions of IoT environments, showcasing the solution's scalability and stability

*2) Bandwidth Savings:* The increasing number of control messages causes performance bottlenecks in the network. Hence, bandwidth saving is a critical factor for scalable content distribution in a distributed hierarchical network. In the figure 10a, we evaluate the bandwidth saving capability in terms of how these solutions fare when content dissemination on the disjoint path's link. Here the disjoint path refers to a particular situation where the number of consumers mismatches with the topology density. Within 10 to 100 consumers, there are no spectacular changes in bandwidth compensation. When we increase the number of consumers to 200, the proposed solution restricts the bandwidth usage significantly up to 20% to 50%. The proposed solution achieves approximately 15% and 40% more than 30% of SdNCN and 15% of SD-NCC. The traditional NDN fails to restore bandwidths as the number of consumers increases.

*3) Communication Overhead:* The average communication overhead is another crucial metric determining the controller's efficiency in processing the control messages when the number of consumers increases. The performance comparison of average and reduced communication overhead throughout the ICN nodes are depicted in Figure 10b and Figure 10c. Besides, the proposed solution's average overhead remains lower than the while we also notice the reduced overhead stays within a range of 41% to 52%. The stable overhead result also validates the effective exchange of flow messages between the controller and the ICN router. The controller detects the inactive ICN nodes while there are disjoint paths and takes immediate actions to recover the communication's inactivity.

## VI. Conclusion and Future Work

This paper investigates the effectiveness of an IC-SDN controller within a clean-slate ICN-IoT network to emphasize its role in discovering and distributing desired content through optimal routing. By leveraging the in-network forwarding and caching capabilities of ICN routers, the proposed controller improves content transmission between producers and consumers. To enhance inter-layer communication, we suggest modifying TLV entries in existing interest and data packets to create request and response packets. This modification facilitates scalable communication between nodes in the data layer and the controller in the control layer. Simulation studies and numerical analyses demonstrate the model's efficiency, showing reductions in packet delay, bandwidth usage, and communication overhead, while simultaneously improving caching efficiency and throughput. Future research will explore the deployment of multiple hierarchical IC-SDN controllers within the ICN-IoT paradigm to further enhance content delivery and caching in scenarios such as blockchain and metaverse.

## References

[1] F. Spinelli and V. Mancuso, "Towards enabled industrial verticals in 5g: a survey on mec-based approaches to provisioning and flexibility," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 596–630, 2020.

[2] Y. Siriwardhana, P. Porambage, M. Liyanage, and M. Ylianttila, "A survey on mobile augmented reality with 5g mobile edge computing: Architectures, applications and technical aspects," *IEEE Communications Surveys & Tutorials*, 2021.

[3] Y. Xu, G. Gui, H. Gacanin, and F. Adachi, "A survey on resource allocation for 5g heterogeneous networks: Current research, future trends and challenges," *IEEE Communications Surveys & Tutorials*, 2021.

[4] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[5] D. E. Sarmiento, A. Lebre, L. Nussbaum, and A. Chari, "Decentralized sdn control plane for a distributed cloud-edge infrastructure: A survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 256–281, 2021.

[6] R. Ullah, S. H. Ahmed, and B.-S. Kim, "Information-centric networking with edge computing for iot: Research challenges and future directions," *IEEE Access*, vol. 6, pp. 73 465–73 488, 2018.

[7] V. Jacobson, M. Mosko, D. Smetters, and J. Garcia-Luna-Aceves, "Content-centric networking," *Whitepaper, Palo Alto Research Center*, pp. 2–4, 2007.

[8] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[9] A. Drescher, J. DeHart, J. Parwatikar, and P. Crowley, "Analyzing the performance of icn forwarders on the wire," in *Proceedings of the 7th ACM Conference on Information-Centric Networking*, 2020, pp. 52–58.

[10] B. Liu, H. Dai, W. Xu, T. Yun, and J. Miao, "Scalable hardware content router: Architecture, modeling and performance," in *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. IEEE, 2021, pp. 1–7.

[11] V. Sivaraman, D. Guha, and B. Sikdar, "Optimal pending interest table size for icn with mobile producers," *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1615–1628, 2020.

[12] Z. Li, Y. Xu, B. Zhang, L. Yan, and K. Liu, "Packet forwarding in named data networking requirements and survey of solutions," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1950–1987, 2018.

[13] J. Shi, D. Pesavento, and L. Benmohamed, "Ndn-dpdk: Ndn forwarding at 100 gbps on commodity hardware," in *Proceedings of the 7th ACM Conference on Information-Centric Networking*, ser. ICN '20. ACM, Sep. 2020.

[14] J. Takemasa, Y. Koizumi, and T. Hasegawa, "Vision: toward 10 tbps ndn forwarding with billion prefixes by programmable switches," in *Proceedings of the 8th ACM Conference on Information-Centric Networking*, ser. ICN '21. ACM, Sep. 2021.

[15] H. Yuan and P. Crowley, "Scalable pending interest table design: From principles to practice," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*. IEEE, Apr. 2014.

[16] Q.-Y. Zhang, X.-W. Wang, M. Huang, K.-Q. Li, and S. K. Das, "Software defined networking meets information centric networking: A survey," *IEEE Access*, vol. 6, pp. 39 547–39 563, 2018.

[17] W. Rafique, A. S. Hafid, and S. Cherkaoui, "Complementing IoT Services Using Software-Defined Information Centric Networks: A Comprehensive Survey," *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23 545–23 569, Dec. 2022.

[18] ——, "SoftCaching: A framework for caching node selection and routing in Software-Defined Information Centric Internet of Things," *Computer Networks*, vol. 235, p. 109966, Nov. 2023.

[19] D. Kutscher and D. Oran, "RESTful information-centric networking: Statement," in *Proceedings of the 9th ACM Conference on Information-Centric Networking*, ser. ICN '22. New York, NY, USA: Association for Computing Machinery, Sep. 2022, pp. 150–152.

[20] J. Wang, J. Luo, Y. Ran, J. Yang, K. Liu, and S. Guo, "Towards Predictive Forwarding Strategy in Vehicular Named Data Networking," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 3, pp. 3751–3763, Mar. 2023.

[21] D. M. Doe, D. Chen, K. Han, H. Wang, J. Xie, and Z. Han, "DSORL: Data Source Optimization With Reinforcement Learning Scheme for Vehicular Named Data Networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 10, pp. 11 225–11 237, Oct. 2023.

[22] Z. Zhang, C.-H. Lung, X. Wei, M. Chen, S. Chatterjee, and Z. Zhang, "In-Network Caching for ICN-Based IoT (ICN-IoT): A Comprehensive Survey," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14 595–14 620, Aug. 2023.

[23] M. Rayani, A. Ebrahimzadeh, R. H. Glitho, and H. Elbiaze, "Ensuring Profit and QoS When Dynamically Embedding Delay-Constrained ICN and IP Slices for Content Delivery," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 2, pp. 769–782, Mar. 2022.

[24] E. Bardhi, M. Conti, R. Lazzeretti, and E. Losiouk, "Security and Privacy of IP-ICN Coexistence: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, pp. 1–1, 2023.

[25] X. Zhang and Q. Zhu, "Information-Centric Virtualization for Software-Defined Statistical QoS Provisioning Over 5G Multimedia Big Data Wireless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 8, pp. 1721–1738, Aug. 2019.

[26] M. Amadeo, C. Campolo, G. Ruggeri, A. Molinaro, and A. Iera, "SDN-Managed Provisioning of Named Computing Services in Edge Infrastructures," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1464–1478, Dec. 2019.

[27] Y. Xiao, J. Liu, J. Wu, and N. Ansari, "Leveraging deep reinforcement learning for traffic engineering: A survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2064–2097, 2021.

[28] Q. Zhang, X. Wang, J. Lv, and M. Huang, "Intelligent Content-Aware Traffic Engineering for SDN: An AI-Driven Approach," *IEEE Network*, pp. 12–19, 2020.

[29] P. Benedetti, G. Piro, and L. A. Grieco, "A softwarized and MEC-enabled protocol architecture supporting consumer mobility in Information-Centric Networks," *Computer Networks*, vol. 188, p. 107867, Apr. 2021.

[30] G. Nencioni, R. G. Garroppo, and R. F. Olimid, "5g multi-access edge computing: A survey on security, dependability, and performance," *IEEE Access*, vol. 11, pp. 63 496–63 533, 2023.

[31] S. Sharif, M. H. Y. Moghaddam, and S. A. H. Seno, "Adaptive cache content placement for software-defined Internet of Things," *Future Generation Computer Systems*, vol. 136, pp. 34–48, Nov. 2022.

[32] A. Khalid, R. A. Rehman, and M. Burhan, "CBILEM: A novel energy aware mobility handling protocol for SDN based NDN-MANETs," *Ad Hoc Networks*, vol. 140, p. 103049, Mar. 2023.

[33] V. Demiroglou, S. Skaperas, L. Mamatas, and V. Tsaoussidis, "Adaptive Multi-Protocol Communication in Smart City Networks," *IEEE Internet of Things Journal*, pp. 1–1, 2024.

[34] A. Anjum, P. Agbaje, S. Hounsinou, N. Guizani, and H. Olufowobi, "D-NDNoT: Deterministic Named Data Networking for Time-Sensitive IoT Applications," *IEEE Internet of Things Journal*, pp. 1–1, 2024.

[35] Q. Chen, F. R. Yu, T. Huang, R. Xie, J. Liu, and Y. Liu, "Joint resource allocation for software-defined networking, caching, and computing," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 274–287, 2018.

[36] Q. Sun, W. Wendong, Y. Hu, X. Que, and G. Xiangyang, "Sdn-based autonomic ccn traffic management," in *2014 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2014, pp. 183–187.

[37] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "Nox: towards an operating system for networks," *ACM SIGCOMM Comp. Commun. Rev.*, vol. 38, no. 3, pp. 105–110, 2008.

[38] M. Mosko, I. Solis, and C. A. Wood, "Content-centric networking-architectural overview and protocol description," *arXiv preprint arXiv:1706.07165*, 2017.

[39] J. Son, D. Kim, H. S. Kang, and C. S. Hong, "Forwarding strategy on sdn-based content centric network for efficient content delivery," in *2016 international conference on information networking (ICOIN)*. IEEE, 2016, pp. 220–225.

[40] S. Gao, Y. Zeng, H. Luo, and H. Zhang, "Scalable control plane for intra-domain communication in software defined information centric networking," *Future Generation Computer Systems*, vol. 56, pp. 110–120, 2016.

[41] J. V. Torres, I. D. Alvarenga, R. Boutaba, and O. C. M. Duarte, "An autonomous and efficient controller-based routing scheme for networking named-data mobility," *Computer Communications*, vol. 103, pp. 94–103, 2017.

[42] R. Jmal and L. C. Fourati, "An openflow architecture for managing content-centric-network (ofam-ccn) based on popularity caching strategy," *Computer Standards & Interfaces*, vol. 51, pp. 22–29, 2017.

[43] J. Lv, X. Wang, M. Huang, J. Shi, K. Li, and J. Li, "Risc: Icn routing mechanism incorporating sdn and community division," *Computer Networks*, vol. 123, pp. 88–103, 2017.

[44] M. A. U. Rehman, R. Ullah, B.-S. Kim, B. Nour, and S. Mastorakis, "CCIC-WSN: An Architecture for Single-Channel Cluster-Based Information-Centric Wireless Sensor Networks," *IEEE Internet of Things Journal*, vol. 8, no. 9, pp. 7661–7675, May 2021.

[45] M. S. U. Din, M. A. U. Rehman, and B.-S. Kim, "Cidf-wsn: a collaborative interest and data forwarding strategy for named data wireless sensor networks," *Sensors*, vol. 21, no. 15, p. 5174, 2021.

[46] J. Hong, T. You, L. Dong, C. Westphal, and B. Ohlman, *RFC 9138: Design Considerations for Name Resolution Service in Information-Centric Networking (ICN)*. USA: RFC Editor, Nov. 2021.

[47] T. Yu, Z. Zhang, X. Ma, P. Moll, and L. Zhang, "A pub/sub api for ndn-lite with built-in security," *Named Data Networking, Tech. Rep. NDN-0071, Revision*, vol. 1, 2021.

[48] "Named Data Networking (NDN) Packet Format Specification 0.3," https://named-data.net/doc/NDN-packet-spec/current/.

[49] S. Salsano, N. Blefari-Melazzi, A. Detti, G. Morabito, and L. Veltri, "Information centric networking over sdn and openflow: Architectural aspects and experiments on the ofelia testbed," *Computer Networks*, vol. 57, no. 16, pp. 3207–3221, 2013.

[50] S. Signorello, R. State, J. François, and O. Festor, "Ndn. p4: Programming information-centric data-planes," in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*. IEEE, 2016, pp. 384–389.

[51] P. Zuraniewski, N. van Adrichem, D. Ravesteijn, W. IJntema, C. Papadopoulos, and C. Fan, "Facilitating icn deployment with an extended openflow protocol," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, 2017, pp. 123–133.

[52] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese *et al.*, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.

[53] Ruggeri, Giuseppe and Amadeo, Marica and Campolo, Claudia and Molinaro, Antonella and Iera, Antonio, "Caching Popular Transient IoT Contents in an SDN-Based Edge Infrastructure," *IEEE Transactions on Network and Service Management*, vol. 18, pp. 3432–3447, 09 2021.

[54] Y. Hu, C. Serban, L. Wang, A. Afanasyev, and L. Zhang, "Pli-sync: Prefetch loss-insensitive sync for ndn group streaming," in *IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[55] 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, pp. 6684–6695, 2021.

[56] G. Araujo and L. Sampaio, "A scalable, dynamic, and secure traffic management system for vehicular named data networking applications," *Ad Hoc Networks*, vol. 158, p. 103476, 2024.

[57] L. Zhu, M. M. Karim, K. Sharif, C. Xu, F. Li, X. Du, and M. Guizani, "SDN Controllers: A Comprehensive Analysis and Performance Evaluation Study," *ACM Computing Surveys*, vol. 53, no. 6, pp. 133:1–133:40, Dec. 2020.

[58] L. Zhu, M. M. Karim, K. Sharif, C. Xu, and F. Li, "Traffic Flow Optimization for UAVs in Multi-Layer Information-Centric Software-Defined FANET," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 2, pp. 2453–2467, Feb. 2023.

[59] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the evolution of ndnsim: An open-source simulator for ndn experimentation," *ACM SIGCOMM Computer Communication Review*, vol. 47, no. 3, pp. 19–33, 2017.

[60] MATLAB, *9.7.0.1190202 (R2019b)*. Natick, Massachusetts: The MathWorks Inc., 2018.

[61] "ns-3, a discrete-event network simulator for internet systems," https://www.nsnam.org/, 2022.

[62] "ndn-cxx: NDN C++ library with eXperimental eXtensions," https://named-data.net/doc/ndn-cxx/current/, 2022.

[63] A. Afanasyev, J. Shi, B. Zhang, L. Zhang, I. Moiseenko, Y. Yu, W. Shang, Y. Huang, J. P. Abraham, S. DiBenedetto *et al.*, "Nfd developer's guide, revision 10," *Dept. Comput. Sci., Univ. California, Los Angeles, Los Angeles, CA, USA, Tech. Rep. NDN-0021*, 2018.

[64] N. Spring, R. Mahajan, and D. Wetherall, "Measuring isp topologies with rocketfuel," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 133–145, 2002.

[65] O. N. Foundation, "Openflow switch specification, version 1.5.1 ( protocol version 0x06 )," p. 1 to 283, 03 2015. [Online]. Available: https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf

[66] M. Amoretti, R. Pecori, Y. Protskaya, L. Veltri, and F. Zanichelli, "A scalable and secure publish/subscribe-based framework for industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 3815–3825, 2021.

**Md Monjurul Karim** is a postdoctoral research fellow at the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences. He received his Ph.D. in Computer Science and Technology from the Beijing Institute of Technology in 2023. He obtained his M.E. and B.E. degrees in Computer Science and Technology from Northwestern Polytechnical University in Xi'an, China, in 2015 and 2012, respectively. He also worked as a research and teaching assistant at the Southern University of Science and Technology in Shenzhen. His research interests include information-centric networking, software-defined networking, blockchain, deep learning, and intelligent transportation.

**Kashif Sharif** [SM'20] received his M.S. degree in information technology in 2004 from National University of Sciences and Technology, Pakistan, and Ph.D. degree in computing and informatics from University of North Carolina at Charlotte, NC, USA in 2012. He is currently an Associate Professor for research at Beijing Institute of Technology, Beijing, China. Previously he has held positions at UNC Charlotte, USA, and NUST, Pakistan. His research interests include data centric networks, blockchain & distributed ledger technologies, wireless & sensor networks, software defined networks, and 5G vehicular & UAV networks. His work is funded by Beijing Natural Science Foundation. He also serves as associate editor for IEEE Access.

**Sujit Biswas** [M'20] is an Assistant Professor in Computer Science Department, City, University of London, UK, Research Associate at University College London (UCL), and adjunct faculty at Northumbria University, UK. Prior to this position, he was a lecturer in Computer Science and Digital Technologies at the University of East London; a research fellow in Blockchain and AI at the University of Surrey's Center for Vision, Speech, and Signal Processing (CVSSP). He holds a Ph.D. in Computer Science and Technology from the Beijing Institute of Technology, China. His research focuses on consensus algorithms, scalability solutions, and privacy-enhancing techniques within blockchain systems.

**Zohaib Latif** is working as an Assistant Professor at Nazarbayev University, Astana, Kazakhstan. He has worked as an Assistant Professor (Research) and Postdoc Research Associate at Hanyang University, Seoul, South Korea. He completed his Ph.D. in Computer Science and Technology from Beijing Institute of Technology, Beijing, China. He passed his Masters in Electronics and Electrical Engineering from University of Glasgow, Glasgow, United Kingdom in 2009. His research interests include Computer Networks, Software Defined Networking (SDN), Multi-access Edge Computing (MEC), Computational Offloading to MEC, Internet of Things (IoT), and Blockchain.

**Qiang Qu** is a Senior Researcher at the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, and a Full Professor at the University of Chinese Academy of Sciences. Prof. Qu is the Director of the Guangdong Provincial Blockchain and Distributed IoT Security Engineering Research Center and a senior member of the China Computer Federation. He received his PhD from Aarhus University in 2014, under the supervision of Professor Christian S. Jensen. Prof. Qu's extensive experience includes roles at Innopolis University, Carnegie Mellon University, ETH Zurich, and Singapore Management University. His research interests encompass blockchain technology, databases, and data intelligence systems. He has been the principal investigator for numerous projects and currently serves as the Chief Scientist for a project funded by the National Key Research and Development Program of China.

**Fan Li** [M'12] received the Ph.D. degree in computer science from the University of North Carolina at Charlotte, Charlotte, NC, USA, in 2008, the M.Eng. degree in electrical engineering from the University of Delaware, Newark, DE, USA, in 2004, and the M.Eng. and B.Eng. degrees in communications and information system from the Huazhong University of Science and Technology, Wuhan, China, in 2001 and 1998, respectively. She is currently a Professor with the School of Computer Science, Beijing Institute of Technology, Beijing, China. Her current research focuses on wireless networks, ad hoc and sensor networks, and mobile computing. Her papers have won Best Paper Awards from IEEE MASS (2013), IEEE IPCCC (2013), ACM MobiHoc (2014), and Tsinghua Science and Technology (2015). She is a member of IEEE & ACM.