INVITED PAPER

A Survey of Information-Centric Networking: The Quest for Innovation

Hitoshi ASAEDA[†a)], *Senior Member*, Kazuhisa MATSUZONO[†], Yusaku HAYAMIZU[†], Htet Htet HLAING[†], *and* Atsushi OOKA[†], *Members*

SUMMARY Information-Centric Networking (ICN) is an innovative technology that provides low-loss, low-latency, high-throughput, and high-reliability communications for diversified and advanced services and applications. In this article, we present a technical survey of ICN functionalities such as in-network caching, routing, transport, and security mechanisms, as well as recent research findings. We focus on CCNx, which is a prominent ICN protocol whose message types are defined by the Internet Research Task Force. To facilitate the development of functional code and encourage application deployment, we introduce an open-source software platform called Cefore that facilitates CCNx-based communications. Cefore consists of networking components such as packet forwarding and in-network caching daemons, and it provides APIs and a Python wrapper program that enables users to easily develop CCNx applications for on Cefore. We introduce a Mininet-based Cefore emulator and lightweight Docker containers for running CCNx experiments on Cefore. In addition to exploring ICN features and implementations, we also consider promising research directions for further innovation.

key words: *Information-Centric Networking (ICN), Content-Centric Networking (CCN), CCNx, Cefore*

1. Introduction

Data are of critical importance for communications, and there is a need for advanced network architectures that can provide large volumes of data to humans, computers, sensors, and robots or cybernetic avatars quickly and reliably. Traditionally, most data are stored on a server and users fetch data from a central location, which requires significant bandwidth that can be overloaded as the number of users increases. To reduce the network traffic load on a central server, Content Delivery Networks (CDNs) have evolved as a concept for storing content geographically closer to end users.

Cloud computing provides efficient resource management functionality and deploys server images in multiple regions to enhance user experience. Many Internet services have been shifted to the cloud and handle data stored on cloud servers, regardless of the user environment, application demands, or requirements. Because all communications are performed via the cloud, the most significant disadvantages are its high latency, low energy efficiency, and high administrative costs and overhead, caused by location or load estimation mismatches between users and servers.

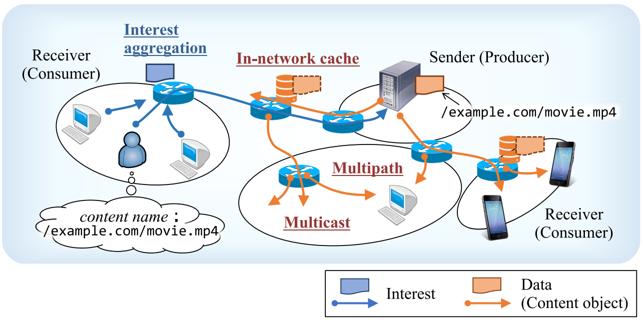

Fig. 1 ICN concept.

Edge computing, in which data are processed and information is provided by servers located near end users, has also attracted significant attention. Edge computing reduces latency and improves the quality of communication services. It is expected that the fusion of cloud and edge computing will enhance future Internet services. However, because such systems are still server dependent, it is difficult to solve the fundamental problem of optimizing server arrangement in terms of data processing capacity, transfer efficiency, and other related factors. To overcome these restrictions, the development of a novel network architecture is essential.

Information-Centric Networking (ICN) is an emerging technology that offers salient features for popular applications and services. It exploits a data-centric protocol that focuses on content delivery, unlike traditional client–server communications in CDNs. To facilitate the retrieval of content located on a network, ICN adopts a name-based communications protocol that detaches content from the original servers and creates new points of access to it using in-network caches allocated on intermediate routers or network nodes in a distributed manner (Fig. 1). This method reduces congestion and enhances delivery performance. ICN embeds functions enabling multicasting, multipath communications, and security at the protocol level, providing a new paradigm for retrieving data with reduced network traffic congestion and latency, enhancing throughput.

In this article, we present a technical survey of ICN functionalities, including (1) in-network caching, (2) routing, (3) transport, and (4) security, as well as a review of recent related work. We introduce an open-source software platform called *Cefore* that is compatible with the CCNx protocol specified by the Internet Research Task Force (IRTF)

Manuscript received April 14, 2023.
Manuscript revised July 17, 2023.
Manuscript publicized August 22, 2023.

[†]The authors are with the Network Architecture Laboratory at the National Institute of Information and Communications Technology (NICT), Koganei-shi, 184-8795 Japan.
a) E-mail: asaeda@nict.go.jp

DOI: 10.1587/transcom.2023EBI0001

to encourage ICN deployment in the real and virtual worlds. We also present experimental results using Cefore running on a network emulator and Docker containers. This article includes some content from our previous article [1] but has been updated with the most recent work.

2. ICN Principles and Protocol Basics

The core ICN principle is that the content desired by users should be available from the nearest routers or nodes in a network without server or cloud involvement, thereby improving the efficiency of conventional IP-address-based communication, communication performance for end users, utilization efficiency of network and server resources, and communication energy efficiency.

CCNx [2], [3] is a prominent ICN protocol whose message types have been defined by the IRTF ICN research group [4]. Another well-known ICN architecture is Named Data Networking (NDN) [5], whose design concept is the same as that of CCNx; however, there are several protocol differences. In this article, we focus on CCNx. To implement such an architecture, a *consumer* who wishes to retrieve content sends a request packet, referred to as *Interest*, with a *content name* to one or more adjacent upstream routers. As shown in Fig. 2, content names are expressed using hierarchical components similar to the URIs commonly used on the Internet. Large content such as videos are generally split into data *chunks*, which are uniquely identified by exact content names expressed in the form of `/example.com/video.mp4/%123` (or `/example.com/video.mp4/Chunk=123`) that consist of a content name prefix and chunk or sequence number. To differentiate contents with identical names, it is also possible to include a version number in the content name.

The key function that facilitates the forwarding and retrieving of content by name is name-based routing, which leverages distributed in-network caches on routers or nodes in a network. Figure 3 illustrates the components in a router. A router that receives an Interest first registers the receiving *face*[†] in its *Pending Interest Table (PIT)*, which maintains the outgoing interface and requested content name. The router then checks whether it holds the specified chunk in its cache space, which is called the *Content Store (CS)*. If the router holds the chunk in its CS, it copies the chunk into a *Data* (or *Content object*) packet and transmits the Data packet through the registered PIT entry. If the router does not hold the chunk in its CS, it checks whether it has a PIT entry with the same content name. If so, it recognizes that it has already received an Interest in the same chunk from downstream that was forwarded to upstream routers. Interest messages specifying the same content name are aggregated at the routers; a new Interest requesting the same chunk is not sent out because the corresponding Data will arrive at the router shortly. This PIT-based outgoing interface management intrinsically supports multicast-like communications.

[†]In CCNx, both the physical interface accessing a device and logical interface accessing an application are called faces.

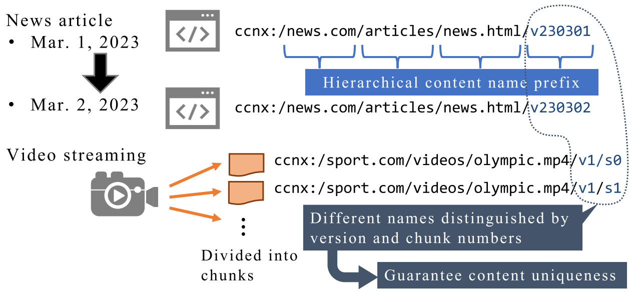
News article
• Mar. 1, 2023
• Mar. 2, 2023
ccnx:/news.com/articles/news.html/v230301
Hierarchical content name prefix
ccnx:/news.com/articles/news.html/v230302
Video streaming
ccnx:/sport.com/videos/olympic.mp4/v1/s0
ccnx:/sport.com/videos/olympic.mp4/v1/s1
Different names distinguished by version and chunk numbers
Divided into chunks
Guarantee content uniqueness

Fig. 2 Hierarchical content name prefix.

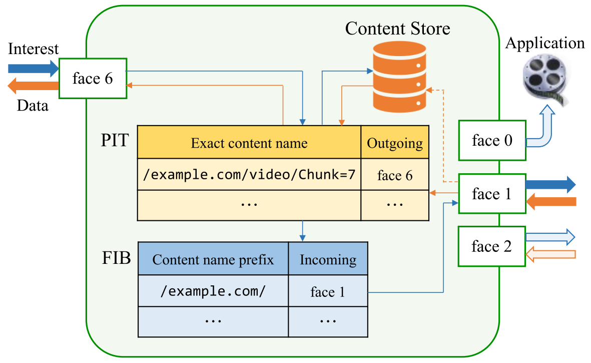

Fig. 3 Internal components of a router.

If the router does not have a PIT entry for the same content, it examines its routing table, called the *Forwarding Information Base (FIB)*, and forwards the Interest to upstream routers. Upstream routers are identified using the longest prefix that matches the content name prefix in the FIB. Interest forwarding is repeated at the intermediate routers on the path to the content *publisher* (or *producer*) in a hop-by-hop manner. Forwarding an Interest to upstream routers enables an intermediate router that has the specified chunk to forward Data packets through its corresponding PIT entries. After a Data packet has been forwarded through a PIT entry, that PIT entry is removed at the corresponding router. Consumers, therefore, trigger Data delivery on the reverse path along which Interests are forwarded. These procedures eliminate the need to access heavily loaded servers. Note that the sequence of processing PIT–CS–FIB is changed or performed in parallel depending on the implementation.

Publishers can optionally encrypt chunks and add signatures with encoded public keys into Data packets. Consumers then verify the content when they receive the Data packet. Additionally, consumers can sign an Interest using their private key to confirm validity, enabling publishers to verify the request.

In-network cache spaces distributed over a wide-area network are used as components of the data plane, and content name-based routing provides the optimal routes for retrieving content. Basic security is also considered because data breaches have unfortunately become common in modern society. These functions are closely related, and hence integrated and linked with protocols to support new applications and compatibility with various network technologies.

3. In-Network Caching

In-network caching is a distinguishing feature of ICN because it helps lower latency and minimize traffic. Cached content can be also served in the event of an outage at the original publisher. Effective cache deployment and replacement strategies play a crucial role in determining the overall performance of ICN. Caching strategies determine which content should be cached, where it should be stored, and when it should be deleted. Although many studies have been conducted on in-network caching, there are still numerous obstacles to be addressed. In this section, we focus on different cache deployment and replacement strategies, and discuss relevant recent studies.

3.1 Cache Deployment

Cache deployment in ICN can be categorized according to different targets. The first target is cache location, which can be further separated into on-path and off-path caching [6], along with routing paths. In on-path caching, returned Data packets are cached at routers along the traversed routing paths to consumers, whereas off-path caching allows Data packets to be cached at any router, regardless of their return paths. The on-path caching strategy is also known as transparent en-route caching [7], which caches all content received at all routers along the forwarding path. Ultimately, the caching decisions made by on-path caching policies are more innate in ICN because the path to the publisher and the cached content is congruent, and thus no additional cache discovery tasks are required. In contrast to on-path caching, off-path caching allows for content to be cached across the entire network to enhance content availability. Cache placement is determined using traffic analysis operations and monitoring methods. Off-path caching can be effective in a small network, but it may introduce scalability issues in cached content discovery tasks.

Many caching mechanisms to optimize various ICN performance metrics, including cache hit ratio, cache delivery distance, and network latency, have been explored. To reduce the cost of network traffic and offer higher caching utility, Huang et al. [8] proposed an on-path collaborative caching scheme in which each router along the path collaborates with other routers to forward content requests, eventually reaching the router that serves the target content. Additionally, the caching utility function calculates the utility of each content before removing the cache, thereby attaining a higher cache hit ratio. The work in [9] explored the correlation between the fresh and cached content of a node to design a path cache approach that allows for the caching of relevant content with higher correlation. It results in a notable enhancement in cache hit ratio and cache memory efficiency. In [10], the authors introduced an off-path content discovery and retrieval system based on content type. The most popular content is indexed and cached in the name resolution server to enable quick identification of data in the CS, which can effectively minimize the network traffic load and latency.

Another cache deployment target is cooperative caching. Strategies that consider this target include the leave copy everywhere (LCE), leave copy down (LCD), move copy down (MCD), and probabilistic caching (ProbCache) strategies. The most straightforward LCE strategy, presented in [11], simply caches Data packets at every intermediate node along the return path to increase the cache hit ratio and minimize latency. However, LCE could result in significant cache redundancy and insufficient cache storage by storing caches in multiple locations. In the LCD strategy, e.g., [12], content is cached at the router on the return route located one hop below where the cache hit has occurred. The idea behind LCD is to reduce the replication of caches of the same content along the route and effectively solve the cache redundancy problem of LCE. Subsequently, MCD [14] was proposed as an alternative to LCD that aims to improve content availability and reduce cache redundancy by discarding copies when a cache hit occurs and moving cached content to the underlying node closer to consumers. In ProbCache [15], content is replicated based on a probability value p and cached at the edge; otherwise, a copy with a probability value $(1 - p)$ will not be kept, and users must retrieve the cached content from the nearest edge router. As an alternative caching scheme for ICN, the authors of [16] proposed an inter-domain cache sharing mechanism to overcome the two-sided market issue when sharing in-network data between two ISP peers. This mechanism is based on Nash bargaining, which reduces traffic costs and pricing problems.

3.2 Cache Replacement

In-network caching is reliant not only upon cache deployment strategies but also replacement policies. It can be consolidated through a popularity-based mechanism to ensure that popular content is available nearby on the network and assumes that frequently requested content is more likely to be accessed in the future. However, because a cache space can reach its maximum capacity, a cache replacement policy to determine which content should be removed from the cache to allocate room for new content chunks is vital. The following cache replacement policies are the typical ones ICN supports.

- Random replacement (RR): RR randomly removes content and replaces it with newly arriving chunks.
- First in first out (FIFO): FIFO always removes the oldest cached chunks upon the arrival of new Data packets.
- Least recently used (LRU): LRU replaces the content that has not been requested for the longest period with new content.
- Least frequently used (LFU): LFU removes the cached content with the lowest frequency of access to store the most popular content and maximize the cache hit ratio.

A lightweight and efficient replacement scheme was presented in [17] based on the LRU policy. It divides data into low-inter-reference and high-inter-reference recency sets to perform replacement with minimum complexity and a high cache hit ratio. A Quality-of-Service aware replacement policy [18] was proposed to improve the network delay and cache utilization in vehicular networks. By classifying the traffic into different classes, the caching policy aims to cache not only the popular content closer to consumers but also the most important and long-term useful data. As another effective caching mechanism, utility-based cache optimization [19], [20] can be considered, which uses fixed utility functions over the cache hit ratio. As in [21], a utility-based replacement policy for ICN has been exploited to improve the cache hit ratio and cache diversity by keeping the most popular and recently used content items in the cache memory.

The choice of an appropriate caching policy is vital because it influences the cache hit ratio (and thus communication performance) and the load on publishers. Caching studies primarily concentrate on cache performance, but improvements in network performance focusing on mobility, throughput optimization, and load balancing for adaptive caching are necessary to maximize overall performance and applicability in deployment scenarios.

4. Routing

In ICN, content-name-prefix-based routing paths are constructed among the routers in a network. Because content can be retrieved from anywhere in the network, multiple paths are naturally supported. Multipath forwarding contributes to load balancing, robustness to packet loss, and data retrieval performance. When routers configure multiple neighboring (upstream) routers in their FIBs, an Interest can be forwarded to one or more neighbors. Data packets are then returned on the reverse path of the Interest packets; the fastest Data packet is forwarded to the consumer and the other Data packets are discarded at the routers because the corresponding PIT entries have been removed. Therefore, the way to coordinate FIB entries is an important factor for realizing efficient data acquisition.

4.1 Routing Protocols

In name-based routing protocols, routers advertise content name prefixes and identifiers to other routers in the network. The open shortest path first for named data (OSPFN) strategy [22] was proposed by the NDN project [23]. OSPFN extends OSPF over TCP/IP for name-based communication environments. OSPFN uses opaque link state advertisements to advertise name prefixes and supports multisource advertisements, unlike the original OSPF. Recently, the NDN project has targeted the development of pure NDN routing protocols and proposed the named data link state routing (NLSR) protocol [24]. NLSR uses Interest/Data packets for the route advertisement process to disseminate name-based routing messages through a network. Dijkstra's SPF algorithm has been adopted as a routing algorithm and extended to deliver hierarchical name prefix information. NLSR partially supports multipath information, which is an important feature of ICN. However, it cannot adequately handle a multisource context if multiple publishers/sources provide the same named content in a network.

Link state control routing (LSCR) [25] has been proposed for addressing multipath/multisource contexts. LSCR uses two types of link state advertisement (LSA): router LSA and anchor LSA. Router LSA is used for advertising the state of a router and routing cost of its outgoing links, and anchor LSA is used to advertise the presence of name prefixes. Unlike previous routing protocols such as OSPFN and NLSR, LSCR can construct multiple paths with low communication/computation overhead, particularly when multiple content servers in the network hold the same name prefix. Another important technical issue in multipath communications is the looping of Interest packets. In ICN, since Interest packets can be forwarded to multiple caching nodes, these packets can potentially loop in a network. To avoid this problem, loop-free import-dependent (LFID) routing [26] has been proposed as a directed-acyclic-graph-based next-hop selection method for NDN. LFID routing can adapt to various network events such as link failures and congestion while eliminating loops.

4.2 Forwarding Strategy

A forwarding strategy determines where to forward received Interest packets. The NDN adaptive forwarding strategy [27] has been developed and implemented in a forwarding daemon implementation called the NFD. In this forwarding strategy, each interface registered in a FIB entry of an NDN router is ranked by a forwarding policy defined by the operators. This strategy also provides a rate limit function by sending control packets called Interest NACKs (negative acknowledgements) to notify routers of congestion. It has been claimed that NDN adaptive forwarding can adapt to various network events such as link failure, congestion, and prefix hijacking. Carofiglio et al. [28] proposed an optimal multipath congestion control and request forwarding strategy for ICN to effectively utilize multipath communication resources. In this proposal, the number of PIT entries queueing on each interface at a router is used to represent the degree of congestion on each path. Weight-based stochastic forwarding allows this proposal to utilize multiple paths efficiently while balancing the congestion on branching paths.

4.3 Scalability

Unlike IP address prefixes, content names (or prefixes) cannot be aggregated in a routing table, and thus a huge number of names (prefixes) may be stored in a FIB. Intelligent and scalable routing protocols are therefore required. Hyperbolic routing (HR) [29] addresses this issue by leveraging greedy geometric routing based on the coordinates of the nodes in hyperbolic network geometry [30], [31]. HR does not maintain traditional forwarding tables and adopts a topology-agnostic approach to reduce computing overhead. HR could contribute to routing scalability, but since HR does not consider topological connectivity when computing the route, it sometimes falls into local optima, which can be avoided through backtracking using NDN forwarding.

Nguyen et al. proposed SmartFIB [32] to mitigate the downsides of route caching. SmartFIB separates the storage of routes into popular prefixes with high hit rates and repetitive route references that store ephemeral or unpopular routes. The advantage of this design lies in its utilization of eFIB, which is an ephemeral route storage space residing in slow memory that acts as both storage for one-off unpopular routes and a filter that protects the main FIB from including unpopular routes.

4.4 Interaction with In-Network Caching

It is important to maximize the utilization of in-network caches from a routing perspective. If a routing protocol advertises all cached content stored in the CS in the control plane, routing convergence is slow or practically impossible. Rossini et al. [33] proposed the nearest routing replica (NRR) as a practical routing method for utilizing the closest off-path cached content. NRR provides high performance when combined with the FIX [34] caching policy, which stores popular content in the network using probabilistic caching. Breadcrumbs [35] is a prominent off-path routing method that provides chunk-level and best-effort off-path cache discovery by utilizing in-network guidance with minimum communication overhead.

Li et al. [36] focused on another approach, combining caching and routing, and proposed a caching scheme called consecutive data chunk caching (ConCache) and a forwarding strategy called adaptive data chunk retrieval (ACUR). In ICN, cached content in a router can be mixed by customized caching policies such as FIX and LRU or request forwarding strategies. This leads to *missing teeth* at the chunk-level (i.e., chunks are not stored consecutively in the router). ConCache stabilizes latency performance by storing chunks consecutively and imposing an ACUR forwarding strategy to forward requested content with consecutive chunks to consumers. This strategy for in-network caching is valid not only at the transport level but also the application level [37]. It has been reported that the consecutiveness of cached content positively affects application layer performance. Therefore, networking layer technologies such as routing/forwarding/caching and high-level layer technologies such as transport and applications are deeply related to networking performance in ICN.

5. Transport

ICN transport is designed to provide flexibility. ICN defines a basic packet forwarding mechanism driven by name-based connectionless communication. ICN transport controls Interest–Data packet exchange through a consumer-driven hop-by-hop dynamic forwarding paradigm by maintaining the flow of Interest and Data packets on a path where one Interest packet requests one Data packet.

Unlike end-to-end connection-oriented TCP transport, ICN can make intermediate routers intervene in transport control and conduct native flexible multipath data retrieval. The forwarding strategies configured at the routers described in the previous section determine which face(s) are used for forwarding Interest to optimize Data reception. ICN transport is therefore another key factor that provides unique congestion and rate control algorithms using adaptive forwarding strategies.

5.1 Congestion Control

5.1.1 Consumer-Driven Approaches

Because consumers typically initiate data flow by sending Interest packets, most data traffic is consumer driven. Several consumer-driven techniques for flow and congestion control have been proposed and evaluated [38]. Most adopt a sliding window algorithm for simplicity and employ the classical and distinctive additive increase multiplicative decrease (AIMD) control at the consumer side. Unlike TCP, traditional congestion detection techniques with duplicate acknowledgements are not applicable in ICN because out-of-sequence Data packets appear frequently based on the nature of dynamic multipath communication. From another perspective, the presence of in-network cache and multiple paths makes it difficult for consumers to set the timeout values appropriately based on frequent variations in the observed round trip time (RTT). Because transport quality may impact the accuracy of RTT-based timeout mechanisms, unexpected performance degradations can occur. In this context, transport mechanisms using explicit congestion signaling have been proposed [39], [40] to improve performance.

To tackle the fundamental problem of unknown but available content sources in ICN transport, Carofiglio et al. [28] proposed a joint multipath congestion control and Interest forwarding mechanism. This approach explicitly accounts for the interplay between multipath flow control, in-network caching, and Interest forwarding at routers. Different routes thorough which Interest/Data travel can be distinguished using an explicit labeling technique [41] in which consumers keep separate estimates of the minimum and maximum RTTs observed on a given route. A consumer adopts delay-based window control according to the observed RTTs on each path. Routers implement a simple forwarding strategy for data traffic load balancing, which uses a randomly weighted algorithm to select an output interface for each incoming Interest to minimize the number of pending Interest packets.

Instead of using a window-based approach, Mahdian et al. [42] proposed a rate-based multipath-aware ICN congestion control approach in which routers update information regarding a fair sharing rate according to the link capacity and average traffic rate, and then insert this information into Data packets. Based on this information, consumers adjust their Interest sending rates. Extensive simulations demonstrated that this algorithm has significantly faster convergence and lower overshoot and oscillation than the classical approaches. It also allows a flow to fully utilize the network resources along all available paths while maintaining fairness among competing flows. Another scheme called PCON [40] adopts rate-based congestion control to exploit active queue management and send congestion signals to consumers, reducing the problem of tuning retransmission timeouts and removing the need for hop-by-hop Interest shaping mechanisms.

5.1.2 Interest Shaping at Routers

Several schemes adopting hop-by-hop Interest shaping [43]–[45] have been proposed to allocate network capacity among different content flows proactively. They provide effective congestion control and mitigate data loss, and may be a better option than traditional TCP-like mechanisms. In [43], the authors introduced an approach for routers to enforce fair sharing of bandwidth among content flows, which discards Interest packets if the incoming rate exceeds the current fair rate. To determine the shaping rate at routers, [44] adopted an estimated max–min fair rate and demonstrated that this scheme is stable and converges to an efficient max–min fair equilibrium. To improve the performance of Interest shaping schemes, a mechanism for bidirectional Interest rate shaping was proposed in [45] to consider the bandwidth consumption of Interests in the reverse direction, yielding near-optimal throughput.

5.2 Forwarding Strategy

An important benefit of ICN transport is adaptive forwarding, facilitated by the forwarding states maintained at routers. Yi et al. [46] proposed an adaptive forwarding strategy combined with a hop-by-hop Interest shaping scheme. The main objective is to select the best-performing face(s) for Interest forwarding to adapt to network conditions, automatically balance loads, and quickly detect and adapt to link failures. A periodic probing mechanism for available faces was also proposed to discover alternative available paths or paths with better performance. The authors demonstrated that a stateful forwarding plane handles short-term link failures and congestion more effectively than IP networks. Additionally, the authors of [47] argued that adaptive forwarding contributes to more scalable routing protocols by relaxing requirements related to convergence time and completeness.

Several probability-based adaptive strategies [48], [49] have been proposed to select faces for Interest forwarding based on a probability distribution and statistical information. In [48], to select appropriate faces with minimal delay, the FIB was modified to maintain the selection probability per face and name prefix, which was determined based on delay statistics. Stochastic adaptive forwarding was also proposed in [49] to provide more generic opportunities to adapt to network conditions by considering additional information such as hop count, transmission cost, and content types.

5.3 Cooperation with In-Network Computing

To maximize the benefits of ICN, various transport mechanisms cooperating with in-network computing have been proposed. ICN is synergistic with network coding because it facilitates the caching of Data packets throughout a network. For instance, to achieve higher quality of experience (QoE) for delay-sensitive streaming, Matsuzono et al. [50] exploited in-network caching and coding techniques to recover lost Data packets quickly and effectively while taking into account a limited allowable delay. The authors of [51] applied similar techniques to adaptive video streaming based on MPEG dynamic adaptive streaming over HTTP, which improved the cache hit ratio at in-network caches by efficiently leveraging multipath data retrieval. This improves QoE and reduces network resource utilization. However, it is desirable to resolve fairness issues and develop effective congestion control mechanisms in addition to improving overall throughput or latency.

ICN can exploit distributed computing resources and functions through automatic load distribution and failure resiliency. Remote method invocation in ICN [52] offers a sophisticated method for in-network computing invocation initiated by consumers who send Interests with a function name, to which the server responds with Data containing a *thunk* name and estimated completion time. The consumer then issues a new Interest with the received thunk name to fetch computation results after waiting for the indicated time. This method can reduce unnecessary Interest traffic and the size of Interest states maintained at routers.

ICN-based service function chaining (SFC) mechanisms [53]–[55], which execute multiple service functions for a content flow, have been investigated to facilitate sophisticated in-network computing through on-the-fly execution of multiple functions in an ordered set as required by a data flow, thereby enabling network-wide distributed computing. To enable highly scalable function calls, the authors of [56] proposed an elastic function offloading network (FON) that integrates ICN in SFC networks. FON can provide network service functionalities such as in-network caching and transcoding for SFC networks.

6. Security

The encryption of sensitive content is an essential and common practice to ensure content security. Basic CCNx [2], [3] supports private key cryptography using a shared secret key under the advanced encryption standard counter (AES-CTR) and advanced encryption standard cipher block chaining (AES-CBC) modes for Interest–Data communications. Additionally, CCNx allows publishers to sign each Data packet cryptographically with data chunk hashing using public key cryptography methods to provide content integrity.

However, various questions and tradeoffs may arise regarding efficiency and scalability, the strength of security algorithms, and privacy. In-network caching is beneficial for content delivery performance, but ubiquitous caching mechanisms can lead to security and privacy concerns because anyone can access stored content if it is not properly secured. The basic CCNx does not provide access control mechanisms because its authorization and authentication models are limited, making it difficult for publishers to control and monitor the content at each node. Therefore, it is necessary to implement additional security measures, including authentication, access control, and integrity, to guarantee content security.

6.1 Encryption

A data chunk can be encrypted by a publisher before distributing it on the network such that consumers with the correct decryption key can decrypt the content. There are various key criteria to consider to achieve fully secure content distribution in an ICN architecture.

- Cryptographic algorithms: Symmetric encryption is an effective solution for data security. However, cache utilization is limited because content is encrypted with a unique secret key for each consumer, which prevents other consumers from decrypting and accessing the same content from the cache. Additionally, consumer requests must reach the publisher for content and key retrieval, requiring publishers to be available continuously. To address this limitation, recent studies have considered public key cryptography to secure content [57]. However, most public key infrastructure (PKI)-based schemes naturally come with complex management infrastructure, and misconfigurations can lead to security vulnerabilities.
- Key management: The dissemination of secret keys through a secure channel is expensive and prone to key exposure in symmetric encryption. Additionally, the implementation of PKI-based schemes relies on a resource-intensive key management system and centralized certificate authority (CA) to manage certificates, incurring additional communication costs. A flexible and efficient key management system should be implemented for ICN to minimize the burden on consumers and other network entities [58].
- System performance: The straightforward utilization of advanced encryption algorithms can negatively impact system performance because encryption and decryption processes can be computationally intensive. A lightweight and reliable solution is required to reduce complexity, particularly in resource-constrained environments.

6.2 Access Control, Authorization, and Revocation

Content can be cached and distributed across multiple nodes, and hence access control is a vital concern. ICN allows publishers to encrypt content with access policies and assign authorization so that only legitimate consumers can access and retrieve content, but authorization delays and efficient revocation are major obstacles to implementing scalable access control mechanisms.

The authors of [57] proposed a secure video streaming service for ICN to improve confidentiality using content encryption and public key signatures. This system restricts viewer access and video quality through encryption based on multi-message/multi-authority ciphertext policy attributes, but questions remain regarding the management of multiple authorities for attribute control and scalable revocation. Wu et al. [59] presented an efficient access control framework to improve content security in information-centric edge networking. This framework utilizes confidentiality enhanced network coding by applying all-or-nothing transformation and encrypting only a small portion of an encoding matrix. Timestamp-based revocation is utilized to prevent revoked users from accessing content, which requires redistributing keys to all legitimate users after the access of some users has been revoked. To achieve robust security and fine-grained access control, a combination of private and public key cryptography is desirable.

6.3 Integrity, Authenticity, and Provenance

Content integrity is an important aspect of ensuring the authenticity of content that could be modified or corrupted in transit. This can be achieved by implementing digital signatures and applying additional cryptographic techniques so that consumers can verify the signature and provenance of the received content.

A privacy-preserving authentication scheme was proposed in [60] using broadcast encryption, in which every user request is authenticated at an edge router based on group signatures and hash chains. This design meets the stated security goals related to data confidentiality, accountability, and user authentication, but scalability issues with respect to the numbers of edge nodes and users remain. The authors of [61] compared different digital signature schemes and proposed feasible acceleration methods to reduce communication costs associated with content integrity and origin authentication. However, their framework adopts batch verification, introducing additional delay and complex digital certificate management problems.

To eliminate the need for certificates, information security continuous monitoring [62] implements a hybrid encryption-based access control mechanism. Content is encrypted using symmetric encryption, and symmetric keys are securely encrypted using identity-based proxy re-encryption at publishers. Edge routers re-encrypt encrypted symmetric keys to share access delegation with legitimate consumers and perform revocation. Additionally, identity-based signatures allow consumers to verify content authenticity. This method provides lightweight computation and communication for ICN entities while fulfilling content-centric security with cache utilization through certificate-free identity-based cryptography.

6.4 Availability and Accessibility

Distributed denial of service (DDoS) attacks and cache-oriented attacks can significantly influence content availability in ICN. An attacker can perform time analysis on cache contents to violate user privacy or request invalid and unpopular content to exhaust network resources. A cluster-based method for detecting cache pollution in secure ICN was proposed in [63]. A router calculates an incoming Interest metric and modifies its Interest popularity table. Then, a novel k-means clustering technique is used to classify Interests into separate clusters. Based on the clustering results, malicious false-locality and location-disruption attacks can be detected using an improved decision tree approach.

Various solutions for detecting and defending against such attacks have been proposed in the literature, but they generally compromise scalability and performance in large-scale networks. ICN requires robust security mechanisms to provide advanced detection for practical applications. Artificial Intelligence and Machine Learning (AI/ML) can be applied to detect anomalies in network traffic and identify suspicious content behavior in distributed ICN. The authors of [64] proposed an attack detection and mitigation mechanism based on user and network behaviors using a trained support vector machine. This scheme can efficiently detect false-locality attacks on content and probabilistically discard them to prevent cache pollution.

6.5 Trust

ICN provides authentication between publishers and consumers but cannot prevent data-poisoning attacks nor detect corrupted data. DCAuth [65] provides data-centric authentication for secure in-network data retrieval. To verify data cached in and transmitted through a network, a secure channel called the suspension chain is established between consumers and publishers through intermediate routers that seamlessly merge CA-based trust and neighbor-based trust with self-certified naming. Because each router along a Data forwarding path authenticates its neighboring routers, hop-by-hop Data transfer through the suspension chain is guaranteed.

Name-based trust management can be delegated using a hierarchical naming structure in ICN that reflects relationships between publishers and consumers. A suitable trust model ensures that the authenticated publisher of a name can sign Data packets and disseminate Data associated with that name through the network [66]. A trust schema was presented in [67], which includes well-defined trust policies to define links for content names and sign keys within the trust context.

Distributed ledger technology (DLT) can be seamlessly integrated with ICN to create an efficient and secure data storage and retrieval environment. The authors of [68] proposed a framework for protecting data in ICN using the immutability, decentralization, and security features of

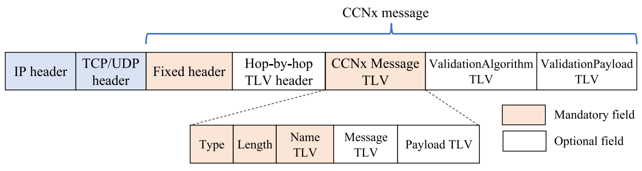

Fig. 4 CCNx-1.0 message containing Type-Length-Value (TLV) fields encoded in an IP packet.

blockchain technology to enhance data security throughout its lifecycle. An ICN-based user-centric in-network caching (UCINC) mechanism [69] cooperating with DLT regulates data caching in the network for users who have access privileges. An attribute-based encryption (ABE) technique such as CP-ABE [70] is applied to control access, and only legitimate users identified based on their attributes (e.g., locations) can retrieve trusted data through the ICN paradigm.

7. Cefore Implementation

7.1 Overview

To implement novel communication services, a combination of technical capabilities and strategic objectives is required. We have developed a software platform called *Cefore* [1], [71] that enables ICN communications and provided it as an open-source software with the BSD 3-clause license. Cefore consists of a no-frills forwarding daemon called *cefnetd* and CS manager daemon called *csmgrd* that provides in-network caches to cefnetd daemon(s). Cefnetd runs on Linux (Ubuntu), macOS, and Raspberry Pi OS, and csmgrd runs on Ubuntu and macOS. Cefore also includes useful network tools such as `cefputfile` and `cefgetfile` for uploading and downloading data, and utilities such as CCNinfo [72].

Cefnetd provides the Interest/Data forwarding function to handle CCNx-1.0 messages [2], [3] encoded in IPv4/v6 packets, as shown in Fig. 4. Cefnetd manages a FIB and PIT and embeds a lightweight FIFO-based caching mechanism (called the *local cache*) using its own memory (RAM).

Csmgrd is a caching daemon that creates cache spaces on its own RAM (called the *memory cache*) or UNIX filesystem (called the *filesystem cache*). Cefnetd connects to csmgrd through a UNIX socket or TCP and use it as their own CS. While the local cache on cefnetd facilitates high-speed data access, csmgrd isolates the caching implementation from cefnetd and reduces the dependence of cefnetd's resources. The filesystem cache increases the capacity of the cache space at a low cost.

Cefore daemons can be customized and enhanced by adding plugin libraries, as shown in Fig. 5. Users can develop the plugin libraries for their proposals and incorporate them into cefnetd and/or csmgrd without drastic changes in the base source code.

Users can set up cefnetd and csmgrd through configuration files such as `cefnetd.conf` and `csmgrd.conf` to allow changes in the parameters (e.g., in-network cache

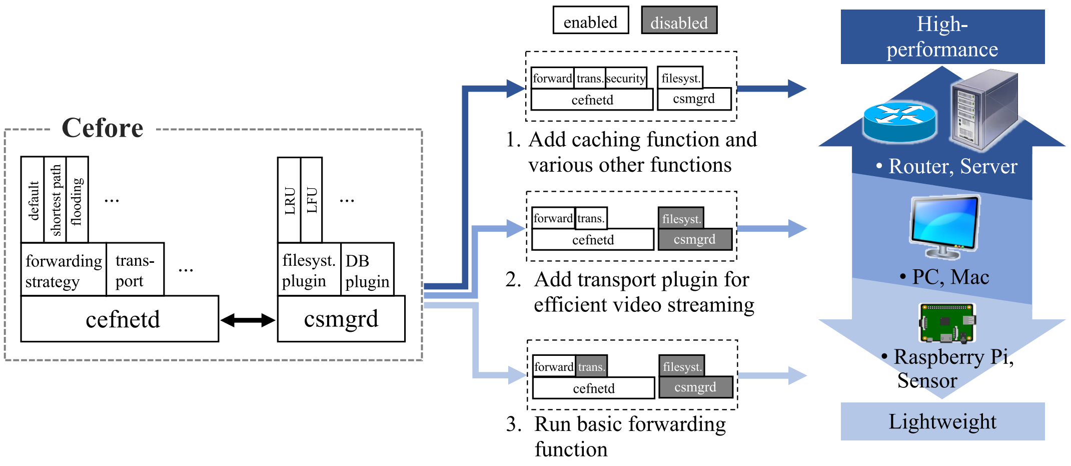

Fig. 5 Cefore implementation extensibility through plugin libraries and external daemons.

Table 1 Maximum throughput of Cefore running on physical environments.

OS	CPU	Memory (GB)	Throughput (Mbps)
Ubuntu 20.04	Xeon Gold 6248R (3.0 GHz × 24)	192	881
macOS Monterey	Intel Core i7 (3.1 GHz × 4)	16	878
Raspberry Pi OS	ARM Cortex-A72 (1.5 GHz × 4)	4	112

size (MB) and whether or not to use a filesystem cache). Users who develop plugin libraries configure their plugins in `plugin.conf`.

For the preliminary experiments, we set up four Cefore machines (ver. 0.10.0b): an Ubuntu PC running cefnetd as a publisher, and an Ubuntu PC, Mac, and Raspberry Pi running cefnetd as a consumer. The publisher and three consumers were connected via 1-Gbps Ethernet links. We measured the maximum throughput of fetching Data (encoded with CCNx-1.0 message TLVs over UDP/IP) using the `cefgetfile` command on each consumer. The Data size was 100 MB, and its block size was 8 KB. The pipeline size of `cefgetfile` was set to 32. Table 1 shows the results.

7.2 Symbolic Interest

Consumer-driven transport in CCNx allows applications to determine transport functionalities flexibly by selecting the Interest types used for Data acquisition. The regular Interest (RGI) defined in CCNx-1.0 specifications [2], [3] identifies content information in the form of data chunks uniquely identified by content names with chunk numbers. For example, if a video has a prefix of `ccnx:/example.com/video.mp4`, the name of the n-th chunk in the video is `ccnx:/example.com/video.mp4/Chunk=`n. RGI suffers from excessive overhead when requesting large content that is divided into multiple data chunks of equal size to the maximum transmission unit, each of which is assigned a unique identifier. To request a 10-Mbps video divided into 1.5 KB chunks, RGI sends approximately 800 requests per second. This not only imposes a large burden on the consumer side, but the transmission of a large number of Interest packets congests the network and increases the chance of Interest and Data packet loss, leading to greater latency for Data retrieval. Large data chunks can cause redundant Data traffic when a single fragment loss occurs because retransmissions lead to entire data chunks traversing the network again. These impacts are significant in broadband applications (e.g., high-definition video streaming), where low latency and high quality are required.

CCNx supports multicasting, meaning that when a router receives identical Interest packets requesting the same chunk, it registers all interfaces in its PIT but forwards only one packet to its upstream router. The upstream router then forwards the Data to all corresponding interfaces registered in its PIT. However, the period over which Interests are aggregated in a router is as short as the RTT. A typical RTT on the order of milliseconds is too short to reap the benefits of Interest aggregation, despite the high synchronicity of identical Interest packet arrivals.

To address these issues, we implemented a unique Interest message type called *symbolic Interest (SMI)* [50], [73]. SMI allows consumers to request Data without specifying the chunk numbers in the names. In the above example, consumers send an SMI with the prefix `ccnx:/example.com/video.mp4`, and intermediate routers create an SMI-specific PIT entry and forward the SMI toward the publisher. The consecutive chunks with the same prefix are then forwarded to the consumers until the specified lifetime (e.g., a few seconds). Unlike regular PIT entries created by RGI, SMI-specific PIT entries are deleted not after the chunks have been transmitted but when the lifetime timer (InterestLifetime) specified in the SMI (e.g., 2 s) expires. If a consumer sends a new SMI well before (e.g., 1 s before) the expiry of the InterestLifetime specified in the last SMI, they update the timer and keep receiving the corresponding Data.

An SMI makes the most of Interest aggregation and multicasting because SMI packets requesting Data with the same prefix are aggregated until the longer PIT entry expires. As long as consumers continue to receive the same named content, the aggregation of Interests and multicasting of Data packets will operate in an optimal manner without strict timing constraints. SMI substantially reduces the amount of Interest traffic and lowers latency by mitigating the effects of Interest loss in the network [50], [73].

By contrast, an SMI is not suitable for every type of application. An SMI is designed to receive real-time streaming or large sized data such as video content divided into multiple chunks. To request content with an exact content name (and chunk number if necessary), an RGI must be used. Additionally, flow balance is not maintained with an SMI. Because chunks of a data stream are continuously forwarded for a specified lifetime, it is not possible to control the rate/flow or request lost packet retransmission. Together with SMIs, the use of RGIs with a rate control mechanism or network coding according to the packet reception status/conditions will address this issue. Furthermore, if an SMI packet is lost, then stream reception stops for a few seconds; therefore, it is usually necessary to transmit the same SMI packets a few more times. Finally, because an SMI does not specify exact content names, it does not acquire chunks from the in-network cache.

To specify an SMI, Cefore currently encodes an SMI with a vendor-specific TLV field classified by a T_ORG type value, which is the National Institute of Information and Communications Technology's Private Enterprise Number (PEN) [74], in the name TLV field (see Fig. 2) to interoperate with other CCNx implementations.

7.3 Cefpyco Library

All source code and libraries for Cefore are written in the C language for optimization reasons. However, when developing applications that work with Cefore, implementation can be an obstacle because the C language requires specific redundant initialization, dynamic memory allocation and deallocation, and the frequent string manipulation of content names. To address this issue, we have developed and released a Python wrapper for Cefore applications called *Cefpyco* [75].

Cefpyco enables the utilization of basic Cefore functions for handling Interest and Data packets by importing Cefpyco modules and making function calls in Python applications running on Cefore. For example, in the C language, approximately 33 lines of code are required to send an Interest packet, but in Cefpyco, the same process can be implemented using only four lines.

Cefpyco consists of two main classes called `CefpycoHandle` and `CcnxPacketInfo`. `CefpycoHandle` acts as an intermediary for communication with Cefore and enables basic ICN communication such as sending and receiving Interests and Data. There are three main methods in `CefpycoHandle` for (1) sending an Interest/Data packet with a specified name (`send_interest`/`send_data`-method), (2) waiting to receive an Interest/Data packet for a specified period (default 4 s) and returning the received packet information (`receive`-method), and (3) registering the name prefix of an Interest that the application wishes to receive (`register`-method). `CcnxPacketInfo` is a class for storing information regarding received packets. The return value of the `receive`-method in `CefpycoHandle` can be used to view information regarding packets received from Cefore in Python code. Specifically, the content name, chunk number, payload contents, and payload length can be checked.

We now describe ways to create a consumer and publisher application that exchanges an Interest with the name `ccnx:/test/Chunk=0` and a corresponding Data packet. For both operations, it is necessary to initialize a connection when starting the application and close the connection with cefnetd when terminating the application. These processes are realized using the `begin`-method and `end`-method in `CefpycoHandle`, respectively. `CefpycoHandle` provides the `create_handle`-method so that processes can be described concisely together using a `with` statement.

On the publisher side, the `register`-method is used to notify cefnetd of the prefix name (i.e., `ccnx:/test`) provided by the publisher application. This creates an FIB for the publisher application in cefnetd and allows the publisher application to receive Interest packets with the name prefix from cefnetd. After detecting an Interest packet, the `send_data`-method is called to return the corresponding Data packet.

On the consumer side, the `send_interest`-method is called to issue the Interest named `ccnx:/test/Chunk=0`. Then, using the `receive`-method, the application only has to wait until the Data packet is returned. If the application wishes to fetch arbitrary content composed of many Data packets, this series of processes can be repeated while the received Data are verified.

Note that since Cefpyco is designed for developing Cefore applications, it does not support changes to the cefnetd and csmgrd daemons' internal behaviors, such as forwarding strategies and cache replacement policies.

8. Experiments

8.1 Cefore Network Emulator

For large-scale network experiments, we have developed the *CeforeEmu* network emulator based on Mininet [76]. By exploiting Mininet's lightweight approach of using OS-level virtualization features, CeforeEmu creates a virtual network topology with Linux containers that serve as virtual hosts on a Linux machine, where Cefore daemons can be launched. A network simulator such as ndnSIM [77] creates an environment that mimics the behaviors of ICN protocols and runs with simulation code and configurations that are different from the ones used in the real world. By contrast,

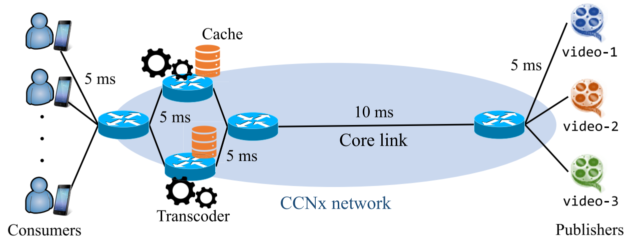

Fig. 6 Emulation environment setup in CeforeEmu.

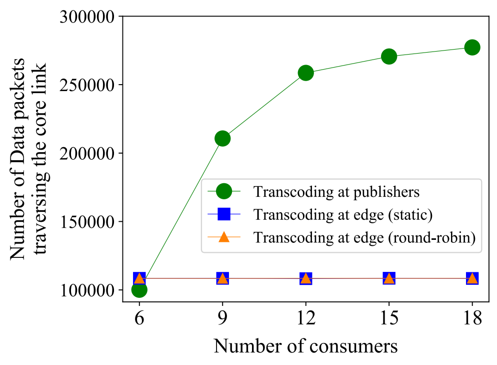

Fig. 7 Amount of Data traffic measured on CeforeEmu.

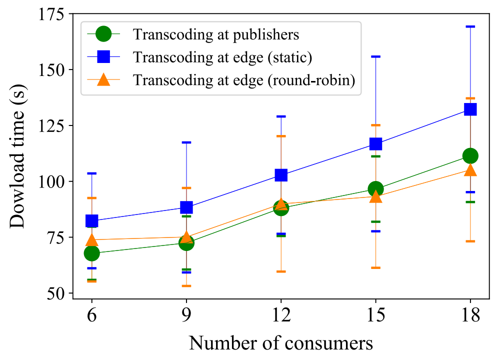

Fig. 8 Download time with/without transcoding using edge node(s).

CeforeEmu runs with Cefore daemons and verifies the protocols, in-network caching mechanisms, and configurations in the realistic virtual network environments. CeforeEmu enables actual running code (i.e., CCNx applications) to be tested and facilitates large-scale experiments.

To utilize CeforeEmu, Mininet must be installed in advance on a Linux machine using the `apt-get` command. Mininet can be readily installed on different operating systems using a virtual machine such as VirtualBox® or VMware® Fusion. This is accomplished by downloading the respective image files [76].

For experiments on CeforeEmu, users configure network parameters (e.g., the number of hosts or link delay/bandwidth/loss rate) in `cefemu.conf` to set up a virtual network topology and configure Cefore parameters in the common `cefnetd.conf`, `csmgrd.conf`, and `plugin.conf` files to run cefnetd and csmgrd on the virtual network.

We introduce an experiment for video data transmission on a virtual network on CeforeEmu, as illustrated in Fig. 6. In this experiment, three videos have been published and consumers invoke the `cefgetfile` command with RGI to fetch the video data by specifying a randomly selected content name (e.g., `ccnx:/video-1`). Each video content consists of 36,000 chunks, and the pipeline size of `cefgetfile` is set to 32. According to the playback capabilities of each consumer, consumers can specify the video quality as `high`, `middle-1`, `middle-2`, `low-1`, or `low-2` in addition to the content name, e.g., `ccnx:/video-1/middle-1`, for the next video data transmission.

In this experiment, when the publisher or one of the two edge nodes that caches the original video data (i.e., chunks) receives an Interest that specifies a video quality other than `high`, they transcode the chunk to the specified level and forward the Data packet to the consumer. Two edge nodes capable of transcoding have been set up, but each Interest is forwarded to and transcoded by the selected edge node in (1) a static manner or (2) a round-robin fashion. The transcoded chunk is stored in the edge node's cache and shared with other consumers for the upcoming requests.

Both transcoding and caching are expected to reduce the traffic (and bandwidth consumption) on the core link and provide lower-latency communication when compared with the original video data transmission. Figure 7 presents the amount of Data traffic on the core link depending on the number of consumers. If the number of consumers increases, the amount of traffic largely increases when transcoding is applied by the publisher, while it is stable when transcoding is applied by an edge node because the original video data are cached and reused. Figure 8 shows that the download time is highest when transcoding is processed at the fixed edge node. Because a single edge node is assigned both transcoding and caching duties for all transmitted video chunks without the ability to delegate tasks to other nodes, it is consistently overloaded. When the heavy tasks are shared among two edge nodes, the download time decreases. It is expected that more intelligent distributed processing shared among several edge nodes will yield more effective data forwarding in ICN.

8.2 Cefore on Docker Containers

Cloud-native ecosystems such as Docker [78] and Kubernetes [79] are emerging network softwarization (or microservice) technologies for edge/fog computing. Docker containers are beneficial for the operation and management of software resources and provide a way to deploy developed Cefore functions on a real network infrastructure quickly and easily, as follows.

- A Docker container is lightweight; therefore, we can build many containers on one physical machine. This enriches evaluation scenarios and improves the scalability of experiments.
- Docker containers can be easily and quickly initiated and terminated. This facilitates the convenience of testing and evaluating ICN services.

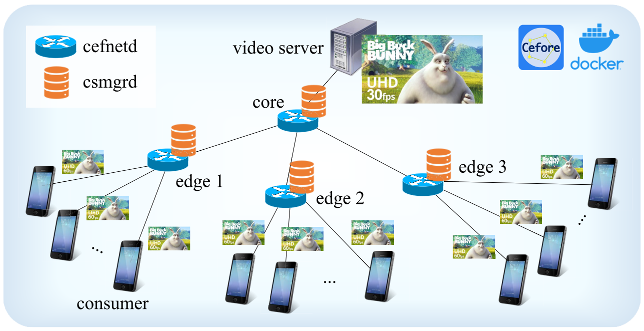

Fig. 9 4K multicast video streaming over the Docker-based Cefore platform.

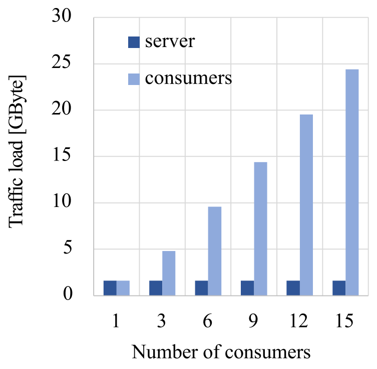

Fig. 10 Traffic load in 4K video streaming.

- Operational tools such as docker-compose[†] can be used to flexibly and quickly set up Docker containers for multiple ICN nodes providing different functions to coexist in a network.

We introduce the simple and rapid construction and deployment of ICN infrastructures and applications based on our recent work [80]. Figure 9 presents a model of a sample application on a Cefore platform. There are three types of microservices for the consumer, router, and publisher of a 4K/30 fps video. We converted an MP4 video downloaded from a website[††] into a transport streaming (MPEG-TS) file, and the publisher generates Data packets with a constant bit rate to perform real-time video streaming. We set the sending rate to 20 Mbps, which is equal to the encoding bitrate of 4K video. Each consumer sends SMI to receive the video stream from the router and publisher continuously.

The throughput performance is 20.9 Mbps at each consumer and one edge router (edge 1), which means that all consumers can receive the video stream with sufficient throughput. The edge routers receive Data packets for only one stream, indicating that multicasting works well at the router, which does not need to download the same video streams from the publisher.

Figure 10 presents the traffic load characteristics. We varied the number of consumers from one to fifteen and

[†]https://docs.docker.com/compose/
[††]https://peach.blender.org/

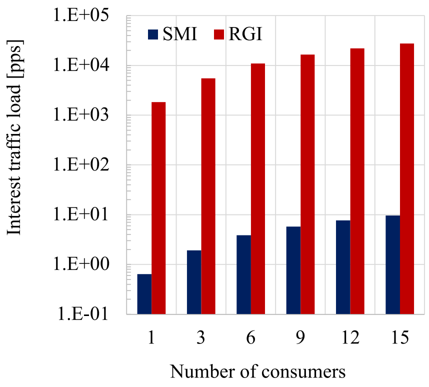

Fig. 11 Comparison between SMI and RGI.

measured the volume of transferred data traffic at the core router and consumers. As the traffic volume of all consumers accumulates, the traffic load of the core router does not increase with the number of consumers. With SMI, we can successfully aggregate Interests at the PIT on the branching router, even if the Data reception timings of consumers are different. These results confirm that efficient multicast communications can be well implemented in the Cefore platform.

Next, we conducted a comparative analysis of SMI and RGI, as presented in Fig. 11. While RGI makes a Data packet retrieval for one Interest transmission, SMI keeps sending Data packets until it times out. Like TCP's need to periodically send keepalive messages, SMI also requires the periodic transmission of Interest messages until a timeout occurs. However, SMI's approach significantly reduces overhead, leading to improved QoE, reduced latency, and less network congestion.

9. Conclusion

This article presented a survey of Information-Centric Networking (ICN) and its four major functionalities, namely (1) in-network caching, (2) routing, (3) transport, and (4) security. Recent related work was also reviewed. We also introduced an open-source software package called Cefore that provides a platform for CCNx-1.0-based communications. A network emulator called CeforeEmu and Docker containers running Cefore were also introduced.

ICN is a novel approach that evolves network infrastructure by introducing uniquely named content for data access. Although ICN can overcome various limitations and address certain issues associated with the Internet, it does not necessarily aim to replace current IP networks or infrastructures. It is feasible to implement ICN on top of IP networks and fuse it with various current and future networking technologies such as future mobile networks, where many high-mobility users require high data rates and low latency because ICN-based consumer mobility can be handled without updating location information/identifiers.

To determine how ICN can be implemented in next-generation services, researchers have been expanding upon conventional research topics such as in-network caching and routing algorithms. The mechanisms we presented in this article are appropriate, and there is no doubt that these conventional functions are the distinguishing features in ICN and provide various advantages. However, studies on fusion with modern and future networks such as edge computing and DLT as well as emerging applications such as connected cars, smart cities, and the metaverse will become more important in future.

At a paradigmatic level, we have a chance to move forward from theoretical research to practical studies because we now have actual running code and the experimental networks aforementioned. We hope to explore the extent of experimental studies that are the most intriguing with respect to ICN deployment.

References

[1] H. Asaeda, A. Ooka, K. Matsuzono, and R. Li, "Cefore: Software platform enabling content-centric networking and beyond," IEICE Trans. Commun., vol.E102-B, no.9, pp.1792–1803, Sept. 2019.

[2] M. Mosko, I. Solis, and C. Wood, "CCNx messages in TLV format," IRTF RFC 8609, July 2019.

[3] M. Mosko, I. Solis, and C. Wood, "CCNx semantics," IRTF RFC 8569, July 2019.

[4] "Information-Centric Networking Research Group," Available at: https://irtf.org/icnrg, accessed on April 10, 2023.

[5] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," SIGCOMM Comput. Commun. Rev., vol.44, no.3, pp. 66–73, July 2014.

[6] M. Dräxler and H. Karl, "Efficiency of on-path and off-path caching strategies in information centric networks," Proc. IEEE International Conference on Green Computing and Communications, pp.581–587, Besancon, France, 2012.

[7] P. Krishnan, D. Raz, and Y. Shavitt, "The cache location problem," IEEE/ACM Trans. Netw., vol.8, no.5, pp.568–582, Oct. 2000.

[8] L. Huang, Y. Guan, X. Zhang, and Z. Guo, "On-path collaborative in-network caching for information-centric networks," IEEE Conference on Computer Communications Workshops, 2017.

[9] D. Man, Q. Lu, H. Wang, J. Guo, W. Yang, and J. Lv, "On-path caching based on content relevance in information-centric networking," Computer Communications, vol.176, pp.272–281, 2021.

[10] S. Bayhan, L. Wang, J. Ott, J. Kangasharju, A. Sathiaseelan, and J. Crowcroft, "On content indexing for off-path caching in information-centric networks," Proc. ACM Conf. Inf.-Centric Netw., 2016.

[11] V. Jacobson, D.K. Smetters, J.D. Thornton, M.F. Plass, N.H. Briggs, and R.L. Braynard, "Networking named content," Proc. ACM CoNEXT, Rome, Italy, pp.1–12, Dec. 2009.

[12] D. Rossi and G. Rossini, "Caching performance of content centric networks under multi-path routing (and more)," Dept. Comput. Sci. Netw., Telecom ParisTech Ecole, Paris, France, Technical Report, 1, 2011.

[13] K. Cho, M. Lee, K. Park, T.T. Kwon, Y. Choi, and S. Pack, "WAVE: Popularity-based and collaborative in-network caching for content-oriented networks," Proc. IEEE INFOCOM Workshops, 2012.

[14] N. Laoutaris, S. Syntila, and I. Stavrakakis, "Meta algorithms for hierarchical Web caches," IEEE International Conference on Performance, Computing, and Communications, 2004.

[15] I. Psaras, W.K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," Proc. ACM SIGCOMM ICN workshop, New York, USA, Aug. 2012.

[16] X. Shao, H. Asaeda, M. Dong, and Z. Ma, "Cooperative inter-domain cache sharing for information-centric networking via a bargaining game approach," IEEE Trans. Netw. Sci. Eng., vol.6, no.4, pp.698–710, Sept. 2018.

[17] X. Wei and Y. Liu, "Low inter-reference recency set replacement policy in named data networking," Proc. IEEE International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), pp.1155–1160, Chongqing, China, Dec. 2020.

[18] H. Khelifi, S. Luo, B. Nour, and H. Moungla, "A QoS-aware cache replacement policy for vehicular named data networks," Proc. IEEE Globecom, Waikoloa, USA, Dec. 2019.

[19] G. Neglia, D. Carra, and P. Michiardi, "Cache policies for linear utility maximization," Proc. IEEE INFOCOM, Atlanta, USA, May 2017.

[20] M. Dehghan, L. Massoulie, D. Towsley, D. Menasche, and Y.C. Tay, "A utility optimization approach to network cache design," IEEE/ACM Trans. Netw., vol.27, no.3, pp.101–1027, 2019.

[21] Q.N. Nguyen, J. Lopez, T. Tsuda, T. Sato, K. Nguyen, M. Ariffuzzaman, C. Safitri, and N.H. Thanh, "Adaptive caching for beneficial content distribution in information-centric networking," Proc. International Conference on Information Networking (ICOIN), Barcelona, Spain, Jan. 2020.

[22] L. Wang, et al., "OSPFN: An OSPF based routing protocol for named data networking," NDN Technical Report NDN-0003, July 2012.

[23] "Named Data Networking," Available at: http://named-data.net/, accessed on April 10, 2023.

[24] L. Wang, V. Lehman, A.K.M.M. Hoque, B. Zhang, Y. Yu, and L. Zhang, "A secure link state routing protocol for NDN," IEEE Access, vol.6, pp.10470–10482, 2018.

[25] E. Hemmati and J.J. Garcia-Luna-Aceves, "A new approach to name-based link-state routing for information-centric networks," Proc. ACM ICN'15, pp.29–38. 2015.

[26] K. Schneider, B. Zhang, and L. Benmohamed, "Hop-by-hop multipath routing: Choosing the right nexthop set," Proc. IEEE INFOCOM 2020, Toronto, Canada, pp.2273–2282, July 2020.

[27] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," SIGCOMM Comput. Commun. Rev., vol.42, no.3, pp.62–67, July 2012.

[28] G. Carofiglio, M. Gallo, L. Muscariello, M. Papalini, and S. Wang, "Optimal multipath congestion control and request forwarding in Information-Centric Networks," Proc. IEEE ICNP 2013, Goettingen, Germany, pp.1–10, 2013.

[29] V. Lehman, A. Gawande, B. Zhang, L. Zhang, R. Aldecoa, D. Krioukov, and L. Wang, "An experimental investigation of hyperbolic routing with a smart forwarding plane in NDN," Proc. IEEE/ACM IWQoS 2016, Beijing, China, pp.1–10, June 2016.

[30] M. Boguñá, F. Papadopoulos, and D. Krioukov, "Sustaining the Internet with hyperbolic mapping," Nat. Commun., vol.1, 62, Sept. 2010.

[31] F. Papadopoulos, D. Krioukov, M. Boguna, and A. Vahdat, "Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces," Proc. IEEE INFOCOM, San Diego, USA, pp.1–9, March 2010.

[32] D. Nguyen and H. Asaeda, "SmartFIB: Low-replacement route storage for large name space forwarding," Proc. IEEE CCNC, Las Vegas, USA, Jan. 2020.

[33] G. Rossini and D. Rossi, "Coupling caching and forwarding: Benefits, analysis, and implementation," Proc. ACM ICN 2014, New York, USA, pp.127–136, 2014.

[34] S. Arianfar, P. Nikander, and J. Ott, "On content-centric router design and implications," Proc. ACM ReARCH'10, New York, NY, USA, Nov. 2010.

[35] E.J. Rosensweig and J. Kurose, "Breadcrumbs: Efficient, best-effort content location in cache networks," Proc. IEEE INFOCOM 2009, Rio de Janeiro, Brazil, pp.2631–2635, April 2009.

[36] R. Li, K. Matsuzono, H. Asaeda, and X. Fu, "Consecutive caching and adaptive retrieval for in-network big data sharing," Proc. IEEE ICC, Kansas City, USA, May 2018.

[37] Y. Hayamizu, K. Goto, M. Bandai, and M. Yamamoto, "QoE-aware bitrate selection in cooperation with in-network caching for

information-centric networking," IEEE Access, vol.9, pp.165059–165071, Dec. 2021.

[38] Q. Chen, R. Xie, F.R. Yu, J. Liu, T. Huang, and Y. Liu, "Transport control strategies in named data networking: A survey," IEEE Commun. Surveys Tuts., vol.18, no.3, pp.2052–2083, 2016.

[39] G. Carofiglio, L. Muscariello, M. Papalini, N. Rozhnova, and X. Zeng, "Leveraging ICN in-network control for loss detection and recovery in wireless mobile networks," Proc. ACM ICN, Kyoto, Japan, Sept. 2016.

[40] K. Schneider, C. Yi, B. Zhang, and L. Zhang, "A practical congestion control scheme for named data networking," Proc. ACM ICN, Kyoto, Japan, Sept. 2016.

[41] G. Carofiglio, M. Gallo, L. Muscariello, and M. Papali, "Multipath congestion control in content-centric networks," Proc. IEEE INFOCOM NOMEN Workshop, 2013.

[42] M. Mahdian, S. Arianfar, J. Gibson, and D. Oran, "MIRCC: Multipath-aware ICN rate-based congestion control," Proc. ACM ICN, Kyoto, Japan, Sept. 2016.

[43] S. Oueslati, J. Roberts, and N. Sbihi, "Flow-aware traffic control for a content-centric network," Proc. IEEE INFOCOM, Orlando, USA, March 2012.

[44] G. Carofiglio, M. Gallo, and L. Muscariello, "Joint hop-by-hop and receiver-driven interest control protocol for content-centric networks," SIGCOMM Compu. Commun. Revl., vol.42, no.4, pp.491–496, 2012.

[45] Y. Wang, N. Rozhnova, A. Narayanan, D. Oran, and I. Rhee, "An improved hop-by-hop interest shaper for congestion control in named data networking," SIGCOMM Comput. Commun. Rev., vol.43, no.4, pp.55–60, 2013.

[46] C. Yi, A. Afanasyev, and I. Moiseenko, "A case for stateful forwarding plane," Elsevier Comput. Commun., vol.36, no.7, pp.779–791, 2013.

[47] C. Yi, J. Abraham, and A. Afanasyev, "On the role of routing in named data networking," Proc. ACM ICN, Paris, France, Sept. 2014.

[48] H. Qian, R. Ravindran, G.-Q. Wang, and D. Medhi, "Probability-based Adaptive Forwarding Strategy in Named Data Networking," Proc. IFIP/IEEE International Symposium on Integrated Network Management (IM), Ghent, Belgium, May 2013.

[49] D. Posch, B. Rainer, and H. Hellwagner, "SAF: Stochastic adaptive forwarding in named data networking," IEEE/ACM Trans. Netw., vol.25, no.2, pp.1089–1102, 2017.

[50] K. Matsuzono, H. Asaeda, and T. Turletti, "Low latency low loss streaming using in-network coding and caching," Proc. IEEE INFOCOM, Atlanta, USA, pp.1–9, May 2017.

[51] J. Saltarin, E. Bourtsoulatze, N. Thomos, and T. Braun, "Adaptive video streaming with network coding enabled named data networking," IEEE Trans. Multimedia, vol.19, no.10, pp.2182–2196, 2017.

[52] M. Król, K. Habak, D. Oran, D. Kutscher, and I. Psaras, "RICE: Remote method invocation in ICN," Proc. ACM ICN, Boston, USA, pp 1–11, Sept. 2018.

[53] M. Arumaithurai, J. Chen, E. Monticelli, X. Fu, and K.K. Ramakrishnan, "Exploiting ICN for flexible management in software-defined networks," Proc. ACM ICN, 2014.

[54] L. Liu, Y. Peng, M. Bahrami, L. Xie, A. Ito, S. Mnatsakanyan, G. Qu, Z. Ye, and H. Guo, "ICN-FC: An information-centric networking basedframework for efficient functional chaining," Proc. IEEE ICC, Paris, France, May 2017.

[55] K. Kanai, T. Tsuda, H. Nakazato, and J. Katto, "Information-centric service mesh for autonomous in-network computing," Proc. ACM ICN, poster session, Osaka, Japan, Sept. 2022.

[56] Y. Hayamizu, K. Matsuzono, T. Hirayama, and H. Asaeda, "Design and implementation of ICN-based elastic function offloading network for SFC," Proc. IEEE NetSoft, Online, June 2021.

[57] F. Khan and H. Li, "Ensuring trust and confidentiality for adaptive video streaming in ICN," J. Commun. Netw., vol.21, no.6, pp.539–547, 2019.

[58] D.K. Smetters, P. Golle, and J.D. Thornton, "CCNx access control specifications," PARC, Technical Report, July 2010.

[59] D. Wu, Z. Xu, B. Chen, Y. Zhang, and Z. Han, "Enforcing access control in information-centric edge networking," IEEE Trans. Commun., vol.69, no.1, pp.353–364, 2021.

[60] K. Xue, P. He, X. Zhang, Q. Xia, D.S.L. Wei, H. Yue, and F. Wu, "A secure, efficient, and accountable edge-based access control framework for information centric networks," IEEE/ACM Trans. Netw., vol.27, no.3, pp.1220–1233, 2019.

[61] Y. Yu, Y. Li, X. Du, R. Chen, and B. Yang, "Content protection in named data networking: Challenges and potential solutions," IEEE Commun. Mag., vol.56, no.11, pp.82–87, Nov. 2018.

[62] H.H. Hlaing and H. Asaeda, "Ensuring content integrity and confidentiality in information-centric secure networks," Proc. IEEE CCNC, Las Vegas, USA, Jan. 2023.

[63] A. Gupta and P. Nahar, "Detection of cache pollution attacks in a secure information-centric network," Data Analytics and Management, LNDECT, vol.54, pp 377–397, Springer, 2021.

[64] Y. Cao, D. Wu, M. Hu, and S. Chen, "Detection and defense schemes for cache pollution attack in content-centric network," Proc. ICENAT 2022, pp.614–629, Springer, 2022.

[65] R. Li, H. Asaeda, and J. Wu, "DCAuth: Data-centric authentication for secure in-network big-data retrieval," IEEE Trans. Netw. Sci. Eng., vol.7, no.1, pp.15–27, Sept. 2018.

[66] C. Tschudin, E. Uzun, and C.A. Wood, "Trust in information-centric networking: From theory to practice," Proc. ICCCN, Waikoloa, HI, USA, pp.1–9, 2016.

[67] Y. Yu, A. Afanasyev, D. Clark, K. Claffy, V. Jacobson, and L. Zhang, "Schematizing trust in named data networking," Proc. ACM ICN 2015, pp.177–186, San Francisco, USA, Sept. 2015.

[68] R. Li and H. Asaeda, "A blockchain-based data lifecycle protection framework for information-centric network," IEEE Commun. Mag., vol.57, no.6, pp.20–25, June 2019.

[69] H. Yamanaka, Y. Teranishi, Y. Hayamizu, A. Ooka, K. Matsuzono, R. Li, and H. Asaeda, "User-centric in-network caching mechanism for off-chain storage with blockchain," Proc. IEEE ICC, June 2022.

[70] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute based encryption," Proc. 28th IEEE Symposium on Security and Privacy, pp.321–334, Oakland, USA, 2007.

[71] "Cefore: Information-centric networking platform," Available at: https://cefore.net, accessed on April 10, 2023.

[72] H. Asaeda, A. Ooka, and X. Shao, "CCNinfo: Discovering content and network information in content-centric networks," IRTF RFC 9344, Feb. 2023.

[73] K. Matsuzono and H. Asaeda, "NRTS: Content name-based real-time streaming," Proc. IEEE CCNC, Las Vegas, USA, Jan. 2016.

[74] "Private Enterprise Numbers (PENs)," IANA, https://www.iana.org/assignments/enterprise-numbers/

[75] "Cefpyco," Avaiable at: https://github.com/cefore/cefpyco/, accessed April 10, 2023.

[76] "Mininet," Available at: http://mininet.org/, accessed April 10, 2023.

[77] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the evolution of ndnSIM: An open-source simulator for NDN experimentation," ACM SIGCOMM Comput. Commun. Rev., vol.47, no.306, pp.19–33, Sept. 2017.

[78] https://www.docker.com/, accessed on April 10, 2023.

[79] https://kubernetes.io/, accessed on April 10, 2023.

[80] "Tutorial: CCNx-based Cloud-Native Function: Networking and Applications," Available at: https://conferences2.sigcomm.org/acm-icn/2022/tutorial-cefore.html, accessed on April 10, 2023.

Hitoshi Asaeda is currently a Director of the Network Architecture Laboratory, National Institute of Information and Communications Technology (NICT) and also a Collaborative Professor with the Graduate School of Informatics and Engineering, the University of Electro-Communications (UEC). He holds a Ph.D. degree from Keio University. He was previously with IBM Japan, Ltd. and a Research Engineer Specialist at INRIA Sophia Antipolis, France. He was a Project Associate Professor at Keio University from 2005 to 2012. He was a Guest Editor-in-Chief of the special series of IEICE Trans. Commun. in 2016. He was a Chair of the IEICE Technical Committee on ICN from 2017 to 2019. He served as a General Chair of IEEE/ACM IWQoS 2021 and ACM ICN 2022 and has been a TPC member for premier conferences such as IEEE INFOCOM, WCNC, and ACM ICN. He was a Program Officer for several international projects and has been actively working in the IETF standards body. He received the IEICE Communications Society Outstanding Contributions Award in 2019. His research interests include ICN, network coding, high-quality streaming, and large-scale testbeds. He is a Senior Member of the IEEE and a Member of the ACM.

Kazuhisa Matsuzono is a Senior Researcher at the Network Architecture Laboratory, NICT. He received the Ph.D. from Keio University in 2012. He was a Post-doctoral Fellow at INRIA in 2013. His research interests include transport protocols for multimedia flows, network coding, and information-centric networks. He has been a Vice-Chair of the IEICE Technical Committee on ICN since 2019. He is a Member of the IEEE.

Yusaku Hayamizu is a Researcher at the Network Architecture Laboratory, NICT. He received his B.E., M.E., and Ph.D. degrees in engineering from Kansai University in 2014, 2016, and 2019, respectively. His research interests include computer networks, information-centric networks, traffic control, in-network computing, and network softwarization. He is the recipient of Best Paper Awards from the 2017 IEEE CQR Workshop and the 2018 IEEE LANMAN Symposium. He is a Member of the ACM and IEEE.

Htet Htet Hlaing received a Ph.D. degree from Kanazawa University in 2021 with a MEXT Scholarship. Following her doctoral studies, she has continued her research as a Researcher at the Network Architecture Laboratory at NICT. Her research interests include information security and privacy, ICN, and distributed ledger technologies. She is a Member of the IEEE.

Atsushi Ooka is a Researcher at the Network Architecture Laboratory at NICT. He received his M.E. and Ph.D. degrees from the Graduate School of Information Science and Technology, Osaka University, in 2014 and 2017, respectively. His research interests include the design and implementation of routers in content-centric networking. He is a Member of the IEEE.