Journal of Information Processing Vol.25 45–55 (Jan. 2017)

[DOI: 10.2197/ipsjjip.25.45]

Regular Paper

Effectiveness of Service-oriented Router for ISP-CDN Collaboration

Janaka L. Wijekoon[1,a)] Erwin H. Harahap[2,b)] Rajitha L. Tennekoon[1,c)] Hiroaki Nishi[3,d)]

Received: April 5, 2016, Accepted: October 4, 2016

Abstract: This article discusses a novel method to strengthen the collaboration between Internet service providers (ISPs) and content delivery networks (CDNs). CDNs are becoming the primary data delivery method in information communication technology environments because information sharing via networks is becoming the driving force of the future Internet. Moreover, it is anticipated that network routers will be equipped with additional processing power and storage modules for providing efficient end-user services. Consequently, this article studies the effectiveness of introducing a Service-oriented Router (SoR) to strengthen the ISP-CDN collaboration to leverage DNS-based request redirection in CDNs. In contrast, the proposed method yields better performance in user redirection and network resource utilization, suggesting that using SoR may a future business model which addresses adequate ISP-CDN collaboration.

Keywords: content delivery network (CDN), Service-oriented Router (SoR), ISP-CDN collaboration, SLRouting, CDN redirection.

1. Introduction

The study [1] provides live statistics of the Internet usage, and the statistics show that more than 40% of the world population has an Internet connection today. Furthermore, the statistics show that people are interested in information sharing via networks, e.g., 500,000 tweets per minutes. Recent traffic analysis studies indicate that this traffic is hosted by online service providers (OSPs) including large content providers, e.g., Google and Yahoo, and content delivery networks (CDNs), e.g., Akamai and Netflix [2], [3]. Moreover, such studies incur that the OSPs use a large number of Internet service provider (ISP) data centers to build their networks near their subscribers according to either proximity or end-to-end latency [3].

In general, CDNs use modified DNS architectures, such as FreeFlow DNS [4], [5], to redirect their subscribers to the nearest content server, i.e., CDN request redirection (RR) [4], [6]. However, CDNs redirect their subscribers without being acutely aware of the ISP network conditions [7]. ISPs, on the other hand, blindly route huge volumes of traffic introduced by CDNs without considering the traffic matrix of CDNs as a parameter for route selection [8], [9]. Yet, Ref. [10] anticipates that the network resource utilization, i.e., network path selection, is highly contingent of the traffic matrix. Therefore, the effectiveness of appropriate RR and network resource utilization is questionable in the absence of a reliable collaboration between the ISP and CDN, e.g., Ref. [1] states that averagely a Google query has to travel about 1,500 miles to a data center and back return to the end-user.

To this end, several recent studies [7], [8], [9], [11] indicated the necessity of ISP-CDN collaboration for foreseeable improvements of either RR or network resource utilization. Similarly, we discussed the effectiveness of edge routers for accelerating the CDN RR process [12]. Given the network routers are the building blocks of information communication technology (ICT) networks, there is an emerging technological trend of strategic improvements for network routers to scale-up end-user services in future networks [13], [14], [15]. Consequently, in Refs. [16], [17], Inoue et al. proposed the Service-oriented Router (SoR), which is a router consisting of a packet analyzing processor and an ample in-network storage module. The SoR was proposed with the aim of providing content-based services by analyzing packet streams and storing necessary packet stream information in high-throughput databases [18].

This study discusses the notion of introducing SoRs as the gateway routers of an ISP network as an improvement of our former study presented in Ref. [12]. In general, ISP network-state information is useful in eliminating bottlenecks and avoiding flash crowds, whereas CDN server-state information is helpful in determining the least-busy servers [10]. In essence, ISPs and CDNs collaboratively can use such information to select best servers to their subscribers, and thus, forward the subscribers to the selected server through least busy network paths. Therefore, as illustrated in **Fig. 1**, we strategically placed the SoRs at both user and con-

[1] Hiroaki Nishi Laboratory, Department of System Design Engineering, Faculty of Science and Technology, Keio University, Yokohama, Kanagawa 223–8522, Japan
[2] Department of Mathematics, Faculty of Mathematics and Natural Sciences, Bandung Islamic University, Jl. Hariangbanga No.2, Tamansari, Bandung Wetan, Kota Bandung, Jawa Barat 40132, Indonesia
[3] Department of System Design Engineering, Faculty of Science and Technology, Keio University, Yokohama, Kanagawa 223–8522, Japan
[a)] janaka@west.sd.keio.ac.jp
[b)] erwin2h@unisba.ac.id
[c)] rajitha@west.sd.keio.ac.jp
[d)] west@sd.keio.ac.jp

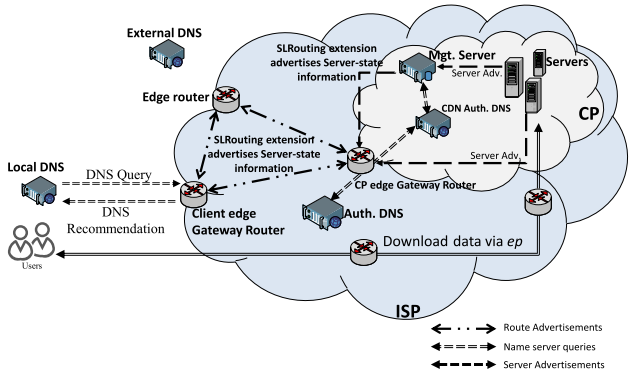


Fig. 1 Proposed network infrastructure with SoRs.

tent providers edges to collect and share both ISP network-state and CDN server-state information.

SoRs collect the ISP network-state information using the routing tables; this study assumes the ISPs are configured with SLRouting protocol [19]. Moreover, the SLRouting protocol is updated to advertise the CDN server-state information to the provider's edge gateway routers, i.e., SoRs. Consequently, working as an intermediary, SoRs collect necessary information from ISPs and CDNs, and use the collected information to leverage DNS-based CDN RR. Particularly, SoRs use information to reduce the delay caused by hierarchical name server resolution process, which is a fundamental limitation of DNS-based CDN RR discussed in Refs. [5], [6]. Thereby, SoRs attempt to accelerate the connection initiation process, and thus, adapt the content server changes on small time scales.

Furthermore, the packet analyzing capability of the SoRs [20], [21], [22] is used to provide on the fly content-based packet redirection; redirect packets by analyzing application layer information, i.e., information provided in unified resource locators (URLs). We implemented a prototype of the proposed architecture in network simulator-3 (ns-3) [23] by using the topology information of the WIDE network [24]. Furthermore, we compared the proposed collaborative redirection method with DNS-based CDN RR which is the primary redirection method used by most CDNs [5].

The rest of the paper is organized as follows. Section 2 explains the proposed method in detail. Moreover, Section 2 provides a detailed explanation of SoR and its principal components (see **Fig. 2**). SoR is currently an ongoing research project, and hence, we introduce a software SoR in this study for explanation purposes. Furthermore, Section 2 provides how proposed method utilizes SLRouting protocol for network path selection and for collecting content server information. Section 3 elaborates the message flow, packet interception, and recommendation process of the proposed architecture. The simulation scenarios and the yielded results are reported in Section 4 followed by discussing the business facts, technical facts, and limitations of the proposed architecture in Section 5. Section 6 briefs the related work for this study, and Section 7 concludes the article.

2. Methodology

The increasing challenges of information sharing in ICT networks and confronted end-user experience improvement motivated us to propose the network architecture illustrated in Fig. 1. From the users' point of view, ISPs and CDNs should operate as a single system that ensures fast and uninterrupted data delivery to their subscribers. As previously stated, various studies such as Refs. [7], [8], [9], [11] pointed out the necessity of ISP-CDN collaboration to achieve this objective. ISP-CDN collaboration can be explained as the process that ensures the end-users are redirected to the best server via the best path. Therefore, as depicted in Fig. 1, we propose a novel approach that uses gateway routers to strengthen the ISP-CDN collaboration, and thus, improving the end-user experience.

2.1 Placing SoR as an Gateway Router

In a computer network, routers are indispensable devices that maintain the network connectivity and "traffic directing" function. In particular, a gateway router, i.e., a router that maintains the connection between the ISP and end hosts, is significant because it conserves the traffic-forwarding policies of the data plane [25]. Therefore, we propose to use gateway routers to collect the ISP topology and CDN traffic information in order to collaboratively use such information for providing foreseeable CDN services.

Although conventional routers have the packet-forwarding capability, they lack the packet-analyzing capability and in-network storage modules. Therefore, in the proposed network, gateway routers have been replaced by SoRs. In strategy, gateway routers are selected because 1) end-users and servers can access corresponding gateway router within a one-hop distance, 2) edge computing is gaining momentum [26], and hence, edge routers are used to provide content-based services [14], [15], and 3) gateway routers can resolve DNS queries within a one-hop distance; queries need not be sent to the public network (i.e., ISP authoritative DNS) each time the local DNS server queries the hierarchy of name servers to find the nearest content server [5].

2.2 SoR and its Main Components

SoR, a router with an ample storage module and packet processing cores, has been proposed with the aim of providing content-based services, i.e., edge computing services, from the router [16], [17]. Consequently, SoR performs deep packet inspection (DPI) to analyze packet content and stores the necessary information in high-throughput databases [18]. The stored information is then used to provide content-based services, i.e., recommendation, to the end-users from the edge routers as explained in Refs. [20], [27]. Lately, Akamai, Cisco, and Qualcomm proposed methods of using edge routers to provide end-user services by assuming the edge computing services [26], [28].

In a concept similar to SoR, major router manufacturers (e.g., Cisco, Juniper, and Alaxala) have recently revealed programming interfaces that allow packet manipulation by third-party applications and the addition of new services to routers [29], [30], i.e., process Internet of Things data at the edge routers [26]. In addition, studies such as Ref. [31] proposed to implement a high-performance router by shrinking a rack of computers into a single hardware box. Further, Intel introduced a packet processing framework, Intel data plane development kit (DPDK) [32],

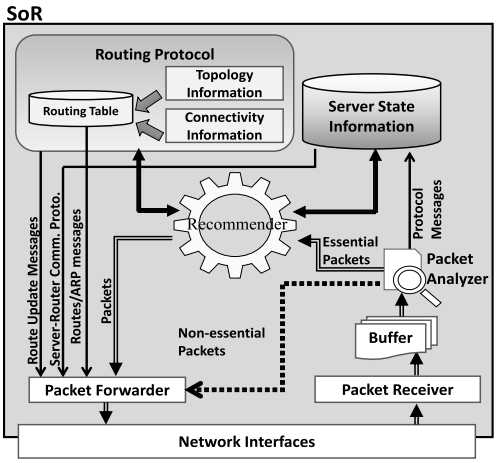


Fig. 2 SoR prototype designed using ns-3.

to analyze packets and simultaneously maintain high throughput packet flow; Intel claimed that they could analyze packets by providing a throughput of up to 160 Gbps [33].

Studies such as Refs. [28], [34], [35] projected such trends and used cached content on routers for future CDN implementations. Similarly, SoR uses the DPI module and the attached in-network storage module for collecting and storing both server and network-state information, which uses to provide end-user services. However, the SoR introduced in Refs. [16], [17] continues to be a topic of research and development; its software and hardware components are currently being developed to create a complete router [21], [22], [36]. Consequently, as depicted in Fig. 2, we designed a prototype of the SoR as an ns-3 module. The implemented ns-3 SoR module consists of a routing module, server-router communication module, packet analyzer module, and a database to store the network and server-state information.

2.2.1 Packet Capturing and Analyzer Module

The SoR packet analyzer module is designed to perform DPI over the packets that pass through SoR interfaces. As illustrated in Fig. 2, the SoR packet analyzer analyzes the five tuples of a packet: source IP, source port, destination IP, destination port, and transport-layer protocol to classify packet flows. Further, it analyzes the packet for specific header information, i.e., DNS header, if necessary. The analyzer module forwards the packets to either the routing or the recommender module by classifying the packets into categories such as DNS request messages and UDP data packets.

2.2.2 Recommender Module

The recommender module is designed to handle packets that are forwarded from the analyzer module. In this implementation, the recommender module primarily receives DNS query packets, or UDP end-user data request packets from the analyzer module. This module is responsible for providing two services: 1) intercept the hierarchical DNS resolution process [6] for fast connection initiation and quick recognition of server changes, and 2) analyze the application layer information, i.e., URI, of data packets and redirect the packets to the best server based on the server-state information. A detailed explanation can be found in Sections 3 and 4.

2.2.3 Routing Module

The routing module is responsible for finding the network routes (e.g., shortest path, minimal delay path, or low-weight path) and forwarding packets to the next hop towards their destinations. SoRs can be configured with any routing protocol. However, as stated in Ref. [37], the regular "shortest path routing protocols" are a lack of adapting to dynamic networks such as CDNs. Therefore, in this study, as presented in Fig. 1, we used the SLRouting protocol [19] as the interior gateway routing protocol (IGP) of the proposed network.

2.2.4 Server-router Communication Module

The server-router communication protocol has been introduced to collect and store server-state information (e.g., an average packet waiting time, server utilization, etc.) at the SoRs. This module is designed as an extension of the SLRouting protocol. An assumption was made that CDNs agree to share up-to-date server-state information; then, we infer that the network administrators configure CDN management servers to periodically advertise the server-state information to the corresponding gateway SoR (see Fig. 1).

2.2.5 SLRouting Protocol

The SLRouting protocol was initially introduced as a delay-based hybrid interior gateway routing protocol (IGP) [19]. SLRouting calculates a composite route metric by using the states of the network components including servers, routers, and network links. Then, the composite metric is used to compute the minimal delay path, i.e., effective path (ep), between two endpoints, i.e., routers.

In this study, as previously stated, the SLRouting protocol is used as the IGP of the proposed network. Thus, SoRs access network-state information by using the routing protocol, i.e., the routing table. Moreover, SoRs use the server-router communication protocol, which is an extension of SLRouting, to obtain server-state information and store the information in server-state information databases; a detailed explanation can be found in Section 2.2.6.

In order to summarize the theory explained in Ref. [19], let us consider a network represented by Graph $G = (V, E)$, where V denotes the set of nodes and E denotes the set of directed physical links. **Figure 3** describes the symbols used in this explanation. Let M_{ij} denote the cumulative traveling delay of flow (i,j) from node i to j, where $i,k,j \in V$. Flows are carried on end-to-end paths consisting of intermediate routers and links, and M_{ij} is the summation of the propagation delay through the intermediate routers and links.

As illustrated in Fig. 3, we assumed that servers and routers can observe by using the M/M/1 queueing model, where the Poisson process determines the arrival rate (λ), and the job service (μ) has an exponential distribution [38], [39]. By applying the Little theorem, $\mathcal{T} = 1/(\mu - \lambda)$ [38]; the average waiting times ($\mathcal{T}$) at the server ($\mathring{\mathrm{d}}_j(t)$) and the router ($\hat{\mathrm{d}}_k(t)$) are calculated by using Eqs. (1) and (2), respectively. j_n and k_n denote the server and router indexes respectively, and $n = 1, 2, 3, \ldots$ and $n \in ep$.

$$\dot{d}_{j_n}(t) = 1/(\mu_{j_n}(t) - \lambda_{j_n}(t)) \tag{1}$$

$$\hat{d}_{k_n}(t) = 1/(\mu_{k_n}(t) - \lambda_{k_n}(t)) \tag{2}$$

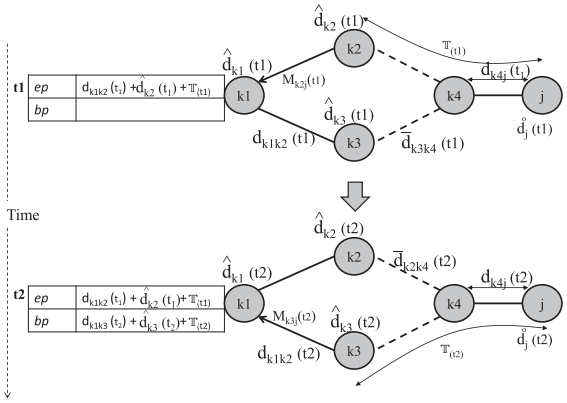


Fig. 3 Basic concept of the SLRouting protocol (solid lines denote direct links, dotted lines denote the distance). k represents nodes, and j represents the destination; j can be either a route or a server.
Router Metric ($\hat{d}_k$): average waiting time at the router,
Link Metric (d_{kj}): average travel time through the link,
Server Metric ($\mathring{d}_j$): average waiting time at the server

The packet traveling delay of a link depends on two major factors: 1) packet transmission delay (T_{trans}) and 2) packet propagation delay (T_{prop}). T_{trans} of a link kj depends on the available bandwidth, $L_b(t) = L_c - L_l(t)$ of a link, where L_l denotes the used bandwidth and L_c represents the total bandwidth [10]. In this case, for a packet of size Z that propagates through the link, T_{trans} can be calculated by using Eq. (3).

$$T_{trans}(t) = Z/L_b(t) \tag{3}$$

Further, T_{prop} of a link depends on the length and medium of the link [39]. Consequently, the total traveling delay ($d_{kj}(t)$) of a given link, kj, can be written by using Eq. (4).

$$d_{kj}(t) = T_{prop_{kj}}(t) + T_{trans_{kj}}(t) \tag{4}$$

Let us consider the scenario presented in Fig. 3 in order to understand the application of the distributed Bellman-Ford algorithm to calculate the minimal delay path (ep). Let us suppose that the node k_1 receives route updates at $t1$ and $t2$. At $t1$, when $k1$ receives the new destination, j, from k_2, node k_1 calculates the cumulative propagation delay by using Eq. (5), where $\mathbb{T}_{t1} = \bar{D}^{k_4}_{k_2 j}(t1)$; minimal delay between nodes k_2 and j via node k_4. Thus, k_1 considers the calculated cumulative time as the best path to the new destination j.

$$M_{k_1 j}(t1) = d_{k_1 k_2}(t1) + \hat{d}_{k_2}(t1) + \mathbb{T}_{t1} \tag{5}$$

Similarly, at the time $t2$, when k_1 receives a route update for destination j from k_3, the new propagation time through k_3 will be calculated by using Eq. (6), where $\mathbb{T}_{t2} = \bar{D}^{k_4}_{k_3 j}(t2)$; minimal delay between nodes k_3 and j via node k_4.

$$M_{k_1 j}(t2) = d_{k_1 k_3}(t2) + \hat{d}_{k_3}(t2) + \mathbb{T}_{t2} \tag{6}$$

Assuming that $M_{k_1 j}(t1) < M_{k_1 j}(t2)$; k_1 uses the route via k_2 as the ep and the route via k_3 as the backup path for destination j. Subsequently, the cumulative propagation delay of ep, i.e., from k_1 to j, can be written as Eq. (7), where $\bar{d}_{k_2 k_4}$ is the minimal delay path between the distance nodes k_2 and k_4.

$$\mathbb{T}(t) = K1\Big[\dot{d}_j(t)\Big] + K2\Big[d_{k_1 k_2}(t) + \bar{d}_{k_2 k_4}(t) + d_{k_4 j}(t)\Big] + K3\Big[\hat{d}_{k_2}(t) + \hat{d}_{k_4}(t)\Big] \tag{7}$$

In general, the composite metric to reach a destination j from a node i can be simplified as Eq. (8), where n and m are the indexes of the network components in ep, $n \neq m$, $i \neq j$, and i can be considered as a router.

$$M^{k_n}_{ep_{ij}}(t) = K1\Big[\dot{d}_j(t)\Big] + K2\Big[\sum_{n,m \in ep} d_{k_n k_m}(t)\Big] + K3\Big[\sum_{n \in ep} \hat{d}_{k_n}(t)\Big] \tag{8}$$

It should be noted that $K1$, $K2$, and $K3$ are controllable coefficient values, which are introduced to provide control in calculating the composite metric [19].

2.2.6 Collecting Server-state Information

In CDNs, as explained in Ref. [6], CDN management servers, e.g., Auth. DNS servers [40], determine the conditions of the content servers by (periodically) accessing the server-states information of the content servers. The management servers then create redirection policies and recommend content servers to their subscribers using DNS resolution [6], [41]. We also assumed the same approach in the proposed method, and hence, as illustrated in Fig. 1, we updated the management servers to advertise the server-state information to the corresponding gateway router as well.

Servers used the introduced server-router communication module to advertise server-state information to their corresponding gateway router. The server-router communication module is designed and implemented as an extension of the SLRouting module. Servers, i.e., management or content, use advertisement messages, which are designed similar the Route Update Messages presented in Ref. [19], to advertise the server-state information. The advertisement message can be attached to the SLRouting route advertisement header using the process explained in Ref. [19]. The server advertisement message consist of IP address, supporting URL(s) informational, average service rate (μ), average packet arrival rate (λ) of a content server, and a sequence number.

In this study, we configured the management servers to advertise the server-state information at each 120 s interval. SoRs store server-state information on the server-state information database; the stored information includes the IP address, URL, URI(s), server-state, and time of the last update. The state of the server indicates the current waiting time ($T_{S_n}(t)$) of the server (i.e., the average duration that a packet must spend on the server until the packet is served). The SoRs use Eq. (1) to calculate $T_{S_n}(t)$.

With the aid of the SLRouting protocol, the proposed method makes sure that all SoRs maintain up-to-date server-state information by advertising the server-state information among the SoRs. Note that SoRs use both periodic and triggered update advertisements to ensure quick distribution of the server-state information among the other SoRs in the network. In a situation where an SoR does not receive update information about a particular server record for 150 s, the record will be removed from the server-state information database and notify the neighbors.

Fundamentally, SoRs use the collected information to recommend the IP address of the least-busy servers by intercepting the

recursive hierarchical DNS resolution process. Moreover, SoRs use the collected information for content-aware packet redirection. In brief, SoRs analyze the application layer information, i.e., URL and the content ID, of the data-request packets to select the least-busy server from the server-state information database. The SoRs then forward the data packet to the selected server.

3. Recommendation and Message Flow

This section discusses the message flow for the proposed network architecture. In general, when a packet reaches gateway router, the gateway router uses the packet analyzer module to classify the packet as a DNS request packet or a data request packet. And the gateway router uses the recommender module to execute necessary packet-handling decision; DNS recommendation or content-aware redirection. **Figure 4** presents the main message flow of the proposed network architecture.

3.1 DNS Recommendation

DNS-based end-user redirection is the most well-known redirection method used by CDNs [4], [6], [40], [42]. A detailed explanation about the DNS-based redirection is given in Refs. [5], [40], and the same approach is assumed in this implementation. Originally, in DNS systems, TTL values were defined to guarantee the cache time of the DNS. In general, TTL values are in the order of minutes or hours [43]. However, with the introduction of DNS-based redirection in CDNs, the TTL values are set to smaller increments to aid CDN load-balancing and traffic flow optimization [6], [41]. On the other hand, as pointed out in Ref. [40], CDNs do not reveal their server selection criteria for their proprietary reasons.

Alternatively, Ref. [40] states that "connection monitoring" method is one of the well-known methods used by CDNs to approximate the locations content servers to their clients. However, connection monitoring method approximates TTL values of the DNS messages based on the RTT between clients and servers. The limitation of this approach is that it does not consider the state of a content server (e.g., average packet waiting time at the server) as a parameter to calculate the TTL values. Such a phenomenon forces the large number of DNS query requests to be transmitted to the public network instead of served by the local DNS.

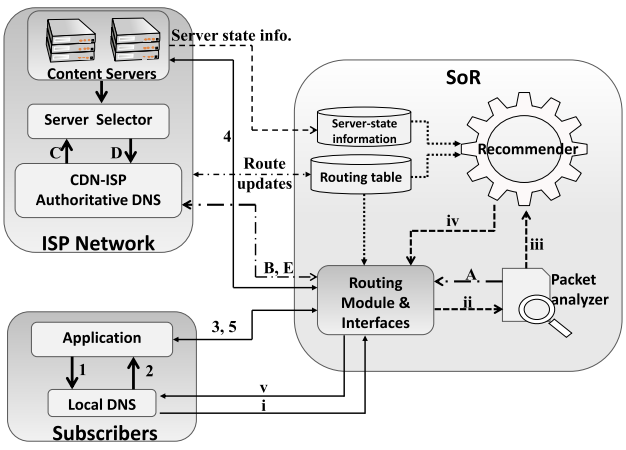


Fig. 4 Message flow among local network, gateway router, and ISP network.

When a given DNS strictly adheres to the calculated TTL, the local name servers must frequently traverse a hierarchy of name servers to resolve and to inform the subscribers about the change in the servers [4]. Generally, in the DNS-based redirection, the local name server recursively queries a hierarchy of name servers until it reaches the authoritative name server for the CDN domain [4], [5]. The caveat is the time that is spent in finding the authoritative name server. This factor influences the connection initiation and rapid adaptation to the changes occurring in servers.

In order to address such limitations, the analyzer module of the edge router, i.e., the gateway router between the end-users and the ISP, is programmed to intercept DNS query packets that are transmitted to the ISP network from the local network. The intercepted DNS query packets will be forwarded to the recommender module (step "ii" of Fig. 4). If the analyzer module receives a DNS response packet, the packet will be forwarded to the routing module without being intercepted. Moreover, if the DNS uses security protocols (e.g., DNSCrypt or DNSSEC), the SoR does not intercept such encrypted DNS packets. Consequently, if the SoR does not intercept a DNS query, such requests assume the general CDN name server resolution process. This approach allows the proposed network to co-exist with the generic CDN architectures.

After receiving a DNS query packet, the recommender module finds a match in the server-state information database using the URL mentioned in the DNS query. If a match is found, the recommender module retrieves the server utilization ($\rho_S(t)$) and the average waiting time at the server ($T_{S_n}(t)$) from the server-state record. Subsequently, the recommender module generates a DNS response packet and sends it to the local DNS server. The recommender module uses Eq. (9) to calculate the corresponding TTL value for the DNS response packet.

$$TTL = K \times t_S \times \left(\frac{1}{\rho_S}\right) \tag{9}$$

Equation (9) is proposed to use the average packet waiting time at a server as a parameter of calculating the TTL value of the DNS records. Consequently, at a given time (t), we used $T_{S_n}(t)$ as proportional to the TTL value, and thereby, calculate TTL values according to the average packet waiting time at a server rather than approximating the TTL values for the DNS messages [40]. The constant K changes according to ($T_{S_n}(t)$) and is introduced to convert the TTL value to the order of seconds. Based on several trial-and-error methods, we decided to use $K = 1{,}000$ in this study. The inverse of the server utilization ($\rho_S(t)$) is used in Eq. (9) to adjust the TTL value according to the server usage, i.e., small values of TTL for busy servers and large values of TTL for idle servers. Thus, using a simple derivation, we found that Eq. (9) demonstrates a better ability to adapt to conditions (e.g., server changes) and to recommend content servers to the end-users adequately.

3.2 Intercepting Data Request Packets

When an SoR receives a data request packet, which uses UDP as the transport layer protocol (e.g., video, game, and TV streaming), on the fly, the SoR recommends the least-busy server ac-

cording to the content of the packet as follow.

The analyzer module first verifies the protocol and forwards the packet to the recommender module. When the recommender module receives the packet, the recommender module obtains the requested URI from the packet. The recommender module then finds the server-state records that match the URI, selects the server record with the least packet waiting time ($T_{S_n}(t)$), and redirects the packet as follows:

- If the URI of the packet matches only one server-state record in the database, since the destination IP address is same as the IP address of the selected server-state record, the packet will be forwarded to the routing module without further modifications.
- If the URI of the packet matches several server-state records in the database, the best server for the particular content request must be selected. Therefore, the recommender module checks for the destination IP address of the packet. If the destination IP address matches the server-state record that has the minimum packet waiting time, the packet will be forwarded to the routing module. Otherwise, the recommender module changes the destination address of the data packet to the best server available in the server-state database based on minimum packet wait time, and forwards the packet to the routing module.

However, in the current implementation, the analyzer does not intercept the TCP packets and does not redirect the TCP connections on the fly due to the protocol complexity. Hence, the analyzer module directly forwards such packets to the routing module. In fact, the proposed network architecture assumes the solutions proposed in Refs. [44], [45], [46] for on the fly TCP connection management.

3.3 Message Flow in the Proposed Network

Now we describe the operation of our working prototype and its interaction with a CDN. We illustrate the basic system diagram of the proposed network architecture, including the flow of messages, in Fig. 4

3.3.1 When Local DNS Has the IP Address

(1) If an end-user wishes to download content from the CDN, the end-user contacts the local DNS for name server resolution. (2) Then, the local DNS replies the IP address to the end-user. (3) After receiving the IP address from the local DNS, the end-user requests the data from the relevant content server. At this point, the SoR intercepts the data request packets. (ii) The packet analyzer verifies the protocol type. (A) If the protocol type is not supported, the analyzer module returns the packet to the routing module and resumes the conventional packet routing. (iii) Otherwise, the analyzer module forwards the packet to the recommender module. Then, the recommender module verifies the packet is destined to the best server by referring the server-state information database. If the packet is not directed to the best server, the destination address of the packet will be changed to the best server. (iv) The recommender module forwards the packet to the routing module. (4 and 5) Finally, the routing module forwards the packet to the assigned server.

3.3.2 When Local DNS Does Not Have the IP Address

(i) The local DNS sends the DNS query packet to the CDN authoritative DNS of the ISP. At this point the gateway SoR intercepts the DNS query packet. (ii) The packet analyzer verifies the message type. (A) If the message type is not supported, the analyzer module returns the packet to the routing module and resumes the conventional DNS resolution process. (iii) Otherwise, the analyzer module forwards the packet to the recommender module. At this point, the recommender module accesses the server-state information database and the routing table to obtain the best IP address match for the DNS query. Then, the recommender module calculates a TTL value according to the state of the selected server. (iv and v) Next, the recommender module generates a DNS response packet and sends it to the Local DNS. In the end, (2) the Local DNS sends the DNS response packet to the end-user. (3, 4, and 5) Finally, the end-user downloads the content.

3.3.3 When Neither Local DNS Nor SoR Has the IP Address

(i) The local DNS sends the DNS query packet to the CDN authoritative DNS of the ISP. At this point the gateway SoR intercepts the DNS query packet. (ii) The packet analyzer verifies the message type. (A) If the message type is not supported, the analyzer module returns the packet to the routing module and resumes the conventional DNS resolution process. (iii) Otherwise, the analyzer module forwards the packet to the recommender module. When the recommender module determines that the SoR does not have an IP address, (A and B) DNS query is forwarded to the CDN authoritative name servers of the ISP. (E and v) The authoritative name server resolves the IP address and reply to the local DNS. (2) The local DNS sends the IP address to the end-user. (3, 4, and 5) Finally, the end-user downloads the content.

4. Simulation Set-up and Experimental Results

Exposing internal router-level topological information as well as the server-level infrastructural information remain unsafe due to the vulnerabilities the ISP is facing. As a consequence, ISPs make topological data are publically unavailable. Consequently, we used an education backbone network presented in **Fig. 5** [24] to implement the first prototype implementation of the proposed collaborative infrastructure. Though the topology is not an ISP network, we arranged realistic simulations by configuring the same network conditions (e.g., link speeds) as provided in Ref. [24] to study the behavior of the SoR as a gateway

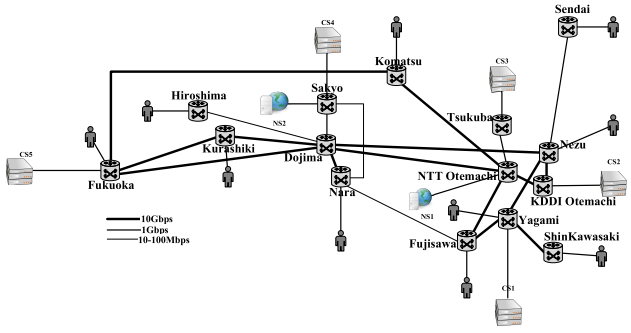


Fig. 5 An instance of the topology.

Table 1 Simulation parameters.

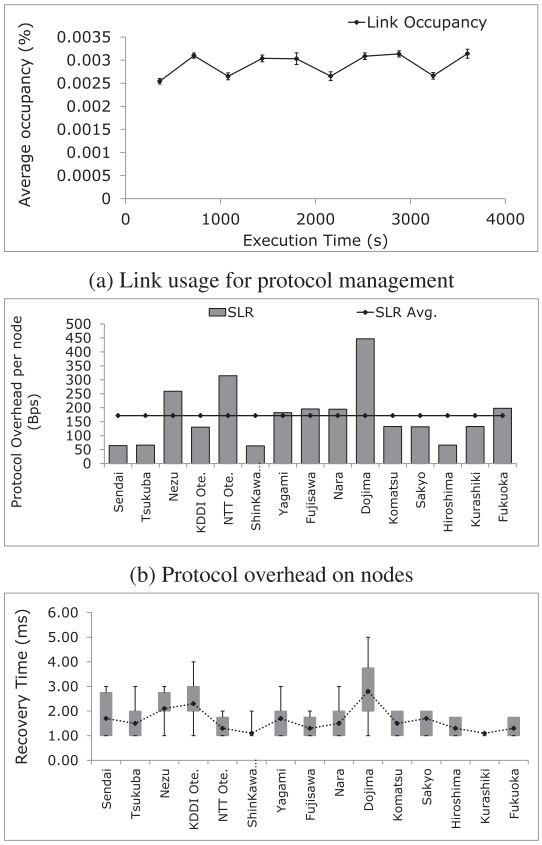
Parameter	Used value
Route update interval	20 s
Server-router communication interval	120 s
Server record expiration time (SoR)	150 s
Simulation time	4,000 s
Number of SoRs	13
Number of regular routers	2
Number of traffic generators	10
Number of servers	5
Number of content	1,000
Average content size	256 B
Number of Auth. DNS	2
TTL avg. (in DNS-based method)	20 s–200 s
Link Bandwidths	configured according to Ref. [24]

router. Moreover, we used the parameters presented in **Table 1** to implement the simulations. The network was configured using SLRouting protocol, and the periodic advertisement interval of SLRouting was used as 20 s. For the simplicity of presentation, we assumed that one content provider operates within an ISP network.

We assumed 1,000 items of content and the contents were fully replicated among the content servers. We strategically placed five content servers in the simulation network based on the popularity of the cities. The content servers were programmed to advertise the server-state information at each 120 s time interval. Furthermore, the SoRs used triggered advertisements for fast propagation of the server-state information among the edge routers. Note that the server records were invalidated if particular server record is not updated within 150 s. We used two authoritative DNS servers. 13 SoRs were used as the edge SoRs of both end-users and content servers.

Supposedly, to simulate the co-existence between SoRs and regular routers in the proposed collaborative infrastructure, as depicted in Fig. 5, several regular routers were also placed in the network. Regular routers do not have the capabilities of the SoRs, and therefore, the regular routers were not placed as the edge routers of any client or a content server. Hence, the regular routers were used as core routers in the network, e.g., Dojima router and NTT Otemachi Router. In order to maintain backward compatibility with the existing Internet architecture, we assumed that content is requested by using a URI; the URI consists of the server address, port, and content ID, which symbolizes the content.

The end-users were created as random traffic generators, and ten end-users were randomly placed on the implemented network. Although the number of traffic generators was ten, that was sufficient to understand the behavior of the SoR as an intermediary for ISP CDN collaboration in the first prototype implementation. The end-users were programmed to resolve the DNS when the TTL values expired. They generated data-request packets based on an assumption of a video streaming service, which is a well-known application in CDNs [47], [48]. Subsequently, the end-users generate UDP-based data request messages as explained in Ref. [49], and the data request message carries the intended URI, i.e., rtsp://<SurrogateServer_Address>:<Port#>/<contentID>, that the end-user wanted to fetch. Note that the sizes of both request and response packets were averagely 256 B.



(a) Link usage for protocol management

(b) Protocol overhead on nodes

(c) Route recovery time

Fig. 6 SLRouting performance evaluation.

4.1 Evaluation Results

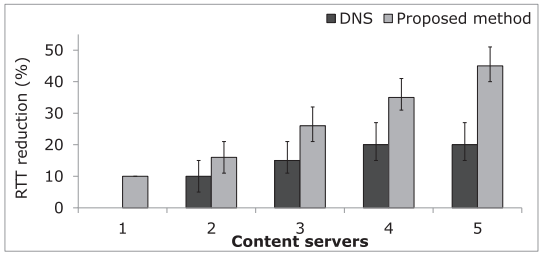
4.1.1 Effectiveness of the SLRouting Protocol

Initially, we measured the effectiveness of the SLRouting protocol and its extension, server-router communication protocol, for the proposed network architecture. For that, we configured both protocols to the topology given in Fig. 5 and simulated the topology for a one-hour period.

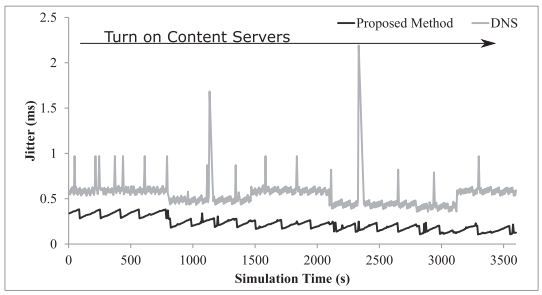
First, we measured the link usage for route management. The obtained results are shown in **Fig. 6** (a). According to the figure, the protocol messages, including server-router communication messages utilized 0.003% of the link capacity on an average. Given the fact that current network links are on a gigabit scale, above link usage is almost negligible.

Then, we analyzed the protocol overhead for the routers by measuring the size of the protocol messages processed by each router. The obtained results are plotted in Fig. 6 (b). According to the figure, the average protocol message size is 170 bytes. It is observed that the protocol displaces an above-average burden on some routers, e.g., Nezu, Dojima, and NTT Otemachi. The reason for this behavior is that, as explained in Ref. [19], route summarization has not yet been implemented. Therefore, the sizes of the route tables and the sizes of the protocol messages increase.

The link recovery time is vital for the traffic engineering (TE) of an ISP [10], and eventually, such techniques improve the per-



(a) RTT reduction (%)



(b) Jitter variance

Fig. 7 End-user performance evaluation.

formance for the end-user. Therefore, we measured the link recovery time of the SLRouting protocol by randomly disconnecting some links and by shutting down some routers. The results obtained are plotted in Fig. 6 (c). According to the figure, the average recovery time of SLRouting is approximately 2.5 ms. The reason for this behavior is that, as explained in Ref. [19], SLRouting maintains the backup route for each destination address. Therefore, when a router receives a "BROKEN" route record, it quickly replaces the route with the necessary backup route.

4.1.2 End-user Experience

We evaluated the end-user experience by measuring the reduction in round trip time (RTT) and the variation in jitter. The obtained results are plotted in **Fig. 7**. According to Fig. 7 (a), as the number of content servers increases, in the collaborative architecture, the RTT decreases by up to 45%. The reason for this behavior is that, when new content servers were introduced in the network, the SoRs were able to adapt rapidly to the topology changes with the aid of the server-router communication protocol. Thus, as explained in Section 3, the SoRs were able to recommend new content servers to the end-users without depending on the hierarchy of name servers.

4.1.3 Jitter Variance

Next, we studied the jitter variance by using a random end-user. The obtained results are plotted in Fig. 7 (b). According to the figure, the proposed method displays averagely 40% of jitter reduction when compared with DNS-based redirection for the entire simulation period. The main reason for this behavior is that the gateway SoRs were able to redirect the end-users to the least-busy servers by using the minimal delay paths. Therefore, the packets reached their intended destinations by avoiding the congested links and routers. In our experiments, we simulated a streaming service, and hence, we can conclude that the proposed method can be used effectively for streaming applications, which is a primary application of network data sharing.

Table 2 DNS request distribution.

	DNS-based method	Proposed method
Resolved within local network	58% (approx.)	85% (approx.)
Passes to the ISP	40% (approx.)	15% (approx.)
TTL	20–200 s (predefined)	10–40 s
Resolution time (avg.)	Order of hundred milliseconds	Order of ten milliseconds

Table 3 Response to server changes.

Executed at	Scenario	Time taken to redirect	RTT
0th second	Start Simulation and Main server	0.2 s	0.07 s
100th second	Start CServer: 1	0.05 s	0.012 s
200th second	Start CServer: 2	0.04 s	0.006 s
300th second	Stop CServer: 1	0.02 s	0.024 s
400th second	Start CServer: 1	0.01 s	0.006 s

4.1.4 DNS Resolution

We analyzed the proposed network architecture for DNS resolution. The analysis data are presented in **Table 2**. According to the table, in the proposed network, 85% of the name server queries are resolved within the local network, either from the local DNS or from the gateway SoR. The reason is that, when Local DNS misses the DNS query, the SoR between the end-user and the provider network was able to recommend the IP addresses to the local DNS without forwarding the DNS queries to the public network. Consequently, the name server resolution time was reduced to the order of milliseconds. SoRs missed 15% of the DNS requests due to protocol management, i.e., delete server-state information records upon expiration. However, those missed DNS queries were able to resolve by contacting the authoritative DNS, which is similar to the regular DNS resolution process. This suggests that the collaborative method successfully maintains consistent data delivery by solving the hierarchical DNS resolving problem.

4.1.5 Response to Server Changes

We continued the experiments to demonstrate the effectiveness of ISP-CDN collaboration for server change situations. We turned the content servers on and off during the simulation and measured the redirection time and RTT for a randomly selected end-user. Simulation scenarios and the obtained results are given in **Table 3**. According to the table, in the beginning, the proposed method waited about 200 ms to identify the main server. The reason is that the DNS cache was miss at both local DNS and user edge SoR, and thus, end-user had to use hierarchical DNS resolution process to find the appropriate content server.

However, results given in Table 3 suggest that the proposed method required on average 64 ms to recognize the changes and redirect the end-users to content servers. The reason is that the SoRs were able to collect up-to-date server-state information and use this information to suggest IP addresses for the end-users without being strongly dependent on the hierarchical name server resolution process. Consequently, as Table 2 shows, 85% of the name server queries were resolved within the local network, and thus, the name server resolution time reduced to the order of milliseconds. Implicitly, SoRs are successfully working as an in-

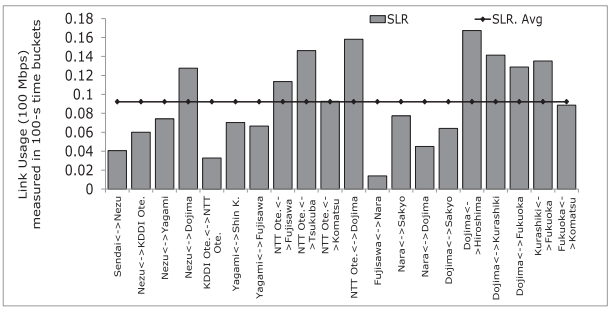


Fig. 8 Network link utilization.

termediary to make a collaboration between ISP and CDNs, and hence, SoRs were able to use the ISP-CDN collaboration to suggest server changes to the end-users within small time scales.

4.1.6 Network Resource Utilization

Now we study the network resource utilization by measuring the network link usage for data delivery. The corresponding results are plotted in **Fig. 8**. According to the figure, the average use of the links for data transfer was approximately 10 Mbps in the proposed network. Thus, the proposed approach was able to exploit almost every link in the topology. The reason is that the SLRouting computes the route matrix according to the state of the network components. It does not depend on the capacity of the links for route metric calculation. Therefore, if a router identifies that a network path (i.e., a route) is busy and the propagation time is high, the router quickly switches to the backup path, which is the next best route available to reach the destination [19].

5. Discussion

This section discusses the remaining technical and business facts of the proposed architecture.

Kamiyama et al. stated that ISP-CDN cooperation is the key to providing faster data delivery [50]. Furthermore, Refs. [7], [47] show that the ISPs should perform adequate routing and end-user redirection in small time scales, and the traffic matrix should be considered as a parameter for the network path selection [51]. Subsequently, the obtained results showed that, by placing SoRs at the edges of ISP networks, the SoRs could be used to leverage DNS-based CDN redirection to achieve end-user redirection in small time scales. Moreover, SoRs uses SLRouting protocol for effective network resource utilization by contemplating ISP and CDN state information as the parameters for network path selection. Therefore, the development of SoR [21], [22], [36] ensures that the proposed collaborative architecture will be a successful business model to which both ISPs and CDNs can use it to guarantee their subscribers download data from the best server via the minimal delay paths.

It is indisputable that packets, e.g., end-user data requests and DNS queries, should be processed at edges of the networks, and hence, edge routers are becoming vital devices to process data and provide adequate services. Nonetheless, edge computing [26] and network virtualization [35], [52] are also gaining momentum among researchers and vendors. To this end, routers should have sufficient processing power and memory to ensure proper performance of the proposed network. According to Refs. [29], [30], [31], [33], routers with high-performance processors and in-network storages are already available in the market. Meanwhile, as an ongoing area of research, SoRs are getting improved and will be available in commerce in the near future. Consequently, in the near future, ISPs can lease network slices to CDNs and CDNs can use SoRs as an intermediary to strengthen the ISP-CDN collaboration.

In the current prototype implementation, SoRs do not intercept TCP connections to provide on the fly TCP redirection because TCP is a stateful protocol and the cost of protocol management is high. Research groups, including commercial vendors and the research community, are investigating possible solutions to intercept and redirect a TCP connection from the routers. For example, Hitachi presented a "WAN accelerator" in Ref. [44], and Cisco provided "multipath TCP" in Ref. [45]. Further, Surton et al. proposed "Man-in-the-Middle TCP recovery" in Ref. [46], and the authors suggested TCP redirection from the routers. Assuming that future CDNs and ISPs use such technologies, SoRs can be used to enhance the on the fly TCP redirection successfully.

6. Related Work

The necessity for cooperation between network and content providers has been a popular topic of discussion within the research community and among vendors. The authors of Ref. [53] proposed a communication portal between ISPs and P2P applications; this portal can be used by P2P applications to determine ISP-based network information to reduce the cost for network providers without sacrificing performance. Another study [54] proposed an Oracle service run by the ISP; the users can use the ranked neighbor to improve the performance metrics. In Ref. [7], authors proposed ISP-CDN collaboration. Authors used the traffic matrix of the CDN for TE, and hence, proposed to redirect end-users to the nearest content servers in small time scales.

The study that is most similar to ours is [55]. In Ref. [55], Poese et al. selected content servers by utilizing network views collected from an ISP; thus, the end-users were redirected to the most efficient server instead of the nearest server. The authors revealed that the ISP-CDN collaboration could be implemented in real ISP networks by simulating their system in the Tier-1 ISP. However, the authors were not interested in using the content server information for network path selection. Further, their proposal is considerably dependent on hierarchical DNS and does not discuss the problems of latency and the lag in adapting to the server changes due to hierarchical DNS resolution.

In Ref. [34], the ISP-centric content delivery (iCODE) architecture was studied; it was designed to cache the content in intermediate routers in the ISP network. iCODE assumed routers would have in-network storage modules, which are similar to SoRs. iCODE used the swarming technique to download the data from routers to end-users. However, the authors were only interested in caching content on the routers. They do not consider the topic of content-based services by assuming more intelligent routers in their system.

In Ref. [56], the Data-oriented network architecture (DONA) was proposed. DONA resolves the content by using flat and self-certifying names instead of URLs. Hence, in the DONA architecture, DNS name resolution is replaced by a name-based any-

cast. In this architecture, name resolution is performed by using a network entity called resolution handler (RH). In fact, the ISP must replace the DNS servers by RHs to use DONA in a network. Although the DONA architecture guarantees the perfect global availability of content, the system functions only when RHs are deployed within on all ISPs.

7. Conclusion

This paper discusses to strengthen the ISP-CDN collaboration using an SoR as the gateway router in an ISP network. SoRs, working as an intermediary, collect and store network topology and server-state information with the aid of the SLRouting protocol. Collaboratively, the SoRs used the ISP's and CDN's information to leverage the DNS-based CDN redirection; recommend content servers to the end-users in small time scales. Consequently, when compared with DNS-based redirection, data download latency decreased by 40–50%, and the end-users were able to adapt to the server changes within small time scales. Further, SoRs used the ISP-CDN collaboration information as the parameters of SLRouting protocol to provide minimal delay network paths by utilizing almost every link in the topology. Consequently, the edge SoRs were able to reduce the end-user jitter by 40% when compared with DNS-based redirection.

In the future, along with the hardware and software development of SoR, we will introduce the proposed system into an ISP and conduct a comprehensive evaluation of an ISP network with real ISP traffic. In conclusion, the proposed method creates the possibility of a new business model that strengthens the ISP-CDN collaboration to leverage the CDN RR for faster data delivery.

Acknowledgments This work was partially supported by funds of SECOM Science and Technology Foundation, MEXT Scholarship for Research Students, Keio University Doctorate Student Grant-in-Aid Program, and the Keio University KLL Ph.D. Research Grant Program.

References

[1] InternetLiveStats: Internet Live Stats, Real Time Statistics Project (Worldometers and 7 Billion World) (online), available from ⟨http://goo.gl/EzIAer⟩ (accessed 2016-03-20).
[2] Valancius, V., Ravi, B., Feamster, N. and Snoeren, A.C.: Quantifying the Benefits of Joint Content and Network Routing, *SIGMETRICS Perform. Eval. Rev.*, Vol.41, No.1, pp.243–254 (2013).
[3] Su, A.-J., Choffnes, D.R., Kuzmanovic, A. and Bustamante, F.E.: Drafting Behind Akamai: Inferring Network Conditions Based on CDN Redirections, *IEEE/ACM Trans. Netw.*, Vol.17, No.6, pp.1752–1765 (2009).
[4] Nygren, E., Sitaraman, R.K. and Sun, J.: The Akamai Network: A Platform for High-performance Internet Applications, *SIGOPS Oper. Syst. Rev.*, Vol.44, No.3, pp.2–19 (2010).
[5] Akamai: Fast internet content delivery with FreeFlow, Akamai (online), available from ⟨http://goo.gl/A12sdq⟩ (accessed 2016-03-20).
[6] Niven-Jenkins, B., Velocix, E. and Brandenburg, R.: Request Routing Redirection Interface for CDN Interconnection (online), available from ⟨https://goo.gl/EKvv3T⟩ (accessed 2016-03-20).
[7] Poese, I., Frank, B., Smaragdakis, G., Uhlig, S., Feldmann, A. and Maggs, B.: Enabling Content-aware Traffic Engineering, *SIGCOMM Comput. Commun. Rev.*, Vol.42, No.5, pp.21–28 (2012).
[8] Valancius, V., Ravi, B., Feamster, N. and Snoeren, A.C.: Quantifying the Benefits of Joint Content and Network Routing, *SIGMETRICS Perform. Eval. Rev.*, Vol.41, No.1, pp.243–254 (2013).
[9] Jiang, W., Zhang-Shen, R., Rexford, J. and Chiang, M.: Cooperative Content Distribution and Traffic Engineering in an ISP Network, *SIGMETRICS Perform. Eval. Rev.*, Vol.37, No.1, pp.239–250 (2009).
[10] Wang, N., Ho, K., Pavlou, G. and Howarth, M.: An overview of routing optimization for internet traffic engineering, *Communications Surveys Tutorials, IEEE*, Vol.10, No.1, pp.36–56 (2008).
[11] Choffnes, D.R. and Bustamante, F.E.: Taming the Torrent: A Practical Approach to Reducing Cross-isp Traffic in Peer-to-peer Systems, *SIGCOMM Comput. Commun. Rev.*, Vol.38, No.4, pp.363–374 (2008).
[12] Wijekoon, J., Harahap, E., Takagiwa, K., Tennekoon, R. and Nishi, H.: Effectiveness of a Service-oriented Router in Future Content Delivery Networks, *2015 7th International Conference on Ubiquitous and Future Networks* (*ICUFN*), pp.444–449 (2015).
[13] Kelly, J., Araujo, W. and Banerjee, K.: Rapid Service Creation Using the JUNOS SDK, *SIGCOMM Comput. Commun. Rev.*, Vol.40, No.1, pp.56–60 (2010).
[14] Ohara, Y., Yamagishi, Y., Sakai, S., DattaBanik, A. and Miyakawa, S.: Revealing the Necessary Conditions to Achieve an 80Gbps High-Speed PC Router, *Proc. 11th ACM SIGCOMM Asian Internet Engineering Conference*, *AINTEC'15* (2015).
[15] Kim, J., Jang, K., Lee, K., Ma, S., Shim, J. and Moon, S.: NBA (Network Balancing Act): A High-performance Packet Processing Framework for Heterogeneous Processors, *Proc. 10th European Conference on Computer Systems*, *EuroSys'15*, pp.22:1–22:14 (2015).
[16] Inoue, K., Akashi, D., Koibuchi, M., Kawashima, H. and Nish, H.: Semantic router using data stream to enrich services, *International Conf. on Future Internet Technologies* (*CFI08*) (2008).
[17] Inoue, K. and Nishi, H.: Semantic router using data stream to enrich services, WestLab (online), available from ⟨http://goo.gl/GmHnnV⟩ (accessed 2016-03-20).
[18] Oyamada, M., Kawashima, H. and Kitagawa, H.: Continuous Query Processing with Concurrency Control: Reading Updatable Resources Consistently, *Proc. 28th Annual ACM Symposium on Applied Computing*, *SAC'13*, pp.788–794 (2013).
[19] Wijekoon, J.L. and Nishi, H.: SLRouting: Server Link Router State Routing Protocol Design and Implementation, *Proc. Asian Internet Engineering Conference*, *ACM-SIGCOMM AINTEC'15*, pp.1–8 (2015).
[20] Masuda, K., Ishida, S. and Nishi, H.: Cross-site recommendation application based on the viewing time and contents of webpages captured by a Network Router, *The 14th International Conference on Internet Computing and Big Data* (2013).
[21] Harvath, A. and Nishi, H.: Parallel Packet Processing on Multi-core and Many-core Processors, *The 21st International Conference on Parallel and Distributed Processing Techniques and Applications*, *PDPTA'15 in WORLDCOMP2015*, pp.129–134 (2015).
[22] Takagiwa, K. and Nishi, H.: Local Trend Detection from Network Traffic Using a Topic Model and Network Router, *The 16th International Conference on Internet Computing and Big Data*, pp.53–59 (2015).
[23] ns 3: ns-3, ns-3 Consortium (online), available from ⟨https://goo.gl/JzLuR3⟩ (accessed 2016-03-20).
[24] WIDE: WIDE Internet, WIDE Internet (online), available from ⟨http://goo.gl/VAzzZC⟩ (accessed 2016-03-20).
[25] Farrel, A.: Chapter 1 - Overview of Essentials, *The Internet and Its Protocols* (Farrel, A., ed.), The Morgan Kaufmann Series in Networking, Morgan Kaufmann, Burlington, pp.1–21 (2004).
[26] Noronha, A., Moriarty, R., O'Connell, K. and Villa, N.: Attaining IoT Value: How To Move from Connecting Things to Capturing Insights Gain an Edge by Taking Analytics to the Edge (online), available from ⟨http://goo.gl/fL1Nol⟩ (accessed 2016-03-20).
[27] Wijekoon, J., Ishida, S., Harahap, E. and Nishi, H.: Service-Oriented Router-Based CDN System: An SoR-Based CDN Infrastructure Implementation on a Real Network Environment, *Computer Software and Applications Conference Workshops* (*COMPSACW*), *2013 IEEE 37th Annual*, pp.742–747 (2013).
[28] Higginbotham, S.: In a distributed world cache is king. Why routers are becoming the new server (online), available from ⟨https://goo.gl/6OLGEB⟩ (accessed 2016-03-20).
[29] Cisco: DevNet, Application eXtension Platform (AXP), Cisco, Cisco (online), available from ⟨https://goo.gl/Ul3afV⟩ (accessed 2016-03-20).
[30] Kelly, J., Araujo, W. and Banerjee, K.: Rapid Service Creation Using the JUNOS SDK, *SIGCOMM Comput. Commun. Rev.*, Vol.40, No.1, pp.56–60 (2010).
[31] Kim, J., Jang, K., Lee, K., Ma, S., Shim, J. and Moon, S.: NBA (Network Balancing Act): A High-performance Packet Processing Framework for Heterogeneous Processors, *Proc. 10th European Conference on Computer Systems*, *EuroSys'15*, pp.22:1–22:14 (2015).
[32] Intel: Intel Data Plane Development Kit (Intel DPDK) Overview Packet Processing on Intel Architecture, Intel (online), available from ⟨http://goo.gl/POGo51⟩ (accessed 2016-03-20).
[33] Intel: Delivering 160Gbps DPI Performance on the Intel Xeon Processor E5-2600 Series using HyperScan, Intel (online), available from ⟨http://goo.gl/RhLeay⟩ (accessed 2016-03-20).

[34] Cho, K., Jung, H., Lee, M., Ko, D., Kwon, T. and Choi, Y.: How can an ISP merge with a CDN?, *Communications Magazine, IEEE*, Vol.49, No.10, pp.156–162 (2011).

[35] Koponen, T., Chawla, M., Chun, B.-G., Ermolinskiy, A., Kim, K.H., Shenker, S. and Stoica, I.: A Data-oriented (and Beyond) Network Architecture, *SIGCOMM Comput. Commun. Rev.*, Vol.37, No.4, pp.181–192 (2007).

[36] Yamaguchi, F. and Nishi, H.: High-throughput and Low-cost Hardware Accelerator for Privacy Preserving Publishing, *2014 IEEE 22nd International Symposium on Field-Programmable Custom Computing Machines* (2014).

[37] Jiang, W., Zhang-Shen, R., Rexford, J. and Chiang, M.: Cooperative Content Distribution and Traffic Engineering in an ISP Network, *SIGMETRICS Perform. Eval. Rev.*, Vol.37, No.1, pp.239–250 (2009).

[38] Little, J.D.C.: Little's Law As Viewed on Its 50th Anniversary, *Oper. Res.*, Vol.59, No.3, pp.536–549 (2011).

[39] Twain, M.: Chapter 3 - Routing Protocol Framework and Principles, *Network Routing* (Medhi, D. and Ramasamy, K., eds.), Morgan Kaufmann, pp.194–236 (2007).

[40] Hofmann, M. and Beaumont, L.R.: *Content Networking: Architecture, Protocols, and Practice* (*The Morgan Kaufmann Series in Networking*), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2005).

[41] Shaikh, A., Tewari, R. and Agrawal, M.: On the effectiveness of DNS-based server selection, *INFOCOM 2001, 20th Annual Joint Conference of the IEEE Computer and Communications Societies, Proceedings, IEEE*, Vol.3, pp.1801–1810 (2001).

[42] Krishnan, R., Madhyastha, H.V., Srinivasan, S., Jain, S., Krishnamurthy, A., Anderson, T. and Gao, J.: Moving Beyond End-to-end Path Information to Optimize CDN Performance, *Proc. 9th ACM SIGCOMM Conference on Internet Measurement Conference*, *IMC'09*, pp.190–201 (2009).

[43] Mockapetris, P.: DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION, IETF (online), available from ⟨https://goo.gl/ZkfFVV⟩ (accessed 2016-03-20).

[44] Hitachi: Hitachi Ltd. to Launch WAN Accelerator Family, Hitachi (online), available from ⟨http://goo.gl/VLk149⟩ (accessed 2016-03-20).

[45] Pearce, C.: MULTIPATH TCP, PWNING TODAY'S NETWORKS WITH TOMORROW'S PROTOCOLS (online), available from ⟨https://goo.gl/FkacVo⟩ (accessed 2016-03-20).

[46] Surton, R., Birman, K., Broberg, R. and Marian, T.: Man-in-the-Middle TCP Recovery (online), available from ⟨http://goo.gl/VN6Ph3⟩ (accessed 2016-03-20).

[47] Pathan, M. and Buyya, R.: A Taxonomy and Survey of Content Delivery Networks, *University of Melbourne, Technical Report GRIDS-TR-2007-4* (2007).

[48] OBI, F.: Broadband Performance (online), available from ⟨https://goo.gl/pLF6pI⟩ (accessed 2016-03-20).

[49] Schulzrinne, H., Rao, A. and Lanphier, R.: Real Time Streaming Protocol (RTSP) (online), available from ⟨https://goo.gl/oHXdyk⟩ (accessed 2016-03-20).

[50] Kamiyama, N., Mori, T., Kawahara, R. and Hasegawa, H.: Optimally Designing ISP-Operated CDN, *IEICE Trans. Communications*, Vol.E96-B, No.3, pp.790–801 (2013).

[51] Antic, M., Maksic, N., Knezevic, P. and Smiljanic, A.: Two phase load balanced routing using OSPF, *IEEE Journal on Selected Areas in Communications*, Vol.28, No.1, pp.51–59 (2010).

[52] ITU-T: Study Group 13, Future networks including cloud computing, mobile and next-generation networks, ITUT (online), available from ⟨http://goo.gl/DLz6OD⟩ (accessed 2016-03-20).

[53] Xie, H., Yang, Y.R., Krishnamurthy, A., Liu, Y.G. and Silberschatz, A.: P4P: Provider Portal for Applications, *SIGCOMM Comput. Commun. Rev.*, Vol.38, No.4, pp.351–362 (2008).

[54] Aggarwal, V., Feldmann, A. and Scheideler, C.: Can ISPS and P2P Users Cooperate for Improved Performance?, *SIGCOMM Comput. Commun. Rev.*, Vol.37, No.3, pp.29–40 (2007).

[55] Poese, I., Frank, B., Ager, B., Smaragdakis, G. and Feldmann, A.: Improving Content Delivery Using Provider-aided Distance Information, *Proc. 10th ACM SIGCOMM Conf. on Internet Measurement*, *IMC'10*, pp.22–34 (2010).

[56] Mohit, C., Andrey, E. and Koponen, T.: Data Oriented Network Architecture (DONA) (online), available from ⟨https://goo.gl/I7H8Y0⟩ (accessed 2016-03-20).

Janaka L. Wijekoon acquired his B.S. degree from SLIIT, Sri Lanka in 2010, and M.S. and Ph.D. degrees from Keio University, Japan, in 2013 and 2016, respectively. Currently, he is a Post-doctoral researcher affiliated with Hiroaki Nishi laboratory, Department of System Design Engineering, Keio University, Japan. He is researching towards a collaborative infrastructure for service-based future Internet including infrastructure design, device state-aware routing, and high performance routers for edge computing.

Erwin H. Harahap received his B.S. and M.S. degrees from Padjadjaran University, Indonesia, in 1994 and from Keio University, Japan, in 2010. Since 2011, he has been a Ph.D. student in Hiroaki Nishi laboratory, Keio University, Japan. He is researcher and lecturer in Mathematics Department, Bandung Islamic University, Indonesia. His research interests include network management systems, artificial intelligence with bayesian networks, network modeling Design and Simulations, and content distribution network.

Rajitha L. Tennekoon received his B.S. (Hons) degree in computer systems and networks and the M.S. degree in information systems security from the Sheffield Hallam University, UK, in 2009 and 2010, respectively. Since 2013, he has been working toward a Ph.D. degree at the Hiroaki Nishi Laboratory, Keio University, Japan. His research interests include information systems security, network security, future Internet technologies, deep packet inspection (DPI), and service-based Internet infrastructure design.

Hiroaki Nishi received his Ph.D. degree from Keio University, Japan, in 1999. Since then, he was a researcher in Real World Computing Partnership. Since 2002, he was a researcher at the Central Research Laboratory, Hitachi Ltd. He joined Keio University in 2003. Since 2014, he has been a Professor at Keio University, and a Visiting Professor at National Institute of Informatics (NII), Japan. He is investigating the Next-generation IP router architecture and Effective Network Systems for the future smart community.