

How Can an ISP Merge with a CDN?

Kideok Cho, Hakyung Jung, Munyoung Lee, Diko Ko, Ted "Taekyoung" Kwon, and Yanghee Choi,
Seoul National University

ABSTRACT

As delivering contents has become the dominant usage of the Internet, efficient content distribution is one of the hottest research areas in the network community. In future networks, it is anticipated that network entities such as routers will be equipped with in-network storage due to the trend of ever decreasing storage cost. In this article, we propose a novel content delivery architecture called ISP-centric content delivery (*iCODE*) by which an ISP can provide content delivery services as well. *iCODE* can provide efficient content delivery services since an ISP can cache contents in routers with storage modules considering traffic engineering and the locality of the content requests. Compared to CDN and P2P systems, *iCODE* can offer reduced delivery latency by placing the contents closer to end hosts, and incentives to ISPs by reducing inter-ISP traffic and allowing traffic engineering. We also discuss the technical and business issues to realize the *iCODE* architecture.

INTRODUCTION

Over the past decade, more and more traffic on the Internet is attributed to content-oriented services such as file download and web access [1]. The original Internet architecture, however, is designed for host-oriented services like remote login. The discrepancy between the endpoint-based TCP/IP protocol suite and content-oriented user demands has brought some problems. For instance, when multiple users close to each other download the same content from a distant server, each download will take place separately and hence take a long time to finish. Furthermore, if there is a surge of user access on a server (so-called flash crowd), it will overwhelm the server, which results in low throughput or even unavailability. The above inefficient delivery and lack of service availability happen because the Internet does not know the contents it carries.

To efficiently provide content delivery services with the current Internet architecture, there have been two representative solutions: peer-to-peer (P2P) systems like BitTorrent and content delivery networks (CDNs) like Akamai. A P2P system is a distributed overlay network composed of cooperative end hosts (or peers). Since peers upload/download contents among themselves, there is no need for a centralized server responsible for contents distribution, which makes P2P

systems scalable. In this way, P2P systems can handle a surge of user demands for a particular content. One of the latest well-known P2P systems is BitTorrent, which adopts *swarming*. The swarming technique allows a peer to receive multiple pieces of content from multiple peers in parallel, which even lowers the overhead of content transfer on each participating peer. However, since a P2P network is blind to the underlying network connectivity and a peer's topology information is limited, a peer in the P2P network may download the content from a distant peer even if a peer in close proximity has the content. Also, if a peer downloads the content from a peer residing in another ISP, it will increase the incoming traffic from another ISP [2], which in turn will have a negative impact on the contract between ISPs. To reduce the inter-ISP traffic volume, ISPs should be involved in the peer selection process [3]. Moreover, the crucial weakness of P2P systems is unavailability, since peers are not stably connected to the P2P systems.

CDNs have been a successful business model, providing stable and efficient content delivery services leveraging distributed data centers (often worldwide). A CDN provider distributes copies of contents to its servers in data centers upon request of content providers. When a user requests a content, the request is redirected to the CDN server in closest proximity for fast content delivery. In other words, a CDN provider pushes (or copies) the contents from the origin server to multiple (geographically distributed) servers toward end hosts. However, since the CDN provider coordinates its CDN servers to service end hosts independent of the ISP, the overall performance of the ISP network is not optimal from the traffic engineering perspective [4]. Also, the CDN service might not be affordable to small-scale content providers due to the cost issue, which means that the CDN service cannot be flexibly provided to a wide range (in terms of traffic demand) of content providers.

Recently, there are emerging technical trends worth noting. The storage cost decreases steadily and exponentially, which enables many network devices (e.g., access points and set-top boxes) to be equipped with ample storage modules. Also, major router manufacturers (e.g., Cisco and Juniper) expose programming interfaces that allow packet manipulation by third-party applications that add new services to routers [5]. Projecting this trend, it is expected that network devices will be able to cache contents in the

foreseeable future [6, 7]. That is, it is possible for routers or network entities to exploit the attached in-network storage modules for content caching and content delivery.

This article develops this idea, and proposes an efficient and flexible content delivery architecture, called ISP-centric content delivery (*iCODE*). *iCODE* caches contents by exploiting storage modules attached to routers and delivers the contents to end hosts by leveraging swarming for stable and fast delivery. The main advantages of *iCODE* are summarized as follows:

- **User experiences:** By placing contents at reliable network entities usually closer to the end hosts, *iCODE* achieves stable and reduced-latency content transfer.
- **Traffic engineering:** When *iCODE* places the cached contents into its routers, it can perform traffic engineering considering:
 - The network topology and the bandwidth capacity of individual links.
 - The popularity and spatial/temporal locality of contents.
- **Incentives to ISPs:** Once contents from other ISPs are downloaded, they can be cached inside the ISP (which deploys *iCODE*) using in-network storage. In this way, *iCODE* reduces the inter-ISP traffic and hence gives the ISP financial advantages.
- **New business model:** The *iCODE* architecture allows the ISP to provide flexible CDN services for content providers with various traffic demands.
- **Incremental deployment:** Two facts facilitate an incremental deployment of *iCODE*:
 - The *iCODE* architecture retains backward compatibility with the current Internet.
 - A single ISP can provide the *iCODE* service independent of other ISPs.

The remainder of this article is organized as follows. We describe the details of the *iCODE* architecture. We present simulation results, and discuss the technical and business issues of *iCODE*. We compare the related proposals on content-oriented networking with *iCODE*. We conclude this article with future work.

/iCODE ARCHITECTURE

/iCODE OVERVIEW

The *iCODE* architecture is illustrated in Fig. 1. To provide efficient and flexible content delivery services, the *iCODE* architecture employs the swarming technique used in BitTorrent and exploits in-network storage modules in network entities such as routers. The network entities cache the contents and service content requests from users. We anticipate that *iCODE* will provide a feasible business model to ISPs.

To retain the backward compatibility with the current Internet, we assume that every content is identified by a uniform resource identifier (URI) (e.g., server1.com/movie/scene1.mpg). When a user requests a particular content, the *iCODE* architecture will intercept the request and check if the requested content is cached inside the ISP network. If the content exists, *iCODE* returns the addresses of multiple routers holding the content, and the user will download the content

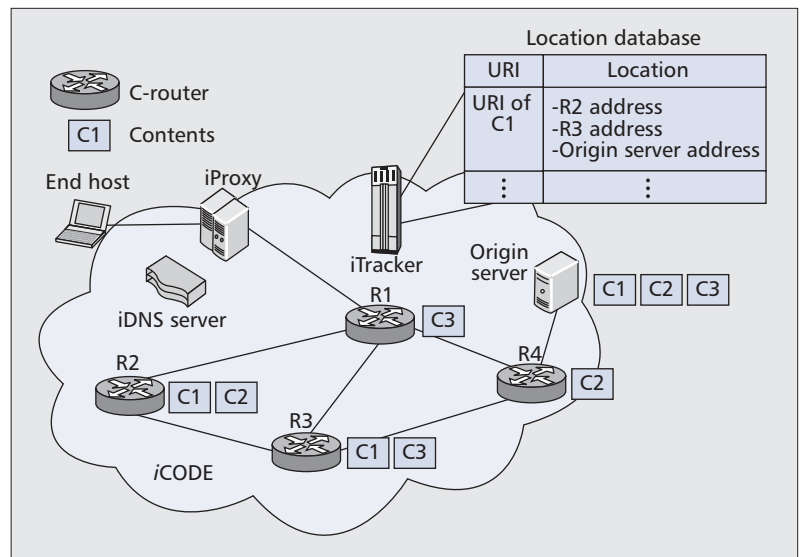


Figure 1. An ISP operating the *iCODE* architecture.

from the multiple routers through parallel transmissions. Otherwise, the content request will follow the current Internet practice (i.e., client-server model), but the downloaded contents may still be cached inside the ISP.

For the sake of exposition, we assume that one physical network (or autonomous system) is owned and operated by a single ISP. We assume that an ISP has many routers with storage modules for content caching [6, 7], which are called content routers or C-routers for short. If *iCODE* concludes that a particular content is popular and has to be cached, it will store the content at multiple C-routers. In this way, a subsequent request for the same content can be serviced from the C-routers within the ISP network.

/iCODE COMPONENTS

We explain the main components of *iCODE* before describing the detailed *iCODE* operations.

***iCODE* Proxy (*iProxy*):** An *iProxy* receives a content request from an end host and sends the lookup request to an *iTracker* (see below) to check whether the content is cached. If the *iTracker* knows the location of the content, it will reply with the location (IP addresses of C-routers holding the content). Then the *iProxy* will download the content, which, in turn, is delivered to the end host. If the *iTracker* does not know the content, the *iProxy* performs domain name system (DNS) resolution for the specified universal resource identifier (URI) and sends the content request to the corresponding server. The *iProxy* is a functional entity that can be collocated with an end host, a middlebox like a network address translator (NAT), or an access router. The main reason the *iProxy* is logically separated from the end host is to adopt the swarming technique transparently to the end host. That is, *iProxy* may have to receive multiple chunks of the content from multiple C-routers in parallel, just like BitTorrent.

***iCODE* Tracker (*iTracker*):** An *iTracker* is operated by an ISP and is responsible for managing contents inside the ISP. That is, it man-

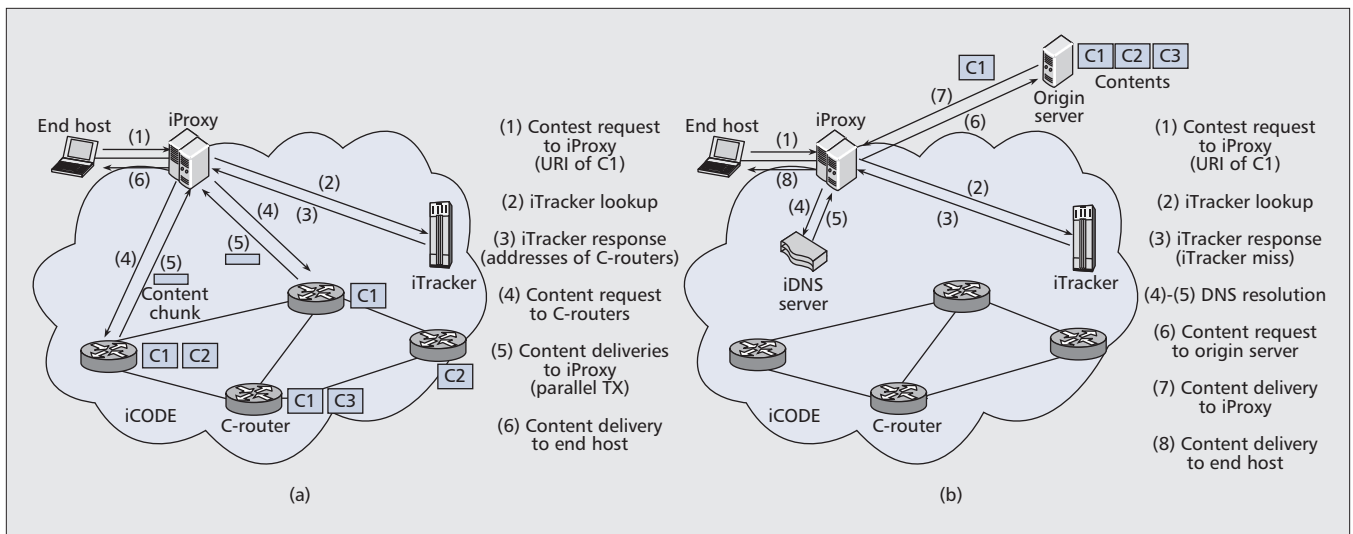


Figure 2. Operations of iCODE when the request comes from the end host inside the ISP.

ages which contents to cache and where to cache them. The *iTracker* maintains the mapping between the content URI and its location (IP addresses of C-routers) in the location database as shown in Fig. 1. Also, the *iTracker* maintains the mapping for the contents stored in the origin server which subscribes to the *iCODE* service. If there are too many contents in a single ISP, there should be multiple *iTrackers* in the ISP. *iTracker* can be implemented in a distributed manner to avoid a single point of failure. When the lookup request arrives from the *iProxy*, if the *iTracker* knows where the content is, it will choose multiple C-routers to deliver the requested content. Note that the *iTracker* can be extended to support ISP-friendly P2P services similar to P4P [3] by tracking the contents of end hosts within the ISP network.

Origin server: An origin content server maintains the content published by the content provider. The origin server registers the metadata of the content to the *iTracker* for the *iCODE* service.

Content router (C-router): A C-router is a router that has a storage module, which can cache copies of contents. It performs content delivery services (in addition to packet routing) upon the request from the *iProxy*. The *iTracker* will manage which contents will be cached and replaced (if the storage is full) at individual C-routers.

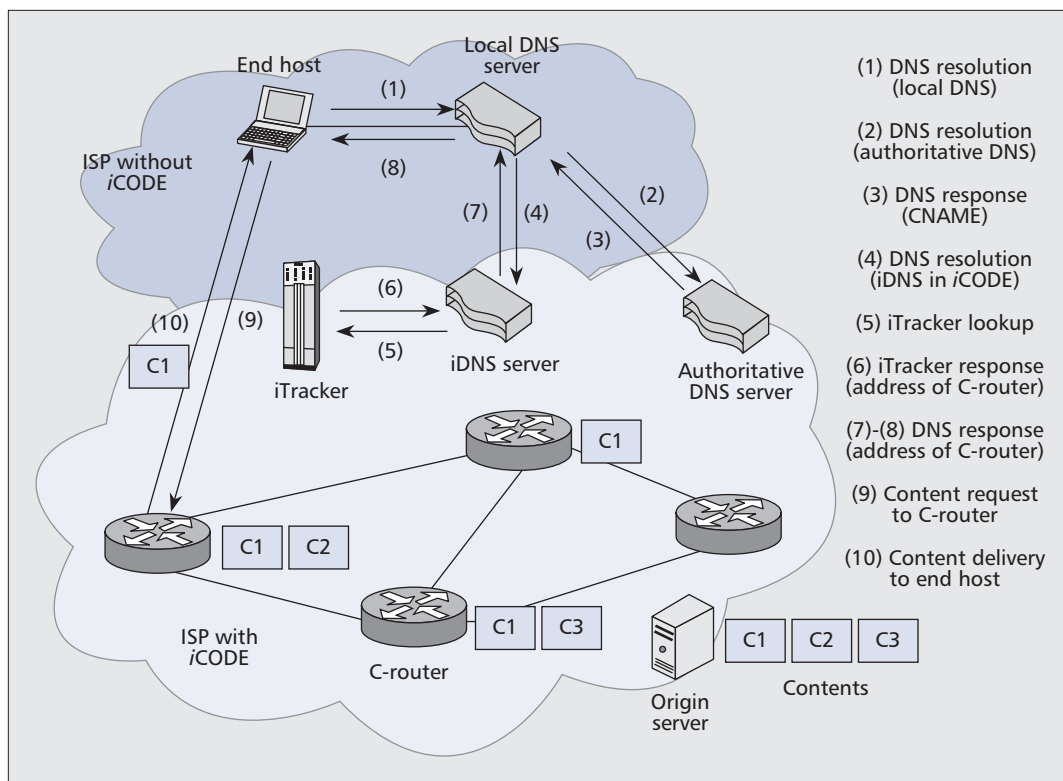
iCODE DNS (iDNS): An ISP providing the *iCODE* service maintains *iDNS* servers for Domain Name Service (DNS) query redirection. When an end host outside the ISP requests a content whose origin server belongs to the ISP, the DNS query will be forwarded to the authoritative DNS server that manages the domain name of the origin server. In the corresponding DNS record, there is a canonical NAME (CNAME) record to redirect the DNS query to the *iDNS* server. The *iDNS* server will contact the *iTracker* to return the IP address of a C-router caching the requested content. Note that the host outside the ISP is not aware of *iCODE*, and hence there is no *iProxy* that performs swarming.

OPERATIONS FOR A CONTENT REQUEST FROM INSIDE THE ISP WITH iCODE

This section explains the operations of *iCODE* when a content request comes from an end host within the ISP that provides the *iCODE* service as illustrated in Fig. 2. The *iCODE* operations are different depending on whether the content copies are stored in the ISP or not.

- The overall operation when there are cached content copies within the ISP is illustrated in Fig. 2a. (1) If an end host wishes to download content, it will send the content request to the *iProxy* in the form of a URI. (2) On receiving the content request, the *iProxy* first consults the *iTracker*, which maintains the location database of contents (which C-routers store the contents.) (3) After checking the location database, the *iTracker* will reply with the IP addresses of the C-routers. (4) After receiving the response, the *iProxy* will request the content to the C-routers. (5) On receiving the content request, the C-routers will transmit the requested content to the *iProxy* leveraging the swarming technique, which improves the download speed and mitigates the burden of individual C-routers. (6) The *iProxy* will forward the received contents to the end host. Note that even if the contents of the origin server are not yet cached in C-routers, the *iTracker* will reply with the IP address of the origin server, since the *iTracker* maintains the metadata of the contents of the origin servers inside the ISP.

- When there is no cached copy within the ISP, *iCODE* operates as illustrated in Fig. 2b. (1) The end host sends a content request to the *iProxy*. (2) Since there is no cached content within the ISP, *iTracker* concludes that the content is missing. (3) In this case, the *iTracker* informs the *iProxy* that the content is missing. (4) Then the *iProxy* will perform DNS resolution to find out the IP address of the origin server. (5) The DNS resolution will return the IP address of the origin server outside the ISP. (6)-(7) The *iProxy* will establish a TCP/IP session with the origin server to retrieve the requested content. (8) After finishing content download, the *iProxy* forwards the content to the end host. In case of successful downloading, the *iProxy* informs the *iTracker* of the result. Considering the download history of



If *iCODE* concludes that a particular content is popular and has to be cached, it will store the content at multiple C-routers. In this way, a subsequent request for the same content can be serviced from the C-routers within the ISP network.

Figure 3. Operation of *iCODE* when the request comes from an end host outside the ISP.

the contents, the *iTracker* may decide to keep copies of the contents at some C-routers.

OPERATIONS FOR A CONTENT REQUEST FROM OUTSIDE THE ISP WITH *iCODE*

Figure 3 shows the operations of *iCODE* when a content request comes from an end host outside the ISP that provides the *iCODE* service. Suppose that `server1.com` is the URI of the origin server which subscribes to the *iCODE* service by the ISP and the CNAME of the origin server for the DNS redirection is `server1-com.iCODE1.com`. (1) When an end host requests content from the origin server that subscribes to the *iCODE* service, it will first perform the DNS resolution. The end host sends a DNS query to the local DNS server, which in turn consults the authoritative DNS server of the origin server. (2) The DNS query of the local DNS server arrives at the authoritative DNS server for `server1.com`. However, since the origin server subscribes to the *iCODE* service, the DNS query will be redirected to the *iDNS* server. For the redirection, there is a CNAME in the corresponding DNS record in the authoritative DNS server. (3) Accordingly, the authoritative DNS server will reply with CNAME, `server1-com.iCODE1.com`. (4) On receiving the DNS response with the CNAME, the local DNS server of the host proceeds the DNS resolution process by sending a DNS query to the *iDNS* server maintained by the ISP. (5) The *iDNS* server consults the *iTracker* to locate the content (which C-router caches the content.) (6) After checking the location database, the *iTracker* replies with the IP address of the C-router caching the content. As the end host does not use an *iProxy*, it can download the content only from the single source. Note

that the *iTracker* can choose the C-router considering latency, current traffic load, and traffic engineering. (7)–(8) Now, the DNS response containing the IP address of the C-router is forwarded to the end host through the *iDNS* server and the local DNS server. (9) On obtaining the IP address of the C-router, the end host sends a content request. (10) The C-router will transfer the requested content to the end host.

SIMULATION RESULTS

We evaluate the performance of *iCODE* by using a discrete event-driven simulator. Our simulation environments are configured as follows. For the realistic performance evaluation, we generated 10 transit-stub topologies by using GT-ITM [8] which is a well-known Internet-like topology generator. They consist of 1 transit domain with 5 routers, and 5 stub domains with 20 routers and 200 end hosts each. There are randomly distributed 1000 contents with 1 Gbyte size, and the number of content requests follows Zipf distribution with parameter 1.0. We compare the average hop counts, link stress, and inter-ISP traffic volume of *iCODE* with those of CDN, P2P, and client-server model. The cache size of all C-routers is set to 10 Gbytes. The caching policy for the *iCODE* is a simple round-robin among C-routers. For the CDN server deployment, we assume the ISP-operated CDN model, which can deploy the server at the best position in the ISP network in terms of hop count. CDN servers store 500 contents (i.e., 50 percent of all contents). For the peer selection in P2P, each peer first selects peers within the same ISP network, similar to P4P [3] (and then adds peers in other ISP networks only when it is not able to find 10

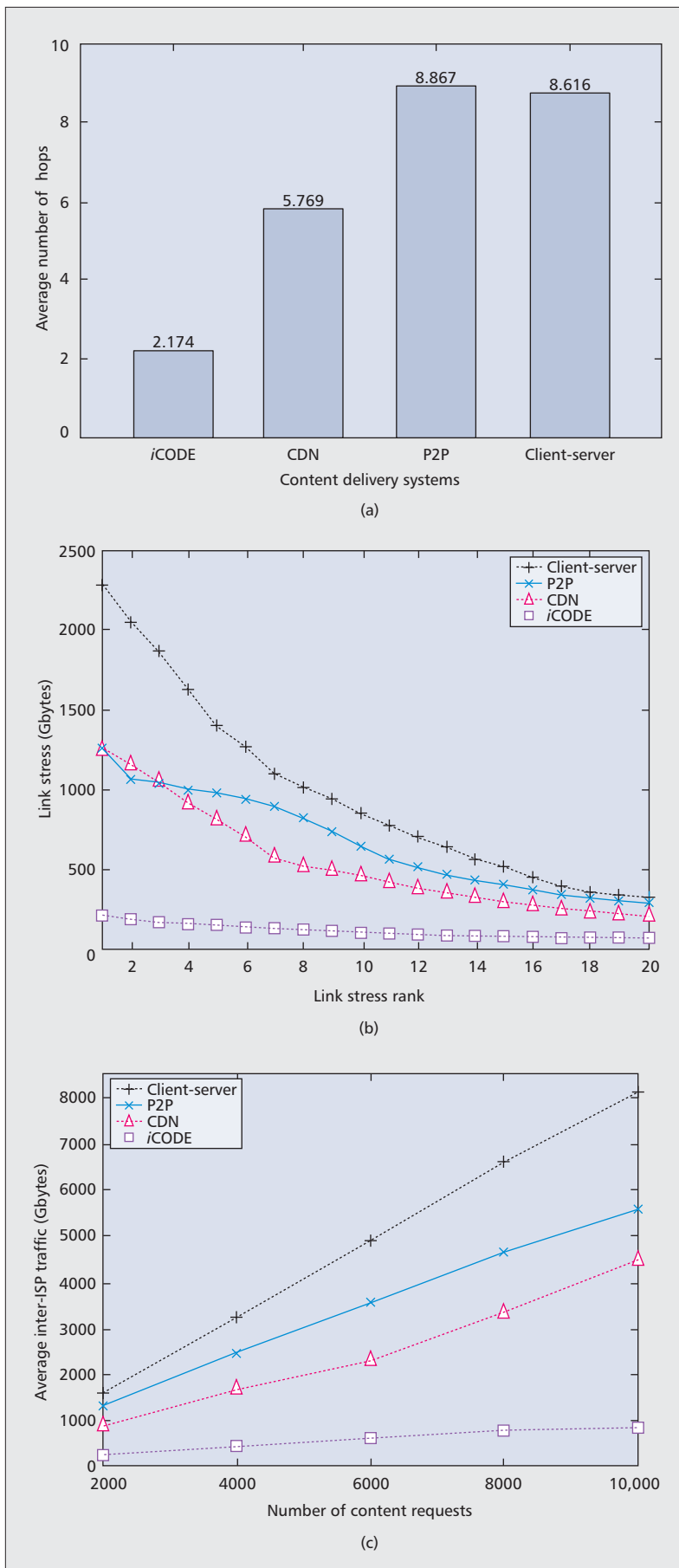


Figure 4. Performance comparison of *iCODE* against CDN, P2P, and client-server model.

peers in the same ISP network). We assume that 100 peers are online on average.

User Experiences: We evaluate the user-experienced performance for each scheme in terms of the average hop counts traversed for the content delivery. As shown in Fig. 4a, users in an ISP with *iCODE* will experience the most reduced transfer delay due to the shortest hop counts. The reason is that the cached content will be delivered from the nearby C-routers inside the network; that is, the C-router is typically closer to the requesting host than CDN servers, peers, or origin servers. In CDN, content requests for contents not stored in the CDN servers will be serviced by the origin servers, resulting in a larger average number of hops than *iCODE*. The client-server model and P2P incur much larger numbers of hops because a considerable amount of contents are originated from servers or peers outside the ISP.

Load Balancing and Traffic Engineering: *iCODE* can achieve load balancing not only among C-routers but also among the links of the ISP network. We measure the link stress for all links, which is defined as the amount of traffic volume passed over a particular link, and plot the 20 links with the highest stress in descending order of link stress in Fig. 4b. *iCODE* uses the links more evenly than other models. Compared to CDN, *iCODE* distributes contents to multiple C-routers; thus, *iCODE* shows lower link stress performance than CDN (80 percent reduction for the top 20 links). CDN can reduce link stress better than the client-server model by using distributed CDN servers. P2P can download from multiple peers, resulting in better performance than the client-server model.

Inter-ISP Traffic Reduction: *iCODE* offers economic incentives to ISPs by reducing inter-ISP traffic. As shown in Fig. 4c, *iCODE* incurs the smallest volume of inter-ISP traffic since *iCODE* services contents from the C-routers inside the ISP once the contents have been retrieved from other ISPs. Even with a large number of content requests, *iCODE* reduces inter-ISP traffic consistently. On the other hand, the other models incur very large amounts of inter-ISP traffic. Since CDN servers cannot store all contents, the CDN model incurs inter-ISP traffic to service requests for contents not stored in CDN servers. P2P results in relatively less inter-ISP traffic than the client-server model since P2P can download part of the content from peers within the same ISP network, whereas the client-server model is fully blind to the underlying network connectivity among ISPs.

DISCUSSIONS

This section discusses remaining technical and business issues of *iCODE*.

Customized Content Delivery Service: The *iCODE* service is affordable to small-scale content providers by supporting a wide range of content providers (in terms of traffic demand). Since the *iCODE* service is provided by the ISP that already provides the Internet connectivity, the content delivery service is possible presumably at a smaller fee than the legacy CDN services. Also, it will be attractive to provide server-load-aware *iCODE* service to the small-scale content pro-

	<i>i</i> CODE	DONA [6]	CCN [7]	PSIRP [10]
Naming	URI	Flat name with a key to authenticate the publisher	Hierarchically encoded binary name based on URI	Flat id of the publication with the scope
Content resolution	<i>i</i> Tracker	Hierarchical resolution handler (RH) topology	Flooding	Rendezvous system
ISP incentive regarding traffic	Inbound and internal network traffic reduction, traffic-aware content delivery	Inbound traffic reduction	Inbound and internal network traffic reduction	Efficient network utilization for multicasting applications

Table 1. Comparisons of innovative content delivery systems.

viders. Since the *i*Tracker participates in every content downloading process, it can control the source of the content transfer: the origin server or the specific C-routers. When the number of content requests is under a certain threshold, the requests are served by the origin server. If the number of requests exceeds the threshold (traffic overload), the requests can be served by the C-routers, which allows small content providers to perform flexible server provisioning.

Incremental Deployment: An ISP can deploy *i*CODE independent of other ISPs. Even when end users residing outside the ISP with *i*CODE request content retrievals, the ISP can service the requests using C-routers caching the requested contents and the DNS redirection mechanism. Also, to transparently provide *i*CODE services to end users, the *i*Proxy performs swarming and in-network cache lookup. Note that *i*Proxy can be implemented as a network entity if the end user's device is not affordable.

***i*CODE Substantiation over Multiple ISPs:** Network virtualization is not only considered one of the key technologies of future networks in International Telecommunication Union — Telecommunication Standardization Sector (ITU-T) Study Group 13, but is also gaining momentum in the research community because of its potential to be a network infrastructure to test various proposals for the future network (e.g., Global Environment for Network Innovations, GENI). Network virtualization is an extension of system virtualization in end hosts to network entities such as routers. An instance of the virtualized fractions of the resources in the network entities is called a slice. Thus, it is possible for an ISP to lease a slice from other ISPs [9]. Assuming that ISPs in the future network will be virtualized, an ISP can substantiate *i*CODE across multiple ISPs by leasing virtualized slices of routers with storage modules from other ISPs.

Swarming: Even though *i*CODE pushes the contents toward end hosts using in-network storage, there might be the transfer overhead on C-routers (e.g., storage access). To mitigate the transfer overhead and not to sacrifice the packet forwarding performance, we adopt the swarming technique. For this, the *i*Tracker will return the IP addresses of multiple C-routers that contain the requested content. As the number of C-routers containing the same content increases, we can lower the transfer overhead of each C-

router. Also, not every C-router needs to hold the entire content; the individual C-routers need only to keep chunks of the content (like downloading different pieces of the same content from multiple peers in BitTorrent).

Cache Management: Obviously, the total storage space is limited, even though the number of C-routers is large and the storage cost is low. We should note that enlarging the storage space of C-routers will incur more upgrade labor cost compared to CDNs due to the geographically distributed nature of routers. Thus, *i*CODE should efficiently manage the in-network storage with the *i*Tracker. As the popularity of a content changes, *i*Tracker can change the number of the content copies inside the ISP, thus allowing cost-effective storage management. Also, the request pattern for a particular content varies spatially and temporally (e.g., time zones, diurnal pattern, spatial locality of contents). Thus, *i*Tracker will be able to flexibly migrate the contents among C-routers considering the above changes.

RELATED WORK

There have been innovative approaches to achieve efficient content delivery in a non-Internet-compatible manner. We compare *i*CODE with these innovative proposals as summarized in Table 1.

Data-Oriented Network Architecture (DONA) [6] proposes to use flat, self-certifying names instead of URLs; hence, DNS name resolution is replaced by a name-based anycast. In DONA, the name resolution is done by a new class of network entities called resolution handlers (RHs). While a failed content request is forwarded to the origin server in *i*CODE, every content request in DONA is forwarded along the hierarchy of RHs, assuming that all the ISPs are DONA-compliant. Although DONA guarantees perfect global availability of contents, the system works only when RHs are deployed over all ISPs. It makes deployment of DONA difficult since ISPs do not have motivation to deploy the RHs. On the contrary, users can benefit from *i*CODE even if only their ISP supports *i*CODE.

Content-centric networking (CCN) [7] extends the URI structure to name contents in a hierarchical manner for human readability, which in turn is mapped into the binary encoding. Content requests are binary-encoded as Interest packets and one

In the future, we will conduct comprehensive evaluations over a large scale testbed by implementing iCODE with real hardware. Also, we will investigate the details of content caching policy and delivery issues in iCODE.

Interest packet solicits one Data packet. The new network entity called a CCN node is somewhat similar to a router in the current Internet; it resolves and forwards Interest/Data packets; it also caches Data packets to reduce the network traffic and hence to enhance the availability. However, there may be many redundant copies in CCN if individual CCN nodes decide to cache their copies, whereas in iCODE the copies of content are managed by the iTracker of the ISP. Also, unlike iCODE, there is no business model for ISPs to introduce CCN except inbound traffic reduction.

Publish-Subscribe Internet Routing Paradigm (PSIRP) [10] is an architecture with the goal of building a publish/subscribebased network. Basically, PSIRP tries to form a multicast tree for each content; the rendezvous system matches the publisher and subscriber and the topology system constructs multicast trees. Even though they propose the use of Bloom filters, Merkle trees and special layer 2 hardware, multicast routing scalability will still be a concern. In iCODE, the scalability problem might happen in iTracker, however, it can be solved with distributed implementation. Also, a business model attractive to ISPs is not identified.

CONCLUSION

This article proposes a new content delivery architecture called iCODE. With the in-network storage modules in routers, iCODE locates the contents closer to the end hosts, resulting in stable and reduced latency of content delivery. Also, iCODE provides incentives to an ISP by reducing the inter-ISP traffic with locally cached contents and by allowing traffic engineering that considers the network status. Furthermore, iCODE opens the possibility of a new business model by which an ISP can support a wide range of content providers. In the future, we will conduct comprehensive evaluations over a large scale testbed by implementing iCODE with real hardware. Also, we will investigate the details of content caching policy and delivery issues in iCODE.

ACKNOWLEDGMENT

We would like to thank Dr. Mostafa Hashem Sherif and the anonymous reviewers for their valuable comments that greatly improved this article. This work was supported in part by NAP of Korea Research Council of Fundamental Science & Technology and in part by the IT R&D program of KCA (10913-05004: Study on Architecture of Future Internet to Support Mobile Environments and Network Diversity). This publication is partially based on work performed in the framework of the project COAST-ICT-248036, which is partially supported by the European Community.

REFERENCES

- [1] ipoque, Internet Study 2008/2009, <http://www.ipoque.com/resources/internetstudies/internet-study-2008,2009>.
- [2] T. Karagiannis, P. Rodriguez, and K. Papagiannaki, "Should Internet Service Providers Fear Peer-Assisted Content Distribution," *Proc. ACM IMC*, Oct. 2005.
- [3] H. Xie et al., "P4P: Provider Portal for Applications," *Proc. ACM SIGCOMM*, Aug. 2008.
- [4] W. Jiang et al., "Cooperative Content Distribution and Traffic Engineering in An ISP Network," *Proc. SIGMETRICS*, June 2009.

- [5] J. Kelly, W. Araujo, and K. Banerjee, "Rapid Service Creation Using the JUNOS SDK," *Proc. ACM SIGCOMM Wksp. PRESTO*, Aug. 2009.
- [6] T. Koponen et al., "A Data-Oriented (and Beyond) Network Architecture," *Proc. ACM SIGCOMM*, Aug. 2007.
- [7] V. Jacobson et al., "Networking Named Content," *Proc. ACM CoNEXT*, Dec. 2009.
- [8] E. W. Zegura, K. Calvert, and S. Bhattacharjee, "How to Model An Internetwork," *Proc. IEEE INFOCOM*, Mar. 1996.
- [9] G. Schaffrath et al., "Network Virtualization Architecture: Proposal and Initial Prototype," *Proc. ACM SIGCOMM Wksp. VISA*, Aug. 2009.
- [10] A. Zahemszky, A. Csaszar, P. Nikander, and C. Esteve, "Exploring the Pub/Sub Routing & Forwarding Space," *Proc. ICC Wksp. Future-Net*, June 2009.

BIOGRAPHIES

KIDEOK CHO (kdcho@mmlab.snu.ac.kr) received his B.S. (magna cum laude) and Ph.D. degrees from Seoul National University, Korea in 2004 and 2011, respectively. Since September 2011, he has been a postdoctoral researcher at Seoul National University. He was a visiting researcher at the Advanced Wireless Networking Laboratory, City University of New York in 2007. He was a recipient of the Korea Student Aid Foundation (KOSAF) scholarship from 2009 to 2010. Since 2006 he has been a technical secretary of the Future Internet Forum, Korea. His research interests include content-oriented networks and wireless/mobile networks.

HAKYUNG JUNG (hkjung@mmlab.snu.ac.kr) received a B.S. degree in computer science and engineering from Seoul National University in 2005, and is working toward a Ph.D. degree at the Multimedia & Mobile Communications Laboratory, Seoul National University. He was a visiting researcher at the Advanced Wireless Networking Laboratory, City University of New York. He was also a student intern at INRIA Rennes, France. His research interest lies in the field of wireless networks as well as content-oriented networks.

MUNYOUNG LEE (mylee@mmlab.snu.ac.kr) received his B.S. degree in computer science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, in 2007. He is currently working toward his Ph.D. degree at the School of Computer Science and Engineering, Seoul National University. His research interests include content-oriented networking and peer-to-peer networks.

DIKO KO (diko@mmlab.snu.ac.kr) received his M.S. degree from School of Computer Science and Engineering, Seoul National University, in 1999. He has worked in the computer gaming industry more than 10 years. He developed the online game "Lineage" in 1998 as a software engineer and created "Yogurting" as the producer in 2005. He also worked as senior director at Cyworld, Inc. in San Francisco, California in 2008. He is currently the CEO of Altwave Lab, Inc. and also working toward a Ph.D. degree at Multimedia & Mobile Communications Laboratory, Seoul National University.

TED "TAEKYOUNG" KWON (tkkwon@snu.ac.kr) is with the School of Computer Science and Engineering, Seoul National University. He was a visiting professor at Rutgers University in 2010. Before joining Seoul National University, he was a postdoctoral research associate at the University of California Los Angeles and City University of New York. He obtained his B.S., M.S., and Ph.D. at Seoul National University. He was a visiting student at IBM T. J. Watson Research Center and the University of North Texas. His research interests lie in future Internet, content-oriented networking, and wireless networks.

YANGHEE CHOI (yhchoi@snu.ac.kr) received his B.S. in electronics engineering at Seoul National University, M.S. in electrical engineering at KAIST, and D.Eng. in computer science at Ecole Nationale Supérieure des Telecommunications, Paris, France, in 1975, 1977, and 1984, respectively. He worked at the Electronics and Telecommunications Research Institute (Korea), Centre National d'Etude des Telecommunications (France), and IBM Thomas J. Watson Research Center (United States) before joining Seoul National University in 1991. He was the dean of the Graduate School of Convergence Science and Technology and president of the Korea Institute of Information Scientists and Engineers. He is president of the Advanced Institutes of Convergence Technologies, and chairman of the Future Internet Forum of Korea. He has published over 600 papers on network protocols and architectures.