HOW CAN AN ISP MERGE WITH A CDN?

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

School of Computer Science and Engineering,
Seoul National University, Seoul, Korea
Email: {kdcho, hkjung, mylee, diko}@mmlab.snu.ac.kr, {tkkwon, yhchoi}@snu.ac.kr

ABSTRACT

*As delivering contents has become the dominant usage of Internet, the efficient content distribution is being one of the hottest research areas in network community. In future network, 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 paper, we propose a novel content delivery architecture called Internet Service Provider (ISP) centric Content Delivery (i*CODE*) by which an ISP can provide content delivery services as well. i*CODE *can provide efficient content delivery services since an ISP can cache the contents in routers with storage modules considering traffic engineering and the locality of the content requests. Compared with CDN and P2P systems, i*CODE *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 i*CODE *architecture.*

Index Terms— Content delivery service, Content router, In-network storage, Future network architecture, Swarming

1. INTRODUCTION

International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) Study Group 13 has recently studied the requirements of future networks by investigating various technologies such as content-oriented network, network virtualization, etc. Indeed, 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 [2]. For instance, when multiple users close to each other download the same content file from a distant server, each download will take place separately and hence will 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 since the Internet does not know the contents it carries.

To provide content delivery service efficiently with the current Internet architecture, there have been two representative solutions: peer-to-peer (P2P) systems (e.g., BitTorrent [3]) and content delivery networks (CDNs) (e.g., Akamai [4]). 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 of a centralized server responsible for contents distribution, which makes P2P systems scalable. In this way, P2P systems can handle a surge of user demands on a particular content file. One of the latest well-known P2P systems is BitTorrent that adopts *swarming*. The swarming technique allows a peer to receive multiple pieces of a content file 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 file from a distant peer even if a peer in close proximity has the file. Also, if a peer downloads the file from a peer residing in another ISP, it will increase the incoming traffic from another ISP [5], which in turn will have a negative impact on the contract between ISPs. To reduce the inter-ISP traffic volume, ISP should be involved in the peer selection process [6]. 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, which provides stable and efficient content delivery services leveraging distributed data centers (often worldwide). A CDN provider distributes the copies of contents to its servers in data centers upon the requests of content providers. When a user requests a content file, the request is redirected to the CDN server in the closest proximity for the fast content delivery. In other words, a CDN provider pushes (or copies) the contents from the origin server to multiple (geographically distributed) servers towards end hosts. However, since the CDN provider coordinates its CDN servers to service end hosts independently of the ISP, the overall performance of the ISP network is not optimal in the perspective of traffic engineering [7]. Also, the CDN service

This work was supported in part by NAP of Korea Research Council of Fundamental Science & Technology and in part by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency) (NIPA-2010-(C1090-1011-0004)).

is not affordable to small-scale content providers, which means that the CDN service can not be flexibly provided to a wide spectrum (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 [8] and set-top boxes) to be equipped with ample storage modules [9]. Also, major router manufacturers (e.g., Cisco and Juniper) expose programming interfaces that allow packet manipulation by third-party applications to add new services at routers [10, 11]. Projecting this trend, it is expected that the network devices will be able to cache the contents in the foreseeable future [12, 13, 14]. 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 paper develops this idea and proposes an efficient and flexible content delivery architecture, called ISP-centric content delivery (*i*CODE). The main advantages of the *i*CODE are summarized as follows.

1. User experiences: By placing contents at reliable network entities usually closer to end hosts, *i*CODE achieves stable and reduced latency of content transfer.

2. Traffic engineering: When *i*CODE places the cached contents into its routers, it can perform traffic engineering considering (a) the network topology and the bandwidth capacities of individual links, and (b) the popularity and the spatial/temporal locality of the contents.

3. Incentives to ISPs: Once contents from other ISPs are downloaded, they can be cached inside the ISP (that deploys *i*CODE) using in-network storage. In this way, *i*CODE reduces the inter-ISP traffic and hence gives the ISP financial advantages.

4. New business model: The *i*CODE architecture allows the ISP to provide the flexible CDN services for content providers with various traffic demands.

5. Incremental deployment: The facts that (a) the *i*CODE architecture retains the backward-compatibility with the current Internet and (b) a single ISP can provide the *i*CODE service independently of other ISPs facilitate an incremental deployment of *i*CODE.

The remainder of this paper is organized as follows. In Section 2, we describe the details of the *i*CODE architecture. Section 3 presents the simulation results, and Section 4 discusses the technical and business issues of *i*CODE. In Section 5, we compare the related proposals on content-oriented networking with *i*CODE. Section 6 concludes this paper with future work.

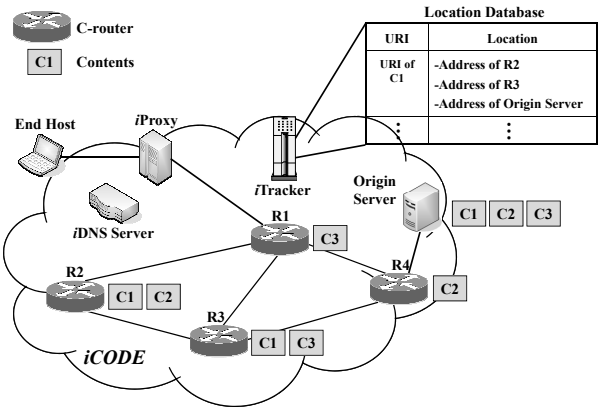


Fig. 1. An ISP operating the *i*CODE architecture is illustrated.

2. *I*CODE ARCHITECTURE

2.1. *i*CODE Overview

The *i*CODE architecture is illustrated in Fig. 1. To provide efficient and flexible content delivery services, the *i*CODE architecture employs the swarming technique adopted from 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 *i*CODE is a feasible business model to ISPs.

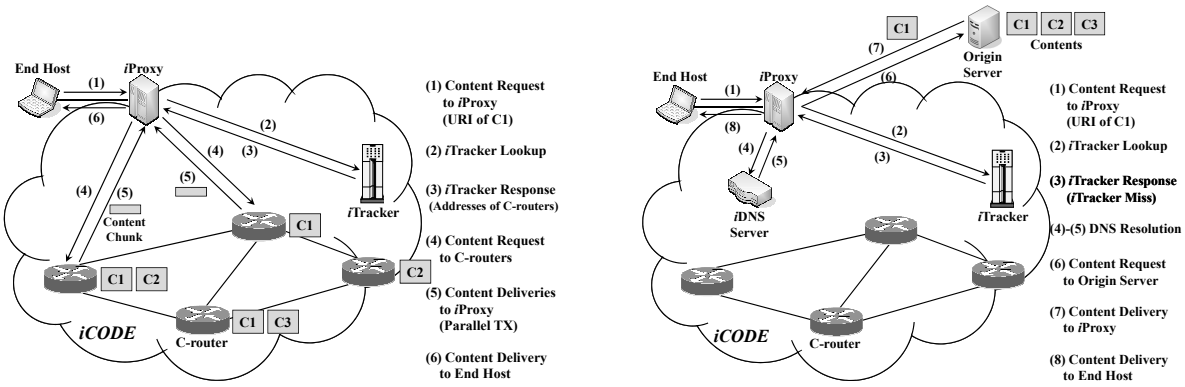
We assume that every content is identified by a uniform resource identifier (URI) to retain the backward-compatibility with the current Internet. When a user requests a particular content file, the *i*CODE architecture will intercept the request and check if the requested content file is cached inside the ISP network. If the content file exists, *i*CODE returns the addresses of multiple routers holding the file and the user will download the content file 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 [12, 13, 14], which are called content routers or C-routers for short. If *i*CODE concludes that a particular content file is popular and has to be cached, it will store the file at multiple C-routers. In this way, the following request for the same content file can be serviced from the C-routers within the ISP network.

2.2. *i*CODE Components

The main components of *i*CODE are as follows.

1. *i*CODE Proxy (*i*Proxy): An *i*Proxy receives a content request from an end host and sends the lookup request



(a) When the requested content is in the ISP.

(b) When the requested content is not in the ISP.

Fig. 2. Operations of *i*CODE when the request comes from the end host inside the ISP.

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

2. *i*CODE Tracker (*i*Tracker): An *i*Tracker is operated by an ISP and is responsible for managing contents inside the ISP. That is, it manages which contents to be cached and where to be cached. The *i*Tracker 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 *i*Tracker maintains the mapping for the contents stored in the origin server which subscribes to the *i*CODE service. If there are too many contents in a single ISP, there should be multiple *i*Trackers in the ISP. *i*Tracker can be implemented in a distributed manner to avoid the single point of failure [15]. When the lookup request arrives from the *i*Proxy and if the *i*Tracker knows where the content is, it will choose multiple C-routers to deliver the requested content. Note that *i*Tracker can be extended to support ISP-friendly P2P services similar to P4P [6] by tracking the contents of end hosts within the ISP network.

3. 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 *i*Tracker for the *i*CODE service.

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

5. *i*CODE DNS (*i*DNS): An ISP providing the *i*CODE service maintains *i*DNS servers for the DNS query redirection. When an end host outside the ISP requests a content file 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 *i*DNS server. The *i*DNS server will contact the *i*Tracker to return the IP address of a C-router caching the requested content. Note that the host outside the ISP is not aware of *i*CODE and hence there is no *i*Proxy that performs swarming.

2.3. Operations for a content request from inside the ISP with *i*CODE

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

1) The overall operation when there are cached content copies within the ISP is illustrated in Fig. 2(a). (1) If an end host wishes to download a content file, it will send the content request to the *i*Proxy in the form of a URI. (2) On receiving the content request, the *i*Proxy first consults the *i*Tracker which maintains the location database of contents. (which C-routers store the contents.) (3) After checking the location database, the *i*Tracker will reply with the IP

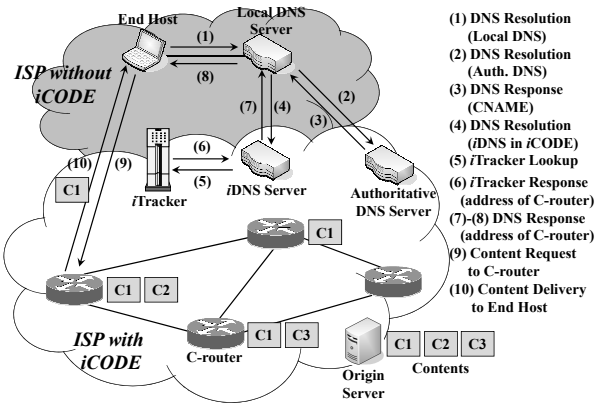


Fig. 3. Operation of *i*CODE when the request comes from the end host outside the ISP.

addresses of the C-routers. (4) After receiving the response, the *i*Proxy will request the content to the C-routers. (5) On receiving the content request, the C-routers will transmit the requested content to the *i*Proxy leveraging the swarming technique, which improves the download speed and mitigates the burden of individual C-routers. (6) The *i*Proxy will forward the received content to the end host. Note that even if the contents of the origin server are not cached in C-routers yet, the *i*Tracker will reply with the IP address of the origin server since the *i*Tracker maintains the metadata of the contents of the origin servers inside the ISP.

2) When there is no cached copy within the ISP, *i*CODE operates as illustrated in Fig. 2(b). (1) The end host sends a content request to the *i*Proxy. (2) Since there is no cached content within the ISP, the cache miss happens at the *i*Tracker. (3) In this case, the *i*Tracker informs the *i*Proxy of the cache miss event. (4) Then, the *i*Proxy 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 *i*Proxy will establish a TCP/IP session with the origin server to retrieve the requested content. (8) After finishing content download, the *i*Proxy forwards the content to the end host. In case of successful downloading, the *i*Proxy informs the *i*Tracker of the result. Considering the download history of the contents, the *i*Tracker may decide to keep the copies of the content file at some C-routers.

2.4. Operations for a content request from outside the ISP with *i*CODE

Fig. 3 shows the operations of *i*CODE when a content request comes from an end host outside the ISP that provides the *i*CODE service. Suppose that *server1.com* is the URI of the origin server which subscribes to the *i*CODE 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 a content file from the origin server that subscribes to the *i*CODE 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 *i*CODE service, the DNS query will be redirected to the *i*DNS 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*.

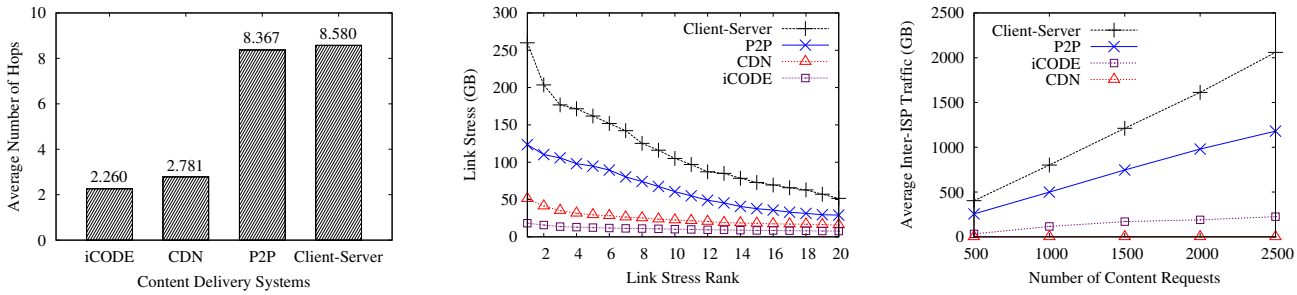
(4) On receiving the DNS response with the CNAME, the local DNS server of the host will proceed the DNS resolution process by sending a DNS query to the *i*DNS server maintained by the ISP. (5) The *i*DNS server will consult the *i*Tracker to locate the content. (which C-router caches the content.) (6) After checking the location database, the *i*Tracker will reply with the IP address of the C-router caching the content. As the end host does not use an *i*Proxy, it can download the file only from the single source. Note that the *i*Tracker 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 *i*DNS 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.

3. SIMULATION RESULTS

We evaluate the performance of *i*CODE by using a discrete event-driven simulator. Our simulation environments are configured as follows. The Internet-like topology is generated using GT-ITM [16]. It consists of 1 transit domain with 5 routers, and 5 stub domains with 200 routers and 200 end hosts each. There are randomly distributed 1,000 contents with 1 GB 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 *i*CODE with those of CDN, P2P, and client-server model. The cache size of all C-routers is set to 10 GB. The caching policy for the *i*CODE is a simple round-robin among C-routers. For the CDN server deployment, we assume the ISP-operated CDN model [17] which can deploy the server at the best position in the ISP network in terms of hop count. For the peer selection in P2P, each peer first selects peers within the same ISP network similar to P4P [6] (and then adds peers in other ISP networks only when it is not able to find 10 peers in the same ISP network).

(1) User Experiences: We evaluate the user-experienced performance of each scheme in terms of the average hop counts traversed for the content delivery. As shown in Fig. 4(a), users in an ISP with *i*CODE will experience the most reduced transfer delay due to the shortest hop counts. Although our simulation assumes that a CDN provider has deployed its server in all ISPs (which is not so likely), *i*CODE can perform better than the ideal CDN service. The reason is that the cached content will be delivered from the



(a) User experiences (Hop counts) (b) Load balancing (Link stress) (c) Inter-ISP traffic reduction (Volume)

Fig. 4. Performance comparison of *i*CODE against CDN, P2P, and client-server model.

nearby C-routers inside the network; that is, the C-router is typically closer to the requesting host than the CDN server. The client-server model and P2P incur much larger number of hops because the considerable amount of contents are originated from the servers or the peers outside the ISP.

(2) Load Balancing and Traffic Engineering: *i*CODE 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 the link stress in Fig. 4(b). *i*CODE and CDN use the links more evenly than P2P and client-server model. Compared with CDN, *i*CODE distributes contents to multiple C-routers; thus *i*CODE shows lower link stress performance than CDN (56.7 % reduction for the top 20 links). P2P can download from multiple peers resulting in better performance than the client-server model.

(3) Inter-ISP Traffic Reduction: *i*CODE offers economic incentives to ISPs by reducing the inter-ISP traffic. As shown in Fig. 4(c), *i*CODE incurs the smallest volume of inter-ISP traffic except CDN [1] since *i*CODE services the content from the C-routers inside the ISP once the content has been retrieved from other ISPs. Even with the large number of content requests, *i*CODE reduces the inter-ISP traffic consistently. On the other hand, both P2P and client-server model incur very large amount of inter-ISP traffic. P2P results in relatively less inter-ISP traffic than client-server model since P2P can download part of the content from peers within the same ISP network, whereas client-server model is fully blind to the underlying network connectivity among ISPs.

4. DISCUSSIONS

This section discusses remaining technical and business issues of *i*CODE.

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

(2) Incremental Deployment: An ISP can deploy the *i*CODE independently 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 provide *i*CODE services to end users transparently, the *i*Proxy performs the swarming and the in-network cache lookup. Note that *i*Proxy can be implemented as a network entity if the end user's device is not affordable.

(3) *i*CODE Substantiation over Multiple ISPs: The network virtualization is not only considered as one of the key technologies of future networks in the ITU-T Study Group 13, but 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) [18]). The network virtualization is an extension of the system virtualization in end hosts to the network entities such as routers. An instance of the virtualized fractions of the resources in the network entities is called a *slice*. So, it is possible for an ISP to lease a slice from other ISPs [19]. 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.

(4) Swarming: Even though *i*CODE pushes the content files towards end hosts using in-network storage, there will be the transfer overhead on C-routers. To mitigate the

[1] Since our simulation assumes that all contents are stored in the CDN servers in advance before content retrievals, there is no inter-ISP traffic for CDN services. In reality, however, contents should be loaded at least one time across the ISP from the origin server to the CDN servers, which may occur comparable inter-ISP traffic with *i*CODE.

Table 1. Comparisons of innovative content delivery systems

	***i*CODE**	**DONA** [12]	**CCN** [13]	**PSIRP** [20]
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*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

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 file. As the number of C-routers containing the same file increases, we can lower the transfer overhead of each C-router. Also, each C-router does not need to hold the entire file; the individual C-routers need to keep only the chunks of the file (like downloading different pieces of the same file from multiple peers in BitTorrent).

(5) Cache Management: Obviously, the total storage space is limited even though the number of C-router is large and the storage cost is not costly. 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 to manage the storage cost efficiently. Also, the request pattern for a particular content file 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.

5. RELATED WORK

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

Data-Oriented Network Architecture (DONA) [12] proposes to use flat, self-certifying names instead of URLs, and 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 the perfect global availability of contents, the system works only when RHs are deployed over all the ISPs. On the contrary, users can benefit from *i*CODE even if only their ISP supports *i*CODE.

Content-centric networking (CCN) [13] 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 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 Data packets in CCN if individual CCN nodes decide to cache their copies whereas the copies of data are managed by the *i*Tracker of the ISP in *i*CODE.

Publish-Subscribe Internet Routing Paradigm (PSIRP) [20] is the architecture with a goal of building a publish/subscribe-based 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 to use Bloom filters, Merkle trees and special layer 2 hardwares, the multicast routing scalability will still be a concern. Also, the business model attractive to ISPs is not mentioned.

6. CONCLUSION

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

7. REFERENCES

[1] ipoque, "Internet study 2008/2009," http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009.

[2] K. Cho, J. Choi, D. D. Ko, T. Kwon, and Y. Choi, "Content-oriented networking as a future internet infrastructure: Concepts, strengths, and application scenarios," in *Proc. CFI*, 2008.

[3] "BitTorrent," http://www.bittorrent.com.

[4] "Akamai," http://www.akamai.com.

[5] T. Karagiannis, P. Rodriguez, and K. Papagiannaki, "Should internet service providers fear peer-assisted content distribution," in *Proc. ACM IMC*, 2005.

[6] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P: Provider portal for applications," in *Proc. ACM SIGCOMM*, 2008.

[7] W. Jiang, R. Zhang-Shen, J. Rexford, and M. Chiang, "Cooperative content distribution and traffic engineering in an ISP network," in *Proc. ACM SIGMETRICS*, 2009.

[8] "Apple time capsule," http://www.apple.com/timecapsule.

[9] D. Ko, K. Cho, M. Lee, H. Kim, T. T. Kwon, and Y. Choi, "Decentralized and autonomous content overlay networking (DACON) with WiFi access points," in *Proc. CFI*, 2010.

[10] "Cisco Application eXtension Platform (AXP)," http://www.cisco.com/en/US/products/ps9701.

[11] J. Kelly, W. Araujo, and K. Banerjee, "Rapid service creation using the JUNOS SDK," in *Proc. ACM PRESTO*, 2009.

[12] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *Proc. ACM SIGCOMM*, 2007.

[13] V. Jacobson, D. Smettersa, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content," in *Proc. ACM CoNEXT*, 2009.

[14] L. Dong, H. Liu, Y. Zhang, S. Paul, and D. Raychaudhuri, "On the cache-and-forward network architecture," in *Proc. IEEE ICC*, 2009.

[15] J. Choi, J. Han, E. Cho, H. Kim, T. Kwon, and Y. Choi, "Performance comparison of content-oriented networking alternatives: A tree versus a distributed hash table," in *Proc. IEEE LCN*, 2009.

[16] E. W. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. IEEE INFOCOM*, 1996.

[17] N. Kamiyama, T. Mori, S. Harada R. Kawahara, and H. Hasegawa, "ISP-Operated CDN," in *Proc. IEEE INFOCOM Workshop*, 2009.

[18] "GENI," http://www.geni.net.

[19] G. Schaffrath, C.Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. Greenhalgh, M. Kind, O. Maennel, and L. Mathy, "Network virtualization architecture: proposal and initial prototype," in *Proc. ACM VISA*, 2009.

[20] A. Zahemszky, A. Csaszar, P. Nikander, and C. Esteve, "Exploring the pub/sub routing&forwarding space," in *Proc. FutureNet*, 2009.