SCAN: Scalable Content Routing for Content-Aware Networking

Munyoung Lee, Kideok Cho, Kunwoo Park, Ted “Taekyoung” Kwon, and Yanghee Choi
School of Computer Science and Engineering
Seoul National University, Seoul, Korea
Email: {mylee, kdcho, kwpark}@mmlab.snu.ac.kr and {tkkwon, yhchoi}@snu.ac.kr

***Abstract*—Since Internet routers are not aware of the contents being forwarded, the same content file is often delivered multiple times inefficiently. Similarly, users cannot exploit a nearby copy of the content of interest unless the content file is serviced by costly content delivery networks. Prior studies on the content-aware routing for efficient content delivery suffer from the scalability problem due to a large number of contents. We propose a scalable content routing, dubbed SCAN, which can exploit nearby and multiple content copies for the efficient delivery. SCAN exchanges the information of the cached contents using Bloom filter. Compared with IP routing, SCAN can offer reduced delivery latency, reduced traffic volume, and load balancing among links.**

***Index Terms*—Scalable content routing, Bloom filter, Parallel transmission**

I. Introduction

A recent measurement study shows that most of the Internet traffic is governed by content retrieval applications such as P2P file sharing or VoD services [1]. That is, users only care about the content itself instead of the source of the content. Since the current Internet routers are not aware of the content being delivered, inefficient content deliveries take place frequently. For instance, a user may retrieve a content file from a distant server, even if there is a nearby copy. Also, content retrievals do not exploit multiple copies of the same content in the networks unless we explicitly leverage peer-to-peer solutions like BitTorrent [2].

To solve the above inefficiency, there have been clean-slate studies on content-oriented networking [3], [4], [5]. They exploit in-network storages attached to routers to cache the contents and support content-aware routing to find a close copy of the requested content at the networking level. Also, with multiple copies, parallel transmission of a content file can be supported to enhance the download speed [4], [5]. To support content-aware routing, each content should have a content identifier (CID), which serves as an index in content routing tables (CRTs) in routers. Since the number of contents in the network can be huge, and CIDs are usually flat ([3]) or difficult to aggregate ([4]), there is a scalability issue.

One way to mitigate the routing scalability issue is to store the information of the subset of the contents, not the entire contents. However, as not all the contents are located, the content reachability problem will happen. We suggest to use the current IP routing as a fallback solution.

In this paper, we propose a scalable routing scheme to realize content-aware networking, dubbed SCAN, which addresses the inefficiency of IP routing. SCAN assumes that a content router (or C-router, for short) can cache contents in its attached storage modules as similar to [3], [4], [5]. In SCAN, C-routers perform both content routing and IP routing; thus, C-routers maintain CRTs as well as the IP routing tables. To guarantee the reachability, the content request will, as a fallback solution, follow the IP routing path to the original server as the current Internet. However, if there is a matched entry in the CRT of any C-router along the path (this process of content routing is called *“scanning”*), the C-router will send *scanning requests* to the next hop (C-router) in the CRT. With the scanning process, close and possibly multiple copies of the requested content can be downloaded for the efficient content delivery.

It is anticipated that the number of contents in a C-router may increase over time due to ever decreasing storage cost. Therefore, even local exchanges of content information among C-routers within a small hop count may cause the routing scalability problem. To alleviate this potential scalability problem, SCAN adopts Bloom filter [6] to compress the CRT information. The remainder of this paper is organized as follows. In Section II, we explain how the CRT is maintained. In Section III, we describe the SCAN routing operations. Section IV presents the simulation results, and Section V compares SCAN with the other proposals on content-aware routing. Section VI concludes this paper.

II. CRT Maintenance

A. Assumptions

We assume that C-routers in the network can cache the copies of the (mostly popular) contents in their storage modules. C-routers perform content routing called *scanning* by sending a *scanning request* in addition to a content request (e.g. an HTTP Get request) in the traditional IP routing. A content request is generated by a host, while a scanning request is generated by a C-router that relays the content request. For scanning, a C-router maintains a local content table (LCT) and a content routing table (CRT), each of which consists of the information of the cached contents at the C-router itself and at neighbor C-routers, respectively. When a content request arrives at a C-router, it performs scanning as follows. It first looks up the LCT to find the cached copy of the requested content. If it fails, it looks up the CRT to find the routing information of the content. To locate the requested content, SCAN uses a CID[1] in content routing and an IP address in IP routing. To extract the CID from the content request, we assume that the deep packet inspection (DPI) functionality is supported by C-routers [7].

Not all routers have to be C-routers for the SCAN routing operations. C-routers can coexist with the legacy IP routers. C-routers exchange CRTs only with other C-routers in an overlay manner. Also, C-routers perform the scanning operations only when they can afford to (e.g. traffic load is not high at the moment). Note that even when all C-routers do not perform the scanning, SCAN can deliver the requested contents to users by IP routing.

B. CRT Compression

The number of contents a C-router can cache may be huge. To handle the scalability issue of CRTs, SCAN compresses the stored content information in a C-router by using Bloom filter (BF) [6]. The BF is a space-efficient random data structure which supports the membership queries for a set of elements. A typical BF is embodied as a series of m-bits. When an element α is to be added to a BF, say S, α is hashed by k independent hash functions (e.g., MD5 or SHA-1). Each hash function returns an integer between 0 and $m-1$. Among m bits of S, the bits corresponding to the results of k hash functions are set to 1. One can decide whether S contains α by checking whether the corresponding k bits are set to 1 or not. However, there is also possibility of *false positive*. To sufficiently lower the possibility of false positive decisions, the size of BF (m) should be large enough compared to the number of stored elements (n) and the number of hash functions (k) (i.e., $k \times n \ll m$).

The key advantage of using BFs is that a BF can condense the membership information into the limited size and check the inclusion of an element. In SCAN, C-routers maintain one BF for each interface in a CRT. BF compresses the information of the contents that are advertised through an interface into the BF of the interface. If k hashed values of the requested content's CID are matched to the BF of a particular interface, the C-router can conclude that the requested content can be located through the interface with high probability. (Note that there might be false positive.)

C. CRT Exchange and Information Decaying

For the scanning operation, C-routers exchange their CRTs periodically. For example, when C-routers C1 and C2 exchange their CRTs, C1 merges BFs of all the interfaces except the interface associated with C2 and sends the merged BF to C2. Note that the contents in C1's LCT will also be inserted into the merged BF. Then, C2 adds the received information to the BF of the interface connected to C1. Similarly, C2 sends its own CRT to C1, and C1 will reflect it into the corresponding BF. If C1 and C2 are not direct neighbors, they can exchange the CRTs over other C-routers overlay. Note that a loop can be formed for the same CID. When a C-router receives a scanning request for the same content, it simply discards the request.

[1]We assume that every content is specified by a CID (e.g., an HTTP URI).

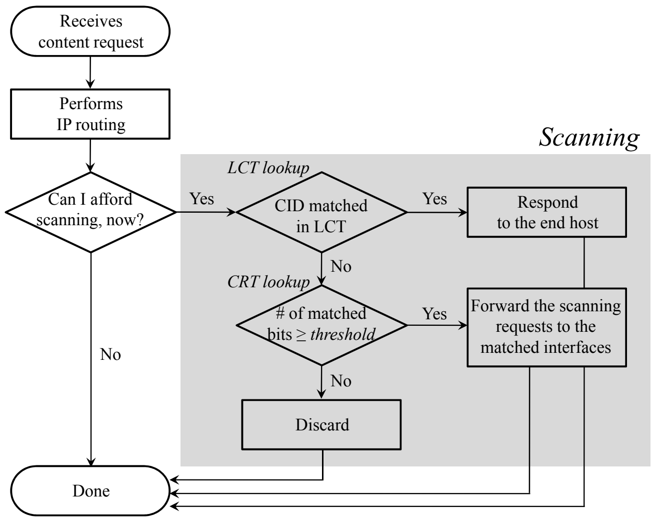


Fig. 1. The forwarding decision at a C-router is described.

If CRTs, represented by BFs, are distributed over multiple hops, it is possible that all the bits of the BF are set to 1 as the bits for the large number of contents will be accumulated. It will raise false positives significantly, and the BF may not be effective. For this, each C-router will decay all the bits (set to 1) of a BF probabilistically before exchanging CRTs as similar to [8]. For example, if the decaying probability is 0.5, half of the bits 1 will be changed to bits 0. In this way, as the traversed hop count of a C-router's CRT increases, the information of the cached contents in the C-router (i.e., hashed bits) will decrease.

Due to the content information decay, it is possible that there is no match for the requested content in the BFs of a C-router even if the content file is cached in a C-router two or more hops away. To handle this situation, the scanning can be performed based on the threshold rather than the perfect match. If the number of matched bits of the scanning request to any of the BFs is over the threshold, the scanning request may still be forwarded to the matched interface. The threshold will be adjusted depending on the hop count from the soliciting host and the hop count from the C-router that generates the scanning request. The scanning is stopped when the number of matched bits is smaller than the threshold. The forwarding mechanism among C-routers is detailed in Section III-A.

III. SCAN Routing

A. Forwarding Decision

When a content request arrives at a C-router, it makes the forwarding decision as described in Fig. 1. The C-router first performs the traditional IP routing: it checks the destination IP address and forwards the request towards the destination. After performing the IP routing (and copying the request message), the C-router should decide whether it will perform the scanning or not. Note that only when the C-router can

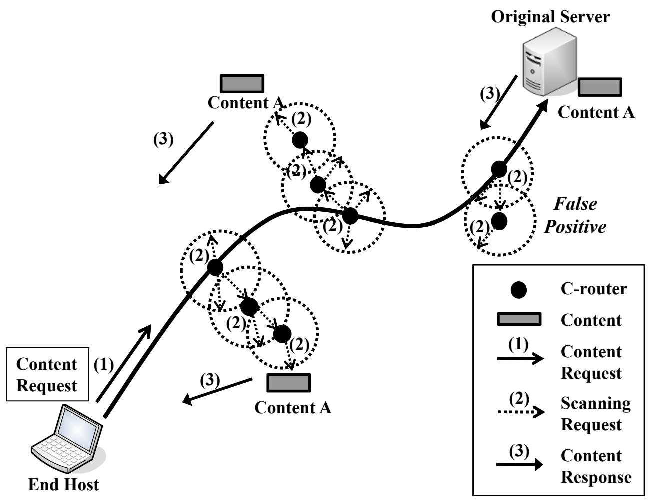


Fig. 2. The SCAN routing operation is illustrated.

afford to (e.g. light traffic load), the scanning operation will be performed since it is mandatory for the C-router to do IP packet forwarding.

During the scanning operation, its LCT is first looked up to check whether the requested content is cached. If cached, the C-router will respond to the soliciting host to notify the existence of the cached copy. If not, it will check the BFs associated with every outgoing interface except the one over which the content request comes in. If there are matched interfaces (i.e. the number of matched bits is equal to or greater than the threshold), the scanning request is generated and sent to the matched interfaces.

Since the BF information is decayed during the CRT exchange, the C-router will send the scanning request if the number of matched bits of a BF is over the predefined threshold. The threshold is defined as $\delta \times (level + depth)$, where δ is the unit value of threshold, the $level$ is the hop count of regular routers (including C-routers) that the original content request has traversed[2], and the $depth$ is the hop count of C-routers that the scanning request has traversed. The rationale behind the $level$ is to perform the scanning more thoroughly near the end host. In addition to the $level$ information, the scanning request also leverages the $depth$ information to decide the threshold of matching/forwarding. When a C-router first sends a scanning request, $depth$ is initialized to 0. The $depth$ value will be incremented by 1 as the scanning request traverses at each C-router. Note that when a C-router receives the scanning request, it does not perform IP routing since its purpose is to find an additional copy.

B. Illustration of SCAN Operation

This section explains the operation of SCAN as illustrated in Fig. 2. (1) An end host sends a content request to the original server of the content. (2) When a C-router, which is along the path to the original server, receives the content request, the C-router first forwards the content request message towards the original server like a regular IP router. After forwarding the content request, the C-router *scans* its LCT and CRT. The existence of the CID in the LCT indicates that the C-router currently caches the content. For the BFs in the CRT, scanning requests of the content are sent if the number of matched bits of the content is equal to or higher than the threshold. The scanning requests may eventually reach multiple C-routers caching the same requested content. In this way, SCAN may find multiple copies of the requested content.

(3) When a C-router caching the content receives the scanning request, it sends a response message to the end host (who issued the content request) to notify the existence of the requested content. On receiving the content request, the original server also sends a response message to the end host. After receiving the response messages, the end host will decide from which C-routers and the original server it will download the content. Although the detailed download procedure from multiple sources in parallel is out of the scope, one feasible mechanism can be described as follows.

The end host may select a set of close C-routers (and possibly the original server as well) based on selection metrics such as hop count or latency [9]. Then the end host establishes connections to the chosen C-routers. For efficient content retrievals, the end host can utilize HTTP byte-range requests for partitioned transmissions [10]. By leveraging the parallel transmissions, the end host downloads the content from multiple C-routers simultaneously.

In the above illustration, we assume that the original server and the end host can support SCAN operations: the original server sends a response, and the end host interprets the response messages. Also, the end host is assumed to download the content from the multiple sources. Note that the first C-router along the path to the original server can undertake the parallel download on behalf of the end host to retain the backward-compatibility with the legacy hosts. As for backward compatibility of the legacy original servers, the first C-router may send a TCP FIN packet to terminate the connection if the content is available from other C-routers.

C. False Positive and Control Overhead

The advantage of using BFs is that a BF can effectively store a large number of CIDs in a fixed size bit string. Also, one can easily verify the inclusion of a CID in a BF. However, the false positive may occur due to the nature of BFs even when a sufficiently large BF is used. False positive decisions in SCAN cause scanning requests to be propagated to irrelevant C-routers as shown in Fig. 2. SCAN mitigates the negative effects of false positive decisions by separating default IP routing and scanning. False positive decisions does not affect the correct content retrieval by leveraging IP routing. Since a content request is always delivered to the original server of the content, SCAN guarantees the reachability of the content regardless of the result of scanning operation.

[2]We assume a C-router can infer the number of hops using the initial TTL values as similar to [9]

TABLE I
AVERAGE HOP COUNT FOR THE CONTENT DELIVERY

	SCAN	**SCAN-FULL**	**SCAN w/o BF**	**IP w/ Caching**	**IP Routing**
Average Number of Hops	3.2	3.2	4.5	4.7	6.8

Additional scanning requests are used to find multiple copies of the requested content. These scanning requests and exchanges of CRTs incur slightly more signaling traffic compared to the traditional IP routing. However, SCAN can reduce the total volume of the network traffic per link significantly by utilizing the cached contents.

IV. Simulation Results

To evaluate the performance of SCAN, we conducted simulations using a discrete event-driven simulator. Our simulation environments are configured as follows. GT-ITM [11] is used to generate an Internet-like topology which consists of 1 transit domain, 5 stub domains, 105 C-routers (collocated with the original servers), and 1,000 end hosts. 10,000 contents are randomly distributed at the original servers, hence each original server stores on average 100 content files. Additionally, the 1,000 popular contents (i.e., top 10%) are randomly replicated (10 copies on average) and distributed at C-routers to emulate the skewed distribution of their copies. We assume there is no cache replacement or migration among C-routers. Every content file is 1 GB size and the unit value of the threshold δ is set to 0.1. The probability distribution of content requests follows the Zipf distribution with parameter 1.0.

We compare SCAN routing with IP routing, IP routing with caching, SCAN without content information compression (denoted by SCAN w/o BF), and SCAN with the full content information (denoted by SCAN-FULL). In IP routing with caching, a router looks up the cache to check whether the requested content is cached or not. If cached, IP routing is stopped and the content will be delivered to the end host. SCAN w/o BF performs the scanning operation but a CRT is maintained without BF (i.e., no compression); thus the amount of content information is limited by memory constraints. Here, the CRT maintains the information of the cached contents from one hop neighbor C-routers only. SCAN-FULL has no memory constraints, and thus keeps the information of all the contents advertised from neighbor C-routers. Hence, there is no false positive. SCAN-FULL is not realistic but used as a reference.

In our simulations, the storage size of every C-router is set to 100 GB for content caching. Each BF is 3,500 bit long with 14 hash functions. Also, the half of the hashed BF diminishes for information decaying. We assume the size of a single content entry without compression is 256 bits. To evaluate the effect of information compression, we set the same memory size for the BFs of a C-router in SCAN and the CRT of a C-router in SCAN w/o BF. Note that the content information can be lost in case of SCAN w/o BF due to the limited memory for the CRT.

A. Content Routing Performance

We evaluate the routing performance of each scheme in terms of the average hop count traversed for the content delivery. As shown in Table I, SCAN-FULL and SCAN outperform other schemes due to the wide-range search of the requested content. Note that even with information decaying, SCAN achieves similar to SCAN-FULL. Compared to IP Routing, SCAN achieves 52.3% less average hop count. IP routing with caching can reduce the average hop count substantially compared to IP routing. However, only one-hop scanning near the IP routing path does not significantly enhance the performance considering SCAN w/o BF with IP routing w/ caching.

SCAN (and SCAN-FULL) reduces a significant amount of network traffic (per link) as shown in Fig. 3(a) due the small number of hop counts required for the content delivery. IP routing w/ caching and SCAN w/o BF often utilize the cached content located at nearby C-routers, and thus reduce the total volume of network traffic compared with IP routing (30.7% and 35.7%, respectively).

B. Load Balancing

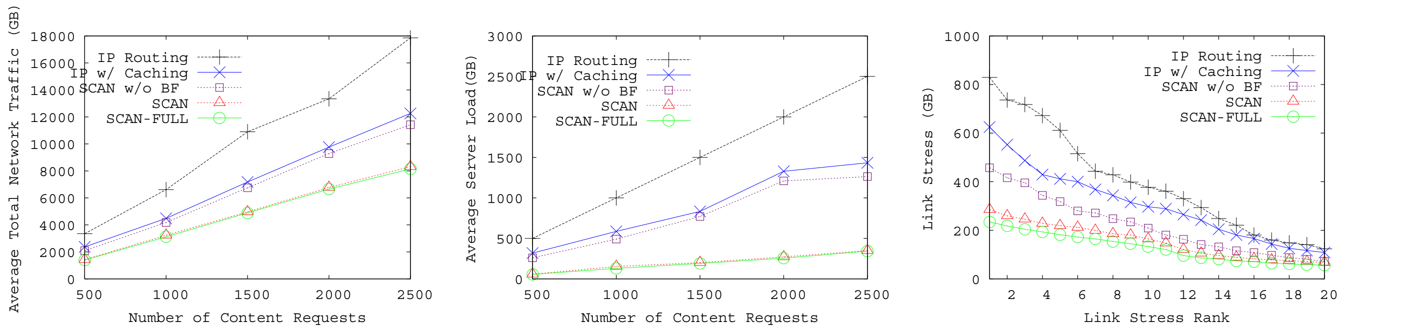
In terms of load balancing, SCAN (except SCAN-FULL) achieves the best performance. Fig. 3(b) shows the average load of the original server as the total number of content requests in a simulation run increases. The original server load is reduced since the requested content is retrieved from the C-routers. Compared with IP routing, SCAN achieves about 86.4% original server load reduction, while SCAN w/o BF and IP w/ caching reduce the burden of original server by 47.3% and 39.6%, respectively. For instance, when there are 2,500 content requests, only about 350 requests are served by the original server in SCAN.

To see another effect of load balancing, we also measure the link stress for individual links, which is defined as the amount of traffic volume passed over a particular link, and plot the top 20 stressed links in the descending order. Fig. 3(c) shows that SCAN (except SCAN-FULL) utilizes the links more evenly than other routing schemes. By utilizing the widely spread multiple copies of the content files, SCAN can achieve better low stress performance which is comparable to that of SCAN-FULL. Since the SCAN w/o BF can search one-hop neighbors along the path in IP routing, it can achieve better load balancing on the links than the IP routing w/ caching.

V. Related Work

To solve the inefficiency in content retrievals in the Internet, there have been efforts to redesign the Internet in a clean-slate manner. Data-Oriented Network Architecture (DONA) [3] and Content-centric networking[3](CCN) [4] redesign the Internet in a content-oriented fashion by proposing new naming and content-aware routing mechanisms. DONA uses flat, self-certifying names instead of URLs, and the content requests are routed over the predefined tree hierarchy of network entities

[3]The name of CCN is changed to Named Data Networking (NDN) [12]



(a) Network traffic reduction (b) Original server load reduction (c) Load balancing among links

Fig. 3. Performance comparison of SCAN against IP routing, IP routing with caching, SCAN without BF, and SCAN-FULL.

(called Resolution Handlers, or RHs) to find the closest copy. Since the root RH should maintain all metadata for all the contents, the scalability problem is a major concern of DONA. Also, DONA does not exploit the multiple copies of the content in the network to enhance the download speed.

CCN extends the URI to name the contents in a hierarchical manner to aggregate the names for the content routing. Since CCN requests a content in the unit of chunk, CCN can support parallel transmission of contents by requesting content chunks simultaneously. However, to guarantee the reachability of content requests, the CID information of all contents should be maintained in the routers, which results in the scalability problem. The problem is even worse when many contents, whose names are not aggregatable, are diversely cached in the routers.

There have been proposals to utilize BFs for the content routing. Scalable Query Routing (SQR) [8] adopts exponentially decaying BFs to route the queries in unstructured P2P networks. Queries will be forwarded to the neighbor peer which has the highest number of matched bits. A content query initially follows a random walk until it meets a peer which has the routing information of the requested content. In SCAN, the content request will follow the IP routing when there is no matched entry in the CRT, which prevents an unnecessary random walk.

In [13], a probabilistic routing algorithm is proposed to enhance the performance of IP routing. Attenuated Bloom filters help to find the nearby copy with the high probability. The traditional IP routing is enhanced through a hybrid approach: first try the probabilistic algorithm, and then follow the deterministic IP routing if needed. In contrast, SCAN utilizes multiple copies in the networks by gradually scanning around the path to the original server.

VI. Conclusions

In this paper, we propose a scalable content routing scheme, dubbed SCAN, for content-aware networking. SCAN achieves both reachability and efficiency by using both IP routing and content routing called scanning. When there are cached copies of a requested content file in the vicinity, scanning is performed to find nearby and/or multiple copies of the requested content for the efficient delivery. Also, to handle the scalability issue of the content routing table, SCAN exchanges the content information that is compressed using Bloom filter. Simulation results show that SCAN delivers the contents to the users faster than other schemes since it can locate one or more closer copies, if any, of the requested content. Also, SCAN reduces the total volume of network traffic and hence provides better load balancing among the links. In future, we will investigate the effects of various caching policies and C-router selection schemes to optimize the performance of SCAN.

Acknowledgment

This work was supported by NAP of Korea Research Council of Fundamental Science & Technology. This publication is based on work performed in the framework of the Project COAST-ICT-248036, which is partially supported by the European Community. The ICT at Seoul National University provided research facilities for this study.

References

[1] ipoque. Internet Study 2008/2009. http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009.
[2] B. Cohen, "Incentives build robustness in BitTorrent," *Proc. Workshop on Economics of Peer-to-Peer Systems*, May 2003.
[3] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *Proc. ACM SIGCOMM*, Aug. 2007.
[4] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content," *Proc. ACM CoNEXT*, Dec. 2009.
[5] K. Cho, H. Jung, M. Lee, D. Ko, T. T. Kwon, and Y. Choi, "How can an ISP merge with a CDN?," *Proc. of ITU Kaleidoscope*, Dec. 2010.
[6] A. Broder and M. Mitzenmacher, "Network Apllications of Bloom Filters: A Survey," *Internet Math.*, vol. 1, no. 4, pp. 485-509, 2005.
[7] N. Hua, H. Song, and T. V. Lakshman, "Variable-stride multi-pattern matching for scalable deep packet inspection," *Proc. IEEE INFOCOM*, 2009.
[8] A. Kumar, J. Xu, and E.W. Zegura, "Efficient and scalable query routing for unstructured peer-to-peer networks," *Proc. IEEE INFOCOM*, 2005.
[9] S. Ren, E. Tan, T. Luo, S. Chen, L. Guo, and X. Zhang, "TopBT: A Topology-Aware and Infrastructure-Independent BitTorrent Client," *Proc. IEEE INFOCOM*, Mar. 2010.
[10] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol. HTTP/1.1. RFC 2616, June. 1999.
[11] E. W. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," *Proc. IEEE INFOCOM*, 1996.
[12] Named Data Networking (NDN), http://www.named-data.net.
[13] S. C. Rhea and J. Kubiatowicz, "Probabilistic Location and Routing," *Proc. IEEE INFOCOM*, 2002.