NDN, Technical Report NDN-0004, 2013. http://named-data.net/techreports.html
Revision 1: July 18, 2013

Scaling NDN Routing: Old Tale, New Design

Alexander Afanasyev
UCLA
alexander.afanasyev@ucla.edu

Cheng Yi
University of Arizona
yic@cs.arizona.edu

Lan Wang
University of Memphis
lanwang@memphis.edu

Beichuan Zhang
University of Arizona
bzhang@arizona.edu

Lixia Zhang
UCLA
lixia@cs.ucla.edu

ABSTRACT

The Named Data Networking (NDN) is a newly proposed design for future Internet architecture, however one commonly raised concern about NDN's feasibility is its routing scalability. In this paper we sketch out a design that makes the NDN routing scale as well as today's IP Internet, potentially significantly better. We apply the well understood concept of *map-n-encap* to the context of NDN so that only ISP name prefixes need to appear in the global routing table. More specifically, our design securely maps each application name to one or more ISP names where the named data resides, and encapsulates the ISP names in NDN's Interest packets to hint the forwarding system of the whereabouts of the requested Data.

1. INTRODUCTION

As Internet applications are increasingly data-centric, a number of new network architecture designs [1,2,3,4] have proposed to use name-based routing. In name-based routing, packets carry data names or data identifiers instead of host addresses, and routers forward packets based on their names or identifiers. Named Data Networking (NDN) [4] is one prominent example. By naming data explicitly and binding the name and data by a cryptographic signature, NDN provides a number of benefits including data security, in-network caching, and better alignment between applications and data delivery. However, one frequently raised concern is NDN's routing scalability [5,6]. If the number of data names is unbounded, how can NDN routing table scale?

We observe that NDN's routing scalability issue is not new; one could ask a similar question about IP. Although IP's address space is finite (2^{32} or 2^{128}), it is larger than any of today's router can hold. IP solves this problem by address aggregation. At the edge of the Internet, hosts and small networks get addresses from their access providers. Since addresses from the same provider can be aggregated into prefixes (i.e., these are provider-aggregatable or PA addresses), routing tables only need to store prefixes instead of individual IP addresses. However, over the years there has been an increasing demand for provider-independent (PI) address prefixes. Such PI prefixes cannot be aggregated with provider prefixes and must be announced separately, leading to increased routing table size [7].

Map-n-Encap [8] was proposed long ago to scale IP routing in face of a large number of PI addresses. The basic idea is to use a mapping system to map a PI destination address to a PA address, and then tunnel the packet (using IP-in-IP encapsulation) to the destination via the PA address. In this way, the core maintains only a limited number of PA address prefixes. This basic approach has led to a number of specific designs including 8+8 [9], LISP [10], ILNP [11], and APT [12], to name a few. However, due to the difficulties in retrofitting new solutions into the operational Internet, so far none of them has been deployed.

NDN is data-centric, and every application names its data in a provider-independent way.[1] Thus in NDN, the problem of PI name prefixes is much worse. We propose to apply the same Map-n-Encap idea to scale NDN routing.[2] Given an Interest packet carrying a PI application name, we first look up the mapping system to find the corresponding PA information, and then forward the packet based on such information. The resulting overall picture is that (a) aggregation happens at the edge of the Internet to aggregate application data names into ISP name prefixes, and (b) encapsulation serves as a packet forwarding mechanism. This architecture essentially moves the scalability issue from the routing to the mapping system, and today's DNS can easily handle this scaling challenge. As the mapping happens at the edge of the network, it will have no impact on the network core.

Although we can see great parallelism between NDN and IP regarding the routing scalability problem and solution at a conceptual level, realizing the Map-n-Encap mechanisms in NDN remains a research problem. NDN

[1] Otherwise, applications have to discover local providers before naming their data, and have to rename whenever they move or change service providers.
[2] Since NDN is a new network architecture, we do not have IP's retrofitting problem.

is fundamentally different from IP. It has different packet formats, semantics, and operations, which prevent us from simply mirroring IP's mechanism in NDN. Instead of encapsulating packets, we propose to add a *forwarding alias* to the Interest packet. Application names are used for caching and signature verification, while the forwarding alias, which reflects the service provider of the content producer, serves as a hint to routers about where the packet may be forwarded. At the same time, mapping from PI application names to PA provider name prefixes will be performed using NDN mechanisms through an NDN-based DNS-like system.

2. DESIGN

2.1 Overview

In NDN, applications name their content. Consumers send *Interest* packets with the names of the content that they want to retrieve, and producers reply with *Data* packets carrying the same or more specific names. Routers do routing, forwarding, and caching all based on names. We assume routers run a routing protocol (e.g., extended OSPF or BGP) to build name-based routing tables, but do not assume any specific protocol in this paper.

Given the finite size of router memory, routing table can only hold a subset of all name prefixes. With Map-n-Encap, only *ISP name prefixes* are kept in the default-free zone (DFZ) routing tables. ISPs are networks that provide transit service for their customers or peers. End-sites, who are either sinks or origins of traffic, do not have their prefixes in the DFZ routing table.[3] According to our analysis of BGP data in 2006 [13], the number of ISP's IP prefixes is 22K out of total 209K prefixes in the global routing table. Assuming one name prefix for each IP prefix for ISPs, the name-based routing table would only have tens of thousands of entries; while the vast majority of NDN names used by end-sites will not show up in the global routing table. Another analysis of BGP data shows also that the growth rate of ISP ASes is only 20% of total AS growth rate [14]. Furthermore, it is known that one of the factors that drive IP table growth is prefix splitting [15], a practice used for traffic engineering or limited hijack defense. But in NDN, by maintaining forwarding states and observing Interest/Data two-way traffic, routers can detect packet delivery problems, discover paths with better delivery performance and switch to them, which greatly reduces the needs for prefix splitting. Therefore, compared with today's BGP table, the name-based routing table after Map-n-Encap should be much smaller and grow much slower.

In order to deliver Interests of a particular application name (e.g., `/reddit.com/r/scholar`[4]), routers need to know the ISPs (e.g., AS-2828 XO Communications and AS-4436 nLayer Communications) that serve the producer (i.e., reddit.com). This is done by looking up the application name in a mapping system similar to DNS. Once such information is obtained, Interests will be sent towards the ISPs, which will then route packets towards the content producers via their internal networks. The rest of this section will discuss the forwarding mechanism first, followed by the discussion about the mapping system.

2.2 How To "Encapsulate"

Assuming mapping has been done (see Section 2.3), in this section we discuss two different ways to carry the ISP information in Interest packets: *name concatenation* and *forwarding alias*.

2.2.1 Name Concatenation

The most straightforward way to implement the "encapsulation" step is *name concatenation*. An ISP prefix is prepended to the application name, forming a provider-dependent name (Figure 1), which is used in both Interest and Data packets. The Interest packet has the same format described in original CCN design [16]. The Data packet has the same format too, but is an encapsulated one. The inner packet contains the original application name, content and signature that binds the two. The outer packet contains the provider-dependent name, the inner packet as its content payload, and its own signature (Figure 2a). Data encapsulation can be done by either the producer or its ISP. If the former, the producer needs to publish the same content under each ISP prefix, and needs to change the names and signatures of all its contents when it changes ISPs. In the latter approach, ISPs encapsulate data packets on the fly, making content producers topology-agnostic with the price of increased router load. For example, the border router at XO Communications can encapsulate incoming Data of `/reddit.com/r/scholar` under the name `/xo/reddit.com/r/scholar` and forward them back to the consumer.

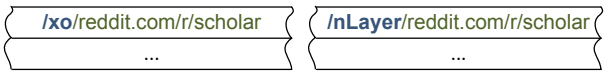

Figure 1: Interests in concatenation approach

Although name concatenation achieves the goal of forwarding packets under Map-n-Encap, it fundamentally changes the names used for fetching data, and this

[3]In practice, when the distinction between ISP and end-sites are blurry, additional economic and/or operational criterion are applied to determine whether a prefix should go into DFZ table.

[4]In our design `/reddit.com` is represented as `/com/reddit`, but for simplicity we will be using the standard notation.

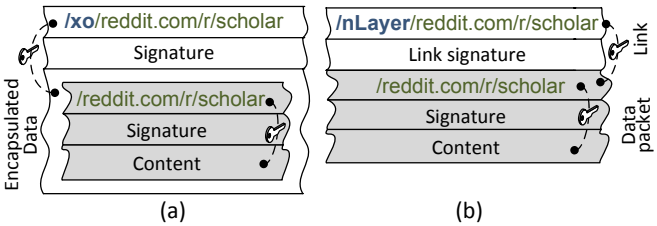

Figure 2: Data in concatenation approach

change leads to a couple of undesirable problems. First, there is extra signing overhead because the original content is signed over twice, i.e., first by the producer's key (inner packet) and then by the ISP's key (outer packet). The extra processing load can lead to serious scalability concerns, since routers need to sign packets at line-speed. This overhead can be reduced by signing only the "link" between the application name and the ISP name, e.g., in Figure 2b, the outer signature binds the two names, not the content. However, this optimization does not fully eliminate the signing overhead.

Second, and probably more critical problem is that name concatenation makes names topology-dependent, which reduces the caching efficiency and content availability. For example, Interests directed to one ISP (e.g., `/xo/reddit.com/r/scholar`) cannot be satisfied by router cache, if the cached Data is under another ISP prefix (e.g., `/nLayer/reddit.com/r/scholar`). In particular, if a mobile host is serving contents while moving, it has to keep track of the ISPs it connects to and append different prefixes to its content names. Consumers will also need to keep track of the producer's ISPs. The Interest with wrong ISP prefixes will not be able to retrieve data or benefit from caching. In another scenario, if the producer is multihomed, and one of its ISPs has failed or the connection to content producer is down, Interests directed to this ISP will be dropped. Routers will not be able to forward them to another ISP that serves the content producer. Therefore, topology-dependent names compromise NDN's principle of being data-centric.

These problems of the name concatenation prompt us to search for better ways to include ISP prefix in Interests/Data.

2.2.2 *Forwarding alias*

In this approach, the ISP prefix is carried in a new field, called *forwarding alias*, in the Interest packet. Application names in Interest and Data packets are intact and are visible to routers, allowing Interests to be satisfied by cached Data regardless of from ISP the Data was initially retrieved. The forwarding alias is used in routing table lookup when Interest packets are being processed by routers. It is a hint provided by the end-host to routers about where to forward the Interest. Figure 3 shows a conceptual view for Interest packets. Data packets have the exactly same format as described in original CCN design [16].

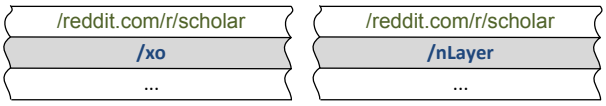

Figure 3: Interests in alias approach

Routers in the DFZ have all ISP prefixes (e.g., `/xo` and `/nlayer`) in their routing tables. All ISP routers also have their customer end-sites prefixes in the routing tables too (e.g., XO and nLayer Communications networks will have an entry for `/reddit.com`).

Pseudocode 1 lists conceptual steps of Interest processing at routers. At a high-level, the algorithm can be summarized by the following four steps. Note that only steps (3) and (4)—lines 10–11 of the pseudocode—represent necessary modifications to NDN's Interest forwarding mechanisms. The rest is the same as the original CCN design.

1. Look up the application name in: cache (line 4), pending interest table (PIT) (line 6), and routing table (line 8).

2. If any lookup of step (1) returned a positive result, proceed with the standard Interest processing. That is, Interest will either be satisfied with previously cached Data (line 5), recorded in the existing PIT entry (line 7), or forwarded according to the forwarding strategy (line 9).

3. *If none of the lookups of step (1) returned a positive result, look up the alias in the routing table (line 10).*

4. *If the lookup of step (3) returned a positive match, forward the Interest accordingly (line 11).*

Pseudocode 1 Interest Processing

```
1:  function Process(Interest)
2:      Name ← Interest.Name
3:      Alias ← Interest.Alias
4:      if Data ← Cache.Find(Name) then
5:          Return(Data)
6:      else if PitEntry ← PIT.Find(Name) then
7:          Record(PitEntry, Interest)
8:      else if FibEntry ← FIB.Find(Name) then
9:          Forward(FibEntry, Interest)
10:     else if FibEntry ← FIB.Find(Alias) then
11:         Forward(FibEntry, Interest)
12:     else
13:         Drop Interest (Interest cannot be satisfied)
14:     end if
15: end function
```

The beauty of the forwarding alias design is that it does not require any additional considerations or changes for NDN data packets or Data packet forwarding. Because Interests create a "breadcrumb" path using application names, application Data packets will easily follow this path back to consumer(s). Also, there is also no additional changes for the producers, other than providing input to the mapping system, indicating which ISPs provide service to the producer.

Carrying forwarding alias is not a step-back from data-centricity of NDN. The alias is nothing more but a hint of where the requested content may reside. For example, if the client expresses an Interest for the Data, which are available locally (e.g., the producer resides inside the same ISP and there are corresponding FIB entries on ISP's routers), then this Interest will be forwarded to the local producer, independent from whether it carries a forwarding alias or not. Also, Interests will pull Data back from the very first router that cached the desired Data, independent from how many ISPs the data producer may connect to.

The binding between the application name and the alias is not signed. That is, the forwarding alias can be easily modified in transit, seamlessly redirecting Interests out of the requested way. This can be used for good, as well as for bad. For example, routers may modify aliases to enforce traffic engineering agreements or to select better paths to producers. At the same time, a misconfigured or compromised router can request all passing-through Interests go to unintended destinations, in attempt to black-hole or eavesdrop traffic, or to DDoS another ISP's network. However, given NDN Interest packets are not signed in general, routers can already modify data names without being detected and perform the same "evil" activities. Thus, introducing the forwarding alias does not introduce any new vulnerabilities into the system.

The forwarding alias approach is an effective mechanism to deliver packets under Map-n-Encap without impacting the major advantages of NDN communication paradigm, including in-network caching. We believe that, for the time being, forwarding alias represents the best tradeoffs among the options to scale unlimited application name space of NDN with the use of conventional routing.

2.3 How To Map

In this section we attempt to answer questions related to the secure mapping between application and ISP names.

2.3.1 *Secure mapping service*

The application to ISP name mapping service should have the following characteristics: (a) mapping information should be securely stored, updated, and queried; (b) mapping system should allow an unlimited namespace for key values (i.e., for application names); (c) information should be stored and managed in a distributed way; (d) system should support a virtually unlimited volume of mapping queries. DNS-style database is an ideal candidate that satisfies all the above-mentioned requirements.

Native NDN mapping system.

The following design sketch demonstrates a way to build a DNS-like database natively in NDN. The sketch assumes that we are using the forwarding alias approach to encapsulate ISP names in Interests, but it can be easily modified to the name concatenation approach as well.

First of all, there should be a dedicated `/root` prefix, which is globally visible and routed to multiple different locations of "NDN-root" servers (which are similar to DNS-root servers). Second, all queries to the mapping system should be performed in an iterative and incremental fashion by expressing Interests for the desired prefix. For example, when trying to obtain routing alias for `/com/reddit/r/scholar`, query (Interest) for `/com` should be made (expressed) first, then `/com/reddit`, then for `/com/reddit/r`, and so forth. The pseudocode 2 highlights the lookup steps.

Pseudocode 2 Mapping Look Up

```
1: function LookUp(Name)
2:     Alias ← /root
3:     for each Prefix ∈ Name do
4:         Interest ← ExpressInterest(Prefix, Alias)
5:         Data ← WaitForData(Interest)
6:         Alias ← Data.Content
7:         if postfix(data.Name) = /* then
8:             return Alias
9:         end if
10:    end for
11:    return Ø  (Empty alias)
12: end function
```

The lookup starts with expressing of Interest for a one-component prefix (e.g., `/com`) with alias `/root` (lines 2–4). This Interest will be forwarded towards one of the `/root` producers and eventually will return Data (from a `/root` producer or cache, line 5–6) containing a forwarding alias (or aliases) for the next lookup iteration (e.g., alias that can be used in Interest for `/com/reddit`). Essentially, the alias here is a provider-dependent name of the producer, acting as a name-server for the queried zone.

The returned Data may have a postfix `/*`, indicating that the search is complete. This, similar to wildcarded domains in DNS, allows aggregate name mappings. For example, if Interest for `/com/reddit` re-

turned Data named /com/reddit/*, then all names started with /com/reddit prefix will use the same routing alias.

Finally, after all prefixes of the name are tried and none of them terminated the search, then lookup is considered to be failed (line 11). This can happen, for example, if data producer does not exist or did not set up its mapping.

Note that our design requires incremental queries, unlike standard DNS where each query (iterative or recursive) contains a full domain name supplied by the user. The reason lies in the fundamental differences between IP and NDN communication. In IP, each request is made to a particular server and there is not a problem if the same query to different servers will return different datasets (e.g., for query reddit.com DNS-root servers will return only name servers for com). In NDN, Interests are sent to the network, not to a particular server. Thus, Interests must identify particular datasets precisely. In particular, Interests for a partial Data cannot have a name of a full Data (Interest for /com/reddit should not return Data for /com). Otherwise, a partial Data (alias for /com) will be cached inside the network as a full Data (/com/reddit) and any further requests for the same name (/com/reddit) will return the cached partial Data.

Although the requirement for the incremental queries may seem to increase query delays (i.e., each resolution may require several round trips), in-network caching should substantially mitigate this problem. That is, while in DNS cache hits are possible only on DNS servers, every router, including a local NDN module, is able to return previously cached data.

The presented design sketch only demonstrates feasibility and simplicity of possible mapping system implementations. More concrete design along with the performance evaluations are part of our future work.

2.3.2 *Queries*

In either of the ISP name encapsulation approaches discussed in Section 2.2, discovering, as well as, concatenating names or adding aliases to Interests is solely a responsibility of data consumers (local NDN module). In case of the mapping service returning several available mappings (/xo and /nLayer in our example), the consumer is free to choose any of them to try first; if no Data is returned within some expected time period, the consumer can re-express Interest using a different mapping. Moreover, the Interest can always be sent without concatenation or forwarding alias at all. A simple learning scheme can be used by the local NDN module to remember which mapping gives the best result and be used as a default option for future Interests. This largely conforms with the end-to-end argument in a sense that end consumers, rather than the network, should be in control of their choices and responsible for getting the data they wanted.

2.3.3 *Updates*

We assume the content producer (or its delegates) holds the responsibility for maintaining and signing the name zones. For updates, we can employ a technique similar to the standard secure DNS updates [17], which can be issued by the local NDN module when application starts or network changes are detected. The question of how local NDN module can discover the connectivity is an additional research question. In the trivial form, a particular name zone can be maintained manually or a local NDN module can be preconfigured to use specific ISP prefixes.

3. DISCUSSION

There is a well known Rekhter's law that relates routing scalability to the congruency between addressing and topology: "Addressing can follow topology or topology can follow addressing. Choose one." [7] That is, no matter how a routing system is designed, it can scale only if either (1) the address structure reflects the topology, or (2) topology is built based on the addresses.

A number of proposed routing protocols, including ROFL [18] and VRR [19], fall into the second category. These designs perform routing operations over virtual (overlay) topologies, which are built based on node IDs (e.g., hashes of node names). Thus their routing table size can scale on the order of $\log N$, where N is the maximum number of node IDs. However, because the topology is virtual, the actual forwarding paths selected by these routing protocols may have a significant stretch. From network operators' perspective, the resulting virtual topology does not obey administrative boundaries nor allow traffic engineering.

Routing schemes from the first category, such as Compact routing [20], Landmark routing [21], and many others, including the existing Internet routing system, are more operator-friendly and easier to implement. In these systems the routing table size scales by letting nodes use topology-dependent addresses (names). Map-n-Encap is a simple and effective way to obtain topology-dependent names. For example, a recent work [22] argued that one could successfully scale routing with flat data names if one just concatenates the (also flat) names of aggregation entities in front of data names, yet another example of Map-n-Encap idea [8].

We believe that routing in NDN should be performed on the physical topology. Based on Rekhter's law, our only option is to scale routing by topology-dependent names. While users and applications need topology-independent names, this conflict can be solved by employing the well known Map-n-Encap approach, introduced almost twenty years ago.

There is an ongoing effort in the NDN project to apply hyperbolic routing [23] to further scale NDN routing. The basic idea is to map ISP names to coordinates in the hyperbolic space based on the underlying physical topology and route Interests based on these hyperbolic coordinates instead of ISP name prefixes. Nonetheless, this approach still requires a mapping system to map application names to ISP names.

One additional benefit of the proposed routing table scaling solution is its potential to support mobility of data publishers. NDN supports mobile data consumers by design, as a mobile can request data from its latest location. However supporting a mobile producer requires that consumers know the mobile producer's latest ISP prefix. Indeed, one currently deployed mobility solution is to rely on DNS as the mapping service to keep the up-to-date information about a mobile's latest location [24]. This solution matches almost exactly what we are proposing. The remaining questions are only about how fast updates can be performed and how a mobile host can discover the prefix of the ISP that it just moves into.

4. CONCLUSION

In this paper, we examined routing scalability problem of both IP and NDN and concluded that they are essentially the same problems. As a result, we proposed application of the old map-n-encap idea to scale NDN routing, where mapping of application names to ISP names leads to a manageable size routing table on transit core routers (i.e., they will contain only entries for ISP names). In the context of NDN, we designed a way to encapsulate ISP names in Interests in form of the forwarding alias, which essentially is a hint for routers as to where forward this Interest. To implement a secure mapping between application and ISP names we sketched out the design of a native NDN mapping service.

5. REFERENCES

[1] D. Cheriton and M. Gritter, "TRIAD: A new next-generation Internet architecture," 2000.

[2] T. Koponen et al., "A data-oriented (and beyond) network architecture," in *Proc. of SIGCOMM*, 2007.

[3] S. Tarkoma, M. Ain, and K. Visala, "The publish/subscribe Internet routing paradigm (PSIRP): Designing the future internet architecture," *Towards the Future Internet*, 2009.

[4] L. Zhang et al., "Named data networking (NDN) project," PARC, Tech. Rep. NDN-0001, October 2010.

[5] A. Baid, T. Vu, and D. Raychaudhuri, "Comparing alternative approaches for networking of named objects in the future Internet," in *Proc. of NOMEN*, 2012.

[6] A. Narayanan and D. Oran, "NDN and IP routing: Can it scale?" Proposed Information-Centric Networking Research Group (ICNRG), Side meeting at IETF-82, November 2011.

[7] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB workshop on routing and addressing," RFC 4984, 2007.

[8] S. Deering, "The map & encap scheme for scalable IPv4 routing with portable site prefixes," *Presentation Xerox PARC*, 1996.

[9] M. O'Dell, "8+8—an alternate addressing architecture for IPv6," Internet draft (draft-odell-8+8-00), 1996.

[10] D. Farinacci, "Locator/ID separation protocol (LISP)," Internet draft (draft-farinacci-lisp-00), 2007.

[11] R. Atkinson, S. Bhatti, and S. Hailes, "ILNP: mobility, multi-homing, localised addressing and security through naming," *Telecommunication Systems*, vol. 42, no. 3, 2009.

[12] D. Jen, M. Meisel, D. Massey, L. Wang, B. Zhang, and L. Zhang, "APT: A practical tunneling architecture for routing scalability," UCLA Comp. Sc. Dep., Tech. Rep. 080004, 2008.

[13] D. Massey, L. Wang, B. Zhang, and L. Zhang, "A scalable routing system design for future internet," in *Proc. of SIGCOMM IPv6 and the Future of the Internet workshop*, 2007.

[14] R. Oliveira, B. Zhang, and L. Zhang, "Observing the evolution of internet as topology," in *Proc. of SIGCOMM*, 2007.

[15] A. Afanasyev, N. Tilley, B. Longstaff, and L. Zhang, "BGP routing table: Trends and challenges," in *Proc. of High Tech. and Intell. Systems conf.*, 2010.

[16] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. of CoNEXT*, 2009.

[17] P. Vixie, S. Thomson, Y. Rekhter, and J. Bound, "Dynamic updates in the domain name system (DNS UPDATE)," RFC 2136, 1997.

[18] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, and I. Stoica, "ROFL: routing on flat labels," in *Proc. of SIGCOMM*, 2006.

[19] M. Caesar, M. Castro, E. B. Nightingale, G. O'Shea, and A. Rowstron, "Virtual ring routing: network routing inspired by DHTs," in *Proc. of SIGCOMM*, 2006.

[20] M. Thorup and U. Zwick, "Compact routing schemes," in *Proc. of SPAA*, 2001.

[21] P. F. Tsuchiya, "The landmark hierarchy: a new hierarchy for routing in very large networks," in *Proc. of SIGCOMM*, 1988.

[22] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker, "Naming in content-oriented architectures," in *Proc. of SIGCOMM Workshop on ICN*, 2011.

[23] M. Boguñá, F. Papadopoulos, and D. Krioukov, "Sustaining the internet with hyperbolic mapping," *Nature Communications*, vol. 1, 2010.

[24] Z. Zhu, R. Wakikawa, S. Cheshire, and L. Zhang, "Home as you go: an engineering approach to mobility-capable extended home networking," in *Proc. of AINTEC*, 2011.