Information-Centric Networking:

Seeing the Forest for the Trees

Ali Ghodsi
KTH / UC Berkeley

Teemu Koponen
Nicira Networks

Barath Raghavan
ICSI

Scott Shenker
ICSI / UC Berkeley

Ankit Singla
UIUC

James Wilcox
Williams College

ABSTRACT

There have been many recent papers on data-oriented or content-centric network architectures. Despite the voluminous literature, surprisingly little clarity is emerging as most papers focus on what differentiates them from other proposals. We begin this paper by identifying the existing commonalities and important differences in these designs, and then discuss some remaining research issues. After our review, we emerge skeptical (but open-minded) about the value of this approach to networking.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

General Terms

Design, Performance, Security

Keywords

Information-Centric Networking, Internet Architecture

1 Introduction

In the emerging research topic of what we will call information-centric networking (ICN), many recent papers and research efforts have noted that we should move the Internet away from its current reliance on purely point-to-point primitives and, to this end, have proposed detailed designs that make the Internet more data-oriented or content-centric. This idea is hardly new; the TRIAD paper [15] described an ICN-like design over a decade ago, and should rightfully be considered an important precursor to all recent ICN work.[1] In addition, the 2002 IETF draft by Baccala [5] (written soon after TRIAD) reaffirmed the point that we should move towards a world where "the primitive operation in displaying a web page is no longer an end-to-end connection to the web server, but the delivery of a named block of data". There were relatively few papers that built on these ideas in the years immediately following their publication, suggesting that they were well ahead of their time.

While remarkably prescient, both of these designs used the existing DNS naming system (with its inherent drawbacks) and focused only on basic content delivery without much attention paid to other important issues such as security, streaming media, or faulty servers. The Data-Oriented Network Architecture (DONA) [19], coming roughly five years after these two seminal works, was perhaps the first comprehensive and detailed clean-slate ICN design, advocating the use of self-certifying names (as suggested earlier in [20, 30]) and incorporating advanced cache functionality to address various other ICN issues. Again, there was little immediate follow-up on this work, and it appeared that the topic was never going to capture the full attention of the broader research community.

These worries were unfounded. The recent Content-Centric Networking proposal (CCN) has ignited widespread interest in the ICN area [17]. Several workshops devoted to ICN have been held, and projects such as 4WARD [3], PSIRP/PURSUIT [29], SAIL [24], and COMET [11] have focused on this topic. CCN itself, in the form of the Named-Data Network proposal [33], was one of the four proposals chosen for NSF's FIA program. Despite the rather tepid research interest in the earlier years, ICN is now clearly entering the networking research mainstream.

Unfortunately, the resulting ICN literature is somewhat difficult to absorb, even for those contributing to it. There

Hotnets '11, November 14–15, 2011, Cambridge, MA, USA.
Copyright 2011 ACM 978-1-4503-1059-8/11/11 ...$10.00.

[1] Of course, publish/subscribe systems are far older than this (see, for example, [25, 28]), and ICN designs share much in common with the publish/subscribe paradigm, but TRIAD was the first (to our knowledge) to advocate publish/subscribe as the basic Internet paradigm and to implement it on a global scale.

is little common terminology between these proposals and, because there is no common framework, the focus is often on low-level mechanisms. As a result, many of the papers accentuate the differences between their design and others[2], leaving it as an exercise for the reader to construct the ICN forest out of these individually-proposed trees. And the community has, for the most part, taken it for granted that the ICN forest is worth preserving, without a deeper investigation of its roots. This state of affairs led us to write this paper, with the goal of providing a broader perspective on current ICN designs and revisiting some of the fundamental assumptions that underlie ICN.

We start (Section 2) by reviewing the commonalities in ICN designs, which are broader and deeper than one might have thought. We then describe (Section 3) what we see as the most essential differences in current ICN designs; these issues require more community-wide discussion to resolve, and we hope that identifying them here will help initiate such discussions. We end in Section 4 by presenting several ICN questions that remain unanswered, some of which raise doubts about the efficacy of the ICN approach.

2 Commonalities of Designs

There has been enough buzz about ICN that most readers will be familiar with the general outlines of the approach. However, in order to provide the context necessary for our later sections, we briefly review the common aspects of ICN systems. We restrict our purview to the more recent ICN designs (*i.e.,* we include designs such as CCN, PSIRP, DONA, Curling [8], and 4WARD, but we do not include TRIAD or Baccala's design). At first glance, all these proposals seem unique, with important differences from other ICN proposals. This impression is reinforced by the fact that each has its own distinctive terminology, and they often emphasize different aspects of the design problem (*e.g.,* LIPSIN [18] focuses more on the datapath behavior than other proposals). However, these proposals all share three fundamental principles (described below) which are the essential ingredients for shifting the Internet away from a point-to-point paradigm to a more information-centric one.

Basic Primitives. The publish/subscribe paradigm has been around for over 25 years [28] and is now used in many systems ranging from web services to enterprise information systems [12]. In this paradigm, there are two basic primitives: `PUBLISH`, which enables information providers to advertise the availability of their content, and `SUBSCRIBE`, which enables consumers to request content. These primitives decouple requests and responses in both space and time: that is, the provider and requester of the content need not know each other's location, nor need they be online at the same time. This decoupling is one of the most profound aspects of publish/subscribe systems.

ICN designs are, at heart, merely global-scale versions of the publish/subscribe paradigm. It is therefore not surprising that all ICN designs are built around two basic interface primitives that resemble the notions of `PUBLISH` and `SUBSCRIBE` (and, in fact, PSIRP adopts the publish/subscribe terminology for its two primitives). CCN uses `REGISTER` and `INTEREST` operations, DONA uses `REGISTER` and `FIND` commands, while Curling uses both `REGISTER` and `PUBLISH` for the publish operation and `CONSUME` for the subscribe operation.

Note that ICN and publish/subscribe are not identical. In ICN designs, these primitives act on the name of the object (that is, content is published and subscribed to by name), while publish/subscribe systems can have broader request semantics (such as describing content with various tags and allowing subscriptions to relate to any content described with that tag). Also, ICN systems usually offer both a one-time "fetch" operation (retrieving content previously published under that name) and an ongoing "subscribe" operation (retrieving all future content published under that name). In contrast, most publish/subscribe systems only support the latter.

Universal Caching. In ICN designs, when a network element receives a request for content (from a peer or host), it does one of two actions: *(i)* if it has the data cached, it can respond with the content directly, or *(ii)* if it does not have the content cached, it can request the content from its peer(s) and then cache the content when this request is filled. This caching is universal in three ways:

- It applies to content carried by any protocol, not just content carried by a specific protocol (*e.g.,* HTTP). ICN thus provides a single *uniform* caching paradigm that underlies all content delivery.
- It applies to all content from all users, not just content from content providers who have contracted for the service (as in today's CDNs). This *democratizes* content delivery.
- It is implemented by all ICN nodes rather than just a few specialized caches, making caching *pervasive*. We will return to this issue in Section 4.

Content-oriented Security Model. Since the ICN approach results in content arriving from network elements other than the originating server, the security model cannot be based on where the packet came from; instead, ICN designs must secure the content rather than the path, as suggested in [30,31] and elsewhere. All ICN designs thus adopt a content-oriented security model in which content is signed by the original content provider, so network elements and consumers can verify the validity of the content merely by verifying the signature. There are some fundamental questions about the role of naming in this security model, which we discuss in the next section, but the basic content-oriented security model is shared by all ICN designs.

[2] The NDN proposal [33] is an exception to this, but it goes to the other extreme; it barely mentions related work!

With this background about the three principles shared by all ICN designs, we now turn to areas where these proposals disagree in fundamental ways.

3 Fundamental Differences in Design

There are three important areas where the various ICN proposals differ: naming, interdomain routing, and the location of the narrow waist in an ICN Internet.

Naming. ICN systems enable consumers to request objects by name, not location. In the content-oriented security model described earlier, the following must hold:

- The consumer must know the name of the content they want. That is, they must be able to map between the real-world description of what they want (*e.g.,* CNN headlines) and its corresponding ICN name.
- The consumer must know the content provider's public key, so that she can verify the provenance and integrity of the content.
- The ICN system itself must be able to bind an object's name to the public key of the content provider, so that it can prevent attackers from registering false content. Without this binding, attackers can use false content as a denial-of-service attack.[3]

There are two main naming systems proposed in the ICN literature. The first, which resembles today's DNS names, uses hierarchical human-readable names. The human readability partially addresses the first requirement, and the hierarchical structure helps scalability. A variety of techniques can allow the consumer to know the public key (ranging from personal contacts to webs-of-trust to PKIs), but for the ICN system to be aware of this key requires a globally-agreed-upon PKI to bind names to keys.

The second naming system uses self-certifying names. Here, the key is bound to the name itself, so the ICN system need not use a PKI. These names are not human-readable, so consumers must use other techniques (*e.g.,* search engines, personal contacts, webs-of-trust) to determine the name of the content they want.[4]

Two diametrically opposing viewpoints on the relative merits of these names can be found in [14, 26]. Given the overlap in authorship between this paper and [14], it should come as no surprise that we think that self-certifying names are clearly the superior choice; this form of naming enables the infrastructure to prevent denial-of-service attacks without requiring the infrastructure to understand user trust models.

Interdomain Name-Based Routing. In order to satisfy requests for content, ICN systems must route those requests. There are many different approaches in the ICN literature for doing this name-based routing within a domain, but these differences are largely mechanistic in nature and not of fundamental importance. Where the proposals differ more fundamentally is in how this routing is done between domains. Some (such as CCN) leverage the current Interdomain routing system and build their name-based routing on top of BGP. Others (such as DONA) follow the BGP policy model but do their own name-based routing, and still others (such as PSIRP) develop their own interdomain routing paradigm.

Narrow Waist. IP is the narrow waist of the current Internet architecture. The various ICN proposals differ in whether this remains the case, or whether the ICN layer becomes the new narrow waist. This is often cited as a fundamental difference between ICN *designs* but we contend this is really a broader architectural debate that is largely orthogonal to ICN design details. All of the ICN designs involve hop-by-hop communication between the ICN-layer elements (*e.g.,* Content Routers in CCN, Rendezvous Nodes in PSIRP, Resolution Handlers in DONA, Content-aware Routers in Curling). Since this communication is merely between hops (and does not require global reachability), one could run any of these designs over IP, or as a replacement for IP (running over some L2-like layer that provides local delivery). However, whether one retains or replaces IP as the narrow waist obviously has implications for the overall Internet architecture, and for the performance required in the ICN layer. We revisit this in Section 4.

4 Research Agenda

Having described how the various ICN designs compare, we now discuss future directions for ICN research. We consider three different categories: topics deserving less attention, topics deserving more attention, and topics deserving *immediate* attention.

4.1 Topics Deserving Less Attention

Many of the ICN papers focus on the required mechanisms, and several reviews of the topic (such as in [1]) suggest future work along these lines. We disagree, and think that focusing on ICN mechanisms is not where we should devote our research effort. We say this because, with recent techniques such as HTTP long polling and Comet, HTTP already supports the basic ICN primitives. The following are then needed to turn HTTP into a full-fledged ICN design (obeying the three principles in Section 2):

- *Providing caching for all content delivery*: HTTP is rapidly becoming the protocol-of-choice for most content delivery, so this goal is already within sight.
- *Providing caching at all network elements*: some router vendors have openly discussed the possibility of

[3] This is a denial of service issue, not a correctness issue: the consumer can verify the correctness of the data because it knows the key, but if the ICN system cannot verify the content does indeed correspond to that name then the ICN system may repeatedly deliver false data (and thus not be able to reliably deliver the correct data).

[4] There is a duality between the two approaches: each uses external mechanisms for one binding, but for one that external binding is between names and keys, and for the other it is between real-world content and names.

placing HTTP proxies on all routers. Doing so would immediately provide the democratization of content delivery, and is technically feasible.

- *Content-oriented security model*: this is an issue of content naming and/or a PKI, neither of which are dependent on ICN design details. Note, however, that this security model is quite different than that used in HTTPS.

Turning HTTP into a full-fledged ICN design does not require additional research but instead is merely a question of deploying known solutions. Focusing on clean-slate ICN designs offers few benefits and makes the deployment problem immensely harder. As trenchantly observed in [22], if there is one thing the Internet currently does well, it is delivering content! Given the tremendous success of, and experience with, current content delivery, our research community would probably be well-served by spending less time fine-tuning the mechanisms in clean-slate ICN systems.

4.2 Topics Deserving More Attention

Privacy. There has been much discussion about how ICN changes the security model, from securing the path to securing the content. However, there has been far less attention paid to the unfortunate fact that ICN also greatly changes the privacy model. Today, a client can establish a secure channel between itself and the content provider, and then the nature of the content being requested from that provider is known only to the client and the provider. In ICN, the name of the content being requested is available to all the ICN nodes processing the request. This is intrinsic to ICN, since ICN provides a get-by-name service, so it is impossible to hide the content name from the ICN infrastructure.[5]

We think it is important to explore how this loss of privacy can be ameliorated. See [4] for an initial attempt at one such design, where users and providers collude to prevent the ICN system (or the government) from being able to detect when certain “forbidden" content is being accessed; in short, the scheme fragments content requests into blocks which are then fetched in a spread-spectrum manner to obfuscate the content requests being made. The intricacy of the design required to provide such a minimal improvement in privacy (the ICN system still knows every piece of content being downloaded, but doesn’t realize that some forbidden content can be reconstructed using the seemingly innocuous downloaded content) suggests that privacy will be a formidable challenge for ICN. Therefore, we should also explore how ICN systems can be redesigned to support better privacy. It may be possible to provide special operations or services that enable the tunneling of content between publishers and subscribers in a way that still enables caching.

In addition, there are a variety of attacks on privacy — ranging from censorship (making content unavailable) to persecution for downloading undesirable content (*i.e.,* using ICN exchanges as non-repudiable evidence) — and dealing with these will presumably involve different approaches.

Interdomain Policies. There is a vast literature on BGP policies, ranging from protocol design to economic analysis. And this area has recently entered a new era where (as discussed in [10]) the traditional customer-provider-peer categories are being augmented by the distinction between eyeball and content networks.

As discussed, ICN systems must confront the issue of interdomain policies, and here the eyeball/content distinction is even more important, because ICN systems are expressly designed for name-based content routing. While the exact nature of content-peering policies will be determined by business considerations, academic research can help illuminate what kinds of global connectivity results from an assortment of pairwise business relationships. For instance, is there an ICN equivalent of the Gao-Rexford conditions, which would provide guidelines for what forms of content-peering relationships result in global connectivity and stability?

Scaling, Object Sizes, and the Narrow Waist The number of content objects is huge, and rapidly growing. Any ICN system should be prepared to handle at least 10^{12} objects (based on the current size of the web), and this is an extremely conservative estimate. At this scale, some tradeoffs need to be made, which we briefly summarize here in a greatly oversimplified fashion.

If routing decisions are to be made at packet speeds, then the routing table needs to be relatively small, say on the order of 10^8 entries or smaller.[6] There are two ways to achieve a routing table that is several orders of magnitude smaller than the total number of objects. The first is to achieve high levels of aggregation through hierarchical names.[7] But such aggregation makes multihoming of data hard (as we have painfully experienced in the Internet), and without multihoming, ICN would merely have one entry in its forwarding table; that of the server originating the content. ICN systems would then route requests back towards this server, and would not be aware of any cached copies that were not on this default path to the server. This caching-along-default-path behavior is hardly worth adopting a clean-slate ICN architecture, because turning every router into an HTTP proxy would accomplish this more simply.

The other way routing tables could be small is if the request patterns result in a small working set in the routing table (so one could merely cache routing entries rather than storing

[5] Note that the name of the requester is typically only available to the first-hop ICN node, since each node is involved in a pairwise exchange with the previous ICN node.

[6] This number could be off by an order of magnitude or two, but so could the estimate of the number of objects. Our core assumption is that the number of objects is roughly three orders of magnitude larger than the size of the routing table that can attain packet speeds.

[7] See the discussion in [14] for how one can achieve aggregation without hierarchical names, which somewhat relieves the tension between aggregation and multihoming.

the whole table on the fast path). We return to this topic later when we discuss the effectiveness of caching, but our conclusions therein are not encouraging. For now, we assume the working set size is not small.

This line of reasoning suggests that one either chooses a simple caching-along-default-path design with hierarchical names, or one does not make ICN routing decisions at packet speeds. In the latter case, one can build *scale-out* lookup engines (*i.e.,* clusters that can handle very large routing tables, certainly on the order of 10^{12} objects) [2, 19]. In this scenario, the ICN system has relatively slow name-based routing (slow in terms of throughput; the latency of each lookup is of less concern), but once the data location is known data transfers can be done at packet speeds (much like today, where DNS lookup processing is relatively slow compared to packet speeds).

However, this requires that the content objects be significantly larger than the requests. Assume that the low-level packet transport can handle x packets/sec (globally), and the route-by-name system can process y requests per second (globally). Then on average each request should result in roughly x/y packets. Otherwise, the transmission infrastructure will be grossly underutilized. We understand that this is a radically, perhaps even dangerously, oversimplified model, but we present it only to illustrate the more general point that if there is a speed mismatch between packet-routing and name-routing, then there must also be a corresponding (and inverse) size mismatch.

In Section 3 we discussed the debate about where the narrow waist should be: IP or ICN? This is equivalent to asking whether route-by-name is the lowest-level global network primitive (*i.e.,* the only way to establish global communication) or whether there is a lower-level address-based network primitive that enables global reachability. Our argument above suggests that unless one adopts the cache-along-default-path design, we cannot do the name-based routing fast enough to make ICN the narrow waist. That is, one cannot have the network level primitive be something that cannot be processed at packet speeds.[8]

Thus, there is an interesting tradeoff between naming (hierarchical or not), routing behavior (just route to server, or route to nearest copy), caching behavior (is the working set size small?), the size of objects (which cannot be single packets unless the requests can be handled at packet speeds), and the narrow waist (the waist must be able to operate at line speeds). Here we have suggested (subject to further research, of course) that there is little reason (based on performance) to adopt an ICN design that only caches along path, which implies nonaggregated names, which implies large routing tables, which implies slower name-based routing, which implies large ICN objects, which implies an IP waist. We present this line of reasoning not as a hardened conclusion, but as a conjecture that should be investigated.

4.3 Topics Deserving Immediate Attention

All of the research issues mentioned above address various design questions, but accept without question one of the basic premises of ICN: making content caching an intrinsic and ubiquitous part of the network infrastructure would improve performance. We now question that premise.

In the early days of the web, well before the advent of ICN, there was a flurry of research about web caching [9, 27]. In particular, one popular design goal was that of cooperative caching [13, 21, 23], in which each cache called upon another cache as needed before contacting the content server. In 1999, Wolman *et al.* used large web traces to analyze the effectiveness of cooperative caching [32]. They found that hierarchical or cooperative caching wasn't helpful once the population behind the edge cache grew past a small threshold:

> *...we show that cooperative caching is unlikely to have significant benefits for larger organizations or populations. That is, with current sharing patterns, there is little point in designing highly scalable cooperative-caching schemes; all reasonable schemes will have similar performance in the low-end population range where cooperative caching works.*

More specifically, the authors note that for a user population in the tens of thousands, "a single proxy cache can provide the same benefits" as cooperative caching. These results applied more broadly not just to the traces they examined, but to heavy-tailed object distributions. The authors reached this conclusion by extending the analytical model of Breslau *et al.* who also found that object requests followed a Zipf distribution [7]. These papers closed the door on what was once a fertile area of study.

Other types of content-sharing networks have content request distributions even more unfriendly to cooperative caching [16]:

> *Kazaa is not Zipf. The popularity distribution for large objects is much flatter than Zipf would predict, with the most popular object being requested 100x less than expected. Similarly shaped distributions exist for small objects and the aggregate Kazaa workload. ... the Web is well described by Zipf.*

Content in ICN-based networks is likely to be at best as (un)cacheable as the Web, and quite possibly as bad as Kazaa.

This should be of great and immediate concern to those of us who have advocated ICN designs. While ICN is a new networking paradigm, its claimed performance advantages come from widespread caching, which is essentially what cooperative caching is. The fact that these papers argued successfully against cooperative caching is troubling.

While such analyses were performed a decade ago, Facebook's use of their caches in their image-serving system indicates that the conclusions still hold [6]. Specifically, when

[8] This assumes that the waist must use moderate sized packets.

requests miss the content-distribution network and further miss Facebook's internal cache, images are returned from the storage system, but are *not* cached in Facebook's internal cache, instead caching only at the CDN:

> *...our experience with the NFS-based design showed post-CDN caching is ineffective as it is unlikely that a request that misses in the CDN would hit in our internal cache.*

Amidst all this empirical evidence, it is useful to recall why additional caching doesn't help. There is a reasonable sized set of popular content, and a very long tail of content that is of interest to a small population. Moderately sized edge-caches, which is what today's CDN's use, are easily sufficient to handle the popular content. Once one enters the long tail, the effectiveness of caching increases *logarithmically* with the size of the cache [7]. Changing the overall network architecture in order to tame the exponentially growing world of content with the logarithmic sword of caching seems a classical example of taking a knife to a gunfight: it may make for a great story, but it won't end well.

5 Final Thoughts

We have raised doubts about whether ICN designs would improve network performance. But there are other benefits that ICN designs might bring, such as a better security model, intrinsic routing stability (*i.e.,* loop-freeness), and protection against denial-of-service. The question, then, is whether these benefits require the full ICN approach, or could be obtained more incrementally. For instance, the security model is based on the properties of the name, not the content delivery mechanism (in fact, this security model was proposed well before ICN designs became popular). In addition, one may find other ways to avoid loops and protect against DoS using more state than today's Internet but far less state than a full-blown ICN design. So we end this paper by posing two fundamental questions: *what benefits do we think ICN designs offer, and are ICN designs the best way to achieve those benefits?* The field has focused on the former (though somewhat uncritically, as suggested by our discussion of caching), but we think addressing the latter is equally important.

6 References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A Survey of Information-Centric Networking, February 2011. http://drops.dagstuhl.de/opus/volltexte/2011/2941.

[2] D. G. Andersen, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan. FAWN: A Fast Array of Wimpy Nodes. In *Proc. of SOSP*, 2009.

[3] P. A. Aranda et al. Final Architectural Framework, June 2010. http://www.4ward-project.eu/.

[4] S. Arianfar, T. Koponen, B. Raghavan, and S. Shenker. On Preserving Privacy in Information-Centric Networks. In *Proc of SIGCOMM Workshop on ICN*, 2011.

[5] B. Baccala. Data-oriented Networking. Internet draft, IETF, August 2002.

[6] D. Beaver, S. Kumar, H. Li, J. Sobel, and P. Vajgel. Finding a Needle in Haystack: Facebook's Photo Storage. In *Proc. of OSDI*, 2010.

[7] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web Caching and Zipf-like Distributions: Evidence and Implications. In *Proc. of INFOCOM*, 1999.

[8] W. K. Chai et al. Curling: Content-ubiquitous Resolution and Delivery Infrastructure for Next-generation Services. *Communications Magazine, IEEE*, 49(3), March 2011.

[9] A. Chankhunthod, P. Danzig, C. Neerdaels, M. Schwartz, and K. Worrell. A Hierarchical Internet Object Cache. In *Proc. of USENIX*, 1996.

[10] D. Clark, P. Faratin, P. Gilmore, S. Bauer, A. Berger, and W. Lehr. The Growing Complexity of Internet Interconnection. *Communications & Strategies*, 2008.

[11] COntent Mediator architecture for content-aware nETworks (COMET). http://www.comet-project.org/.

[12] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec. The Many Faces of Publish/Subscribe. *ACM Computing Surveys*, 35(2), June 2003.

[13] L. Fan, P. Cao, J. Almeida, and A. Broder. Summary Cache: A Scalable Wide-area Web Cache Sharing Protocol. In *Proc. of SIGCOMM*, 1998.

[14] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker. Naming in Content-Oriented Architectures. In *Proc of SIGCOMM Workshop on ICN*, 2011.

[15] M. Gritter and D. R. Cheriton. TRIAD: A New Next-Generation Internet Architecture. http://www-dsg.stanford.edu/triad/, July 2000.

[16] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, Modeling, and Analysis of a Peer-to-peer File-sharing Workload. In *Proc. of SOSP*, 2003.

[17] 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.

[18] P. Jokela, A. Zahemszky, C. Esteve Rothenberg, S. Arianfar, and P. Nikander. LIPSIN: Line Speed Publish/Subscribe Inter-Networking. In *Proc. of SIGCOMM*, 2009.

[19] 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. of SIGCOMM*, 2007.

[20] D. Mazières, M. Kaminsky, M. F. Kaashoek, and E. Witchel. Separating Key Management from File System Security. In *Proc. of SOSP*, 1999.

[21] S. Michel, K. Nguyen, A. Rosenstein, L. Zhang, S. Floyd, and V. Jacobson. Adaptive Web Caching: Towards a New Global Caching Architecture. *Computer Networks and ISDN Systems*, 30(22-23), 1998.

[22] L. Popa, A. Ghodsi, and I. Stoica. HTTP as the Narrow Waist of the Future Internet. In *Proc. of HotNets*, 2010.

[23] M. Rabinovich, J. Chase, and S. Gadde. Not All Hits Are Created Equal: Cooperative Proxy Caching over a Wide-area Network. *Computer Networks and ISDN Systems*, 30(22-23), 1998.

[24] Scalable and Adaptive Internet Solutions (SAIL). http://www.sail-project.eu/.

[25] D. Skeen. Vitria's Publish-Subscribe Architecture: Publish-Subscribe Overview, 1998. http://www.vitria.com/.

[26] D. Smetters and V. Jacobson. Securing Network Content. Technical report, PARC, October 2009.

[27] Squid: Optimising Web Delivery. http://squid-cache.org/.

[28] Tibco. Tibco Enterprise Message Service. http://tibco.com/.

[29] D. Trossen et al. Conceptual Architecture: Principles, Patterns and Sub-components Descriptions, May 2011. http://www.fp7-pursuit.eu/PursuitWeb/.

[30] M. Walfish, H. Balakrishnan, and S. Shenker. Untangling the Web from DNS. In *Proc. of NSDI*, 2004.

[31] D. Wendlandt, I. Avramopoulos, D. Andersen, and J. Rexford. Don't Secure Routing Protocols, Secure Data Delivery. In *Proc. of HotNets*, 2006.

[32] A. Wolman, M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. Levy. On the Scale and Performance of Cooperative Web Proxy Caching. In *Proc. of SOSP*, 1999.

[33] L. Zhang et al. Named Data Networking (NDN) Project, October 2010. http://named-data.net/ndn-proj.pdf.