

# A policy framework for the future Internet

Arun Seehra<sup>†</sup>, Jad Naous<sup>‡</sup>, Michael Walfish<sup>†</sup>, David Mazières<sup>‡</sup>, Antonio Nicolosi<sup>§</sup>, and Scott Shenker<sup>¶</sup>

<sup>†</sup>UT Austin   <sup>‡</sup>Stanford   <sup>§</sup>Stevens Institute of Technology   <sup>¶</sup>UC Berkeley, ICSI

## 1 Introduction

This paper is about the Internet’s future, but we begin with its past. The history of network routing began as a topological problem: how does one find the shortest paths in a graph ([11])? However, with the advent of domain-based Internet routing, policy became an important consideration. In fact, policy concerns were embedded in the 1989 requirements document (RFC 1126) that set the groundwork for the first version of BGP:

*Those resources used by (and available for) routing are to be allowed autonomous control by those administrative entities which own or operate them. Specifically, each controlling administration should be allowed to establish and maintain policies regarding the use of a given routing resource. [22]*

Embodying this principle, BGP allows each domain to unilaterally decide which routes it accepts and exports based on the full AS-level path.

Provider control is not limited to the control plane; providers have imposed usage limits and blocked certain types of traffic that they believe would be injurious to their or other networks.

Moreover, ASes are not the only stakeholders in the Internet. There have been many calls for granting sources some control over their packets’ paths (see, for example, [6, 12, 15, 18, 20, 30, 38, 39]). The reasons vary from performance (letting sources find the best quality paths) to preference (letting sources avoid providers they don’t trust) to price (letting sources find the cheapest paths).

For exactly the same reasons, receivers too have an interest in controlling the path of their incoming packets. Receivers also care *who* is sending them packets and may wish to allow only a subset of incoming flows (e.g., when under attack, accept packets only from customers).

While each of these policy considerations seems natural, it isn’t clear how to balance the concerns of the various stakeholders. Take, for instance, the case of a user trying to send email from her hotel room. The user would like her packets to reach her company’s mail server via a reliable and high-bandwidth path. The hotel would like her packets to take the least costly path. The first-hop provider cares that the packets are coming from a paying customer but wants to block all transiting SMTP traffic because they fear that it might be spam. The receiving mail server only wants to receive outgoing SMTP traffic

from company employees. Moreover, it wants all of this traffic to pass through a third-party virus-scanner service to which it has subscribed. All of these are valid policy goals, as they concern the use of the stakeholder’s resource or the fate of their own communication, but it is not clear which of these policy considerations, when they are in conflict, should prevail.

All of the preceding background leads us to the question this paper tries to address: what policy framework should we adopt in a future Internet architecture? This question is one of both policy and mechanism: what policy considerations should the architecture support, and can we build a mechanism to support those considerations?

### 1.1 The nature of policy

Judging by the bevy of architectural proposals that support policy-oriented features such as interdomain policies, source selection of routes, and interposition of middleboxes by endpoints, there appears to be consensus that the various stakeholders have the right to exert some control over their flows, and that these considerations should be reflected in a future Internet architecture. Table 1 lists many, but by no means all, of these proposals. As the table makes clear, while the union of policy considerations is large, the intersection is small: each proposal generally supports only a particular subset of stakeholder control.

As a community striving to design the future Internet, we have two choices:

- Choose one subset of policy considerations and bet that it will be sufficient to meet all policy needs for the foreseeable future.
- Choose to support all reasonable policy considerations, allowing the Internet’s policies to evolve as its usage and organizational structure change.

The first choice, while certainly expedient, seems risky given how unpredictable the Internet has been so far, both in terms of the nature of traffic and the organizational structure of its stakeholders.<sup>1</sup> In fact, we (as a community) have a terrible record in predicting the future of the Internet, and opting for this choice is a gamble that we will finally get it right this time.

Thus, on policy grounds, the second choice is more desirable. However, it poses two challenges: can we identify what constitutes *reasonable* policy considerations,

<sup>1</sup>Recall that the modern ISP-oriented Internet arose in the last fifteen years and is not at all what the Internet pioneers envisioned.

Proposed approach	Policy function								
	dest. control of sender	resource attri- bution	provider policy granularity			src route control	MB* route control	rcvr- invoked MBs*	network- invoked MBs*
			prefix	suffix	subsequence				
Capabilities, filters [7, 23, 36, 37, 40]	x								* MB = middlebox
Visas [13]		x							
Platypus [32]		x				x			
Pathlets [15]				x	x	x			
LSRR, Wiser [4, 24]			x						
MIRO [35]				x					
Src routing [18, 20, 38, 39]						x			
Byzantine routing [6, 26, 29, 30]						x			
NUTSS [17]							x		
DOA, i3 [33, 34]								x	
DONA [21]									x

Table 1—Policy functions provided by many, but not all, network-layer proposals. Many of these proposals cannot be implemented together. The framework in the text is intended to be flexible enough to capture all of these legitimate policy interests.

and can we build a mechanism to support all such policies? In response to the first challenge, we offer the following principle for reasonable policies:

**Policy Principle:** *A communication should be allowed if, and only if, all participants approve. By participants, we mean the sender, the receiver, the carriers, and any other intermediaries.*

This principle posits that non-participants should have no say in whether a communication occurs. This doesn't mean that governments and other third-parties have no say about the nature of communications, only that the Internet architecture itself does not enforce such third-party concerns. These third-party concerns must be addressed by other means, such as the legal system.

Note that this principle gives every participant veto power. This may be overkill (for instance, as in [38], one might think that receivers should only be able to control the path of packets once they have left the Internet's core), but we conjecture (based on our inability to find one) that there is no intermediate position or weakening of this policy principle that supports the desires of all stakeholders. Moreover, just because the Internet architecture allows such control does not mean it will be exercised, as economic and social pressures strongly constrain which policies are enacted. For instance, BGP allows ISPs to pick routes based on the entire AS path, but they rarely exercise more than first-hop preferences.

This brings us to the second challenge: can we build a mechanism that supports such a general set of policies? The goal of the rest of this paper is to convince the reader that the answer to this question is not an obvious "no". To support our case, the sections ahead outline one such mechanism, discuss its feasibility, and consider its use. While certainly not perfect, this mechanism (which we call ICING—Incorporating Consent in the Internet's Next Generation) should at least provide hope that supporting the general policy principle is not a lost cause.

But before outlining ICING, we first ask: what does it mean for a mechanism to support a policy?

## 1.2 The nature of mechanism

When we say that a mechanism supports a policy, we mean that it enforces the set of policy choices agreed to by the participants; that is, if the participants all approve then the communication should proceed, and if one or more participants don't approve then the communication should not happen. However, there are further mechanism requirements. We now state several *mechanism principles* that should guide the design of any future Internet (and that guided our design of ICING).

### Mechanism Principles:

1. *The mechanism should ensure that approved communications occur as described.* This means that if a communication is described as following a particular path and approved as such, the mechanism should enforce that the communication in fact follows that path.
2. *The mechanism should ensure that unapproved communications cannot be initiated.* This means that if one or more of the participants do not approve the communication, then no packets enter the network. That is, the communication is blocked at the source, before the packets consume network resources.
3. *The mechanism should not rely on any central trusted authority.* No long-lived, global architecture can assume the existence of a permanent, single source of authority.
4. *The mechanism should impose fixed and feasible requirements on the data plane.* Clearly the mechanism must be feasible, but it should also give router vendors a fixed target to implement, avoiding the explosion of options and features that force continual respinning of router ASICs.
5. *The mechanism should implement subsets of policy efficiently.* This means that if only a subset of the participants wishes to exert their control over communications, then the mechanism should be able to simplify the control plane. In short, the mechanism should not make the Internet pay for unused generality, at least not on the control plane.

6. *The mechanism should work even in the face of malicious participants.* Enforcing policy is not difficult if all participants cooperate. A hard problem is how to enforce policies in a non-cooperative environment.

## 2 Description of ICING

We describe ICING at a high level and then fill in some details. Unfortunately, we do not have space to give a full description or address many natural questions. However, our technical report [28] supplies a number of the details.

### 2.1 Overview of ICING

ICING divides the network into *realms*. Realms are defined by trust boundaries; no two realms need trust each other. ICING does not change the basic topology and peering model: today’s ASes map naturally to realms. However, the granularity of a realm is variable. For example, a host could be its own realm, and for deploying ICING, it may be useful to regard the current Internet as one realm.

To communicate with a destination, the sender identifies a sequence of realms—a *path*—between it and the destination. Before sending a packet along a path, a sender needs *consent* from each realm. (We take each endpoint to be its own realm; an alternative is described in [28].) To get a realm’s consent, a sender communicates with a general-purpose server separate from the realm’s forwarding hardware (a decomposition inspired by [8, 9, 16]). The sender proposes the path. In making its decision, the server can incorporate arbitrary factors besides the proposed path (billing relationships [32], authentication, etc.). Upon consent, the server issues a *proof-of-consent* (PoC) that authenticates the path.

A packet contains its path and cryptographic values that allow the forwarders to validate the path.

The forwarders in a realm share two keys: a symmetric key (the *PoC key*) and a public/private key pair (the *realm key*). The PoC key allows the entity that makes policy decisions on behalf of a realm to indirectly communicate those decisions to the realm’s forwarders: the PoC is a MAC of the path, using the PoC key. Because packets contain cryptographic values bound to PoCs, a forwarder can verify that its realm issued consent. Any machine that knows the PoC key can issue consent; it need not be located in the realm and can be a *delegate* of the realm. A realm’s name is its public realm key (as in [3, 27]); such self-certifying names [25] do not require a PKI.

**Challenges.** A salient challenge is how a realm earlier in the path can use its public realm key to prove to later realms that it processed a packet, thereby allowing those realms to verify that the packet followed its path. The natural solution, signing packets, would be prohibitive. Below, we describe how ICING meets this challenge. Of course, there are many other challenges and questions

that we do not have space to describe, including how a core realm can avoid issuing consent for every flow that it carries, how PoC retrieval works, how to avoid denial-of-PoC attacks [5], and how the sender learns of paths to propose in the first place. We describe how ICING solves these problems in [28].

### 2.2 Details

Our last mechanism principle requires that ICING enforce policy in a non-cooperative environment. To ensure that ICING is robust in scenarios of varying hostility, we require it to work under a strongly adversarial model of “non-cooperative”, given immediately below.

**Threat model.** We assume that some realms (end-hosts and providers) are controlled by attackers. Such *malicious* realms can deviate arbitrarily from our protocols, including sending arbitrary packets or flooding the links they have direct access to. We make no assumptions on how malicious realms are implemented (they may directly connect to one another and be controlled by a single attacker). Realms that obey the protocol we term *honest*. The protocols that we describe below concern the behavior of honest realms, in particular determining when they have carried or should carry a packet.

**Protocol.** We give a simplified description of the protocol here. This version, unlike the protocol described in [28], treats each realm as a single forwarder.

To uphold the policy principle and the first two mechanism principles, ICING must ensure that a packet transits an honest realm  $R$  only under the following conditions:

1. [Path Validity] The path  $P$  in the packet’s header was previously approved by  $R$ ; and
2. [Provenance Verification] The packet verifiably transited all honest realms before  $R$  in  $P$  and arrived from the realm just prior to  $R$ .

The two conditions do not explicitly constrain a packet’s trajectory *after*  $R$ . But taken together, they imply:

3. [Path Adherence] A packet forking off its valid path  $P$  by skipping an honest realm  $R_{\text{skip}}$  cannot traverse any honest realm that succeeds  $R_{\text{skip}}$  in  $P$ . (For example, a packet cannot skip a required deep packet inspector and appear valid.)

The third and fourth mechanism principles induce further requirements. The solution must not rely on a PKI, prior coordination among realms, or per-packet public key cryptography (which would induce an unacceptable performance penalty). The solution must be amenable to high-speed implementation, such as in forwarding hardware. We believe that the combination of the threat model and all of these requirements is a new technical problem. (For example, [6, 14, 26, 29, 30] assume central coordination, don’t enforce Path Validity, or aren’t amenable to high-speed hardware implementation.)

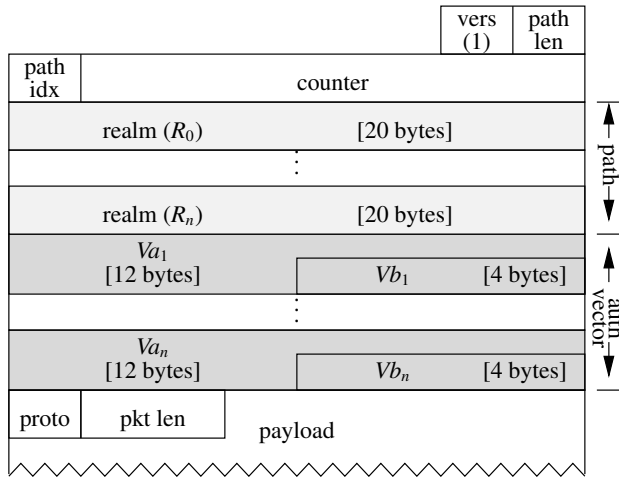


Figure 1—ICING packet format (to follow a 14-byte Ethernet header). The overhead of 36 bytes/realms may seem high, but we explain in §3 why we think that it is not outrageous.

Our high-level approach is: (1) name realms by public keys,  $R_i$ , that fit in packets, and (2) share symmetric keys between each pair of realms, deriving the keys from their names. Briefly, our realm names are points on NIST’s B-163 [2] binary-field elliptic curve group, which we reduce to 160 bits by requiring the top three bits to equal a hash of the lower 160. We use non-interactive Diffie-Hellman key exchange to give each pair of realms  $\langle R_i, R_j \rangle$  a symmetric key,  $k_{i,j}$ , that either realm can derive from the other’s name and its own private key. This provides roughly 80-bit security, comparable to that of 1,024-bit RSA keys [2]. Figures 1 and 2 depict the packet format and proposed protocol constructs.

Packet sending and forwarding follow the pseudocode in Figures 3 and 4. The source is assumed to have one PoC per realm in  $P$  (PoC retrieval happens via the not-discussed control plane). When a realm,  $R_i$ , receives a packet, it performs two steps. First, it checks that the packet took the correct path: it verifies that the  $i$ th entry in the auth vector is equal to the XOR of  $i + 1$  terms, the terms being a packet- and realm-specific authenticator ( $A_i$ ) and  $i$  applications of PRF-96 to the packet contents, one application each under  $k_{0,i}, \dots, k_{i-1,i}$ . Second, it provides proof for the later realms: for each of the remaining entries in the auth vector, it applies PRF-96 to the packet contents (using key  $k_{i,j}$  for the  $j$ th entry) and XORs the result into the given entry. The first time  $R_i$  encounters  $R_j$ , its forwarder must use slow path processing to derive  $k_{i,j}$ . The cost of deriving a  $k_{i,j}$  is a few msec in our experiments [28].

The full design [28] addresses PoC expiration, key management and revocation, and network failure.

### 3 Feasibility

We briefly discuss ICING’s feasibility here, concentrating on two aspects: forwarding speed and packet size. There

$P$	$\langle R_0, R_1, R_2, \dots, R_{n-1}, R_n \rangle$ . A packet’s path.
$M$	$\{\text{vers, cntr, proto, pkt-len, data}\}$ . Its end-to-end contents.
$R_i$	A public key which is also the realm name.
$x_i$	The private key of realm $R_i$ .
$s_i$	The symmetric PoC key used by $R_i$ ’s forwarders to verify packets.
$k_{i,j}$	Symmetric key shared by $R_i, R_j$ , derived through non-interactive Diffie-Hellman key exchange.
$\text{poc}_{P,i}$	$\text{PMAC}(s_i, P)$ . Proof of consent (PoC) to path $P$ by realm $R_i$ .
$V^i$	$\langle Va_1^i, Vb_1, Va_2^i, Vb_2, \dots, Va_n^i, Vb_n \rangle$ . Auth vector when pkt leaves $R_i$ ; lets downstream realms verify provenance.
$A_j$	$\text{PRF-96}(\text{poc}_{P,j}, 0^8 \parallel H(P, M))$ . For notational convenience, let $Va_j^{-1} = A_j$
$Va_j^i$	$\text{PRF-96}(k_{i,j}, i \parallel H(P, M)) \oplus Va_j^{i-1}$ . Proves to $R_j$ that packet has transited $P$ through $R_i$ . Unused if $i \geq j$ .
$Vb_j$	Last four bytes of $A_j$ . Guards forwarder slow path from being invoked spuriously.

Figure 2—Cryptographic values in ICING protocol. PRF-96 is a keyed pseudo-random function that maps 256-bit quantities to 96-bit quantities; it functions as a MAC. Our implementation of PRF-96 uses two applications of AES (details omitted).  $H(\cdot)$  is the bottom 248 bits of CHI-256( $\cdot$ ), a SHA-3 candidate [19].

are of course many other questions, including the required state in forwarders (for caching the  $k_{i,j}$ ), the CPU overhead for the slow path, etc.

**Forwarding speed.** Can forwarders execute the needed cryptographic operations at high speeds, at acceptable logic cost? Our preliminary investigations are promising. We have built a prototype using NetFPGA [1] for the fast path that runs at slightly less than 4 Gbps. For comparison, on this platform, IP forwarding runs at 4 Gbps and uses 43% less logic area than ICING. We expect a custom ASIC to be at least 10 times faster, achieving near-backbone speeds.

**Packet size.** ICING’s packet overhead is 36 bytes per realm so potentially hundreds of bytes per packet. To put this amount in context, we note that our goals in §1–§2 require *some* per-realm byte cost. The naive solution, signing logs in packets with digital signatures, would require roughly 128 bytes/realm and prohibitive processing cost; relative to this baseline, ICING’s overhead is a major improvement. Also, while ICING’s overhead is high for small packets, most bytes travel in large packets; thus, its average overhead (or its effect on total bandwidth consumed) tracks closely its overhead for large packets [28].

As further context, we are proposing ICING for the future. Thus, our present impression of its header’s cost is potentially irrelevant, thanks to technology trends.<sup>2</sup> Indeed, under jumbo frames, ICING’s overhead is negligible. And, hardware trends aside, ICING’s overhead may be an acceptable price for its properties.

<sup>2</sup>For example, research in TCP header compression is now obsolete.

---

```

function SENDPACKET( $P, \text{pocs}, m$ )
//  $P = \langle R_0, R_1, \dots, R_n \rangle$ 
//  $\text{pocs} = \{ \text{poc}_{P,i} = \text{PMAC}(s_i, P) \mid 1 \leq i \leq n \}$ 
//  $m = \{ \text{proto}, \text{pkt-len}, [\text{return path} + \text{PoCs}, ] \text{data} \}$ 
// to guard against replay attacks, init  $\text{cntr}$  per-flow
//  $M = \text{vers} \parallel \text{cntr} \parallel m$ 
for ( $i = 1 \dots n$ ) do
   $A_i = \text{PRF-96}(\text{poc}_{P,i}, 0^8 \parallel H(P, M))$ 
   $Va_i = \text{PRF-96}(k_{0,i}, 0^8 \parallel H(P, M)) \oplus A_i$ 
   $Vb_i = \text{last 4 bytes of } A_i$ 
 $V^0 = \langle Va_1, Vb_1, Va_2, Vb_2, \dots, Va_n, Vb_n \rangle$ 
 $\text{path-idx} = 1$ 
 $\text{pkt} = \text{vers} \parallel \text{path-len} \parallel \text{path-idx} \parallel \text{cntr} \parallel P \parallel V^0 \parallel m$ 
transmit  $\text{pkt}$  to  $R_1$  // may need intrarealm forwarding
cntr++

```

---

Figure 3—Pseudocode for packet construction.  $S = R_0$  constructs a packet to send payload  $m$  along path  $P$ . If the packet is the first in a flow,  $m$  may include a return path and PoCs. Note:  $P$  is 0-indexed while  $V^0$  is 1-indexed.

## 4 Examples

ICING, being designed for generality, can capture many policies that other architectures provide (see Table 1) as well as some new ones, including the following.

**Sink routing.** The literature on source routing is vast, yet almost no proposals (an exception being NIRA [38]) give receivers analogous control, even though they have the same interests as senders (as noted in §1). Thus, we propose *sink routing*, in which the *destination* chooses, or approves of, the entire interdomain path. ICING’s mechanisms naturally enable sink routing.

**Off-path middleboxes.** Under ICING, an end-host can direct traffic headed to it through off-path middleboxes, such as deep packet inspectors (DPIs). However, unlike in previous work (e.g., [33, 34]), the invocation can be *selective* and *enforced*. Thus, a destination domain could require traffic from unknown sources to go through a third-party DPI or DDoS mitigator (e.g., [31]) while permitting other traffic to travel directly to it.

**Exotic routing policies.** ICING allows a participant to approve a path based on any subset of the path. For example, a provider may wish to carry only traffic that has flowed through, or will flow through, a friendly country. Participants can also make decisions based on arbitrary factors, such as whether another entity on the path is a customer and has paid its bill in the last month, whether this customer is allowed to pass through during this time of day, or whether resources are available.

## 5 Summary and discussion

The original designers of BGP captured the policy considerations that were relevant in their day. But we live in different times, with different policy issues and different technology. Given this evolution of requirements

---

```

function RECEIVE( $\text{pkt}$ )
//  $\text{pkt} = \text{vers} \parallel \text{path-len} \parallel \text{path-idx} \parallel \text{cntr} \parallel P \parallel V^{i-1} \parallel m$ 
//  $M = \text{vers} \parallel \text{cntr} \parallel m$ 
   $\text{poc}_{P,i} = \text{PMAC}(s_i, P)$ 
   $A_i = \text{PRF-96}(\text{poc}_{P,i}, 0^8 \parallel H(P, M))$ 
// extract components in  $V^{i-1}$  that we need to verify
  let  $\langle Va_i^{i-1}, Vb_i \rangle = \text{the } i\text{th entry in } V^{i-1}$ 
// following line protects slow path from spurious calls
  check that  $Vb_i$  equals last 4 bytes of  $A_i$ : if not, drop
// following line may require slow path invocation
  compute  $k_{0,i}, k_{1,i}, \dots, k_{i-1,i}$ 
// simulate what earlier forwarders should have done to
// the  $i$ th component of the authorization vector
   $W = A_i$ 
for  $0 \leq j \leq i - 1$  do
   $W = \text{PRF-96}(k_{j,i}, j \parallel H(P, M)) \oplus W$ 
  check that  $W = Va_i^{i-1}$ : if not, drop
// following line may require slow path invocation
  compute  $k_{i,i+1}, \dots, k_{i,n}$ 
// construct  $V^i$ 
   $V^i = V^{i-1}$ 
for  $i + 1 \leq j \leq n$  do
   $Va_j^i = \text{PRF-96}(k_{i,j}, i \parallel H(P, M)) \oplus Va_j^{i-1}$ 
  increment  $\text{pkt.path-idx}$  to  $i + 1$ 
  transmit  $\text{pkt}$  to  $R_{i+1}$  // may need intrarealm fwding

```

---

Figure 4—Pseudocode for packet forwarding.  $R_i$  validates  $\text{pkt}$ , transforms  $V^{i-1}$  to  $V^i$ , and forwards  $\text{pkt}$  to the next realm. Note:  $P$  is 0-indexed while  $V^{i-1}$ ,  $V^i$  are 1-indexed.

and capabilities, we have sought the most general policy framework we thought reasonable and asked whether it could be implemented. To guide this inquiry, we articulated a policy principle and a set of mechanism principles, which led us to a design that, according to our back of the envelope estimates, is not completely infeasible.

For all but one of our principles, we showed (sometimes implicitly) how the design follows from or (mostly) upholds the principle. The one that we did not have space to address (because addressing it relies on delegation and an understanding of ICING’s control plane, two subjects outside our scope) is the fifth mechanism principle, namely that the participants should not have to use the general form of the mechanism if they don’t need its full expressive power. However, the full design of ICING [28] does uphold this principle. In short, under ICING, realms can sub-divide themselves into logical sub-realms, and then delegate control over these sub-realms to specified participants or to the public. Realms can thus disintermediate themselves and avoid approving every flow.

Of course, the opposite is possible too: ICING gives any stakeholder along the path the power to veto that communication, so it is possible that this power will lead to an Internet dystopia with no connectivity and no paths available. However, connectivity is a powerful driver, so not only does this pessimistic outcome seem unlikely

to us, but the policies that ICING makes possible seem *fairer*: ICING empowers senders and receivers, middle-boxes and providers, organizations and ISPs. We cannot predict the future, in terms of where the ultimate power will live, but at the very least ICING provides a neutral foundation on which this tussle [10] could play out.

## Acknowledgments

For constructive critiques of previous drafts, we are grateful to Dave Andersen, Hari Balakrishnan, Dan Boneh, Russ Cox, Mike Dahlin, Nick Feamster, Sanjam Garg, Mark Handley, Phil Levis, Nick McKeown, Guru Parulkar, Vitaly Shmatikov, Jessica Wilson, Emmett Witchel, Nikolai Zeldovich, and the anonymous reviewers. This work was supported by ONR grant N00014-09-10757, by NSF Cybertrust award CNS-0716806, by the Stanford Clean Slate program, and by Intel Corporation, whose gift to Brad Karp supported Walfish and Mazières while they visited Karp at UCL in Autumn 2008.

## References

- [1] NetFPGA: Programmable networking hardware. <http://netfpga.org>.
- [2] Digital signature standard (DSS). Federal Information Processing Standards Publication, November 2008. DRAFT FIPS PUB 186-3.
- [3] D. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker. Accountable Internet protocol. In *SIGCOMM*, Aug. 2008.
- [4] K. Argyraki and D. R. Cheriton. Loose source routing as a mechanism for traffic policies. In *SIGCOMM Wkshp. on Future Directions in Network Architecture*, Sept. 2004.
- [5] K. Argyraki and D. R. Cheriton. Network capabilities: The good, the bad and the ugly. In *HotNets*, Nov. 2005.
- [6] I. Avramopoulos, H. Kobayashi, R. Wang, and A. Krishnamurthy. Highly secure and efficient routing. In *INFOCOM*, Mar. 2004.
- [7] H. Ballani, Y. Chawathe, S. Ratnasamy, T. Roscoe, and S. Shenker. Off by default! In *HotNets*, Nov. 2005.
- [8] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. van der Merwe. Design and implementation of a routing control platform. In *NSDI*, May 2005.
- [9] M. Casado, M. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker. Ethane: Taking control of the enterprise. In *SIGCOMM*, Aug. 2007.
- [10] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden. Tussle in cyberspace: defining tomorrow's Internet. In *SIGCOMM*, Aug. 2002.
- [11] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, Dec. 1959.
- [12] D. Estrin, T. Li, Y. Rekhter, K. Varadhan, and D. Zappala. Source demand routing: Packet format and forwarding specification (version 1). RFC 1940, May 1996.
- [13] D. Estrin, J. Mogul, and G. Tsudik. VISA protocols for controlling inter-organizational datagram flow. *IEEE JSAC*, 7(4), May 1989.
- [14] D. Estrin and G. Tsudik. Security issues in policy routing. In *Proc. IEEE Symposium on Security and Privacy*, May 1989.
- [15] P. B. Godfrey, I. Ganichev, S. Shenker, and I. Stoica. Pathlet routing. In *SIGCOMM*, Aug. 2009.
- [16] A. Greenberg, G. Hjalmtysson, D. A. Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang. A clean slate 4D approach to network control and management. *ACM CCR*, 35(5), Oct. 2005.
- [17] S. Guha and P. Francis. An end-middle-end approach to connection establishment. In *SIGCOMM*, Aug. 2007.
- [18] K. P. Gummadi, H. V. Madhyastha, S. D. Gribble, H. M. Levy, and D. Wetherall. Improving the reliability of Internet paths with one-hop source routing. In *OSDI*, Dec. 2004.
- [19] P. Hawkes and C. McDonald. Submission to the SHA-3 competition: The CHI family of cryptographic hash algorithms. Submission to NIST, 2008. [http://ehash.iaik.tugraz.at/uploads/2/2c/Chi\\_submission.pdf](http://ehash.iaik.tugraz.at/uploads/2/2c/Chi_submission.pdf).
- [20] H. T. Kaur, A. Weiss, S. Kanwar, S. Kalyanaraman, and A. Gandhi. BANANAS: An evolutionary framework for explicit and multipath routing in the internet. In *SIGCOMM Wkshp. on Future Directions in Network Architecture*, Aug. 2004.
- [21] 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 *SIGCOMM*, Aug. 2007.
- [22] M. Little. Goals and functional requirements for inter-autonomous system routing. RFC 1126, October 1989.
- [23] X. Liu, X. Yang, and Y. Lu. To filter or to authorize: Network-layer DoS defense against multimillion-node botnets. In *SIGCOMM*, Aug. 2008.
- [24] R. Mahajan, D. Wetherall, and T. Anderson. Mutually controlled routing with independent ISPs. In *NSDI*, Apr. 2007.
- [25] D. Mazières, M. Kaminsky, M. F. Kaashoek, and E. Witchel. Separating key management from file system security. In *SOSP*, Dec. 1999.
- [26] A. T. Mizrak, Y.-C. Cheng, K. Marzullo, and S. Savage. Fatih: Detecting and isolating malicious routers. In *IEEE DSN*, June 2005.
- [27] R. Moskowitz and P. Nikander. Host identity protocol (HIP) architecture. RFC 4423, May 2006.
- [28] J. Naous, A. Seehra, M. Walfish, D. Mazières, A. Nicolosi, and S. Shenker. The design and implementation of a policy framework for the future Internet. Technical Report TR-09-28, Department of Computer Science, The University of Texas at Austin, Sept. 2009. <http://www.cs.utexas.edu/~mwalfish/icing-tr-09-28.pdf>.
- [29] R. Perlman. *Network layer protocols with Byzantine robustness*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1988.
- [30] R. Perlman. Routing with byzantine robustness. Technical Report TR-2005-146, Sun Microsystems, Aug. 2005.
- [31] Prolexic Technologies, Inc. <http://www.prolexic.com>.
- [32] B. Raghavan and A. C. Snoeren. A system for authenticated policy-compliant routing. In *SIGCOMM*, Sept. 2004.
- [33] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet Indirection Infrastructure. In *SIGCOMM*, Aug. 2002.
- [34] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, and S. Shenker. Middleboxes no longer considered harmful. In *OSDI*, Dec. 2004.
- [35] W. Xu and J. Rexford. MIRO: Multi-path interdomain routing. In *SIGCOMM*, Sept. 2006.
- [36] A. Yaar, A. Perrig, and D. Song. SIFF: A stateless Internet flow filter to mitigate DDoS flooding attacks. In *Proc. IEEE Symposium on Security and Privacy*, May 2004.
- [37] A. Yaar, A. Perrig, and D. Song. StackPi: New packet marking and filtering mechanisms for DDoS and IP spoofing defense. *IEEE JSAC*, 24(10):1853–1863, Oct. 2006.
- [38] X. Yang, D. Clark, and A. W. Berger. NIRA: A new inter-domain routing architecture. *ACM/IEEE Transactions on Networking*, 15(4), Aug. 2007.
- [39] X. Yang and D. Wetherall. Source selectable path diversity via routing deflections. In *SIGCOMM*, Sept. 2006.
- [40] X. Yang, D. Wetherall, and T. Anderson. A DoS-limiting network architecture. In *SIGCOMM*, Aug. 2005.