A Connection Oriented Internet Architecture for Restricting Reachability

Sneha Kasera
School of Computing, University of Utah
Salt Lake City, UT 84112

Abstract— **To aid security in the Internet, we propose a new connection oriented architecture to restrict reachability in the Internet to only those end hosts that explicitly request it. We first describe the various components of our architecture. Next, using qualitative arguments, and some preliminary computations, we show the benefits of our architecture. We also present viable strategies for minimizing connection state at routers, and discuss relevant security issues.**

I. Introduction

The current Internet has served us well. Its philosophy of placing intelligence in the end hosts and keeping the network simple has resulted in the quick development and deployment of a wide range of services and applications. However, the fact that a wide range of applications and services, sometimes critical, are dependent on the Internet has made it an attractive target for malicious attacks. The ease of deployment of new applications without much scrutiny, lack of an understanding of security problems among common users, and complexity of the security solutions, has not helped either. Attackers have been able to successfully flood end hosts, both end users and servers, with unwanted packets that claim resources at end hosts as well as those along the path causing denial-of-service. Attackers have also managed to find ways to infest end hosts with digital pests including worms and viruses. Some of these self-propagating worms could reach millions of Internet hosts in minutes [1]. It is anticipated that in the future these worms will be able to cover a big part of the Internet in seconds [2]. Worm propagation not only affects the end systems but have also been shown to affect the routing stability of the Internet [3]. End host system vulnerabilities are to some extent responsible for propagation of worms. However, we believe that even with advances in end system architectures and software, it might not be possible to remove all end system vulnerabilities. Hence it is not enough to just focus on end system architectures; we must also re-examine the Internet service model and design to fully address this issue.

The current Internet allows all nodes with public locators (IP addresses) to be reached by all the other Internet nodes. The Internet routing protocols exchange information on how to reach all the public IP addresses and create forwarding tables for efficient forwarding of data from one IP address to another. If a node (or a network interface of a node) has a public locator, the routing protocols are expected to find routes to this node from any part of the network. Although this appears to be a great feature for providing unhindered communication between any two nodes with public IP addresses, we believe that this is also the basic weakness of the current Internet service model. The network nodes are vulnerable to receiving unwanted traffic and intrusions exploiting any security vulnerabilities of these nodes. The network nodes could potentially use additional protection mechanisms (e.g. firewalls) but these mechanisms have been successful in preventing attacks only in a limited way.

To aid security in the Internet, we are proposing a fundamental change in the Internet service model and are building a new connection oriented Internet architecture to implement the change in the service model. Our idea is to restrict reachability in the Internet to only those end hosts that explicitly request it. Nodes that provide services need to advertise their presence and hence should be globally reachable. However, reachability to nodes requesting service should be restricted to the serving nodes and that too, only for the duration of the service. We are building an architecture that allows for such selective and restrictive reachability. At a high level, in our architecture, service requesting nodes do not have any IP addresses, public or private. They join the network only to obtain services by setting up connections to the serving nodes. The routers along the path between the service requesting node and the serving node maintain connection state to route packets between these two nodes. The serving nodes also need not have any IP addresses and can offer services through well-known *entry-points*. An entry-point is a router or any intermediate node that has a well-known public IP address. The serving nodes advertise their entry-points through directory services. The serving node sets up connection paths to all the entry points it advertises. When a client decides to obtain the services of a serving node, it sets up a connection path to one of the advertised entry-points, and through it communicates with the serving node. It must be noted that although entry-points do have a globally reachable IP address, a node must first set up a connection to communicate with an entry-point or to a serving node through the entry-point. This means that an attacker node cannot send data packets to an entry-point unless the entry-point accepts a connection from this node[1]. It is possible for an entry point to be co-located with the serving node[2]. Our architecture also provides peer-to-peer, mobile, and multicast

[1] However, a node could freely send connection requests towards an entry-point.

[2] In the current Internet architecture, entry points are essentially co-located with the serving nodes.

communication primitives.

The number of service requesting nodes that are actively *connected* to the network at any given time is likely to be orders of magnitude less than the number of *always on* nodes in the Internet. This implies that much fewer nodes will be vulnerable to DoS and worm propagation attacks in our architecture. Additionally, entry-points, possibly as instructed by the serving nodes, do not receive any data packets from nodes that they do not wish to communicate with, by rejecting connection requests from those nodes.

We would like to emphasize that we are not proposing new security solutions. Rather, we are proposing a new Internet architecture to aid Internet security.

II. Related Work

Anderson *et al* in [4] (and later in [5]) suggested using capabilities to give explicit permission to send data in the Internet. However, this work addresses only flooding attacks. It does not address the problems of non-flooding attacks that use slow-rate data to hack into a remote computer or use up battery resources of a mobile user by not allowing it to *sleep* etc. Our architecture makes service requesting nodes unreachable. However, our architecture can be supplemented with capabilities to prevent flooding attacks on the entry points. In [6], Handley *et al* proposed separating the service requesting node and the serving node address space, with non-global addresses for service requesting nodes, and allowing only explicitly requested traffic towards the service requesting node using the reverse path stored in the request packet. This approach does not store dynamic connection state at routers and hence uses static domain identities, encrypted or not, for storing reverse path in the packets themselves. The reverse paths are static and can be exploited by serving nodes to send unwanted traffic towards the service requesting nodes even after the service requesting node is done obtaining services from the serving node. In our approach, once the service requesting node is done obtaining services, the connection state in routers times out and even the serving nodes cannot reach the service requesting nodes. Handley's approach also introduces additional per packet overhead for appending and removing domain addresses.

Comparison with Off-by-Default: Ballani *et al* [7] proposed that routers should avoid keeping routing state for a node unless it requests explicitly. In the Ballani scheme, if a serving node wishes to be reachable, it spreads routing (and with it reachability) information through all the routers in the network and eventually all routers are able to route (or filter) packets to it. Though the concept of reachability in [7] is very similar to the one (and as fine grained) in our work, the Ballani scheme has the following drawbacks. First, a large number of messages are sent throughout the Internet to convey reachability information for every change. Second, there is a substantial time lag before the reachability change takes effect. This lag is especially crucial when a serving node under DoS attack, wishes to blacklist some nodes. In our architecture, this will be as simple as sending a control message to the entry points to add a filtering rule. Also in the Ballani scheme, a series of changes by a large number of close nodes can potentially send out routing flaps throughout the network. We believe that our architecture addresses the reachability problem in a more comprehensive and fundamental way with less overheads.

Comparison with i3: [8] suggested the use of Internet Indirection Infrastructure (i3) [9] as a DoS prevention mechanism by hiding the actual location of a serving node and instead relying on triggers which can be removed (or changed) in case of a DoS attack. i3's indirection is very similar to our use of rendezvous and entry points. However, i3 provides an overlay solution. It is not intended to be a network-layer architecture. Built on top of IP, i3 does not address the service requesting node reachability problem.

Comparison with SOS: In another overlay solution, SOS [10], important attack targets are protected by allowing traffic to them only through certain "secure" nodes. However, every node close to the attack target must have the capability to filter out attack packets. This is because even though the IP address of an attack target is not known publicly, it still exist and can be leaked out. Once the IP address of the attack target is known, it can be attacked. In our architecture, serving nodes do not have IP addresses and hence traffic cannot be sent towards them without going through the entry points. Only the entry points and not all possible neighbors of a serving node need to have the filtering capability. Moreover, SOS does not protect service requesting nodes. Our architecture is a more complete one in restricting reachability.

III. Architecture

In the current Internet, any node with a public IP address could be sent undesirable packets from any part of the Internet. It is left to the end hosts (or in some cases firewalls placed close to the end hosts) to drop any undesired packets. These packets are typically used to cause denial-of-service attacks on the end hosts or along the path towards the end hosts. They are also used for probing end hosts for vulnerabilities that could be exploited to attack them or to attack other nodes in the network through these end hosts.

The key question we address in this paper is how to ensure that nodes become a part of the network only when they desire service or offer services and not otherwise. In order to achieve this goal, we propose a virtual-circuit like architecture. We start off with a basic model and then build upon it incrementally adding more features later in this section. We consider the network to be made up of *service requesting nodes* that desire services from other nodes, *serving nodes* that provide services, and routers that facilitate the communication between the service requesting and the serving nodes.

A. A Basic Model - Securing the Clients

The components of the basic architecture are described below:

- Service requesting nodes do not have any public locators, permanent or temporary. Serving nodes have locators (IP addresses) that could be reached through global routing information. Since most of the nodes in the Internet are just requesting services from a set of servers, this would render a majority of the nodes in the Internet unreachable. This in itself protects these nodes from Denial-of-Service attacks[3]. Note that since the serving nodes have publicly known locators, they still are globally reachable (and hence not any more secure). However, we will relax this assumption on the serving nodes later in this section to make them secure as well.

- Service requesting nodes send connection request messages towards the serving Nodes. These requests create *soft* connection state at each router along the path from the service requesting node to the serving node. At each router, local identifiers (similar to the virtual circuit identifiers in ATM networks) are assigned to the *connection*. When the connection request reaches the serving node, it must decide whether it wants to accept the connection or not. If the serving node decides to accept the connection, it sends a *connection response* message along the reverse path. The reverse path towards the service request node is traversed using the state previously created by connection request. When the service request node receives the response from the serving node, it can start sending and receiving packets to and from the serving node.
- In order to enhance reliability, the connection request messages are sent periodically to the serving node, which also responds with a connection response. These messages help in dealing with router or link failures. When the service requesting node or the serving node or both are mobile, the soft state also helps creating new paths to current locations of the communicating nodes.

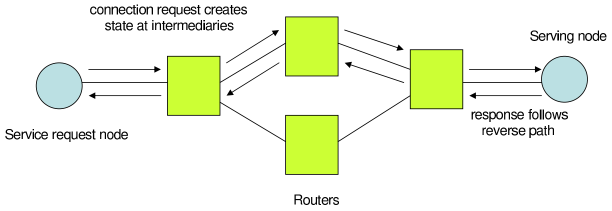

Fig. 1. Service Request Node to Serving Node Connection Setup

Figure 1 shows the connection setup between a Service Requesting Node and a Serving Node. With this architecture as a starting point, we shall augment it to make it more secure while offering richer functionality. Also, as shown in Figure 2, here on when we represent a link between two nodes with dashed and arrowed lines, it actually represents a connection established between the two nodes, which could physically be through any number of intermediate routers.

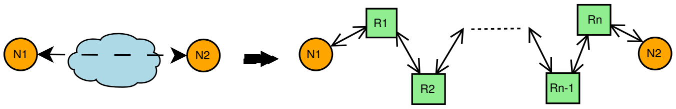

Fig. 2. Representation of a Connection

B. Securing the Serving Nodes

The virtual circuit-like architecture described above naturally protects the end hosts that request service. However, serving nodes are still vulnerable to receiving a large number of unwanted connection requests since their locations are known throughout the network. Protecting serving nodes from unwanted connection requests is a harder problem. This is because a serving node *must* advertise itself to offer services and network routers *must* be able to route packets to it.

To alleviate the threat on serving nodes, we suggest an approach in which each serving node identifies one or more *entry points*. A serving node sets up connections to each entry point and can be reached only through its entry points. Instead of advertising itself directly, a serving node advertises these entry points (through a directory service like DNS). A service requesting node, requiring service from the serving node, sets up a connection with one of these entry points and obtains the service through that entry point. Figure 3 shows the setup to facilitate service requesting node - serving node communication. Depending on the location of it clientele, performance requirements, security concerns, a serving node will be able to change its entry points. The serving node can do this by disconnecting from an old entry point, connecting to a new one, and updating the directory service entry. We note that and entry point could possibly be co-located with a serving node.

The entry points offer numerous other advantages. They are useful for preventing DoS attacks on the serving nodes. They act as a place holder for session control (described in Section III-F). Placement of entry points close to the service requesting nodes will also help in reducing connection state in the backbone as shown in Figure 3. We provide more details on these added advantages later in this section.

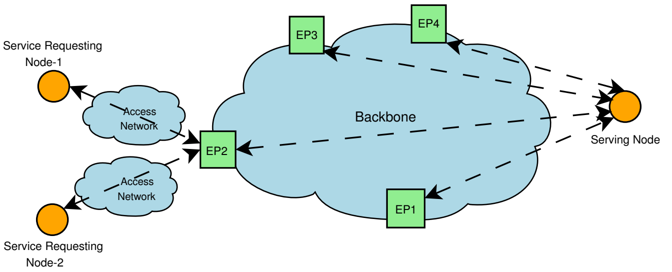

Fig. 3. **Protecting Servers Through Indirection:** The serving node has four Entry Points(EPs). Both service request nodes 1 and 2 choose EP-2 to communicate with the Serving Node. However, EP-2 maintains just one connection with the serving node. It multiplexes data from the two serving requesting nodes over this single connection thereby avoiding per flow state in the backbone.

[3] A significant fraction of all DDoS attacks are against home machines rather than well-known servers [11].

C. Identifiers

In our architecture, end hosts do not have any names of global significance. We identify end users rather than end hosts with names. We use a SIP-like [12] naming mechanism where users are assigned unique identifiers, for example, `alice@somedomain.net`, based on domain hierarchies. The entry points and routers (their network interfaces) have well defined globally known IP addresses. The connections between serving nodes and entry points are assigned identifiers that have local significance only.

D. Peer-to-peer Communication

The current Internet is experiencing a large amount of peer-to-peer traffic and future projections show that this trend will continue in the future. Hence peer-to-peer communication must be a first class citizen in any new Internet architecture. The current Internet has become particularly unsuitable for P2P communication due to the presence of NAT (network address translation) which has split the Internet into one public and many private networks. Numerous techniques (e.g., STUN [13], TURN [14], ICE [15], etc.) have been developed to enable communication in the presence of NATs. The very existence of so many techniques shows that these solutions are more of an afterthought rather than carefully designed ideas. Since the network does not provide the basic infrastructure (STUN server, for example), it is upto the application developer to worry about providing the infrastructure. For example, Skype has its own protocol and infrastructure for bypassing NATs, so the user does not have to worry about the presence of a NAT box. BitTorrent, on the other hand, leaves it to the user to open up certain ports on the NAT box. There is an obvious dichotomy between the *haves* and the *have nots* that has made creating P2P applications not as straightforward as creating client-server applications.

To enable peer-to-peer communication without compromising on security, we use a rendezvous based mechanism for peers to meet each other. When a user decides to be available to other peers, it registers itself at one or more Rendezvous Points (RPs) by establishing connections to the RPs. The location of the rendezvous points are made available to the rest of network users through a registrar directory service. Other peers interested in communicating with this user find out the location of the RPs from the registrar of the user. They then establish connections with an RP and through it, communicate with the user. The RPs are also be used for providing a variety of services including data filtering, user specific firewalls, and caching.

E. Mobility and Multicast Communication

Our architecture supports mobility naturally. First, mobility during a connection is handled by the soft state mechanism. The soft state mechanism creates state along the new paths. State along the old paths expires if not refreshed within a given time. Some packets are likely to be lost while the connection state is established along a new path. These lost packets must be recovered with the help of the transport layer protocol(s) if desired by the applications.

Second, since we do not assign any IP addresses to end hosts there is no confusion between the node's location in a physical network and its IP address. However, if mobile users wish to be reached by others in the network, they must set up connections to one or more rendezvous points and register with their registrar as described above.

Our architecture is also tailor made for supporting source specific multicast [16] communication, where multicast receivers send join messages towards the source creating state in the routers along the path to the source.

F. Benefits

Having defined the architecture, we shall now enumerate its benefits.

Restricted Reachability: Since none of the service requesting nodes have public locators, there is no way for any node in the network to reach them when they are not active. When the service nodes are active, only the nodes along the paths from the service requesting nodes to the serving nodes have state to route packets to the service requesting nodes. However, since this state uses local identifiers it is not possible for a node somewhere else to send packets to the service requesting nodes. Such restricted reachability provides DoS protection.

Since servers usually have much more at stake than a client in the event of a DoS attack, it is equally, if not more important, that they also be protected. In our architecture even the serving nodes do not have publicly known locators. They choose a certain number of entry points, establish connections with them, and advertise the locations of these entry points instead of themselves. This protects them in two ways. First, it provides them with *anonymity of location*. Since no node in the network knows the location of the serving node, no one can send any packet directly to it - packets could be sent only through the entry points. Thus direct flooding of the serving node is prevented. It is also important to note that even the entry points do not know the location of the serving, since it is the server which establishes a connection with them (and not the other way around). Hence even a compromised entry point, that does not have a connection with the serving node, will not be able to harm the serving node when it does not have a connection path to the serving node. Our architecture also provides implicit load balancing for the serving node using the entry points. Even if one (or a few) of the entry points is overloaded (or attacked), the other entry points could share the load. In the current Internet, there is effectively only one entry point per serving node - the upstream access router.

The entry points of serving nodes also provide an ideal infrastructure for the placement of a distributed firewall. The serving node could distribute reachability constraints to these entry points which would in turn filter accesses to the serving node. These reachability constraints could be as specific or generic as the serving node wants them to be. They could be as simple as a list of ports allowed by the server. Or, they could be as complex as providing a blacklist (and/or white-list) of node identifiers denied (or allowed) access to the serving node. Obviously there is a tradeoff between level of security and performance which is upto the serving node to decide. When entry points are empowered to refuse connection requests that do not meet established criteria (e.g., requests that cannot be authenticated), they essentially prevent establishment of undesired data paths and hence prevent undesired data packets from reaching them.

The rendezvous points can also be overloaded with the filtering functionality securing the service requesting nodes further in the case of a peer-to-peer communication.

Session Level Control: The current Internet does not implement any connection level or session level control to drop connections. When a router is congested due to traffic from a large number of TCP connections, it drops packets but the TCP connections are not stopped from continual use of the router link although they receive a very low data rate. We also expect the Internet to be used for a large number of voice over IP calls in the future. These calls require a minimum data rate for any useful voice communication. This requirement cannot be fulfilled without controlling the number of connections serviced at congested routers. We conduct ns-2 simulations to study the benefits of session control. Figure 4 shows how the number of completed file transfer sessions in a 50 second time interval changes with the rate at which new sessions are admitted at a 10 Mbps link. The file sizes are Pareto distributed with mean size equal to 10000 bytes and shape parameter 1.5. New sessions arrive at a mean rate of 5000 sessions per second. As shown in this figure, there is an optimal rate at which the sessions must be admitted. The overall system performance reduces considerably for higher or lower admission rates. We believe that the entry points/routers must implement a *session control* mechanism to control the number of connections that are allowed to use a link especially when the link utilization is very high and number of connections is large. Session level control is harder for non-TCP based applications that do not use any explicit connection setup because routers must now identify the first and last messages of different applications. Moreover, if end-to-end security is deployed using IPsec, even the TCP headers are not visible to the routers. *Explicit connection setup such as the one we propose, overcomes these problems.*

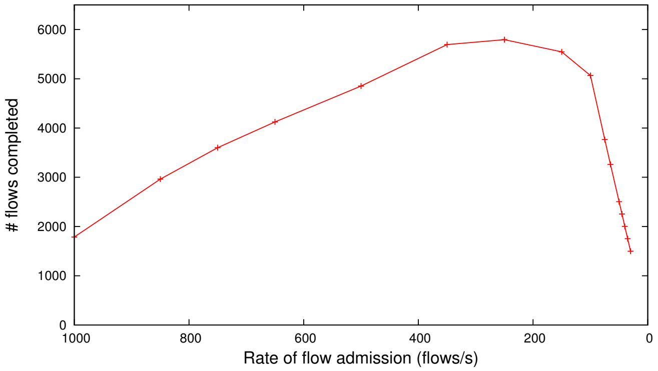

Fig. 4. Benefits of Session Level Control

Other advantages: As explained earlier, our architecture provides support for peer-to-peer, multicast, and mobile communication. Furthermore, our architecture also offers some additional benefits. First, since our architecture is based on forwarding using local identifiers which are smaller than IP addresses, we expect to obtain faster forwarding or switching. Second, as is the case with ATM or label switching networks, our architecture provides a platform to provide quality of service guarantees. Third, we strongly believe that our architecture creates new economic opportunities for businesses, ISPs and application developers. Provision of entry points for serving nodes and rendezvous points for service Requesting nodes will be an important revenue earner for many businesses. Furthermore, additional service will be offered at these nodes (e.g., rate limiting of connections, firewall protection etc.). This, we believe, will help resolve the tussle, as described in [17] between the end nodes and the service providers.

IV. Connection State Management

The success of our architecture is dependent on how well the routers can manage state related to connections. We adopt the following strategies to reduce connection state at routers:

Flow Aggregation: As we move from the edge to the core of the Internet, we expect the connection state to grow very fast. Figure 5 shows how the connection state is expected to change as we move from the source edge network towards a destination edge network. The connection state is expected to be the maximum at the intermediate networks between the edge networks. We would like to aggregate connection state, as much as possible, in the intermediate networks. For this purpose, we propose to build virtual links through the intermediate networks. A virtual link is a virtual circuit connection that is established between routers. These virtual links can be built statically on pre-determined paths across the intermediate networks. They could also be created dynamically across commonly used paths. Thus by creating a hierarchy of virtual circuits we can significantly reduce the state in the intermediate networks. We examine the UUNET backbone network to obtain an estimate of the the number of virtual links required across it to minimize its connection state. We choose UUNET because it has a large number of neighboring Internet service provider (ISP) networks. In [18], the number of neighboring ISPs of UUNET is 2569. This number suggests that a border router will require to establish 2569 virtual links with other border routers to allow connection state aggregation from one UUNET neighbor to another. Thus the backbone routers do not need to establish end-to-end connection state for each flow passing through them. Next, we consider one of

UUNET's neighbors, TWTelecom, that has 277 neighboring ISPs. Then the number of virtual links from all of UUNET's neighbors, other than TWTelecom, to all the neighbors of TWTelecom, will be approximately 2569*277 (≈ 720K). This number is fairly large and suggests that these virtual links should be established dynamically, rather than statically, on a need basis. Currently, we are developing algorithms for finding common paths and for setting up virtual links, both statically and dynamically, in the Internet. These algorithms will be evaluated using Internet topology data obtained from BGP tables.

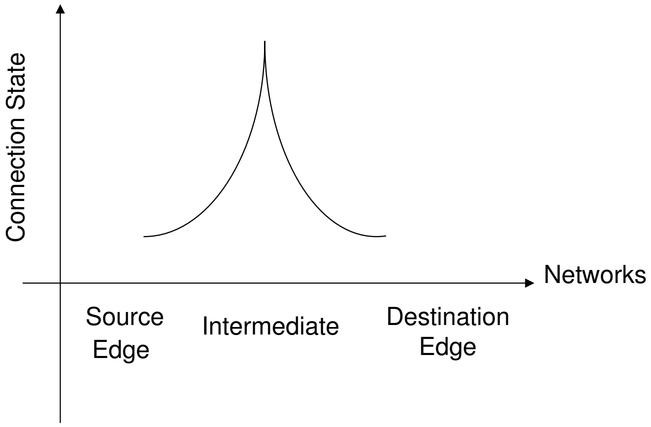

Fig. 5. Connection State vs Network Location.

Place/Invoke Entry Points Points at Strategic Locations for State Reduction: If the entry points for a server are located in different peripheries of the Internet, since there is just one connection between each entry point and the server, the state needed to be maintained by the core routers would reduce by orders of magnitude, making this architecture very scalable. Referring to Figure 3, we can see that the entry point multiplexes data from service requesting node 1 and the service requesting node 2 over this single connection thereby avoiding per flow state in the backbone network. Furthermore, these entry points can be arranged in a hierarchy so that the load is evenly distributed amongst all of them.

V. Additional Security Considerations

Our architecture provides a very powerful capability in the network to restrict reachability to end hosts. However, in order to ensure that only authorized users use the network and that they do not create denial-of-service attacks we include the following additional steps in our architecture development.

- The end hosts and first hop routers must mutually authenticate each other using a challenge and response mechanism (similar to the one used in [19]). This mutual authentication is also used to support user mobility and appropriate network access/usage billing.
- All connection signaling traffic must be authenticated and integrity protected hop-by-hop and at communicating end-hosts. Such mechanisms will not only minimize connection setup attempts but also reduce attempts to maliciously remove established connection state.

In addition to protecting nodes from incoming connections and packet traffic a secure network architecture must also prevent malicious traffic from leaving an end host. Malicious software implanted on end hosts could successfully hijack authenticated sessions, or even falsely authenticate themselves and generate traffic with malicious intent. Even though our architecture makes it hard for such digital pests to contact inactive end hosts or servers, it cannot prevent transmission of malicious traffic from the end hosts when these end hosts are apparently legitimately authenticated. This problem must be addressed through comprehensive security measures on end host operating environments potentially involving additional user involvement. However, development of a good solution for this problem is beyond the scope of this paper.

VI. Conclusions

We presented a new connection oriented architecture to restrict reachability in the Internet. Although the detailed design and evaluation of our architecture are still work-in-progress, we believe that our architecture is viable and necessary for Internet security.

References

[1] D. Moore et al, "Inside the slammer worm," *IEEE Security and Privacy*, vol. 1, no. 4, pp. 33–39, 2003.

[2] S. Staniford et al, "The Top Speed of Flash Worms," in *Proceedings of the ACM Workshop on Rapid Malcode (WORM)*, October 2004.

[3] J. Cowie et al, "Global Routing Instabilities during Code Red II and Nimda Worm Propagation," http://www.renesys.com/projects/bgp_instability, September 2001.

[4] T. Anderson et al, "Preventing internet denial-of-service with capabilities," *SIGCOMM CCR*, vol. 34, no. 1, pp. 39–44, 2004.

[5] X. Yang, D. Wetherall, and T. Anderson, "A DoS-limiting network architecture," in *ACM SIGCOMM*, September, 2005.

[6] M. Handley and A. Greenhalgh, "Steps towards a DoS-resistant internet architecture," *FDNA '04*, August 2004.

[7] H. Ballani et al, "Off by Default!" in *HOTNETS*, 2005.

[8] K. Lakshminarayanan et al, "Taming IP packet flooding attacks," *SIGCOMM CCR*, vol. 34, no. 1, 2004.

[9] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, "Internet indirection infrastructure," in *ACM Sigcomm*, August 2002.

[10] A. Keromytis et al, "Sos: secure overlay services," in *SIGCOMM*, 2002.

[11] D. Moore et al, "Inferring Internet Denial-of-Service activity," in *USENIX Security Symposium*, Aug 2001.

[12] J. Rosenberg et al, "SIP: Session Initiation Protocol," IETF, RFC 2543, June 2002.

[13] ——, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NAT)," IETF, RFC 3489, March 2003.

[14] J. Rosenberg, R. Mahy, and C. Huitema, "Traversal Using Relay NAT (TURN)," IETF," Internet-Draft, September 2005.

[15] J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for the Session Initiation Protocol (SIP)," IETF," Internet-Draft, August 2002.

[16] S. Bhattacharyya, "An Overview of Source-Specific Multicast (SSM)," IETF, RFC 3569, July 2003.

[17] D. Clark, J. Wroclawski, K. Sollins, and R. Braden, "Tussle in cyberspace: Defining tomorrow's internet," in *Proceedings of ACM SICOMM Conference*, Aug. 2002.

[18] N. Spring et al, "Quantifying the causes of path inflation," in *ACM Sigcomm*, August 2005.

[19] M. Buddhikot et al, "Integration of 802.11 and third generation wireless data networks," in *INFOCOM*, April 2003.