Practical Intrusion-Tolerant Networks

Daniel Obenshain*, Thomas Tantillo*, Amy Babay*, John Schultz‡§,
Andrew Newell†, Md. Endadul Hoque†, Yair Amir*‡§, and Cristina Nita-Rotaru¶
*Johns Hopkins University — {dano, tantillo, babay, yairamir}@cs.jhu.edu
¶Northeastern University — {c.nitarotaru}@neu.edu
† Purdue University — {newella, mhoque}@cs.purdue.edu
‡ Spread Concepts LLC — {jschultz, yairamir}@spreadconcepts.com
§ LTN Global Communications — {jschultz, yairamir}@ltnglobal.com

Technical Report Draft CNDS-2016-2 - January 2016
Distributed Systems and Networks Lab - http://www.dsn.jhu.edu

***Abstract*—As the Internet becomes an important part of the infrastructure our society depends on, it is crucial to construct networks that are able to work even when part of the network is compromised. This paper presents the first practical intrusion-tolerant network service, targeting high-value applications such as monitoring and control of global clouds and management of critical infrastructure for the power grid. We use an overlay approach to leverage the existing IP infrastructure while providing the required resiliency and timeliness. Our solution overcomes malicious attacks and compromises in both the underlying network infrastructure and in the overlay itself. We deploy and evaluate the intrusion-tolerant overlay implementation on a global cloud spanning East Asia, North America, and Europe, and make it publicly available.**

I. Introduction

The Internet is becoming an important part of the infrastructure our society depends on, connecting the distributed systems that manage our financial systems, commercial applications, and important aspects of our social interactions. With critical infrastructure control systems for power, gas, and water moving to use IP networks as their communication infrastructure, and with malicious attacks becoming more prevalent and more sophisticated by the day, it is crucial to construct networks that are resilient to the point of intrusion tolerance, able to work even when part of the network is compromised.

In this paper, we present the first practical intrusion-tolerant network service. The service targets high-value applications that need to work at all times, even when part of the network is compromised or under sophisticated attack. Examples include monitoring and control of global clouds, management of critical infrastructure such as the power grid, and military systems such as national nuclear command and control.

Our solution leverages the existing IP network infrastructure, making it practical for deployment. However, native IP network infrastructure cannot provide the resiliency needed for an intrusion-tolerant network. A single IP network is susceptible to failures, attacks, and misconfigurations (malicious or benign) that can render the entire network unusable. Internet routing connecting multiple IP backbones is based on trust and therefore susceptible to routing attacks, such as BGP hijacking [1], [2]. Recent sophisticated DDoS attacks, such as Coremelt [3] and Crossfire [4], can target specific traffic flows and cause them to experience severely degraded quality of service while preventing the Internet from rerouting around the problem.

Our solution uses an overlay approach to leverage the existing IP network infrastructure while providing the required resiliency and timeliness. An overlay running on top of multiple IP networks can tolerate a complete failure of an underlying network and is not bound to Internet routing, allowing it to route around and overcome malicious attacks and compromises at the Internet routing infrastructure in a timely manner. However, the overlay must be constructed with care. By using well-placed overlay nodes, diverse ISP backbones, and multihoming at each overlay node, we can construct a resilient architecture with enough redundancy to prevent anything short of a complete simultaneous meltdown of multiple ISP backbones from interrupting the ability to deliver messages.

While an overlay approach to a resilient networking architecture overcomes attacks and compromises in the underlying IP network infrastructure, the overlay itself is susceptible to compromises. A complete intrusion-tolerant network solution requires combining a resilient networking architecture with an intrusion-tolerant overlay. The remaining challenge and a major novelty of this work is the design and development of a practical intrusion-tolerant overlay that meets the needs of the targeted high-value applications.

We design and construct an intrusion-tolerant overlay that can tolerate arbitrary (i.e. Byzantine [5]) attacker actions, based on the key understanding that no overlay node should be trusted or given preference. We use a Maximal Topology with Minimal Weights, which specifies the overlay nodes, overlay links, and minimal weight allowed on each link, to limit network participation to authorized and authenticated overlay nodes and to prevent routing attacks at the overlay level (e.g. black hole [6] and wormhole [7]). We use source-based routing and redundant dissemination methods to limit the effect that a potentially compromised forwarder can have on the delivery of messages. Specifically, we protect against $K-1$ compromised nodes anywhere in the network by sending along K node-disjoint paths, or provide optimal delivery guarantees,

where messages are delivered as long as a correct path between source and destination exists, by using constrained flooding on the overlay topology. Finally, we prevent compromised nodes from consuming a disproportionate share of resources by enforcing fair network resource allocation at each overlay node.

As described so far, the intrusion-tolerant overlay provides best-effort message forwarding in the presence of compromises. However, high-value applications require messaging semantics stronger than simple forwarding. For example, cloud monitoring requires real-time delivery of a continuous stream of messages to produce an up-to-date picture. Since some monitoring messages convey more critical information than others, in the event of network contention, it is crucial to continue delivering the highest priority messages in real-time at the expense of low priority ones. In contrast, cloud control messages contain critical information that changes the state of the system and must be delivered reliably to maintain consistency.

Inspired by these requirements, we define two intrusion-tolerant messaging semantics: Priority Messaging with Source Fairness provides prioritized timely delivery and Reliable Messaging with Source-Destination Fairness provides reliable delivery. Many applications are served well by one or the other. We are currently investigating how to protect critical infrastructure control systems for the power grid and find that the semantics address several of the needs well. Some applications may require more complex guarantees (e.g. military command and control), which can be supported by creating additional semantics.

The intrusion-tolerant overlay is implemented and released as open source as part of the Spines [8] overlay messaging toolkit. We deploy the implementation on a global cloud that spans 12 data centers from East Asia to North America to Europe. We evaluate the overlay network in two ways. First, we send realistic traffic across the overlay network to evaluate its performance in the presence of compromised nodes. The overlay network ensures fairness and continues to provide the guaranteed semantics in the presence of compromised nodes. Second, we use the deployment as a shadow monitoring system to carry the monitoring messages of the global cloud, where it ran for several months and was used in a limited production capacity. The deployment was able to provide the same timely delivery of monitoring messages as the production monitoring network, validating that the intrusion-tolerant overlay can support high-value applications.

The contribution of this work is inventing the first practical solution to intrusion-tolerant networking. Specifically:

- We describe the resilient networking architecture necessary to support a practical intrusion-tolerant network service on a global scale.
- We describe the principles underlying our design and implementation of a practical intrusion-tolerant overlay: messaging semantics that guarantee fairness and performance in the presence of network compromises, a Maximal Topology with Minimal Weights, and source-based routing.
- We invent and implement two intrusion-tolerant messaging semantics: Priority Messaging with Source Fairness and Reliable Messaging with Source-Destination Fairness.
- We deploy and evaluate the intrusion-tolerant overlay implementation on a global cloud spanning 12 data centers from East Asia to North America to Europe, and present the results. The implementation is publicly available at www.spines.org.

II. Related Work

Prior work has investigated securing Internet routing protocols (surveyed by Papadimitratos et al. [9]), such as integrating security into BGP [10], protecting OSPF with digital signatures [11], and using Public Key Infrastructure and secret keys to authenticate routing updates in generic networks [6]. These works provide security against external attacks, but do not provide intrusion tolerance.

Several works created Byzantine gossip and peer-to-peer (P2P) protocols to disseminate information. Fireflies [12] provides an intrusion-tolerant gossip protocol to maintain full membership information in the presence of Byzantine members, which is used to support a distributed hash table. Castro et al. [13] provide secure node ID assignment, secure maintenance of routing tables, and secure message forwarding for a structured P2P network, assuming no more than a fraction of nodes are Byzantine. BAR Gossip [14] presents a P2P application on top of a Byzantine gossip protocol that provides predictable throughput and low latency for streaming media with high probability. In general, gossip and P2P protocols provide probabilistic message delivery, which is insufficient to support strong deterministic guarantees. In addition, these works assume the underlying network provides a clique of connectivity among the protocol participants, an assumption that can be violated by network compromises.

Other work has provided basic intrusion-tolerant messaging in limited network environments. Probing and flow conservation can be used to determine if routers are behaving maliciously [15], [16], but this places a limit on the location and number of compromises, and assumes correct routing behavior can always be determined. INSENS [17] provides intrusion-tolerant routing in wireless sensor networks by leveraging wireless-specific properties, e.g. compromised nodes have a limited broadcast range.

Previous work investigated routing messages in the presence of Byzantine failures. LITON [18] protects overlay network communication using on-demand node-disjoint routes and HMACs. ODSBR [19] presents a source-based routing scheme that localizes faults to a specific link using disguised probing techniques and re-routes accordingly. Authenticated Adversarial Routing (AAR) [20] successfully routes messages if even one correct path exists between source and destination, however, the limitation of only a single flow and the large initialization overhead are barriers to practical deployment. These works address Byzantine forwarders, but not Byzantine sources. In addition, none of these works were deployed in practice and they focus solely on message delivery, as opposed to guaranteeing messaging semantics.

The SCION work [21] provides a method to protect routing, even in the presence of some compromised nodes. It does this

	LITON	ODSBR	AAR	SCION	Perlman	Our Work
Feasibly (i.e. Internet) Deployable	✓	✓	✓			✓
Protect against link-level tampering	✓	✓	✓	✓	✓	✓
Protect against a single ISP meltdown				✓		✓
Protect against sophisticated DDoS attack				✓		✓
Protect against BGP hijacking				✓		✓
Overcomes Byzantine Forwarders	✓	✓	✓	✓	✓	✓
Overcomes Byzantine Sources					✓	✓
Guarantees Semantics						✓

TABLE I
COMPARISON OF RELATED WORK

by allowing the source and destination to work together to select a path. However, as the SCION work is a "clean-slate" design, deploying it on the Internet is not feasible.

The work most closely related to ours is Perlman's [22], [23], which provides authenticated link state routing in the presence of Byzantine failures. It floods routing updates with source-specific buffers to provide fairness and proposes using node-disjoint paths for data. Perlman bounds the number of nodes in the network to address Sybil attacks, whereas we specify the Maximal Topology with Minimal Weights to prevent Sybil and routing attacks. The fundamental difference with our work is that Perlman only provides best-effort message forwarding at the network level within a single physical network. In contrast, our service guarantees well-defined semantics at the (overlay) network level and is able to leverage multiple underlying networks to provide the needed resiliency in a timely manner. Finally, since Perlman's work is in the context of a physical network, it has practical barriers to deployment.

The most relevant work is compared in Table I.

III. NETWORK AND THREAT MODEL

In this section we describe the network and threat model.

A. Network Model

The intrusion-tolerant network consists of intrusion-tolerant messaging protocols running on top of a resilient networking architecture, which uses an overlay network to leverage several underlying IP networks such as commercial ISP backbones.

The overlay network consists of overlay nodes and logical edges (i.e. overlay links), where each node can be a source that injects new messages, a forwarder, and a destination. Each overlay node has a set of neighbors, i.e. overlay nodes with which it can communicate directly on the overlay without intermediate overlay nodes.

Overlay network communication is authenticated using a Public Key Infrastructure (PKI), where the system administrator and each node in the overlay network has a public/private key pair and knows all the other public keys. The overlay network topology is known by all of the overlay nodes, and changes to the topology can be made by the system administrator.

B. Threat Model

A *correct node* is an overlay node that executes the network protocols faithfully. A *compromised node* is any overlay node that is not correct. Compromised nodes can exhibit arbitrary (Byzantine [5]) behavior. A compromised node has access to all of the private cryptographic material stored at that node. Compromises may be sophisticated and difficult to detect. Rather than detecting and evicting compromised nodes, we provide guarantees even in the presence of such compromises.

A *correct edge* is a logical edge between two overlay nodes that is able to pass messages freely in both directions. A *failed edge* is any logical edge that is not correct. Note that all possible causes of edge failures, including (but not limited to) underlying network link failures and congestion, injected loss, misconfigurations, OSPF attacks, DDoS attacks, BGP hijacking, and physical layer (router/switch) compromises, are covered by this model.

A *correct path* is one consisting of only correct nodes and correct edges.

Overlay node resources. A correct node has sufficient computational resources to keep up with processing incoming messages, but has bounded buffers for storing messages.

Attacker resources. Attackers can compromise overlay nodes and any components of the underlying IP networks. Attackers can have large amounts of network bandwidth, memory, and computation, such as those required by sophisticated large-scale DDoS attacks (e.g. Coremelt [3] and Crossfire [4]). However, we assume the attacker cannot break the cryptographic mechanisms used by our protocols.

We do not assume a specific fractional bound on the number of compromised nodes in the network. However, as a liveness condition there must exist a correct path from source to destination. If this liveness condition is not met, the system will remain correct, but will not be live for that source-destination pair (it may be live for other source-destination pairs).

IV. RESILIENT NETWORKING ARCHITECTURE

We use an overlay approach to build a resilient networking architecture that leverages existing IP network infrastructure while providing the resiliency and timeliness required for a practical intrusion-tolerant network that the Internet cannot natively provide.

A. Overlay Approach to Resilient Networking Architecture

The intrusion-tolerant network cannot be based on a single underlying IP network because that IP network would be susceptible to misconfigurations, attacks, and compromises that could render the entire network unusable. Therefore, it is necessary to use multiple IP networks, which in practice means using the Internet. The Internet is designed to route around problems, including those that affect an entire IP network.

However, Internet routing is based on trust (making it susceptible to routing attacks such as BGP hijacking) and is vulnerable to sophisticated DDoS attacks. For example, the Coremelt [3] and Crossfire [4] DDoS attacks can decimate the service of targeted traffic flows while completely preventing Internet reroutes from taking place, forcing these flows to continuously experience poor service or complete disconnections. Moreover, the tens of seconds to minutes of service interruption during Internet reroutes caused by benign connectivity faults is unacceptable for time-sensitive high-value applications such as cloud and critical infrastructure monitoring; a mechanism that provides faster (ideally near real-time) reroutes is required.

An alternative approach that uses the existing Internet infrastructure without being bound to Internet routing is to use an overlay. Overlay networks can sit on top of multiple underlying IP networks simultaneously and provide the ability to control (and quickly change) the path of messages through the middle of the network by sending them through intermediate overlay nodes. As a result, overlays can overcome failures that render entire underlying IP networks unusable and can overcome Internet routing attacks (e.g. Coremelt and Crossfire) by quickly rerouting messages on alternative paths rather than being forced to use the chosen end-to-end Internet path [24], [25]. In addition, the overlay can implement dissemination schemes that are not possible on the Internet, such as multiple node-disjoint paths. We use resilient overlay topology construction, diverse network providers, and multihoming to construct a networking architecture that can survive anything short of a complete simultaneous meltdown of multiple ISP backbones.

1) Resilient Overlay Topology Construction

To be resilient, the overlay network should contain redundancy: a source and destination should be able to communicate along multiple disjoint paths so that even if some paths fail, other paths can still deliver messages. However, the overlay must be constructed carefully to ensure that disjointness in the redundant overlay topology matches actual physical disjointness in the underlying network infrastructure. Otherwise, multiple overlay links may overlap at the physical level, increasing the risk that a single failure in the underlying network will affect multiple overlay links.

We address this concern by placing overlay nodes in strategic locations, i.e. well-provisioned data centers. ISPs invest strongly in a relatively small number of strategic data center locations by laying independent fiber connections between them. These are ideal locations for overlay nodes, as the links between different pairs of geographically-close data centers are likely to be disjoint at the physical network level.

We can leverage the data center investment and the available map of the backbones to design our overlay topology to follow, more or less, the underlying network topology. We create overlay links between overlay nodes that are directly connected or only a few hops apart on the backbone. There are relatively few underlying network routing options between overlay nodes that are connected in this way, making it likely that messages sent between the two overlay nodes will follow the expected backbone path. This predictability allows the overlay topology to be designed with high likelihood that no overlay links overlap.

Note that overlay nodes are not connected as a clique. This would result in overlay links that correspond to many-hop connections on the backbone. The increased number of routing options for many-hop connections makes the path messages take less predictable, potentially leading to overlay links that overlap at the physical level.

2) Use of Diverse Network Providers

To use multiple underlying IP networks, we can use an approach similar to [26]. That work shows how to assign a small number of diverse software variants to nodes to maximize the expected client connectivity when each variant has some probability of failing completely. By considering different ISPs as the diverse variants, we can use these ideas to choose which single ISP each overlay node should contract with to maximize resilience to one or more ISPs suffering a complete meltdown.

3) Multihoming Using Diverse Network Providers

To further improve resilience, each overlay node can contract simultaneous service from multiple ISPs via multihoming, as shown in Figure 1. Using multihoming, an overlay link is correct as long as at least one combination of the available ISPs on each end of the link can pass messages. Normally, combinations that use the same ISP at both ends are likely to be more resilient, as they are not affected by BGP routing, but any combination can be used.

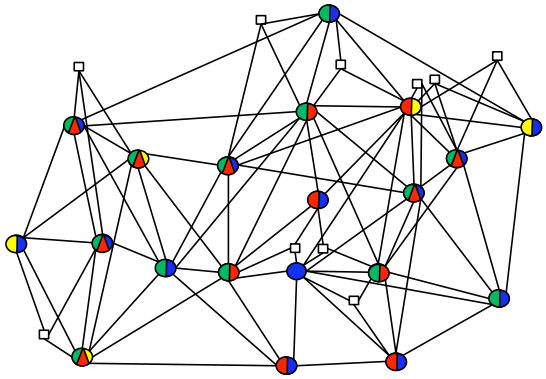

Fig. 1. An overlay network using multihoming. Each color represents a different ISP; a node with multiple colors simultaneously contracts service with multiple ISPs.

B. Attack Resilience

In the event of a BGP hijacking attack, traffic using Internet routes that cross multiple ISPs can be diverted to an attacker-specified destination, but traffic that stays within a single ISP is not affected. Therefore, overlay links that contract service from the same provider on both ends can still pass messages during the attack. The overlay can route messages across different ISPs without relying on BGP by switching between providers

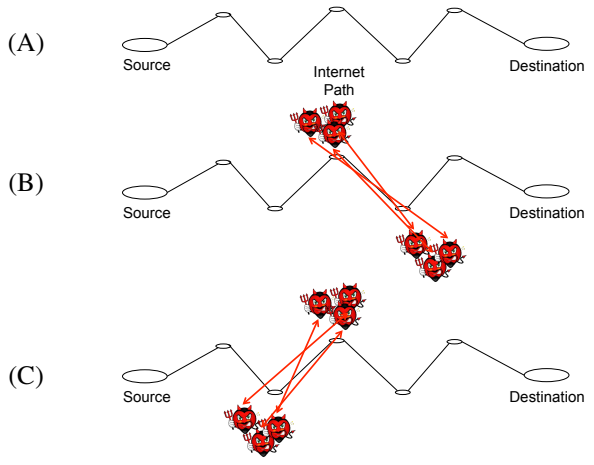

Fig. 2. An illustration of a Crossfire-style DDoS attack. By switching between (B) and (C) above, the attacker can cut off communication between source and destination, while simultaneously ensuring that the path will not be rerouted because no one link is unusable for long.

inside the overlay nodes: a node can receive a message using one provider and send it using a different provider.

The Crossfire [4] and Coremelt [3] DDoS attacks can force a targeted traffic flow or a targeted area to experience severely degraded quality of service or complete disconnection by overwhelming specific links on the Internet path used by the targeted flow or paths leaving or entering the targeted area. Normally, OSPF or BGP will eventually detect the problem on the link and route around it. However, the attack on the flow or area is made persistent by switching between different links on the same path: no one link is attacked for long enough to be detected, but the path as a whole is always unusable. This is illustrated in Figure 2.

In overlay networks, the Crossfire and Coremelt attacks can be used to attack the Internet paths corresponding to the overlay links connecting overlay nodes. However, our resilient networking architecture makes it very difficult for an attacker to cut off communication between a source and a destination. A successful attack must simultaneously affect multiple overlay links (enough to cut the overlay topology between source and destination), attacking each such overlay link on multiple ISPs (enough to cut any combination of ISPs available on that overlay link). This significantly raises the bar for the attacker.

V. Intrusion-Tolerant Overlay Network

A resilient networking architecture based on an overlay approach tolerates compromises and attacks in the underlying IP network infrastructure, but the overlay itself must also be resilient to compromises. We build an intrusion-tolerant overlay network using a Maximal Topology with Minimal Weights, redundant source-based dissemination methods, and intrusion-tolerant messaging protocols that guarantee well-defined semantics.

A. Maximal Topology with Minimal Weights

Each overlay node trusts an offline system administrator to initially distribute a signed Maximal Topology with Minimal Weights (MTMW). The MTMW specifies the overlay nodes and links in the network and the minimal weight allowed on each link. Weights can represent any real-world cost (e.g. latency) and routing decisions minimize weight. Overlay nodes only accept messages from their direct neighbors in the MTMW; non-neighbors communicate through intermediate nodes.

Overlay nodes monitor the links with their neighbors, raise and lower link weights when problems arise and resolve respectively, and disseminate signed routing updates.[1] A node is not allowed to change the weights of non-neighboring links or decrease the weight of any link below its minimal allowed weight. If a node attempts such an action, it is detected, that node is considered compromised, and that update is ignored. As a result, routing attacks (e.g. black hole [6] and wormhole [7]) are prevented, because routing updates which would otherwise have disproportionately attracted traffic towards the node that issued the update are disallowed and ignored.

To change the network topology, the offline system administrator can update, sign, and re-distribute the MTMW. Each MTMW is assigned a unique monotonically increasing sequence number to defeat replay attacks. The offline system administrator can be converted to an online Certificate Authority, if desired.

B. Redundant Source-based Dissemination Methods

We use redundant source-based dissemination methods to limit the effect that a potentially compromised forwarder can have on message delivery. We use K Node-Disjoint Paths and Constrained Flooding because they provide a good spectrum of cost-resiliency tradeoffs.

1) K Node-Disjoint Paths

In the K Node-Disjoint Paths dissemination method, each message is sent across the network K times, via K distinct paths, such that no two paths share any overlay nodes, other than the source and destination [27], [28]. The K separate paths are specified on the message at the source.

Guarantees. K Node-Disjoint Paths tolerates $K-1$ compromised nodes anywhere in the network and any number of failed edges, as long as there exist K node-disjoint paths across the topology after the failed edges are removed.

2) Constrained Flooding

In Constrained Flooding, in the worst case, each message is sent on every overlay link in the overlay network topology (not the entire underlying network). In practice, acknowledgments from the destination and feedback from neighboring overlay nodes prevent the message from being sent to nodes that are already known to have received it, reducing overhead.

Guarantees. Constrained Flooding tolerates any number of compromised nodes and failed edges, as long as there exists a correct path from source to destination. This guarantee is optimal: if a correct path does not exist, a combination of compromised nodes and failed edges have cut the network and no dissemination method can succeed.

C. Intrusion-Tolerant Messaging Semantics

We present two specific intrusion-tolerant messaging semantics: one that provides prioritized timely delivery, and one that provides reliable delivery. These semantics ensure fairness and guarantee performance to the applications using them, even in the presence of compromised nodes.

1) Priority Messaging with Source Fairness

Priority Messaging with Source Fairness (Priority Messaging) is motivated by the real-time demands of monitoring systems. Priority Messaging ensures that each source's highest priority messages are always timely. Within timeliness and buffering constraints, Priority Messaging is as reliable as possible.

Because compromises may be impossible to detect, no source can be given preference over another. Resources must be allocated fairly across sources, and message priority must be considered independently for each source. If we compared priorities across sources, a compromised source could send

[1] We use rate-limiting and overtaken-by-event techniques to limit the impact of spurious routing updates from compromised nodes.

all of its messages with highest priority, potentially starving correct sources.

Protocol description. Each message is assigned a priority, expiration time, and monotonically increasing sequence number by its source and is digitally signed. Each node stores uniquely received messages, identifying and discarding duplicates based on sequence number. If message storage is full on a given outgoing link, an incoming message that needs to be sent on that link causes the oldest lowest-priority message from the source currently using the most storage on that link to be dropped, making room for the newer message. When sending messages on a given link, each active source is treated in a round-robin manner; for a particular source, the oldest highest-priority message from that source is sent. This approach enables full utilization of each link by dynamically allocating resources based on the number of active sources, i.e. the number of sources currently sending on the link.

Since Priority Messaging does not provide ordered delivery, a node must potentially store the sequence numbers of all the messages it has received to recognize duplicates. However, requiring an expiration time (with an enforced upper bound) on each message prevents this storage from growing without bound since expired messages can be discarded.[2]

Service guarantees. Formal specifications and proofs of Priority Messaging guarantees can be found in [30]. Informally, Priority Messaging delivers messages in a timely (near real-time) manner. In the absence of network contention, all messages are timely. In the presence of network contention, Priority Messaging maintains real-time delivery of the higher priority messages by potentially dropping some of the lower priority messages or delivering them with higher latency.

Priority Messaging guarantees fair storage and bandwidth allocation for each source on each outgoing link between two correct nodes; each active source receives either the resources (storage and bandwidth) it requests or its fair share $\left(\frac{total\_resources}{\#active\_sources}\right)$, whichever is smaller. If a source is using less than its fair share, the unused resources are evenly reapportioned among the other active sources that are requesting more than their fair share. Note that even if a correct node receives messages in an unfair way from a compromised neighbor, fairness is still maintained on the correct node's outgoing links.

2) Reliable Messaging with Source-Destination Fairness

Reliable Messaging with Source-Destination Fairness (Reliable Messaging) is motivated by the reliability demands of control messages. Reliable Messaging ensures that messages sent between a source and destination are delivered end-to-end reliably and in order. Within these reliability constraints, Reliable Messaging is as timely as possible.

Because messages must be kept until they are acknowledged by the destination, we cannot use source-based fairness and dynamic storage allocation (as in Priority Messaging). Using those approaches, a compromised destination could refuse to acknowledge messages to block sources from sending to other correct destinations. Therefore, Reliable Messaging must enforce fairness based on source-destination flows and must statically allocate storage across all potential flows, where a flow consists of all traffic between a source and a destination.

Protocol description. Each message is assigned a monotonically increasing sequence number and is digitally signed. Nodes accept and store messages with the next expected sequence number for each flow. Duplicates are easily recognized and discarded, defeating replay attacks. To provide end-to-end reliability, intermediate nodes maintain responsibility for messages until they are acknowledged by the destination. When storage for a particular flow fills, the node stops accepting new messages for that flow, creating *back-pressure* (all the way back to the source) that prevents new messages from entering the network for that flow. When sending messages on a given link, each active flow is treated in a round-robin manner; for a particular flow, the next expected message for that neighbor is sent. As in Priority Messaging, dynamic bandwidth allocation enables full bandwidth usage of each correct link.

Destinations acknowledge messages using *end-to-end (E2E) ACKs*. E2E ACKs are periodically generated (with a configurable E2E timeout), signed, and flooded back through the network by each destination. Each E2E ACK indicates the highest in-order sequence number received from every source node in the network. E2E ACKs allow intermediate nodes to discard acknowledged messages, making room for new ones and clearing back-pressure. These ACKs operate on an overtaken-by-event basis, with intermediate nodes only storing the latest one from each destination. To prevent compromised nodes from spamming E2E ACKs to consume network bandwidth, a correct node only forwards E2E ACKs that indicate progress, and forwards them no more often than the E2E timeout. The choice of E2E timeout presents a trade-off between overhead and responsiveness; longer timeouts preserve more bandwidth for data messages, but make the network take longer to clear back-pressure. As an optimization, nodes can also send *neighbor ACKs* that indicate which messages they have received, to prevent neighbors from sending unneeded messages, improving bandwidth usage.

Reliable Messaging provides reliability even when intermediate nodes crash and recover. Once a recovering node retrieves the latest E2E ACKs from a correct neighbor, it can resume correctly forwarding data messages. This ensures that messages will flow even when the only correct path from source to destination is an eventual path.

Service guarantees. Formal specifications and proofs of Reliable Messaging guarantees can be found in [30]. Informally, Reliable Messaging delivers messages end-to-end reliably and in order for each source-destination flow, even if there is only an eventual path between source and destination.

Reliable Messaging guarantees fair storage and bandwidth allocation for each flow on each outgoing link between two correct nodes. Each active flow receives either the bandwidth it requests or its dynamic fair share $\left(\frac{total\_bandwidth}{\#active\_flows}\right)$, whichever is smaller. For storage, each active flow receives its static fair share $\left(\frac{total\_storage}{\#total\_flows}\right)$. Note that, as with Priority Messaging, compromised neighbors cannot affect the fairness of a correct node's other correct outgoing links.

[2]This requires monotonically increasing clocks and some level of network synchronicity. Previous work has met these conditions with atomic clocks [29].

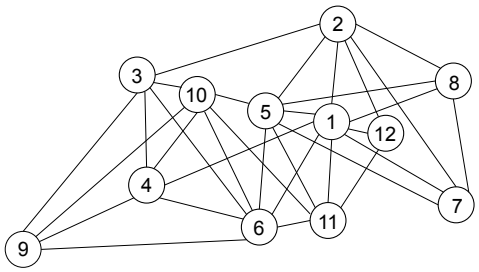

Fig. 3. Global cloud topology spanning East Asia, North America, and Europe

D. Implementation Considerations

Protocol selection. The intrusion-tolerant overlay allows the messaging semantics and dissemination method to be chosen on a message-by-message basis. Currently, the four options are: Priority K-Paths, Priority Flooding, Reliable K-Paths, and Reliable Flooding.

Proof-of-Receipt Link. Neighboring overlay nodes communicate using a Proof-of-Receipt (PoR) link that provides TCP-fair reliable in-order communication. The link maintains cryptographic authentication and integrity (similar to DTLS [31]), using an authenticated Diffie-Hellman [32] key exchange to establish a shared secret key for link-level message integrity. Each side of the link must acknowledge messages with a proof-of-receipt, using a cumulative nonce method [33], to defeat denial-of-service attacks that acknowledge unreceived messages to drive the sender arbitrarily fast.

Diversifying overlay node attack surface. Compiler-based diversity [34] and proactive recovery [35], [36] of overlay nodes diversify the attack surface of each overlay node across space and time, greatly reducing the chance that a single compromise can take down multiple overlay nodes at the same time.

Cryptographic mechanisms. We use the RSA [37], Diffie-Hellman and HMAC [38] (using SHA-256 [39]) implementations from the OpenSSL library [40].

Fuzz testing. We validated the implementation using the Turret platform [41], designed for automatically finding attacks. Turret revealed several bugs in message validation that allowed a compromised node to cause a correct node to crash via specific fields in a message (e.g. receiving an ACK for the maximum sequence number). To date, we have fixed all discovered vulnerabilities.

VI. Deployment and Evaluation

We deploy the intrusion-tolerant overlay implementation on a global cloud [42] (Figure 3) that spans 12 data centers from East Asia to North America to Europe. We do not report the specific latency on each edge for proprietary considerations. This topology contains sufficient redundancy to support at least three node-disjoint paths between any two nodes. We were authorized to use 500 Mbps continuously for a whole year. To ensure that we did not exceed the budget, we set the bandwidth capacity of each of the 32 links in the topology to be 10 Mbps.

We evaluate the overlay deployment's performance and overhead (Section VI-A) and resilience to attacks (Section VI-B) by sending realistic traffic similar to monitoring traffic patterns observed in the cloud, with most messages below 3500 bytes. In addition, we use the deployment as a shadow monitoring system to carry the monitoring messages of the cloud (Section VI-C). Note that all experimental results are obtained by running on the actual global network, except Table II and Figure 8, which are instead measured in a controlled laboratory environment.

	Priority (Mbps)			Reliable (Mbps)		
	Flood	**K=1**	**K=2**	**Flood**	**K=1**	**K=2**
(a)	125	480	425	125	395	395
(b)	45	85	80	40	85	80

TABLE II
MAXIMUM GOODPUT MEASURED WITH: (A) NO CRYPTOGRAPHY, (B) HMACS AND SIGNATURES.

Dissemination Method	**Avg. # Hops**	**Scaled Cost**	**Avg. Path Lat. (ms)**
K=1	1.9	1.0	41.4
K=2	4.4	2.3	43.5
K=3	6.6	3.5	46.6
Naïve Flooding	64.0	34.1	-
Engineered Flooding	32.0	17.0	-

TABLE III
ANALYTICAL COST OF SOURCE-BASED DISSEMINATION ON THE TOPOLOGY. SCALED COST IS AVG. # HOPS NORMALIZED BY COST OF K=1.

Protocol	**Avg. # Hops**	**Scaled Cost**
Priority Flooding	35.8	19.0
Reliable Flooding (w/o End-to-End ACKs)	31.3	16.7
Reliable Flooding	16.3	8.7

TABLE IV
MEASURED COST ON THE TOPOLOGY. SCALED COST IS AVG. # HOPS NORMALIZED BY COST OF K=1.

A. Performance and Overhead

We evaluate the performance and overhead of the intrusion-tolerant overlay in benign environments.

Cryptographic impact on performance. Table II shows the maximum performance for one active flow in the network, obtained in a controlled laboratory environment matching the topology of Figure 3. In (a), cryptographic mechanisms are disabled. In (b), they are enabled. Since performance is strictly CPU bound when using cryptography, adding additional hardware by sharding the incoming traffic across multiple cores or even multiple machines would enable us to reach performance comparable with (a) in Table II.

Communication cost. The communication cost of the intrusion-tolerant protocols is topology-dependent. Tables III and IV show the analytical and experimental cost of source-based dissemination methods and messaging semantics on the cloud topology. The cost associated with sending messages corresponds to the number of edges those messages traverse.

Analytical comparison: In Table III, we report the average cost over all source-destination pairs for each dissemination method. For K-Paths with K=1, equivalent to single-path routing, the average number of hops between two nodes is 1.9 and the average path latency is 41.4 ms. For K=2 (resp. 3), the total number of hops across the paths increases by more than double (resp. triple). Since multiple shortest (latency-wise) paths do not exist, the average latency across the paths increases. Since the cost of Constrained Flooding depends on network message timing and acknowledgements in some of the messaging protocols (e.g. Reliable Messaging), we cannot calculate its analytical cost. Instead, we show the analytical

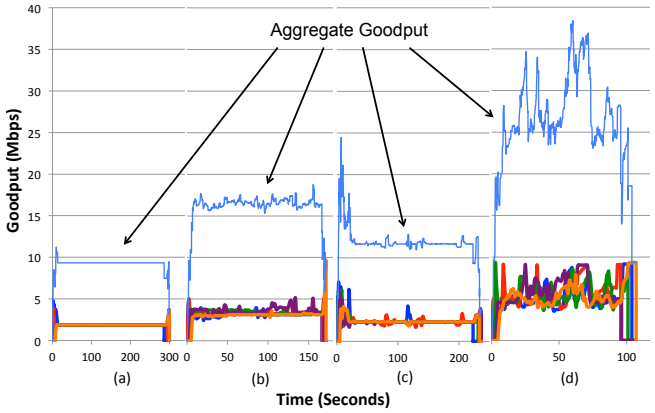

Fig. 4. Experimental goodput for: (a) Naïve Flooding, (b) Priority Flooding, (c) Reliable Flooding (no E2E ACKs), and (d) Reliable Flooding.

cost of Naïve Flooding, where messages traverse each edge in both directions, and Engineered Flooding, where techniques (such as random delay) are used so that messages traverse each edge only once. Since flooding schemes are not path-based, average path latency is not reported.

Experimental comparison: We measure and compare the costs of Priority Flooding, Reliable Flooding without E2E ACKs, and Reliable Flooding (Table IV). The experiments use five randomly selected flows (9-11, 4-5, 7-9, 1-10, and 3-8 in Figure 3) each sending at the maximum link capacity (10 Mbps) to create network contention. Since the experimental costs for K-Paths are very similar to their analytical costs, they are omitted. Note that Reliable Flooding without E2E ACKs is not a correct protocol. It is solely used to evaluate the cost benefits of neighbor ACKs.

The cost of Priority Flooding is between the costs of Naïve Flooding and Engineered Flooding because messages traverse some (but not all) edges in only one direction. Since timeliness is vital for Priority Messaging, a random delay is infeasible and only the natural latency of the network can prevent messages from flowing twice on a given edge. Note that the Priority Flooding cost includes messages that traverse part of the network but do not arrive at the destination due to contention.

In Reliable Flooding, neighbor ACKs eliminate the need to forward messages to neighbors that have already acknowledged them. E2E ACKS eliminate the need to forward messages that have already been acknowledged by the destination. Both types of ACKs can prevent messages from traversing every edge; with E2E ACKs, messages may not even need to reach every node in the network. The cost of Reliable Flooding without E2E ACKs is comparable to the cost of Engineered Flooding, showing the benefits of neighbor coordination. The cost of Reliable Flooding is significantly lower; the E2E ACKs provide global knowledge to nodes, giving the power of flooding for a much cheaper cost.

Aggregate goodput. Figure 4 shows the goodput for Naïve Flooding, Priority Flooding, Reliable Flooding without E2E ACKs, and Reliable Flooding. In Naïve Flooding (Figure 4a), since each message travels on every edge in both directions, each of the five flows gets one fifth of the bandwidth capacity. Priority Flooding (Figure 4b) and Reliable Flooding without E2E ACKs (Figure 4c) both have higher goodput than Naïve

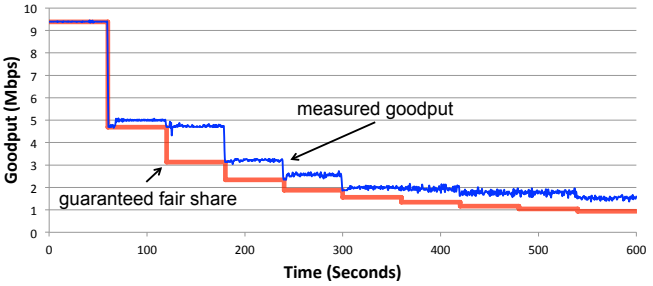

Fig. 5. Performance of one Priority Flooding flow (thin line) with an increasing number of active sources. The thick line shows the guaranteed fair share.

Flooding because they avoid sending messages on some links. Priority Flooding has higher goodput than Reliable Flooding without E2E Acks. Because Priority Flooding drops messages due to contention and different messages are dropped on different paths, not all messages reach all nodes, providing opportunities for more messages to arrive at the destination in aggregate. Reliable Flooding (Figure 4d) has the highest goodput because the E2E ACKs prevent some messages from being sent to regions of the network that do not need them.

Discussion. The cost of Reliable Flooding (Table IV) is about 2.5x the cost of K=3 (Table III), making it an appealing alternative. Initially, we expected that Priority Flooding would cost less and perform better than Reliable Flooding, since it is a best-effort protocol and less rigid. However, the E2E ACKs result in higher aggregate goodput as well as higher goodput for each individual flow. While Reliable Flooding achieves higher goodput, Priority Flooding is still the best choice for applications that require timeliness.

In this global cloud, the monitoring and control traffic amounts to less than 0.1% of the overall traffic. Because the messaging overhead of our protocols on that topology is 2.3x for K Node-Disjoint Paths with K=2 (Table III) and up to 19x for Constrained Flooding (Table IV), the overhead cost of intrusion-tolerant monitoring and control is below 2% of the total traffic. We consider this a tolerable overhead cost because these applications form the infrastructure of the cloud.

B. Resilience to Attacks

We evaluate the performance of the deployment under instrumented attacks.

1) Priority Messaging

Figure 5 shows the performance of a single Priority Flooding flow sending at the maximum link capacity. Every 60 seconds, an additional randomly selected source node begins sending at the same capacity. At each interval, we report the measured performance (thin line) and the guaranteed fair share (thick line) based on the number of active sources. The measured goodput outperforms the minimum guaranteed value because not all links are in full contention at all times.

Figure 6a shows the goodput for a correct Priority Flooding flow (9-11) sending at 1.6 Mbps, with four compromised flows trying to consume bandwidth by each sending at the maximum link capacity (10 Mbps). The goodput of the correct flow is not affected because 1.6 Mbps is less than its fair share with four other active flows. The remaining bandwidth is shared

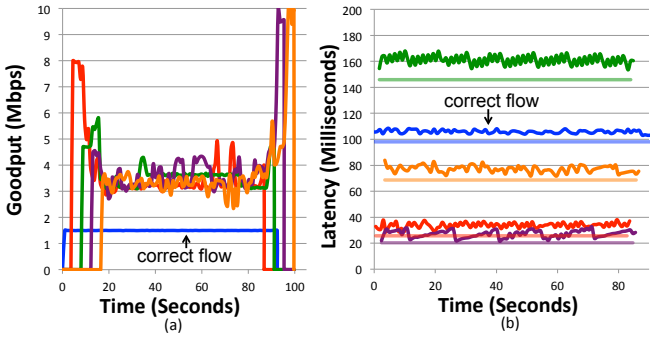

Fig. 6. Priority Flooding (a) goodput and (b) latency.

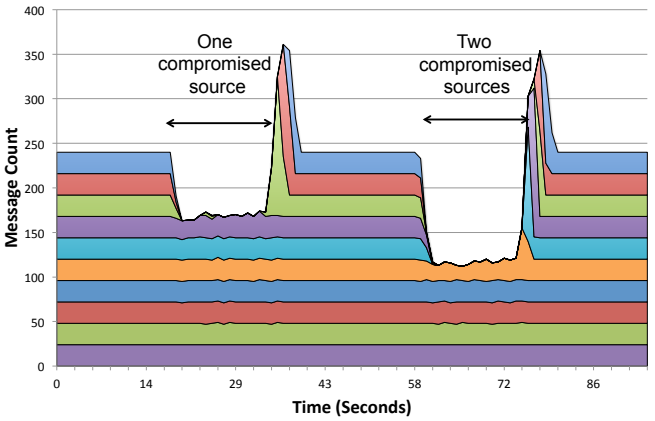

Fig. 7. Priority Flooding under message spamming attack. When compromised nodes attempt to saturate the network with highest-priority messages, the correct node's higher priority messages (lower bands) are preserved.

evenly among the other flows. These results are consistent with the service guarantees of Priority Messaging.

Figure 6b shows the latency experienced by these five flows and compares it to the propagation delay between the source and destination (flat line). While all five flows experience latency close to propagation delay, the correct flow's latency is closer to propagation delay because it sends less than its fair share, so its messages do not wait in queues.

In Figure 7, a correct Priority Flooding flow (7-9), from Europe to East Asia, sends at a rate of 7 Mbps and evenly distributes its messages across ten priority levels (ten colored bands). With no attack, all messages are received at the destination. When one or more compromised nodes attempt to saturate the network with highest-priority messages, the correct node's higher priority messages are preserved at the expense of its lower priority messages. After the attack stops, the lower priority messages still in storage at intermediate nodes are forwarded, resulting in a burst of traffic. As can be seen in the graph, this storage is cleared in order by priority (i.e. an entire priority level is cleared before starting the next lower level). In all cases, the correct flow achieves the bandwidth it requests or at least its fair share, meeting the service guarantees.

2) Reliable Messaging

Figure 8 shows the performance of a single Reliable Messaging flow (7-9), from Europe to East Asia, for both Constrained Flooding and K-Paths, with various loss rates applied to all links in the network. This experiment is emulated to match the topology and latencies of the real cloud to accurately control the injected loss. The flow is able to maintain

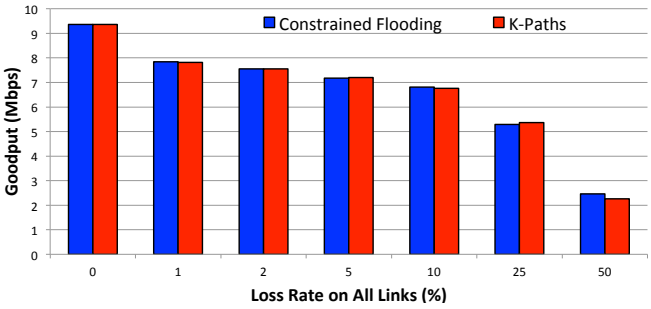

Fig. 8. Performance of one Reliable Messaging flow with loss rates applied to all links in the topology.

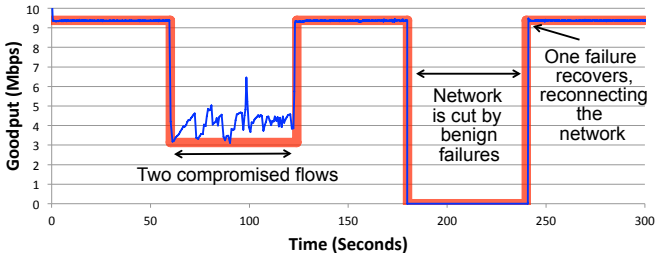

Fig. 9. Performance of one Reliable Flooding flow (thin line), impacted by two compromised flows and by crashes that cut the network.

performance, even under high loss. This is the worst-case flow for loss in the topology because it uses the most hops, and loss is applied on each hop. In fact, this is one of the worst-case flows on the globe for backbone traffic as it spans about half of the earth's circumference.

Figure 9 shows the performance of a single Reliable Flooding flow sending at maximum link capacity over the course of two events: two compromised flows attempting to saturate the network, and a crash-recovery of intermediate nodes that causes a network partition between source and destination. The guaranteed fair share (thick line) is shown for reference in all cases. Throughout the experiment, the flow's goodput meets the service guarantees. During contention, the E2E ACKs result in goodput higher than the guarantee.

C. Shadow Monitoring System

We use the deployment to carry the monitoring messages of the global cloud. The monitoring messages provide a real-time view of the cloud, updating every 1–3 seconds depending on the type of information. This view contains detailed information regarding the status of data centers, the network characteristics (e.g. latency, bandwidth, loss rate) of links between data centers, the status of cloud access points (i.e. clients), and the service characteristics that each client-generated task receives.

The deployment ran for several months as a complete shadow monitoring system for all monitoring messages. It was used in a limited production capacity: monitoring messages carried by the deployed network were processed and displayed in a graphical user interface that shows a real-time view of the cloud. Other alarm and log related features were not implemented. The monitoring messages used Priority Messaging because it meets the real-time requirements. Throughout the deployment, K-Paths (with K=2) and Constrained Flooding were used to evaluate their applicability.

The deployed network was able to provide the same timely

delivery of monitoring messages as the production monitoring network (resulting in an equivalent real-time view of the cloud), but with the addition of intrusion-tolerant guarantees for a tunable higher cost. Based on our experience building and running this cloud, we find the higher cost, even for Constrained Flooding, an acceptable price to pay for the critical messages, given the strong guarantees gained. Further, we are even considering using the deployed network with K-Paths (K=2) to carry data for some select high-value applications.

VII. Conclusion

We presented the first practical intrusion-tolerant network service. The solution uses an overlay approach to overcome malicious attacks and compromises in both the underlying network infrastructure and in the overlay itself. We deployed the overlay on a global cloud spanning East Asia, North America, and Europe, and evaluated its performance carrying realistic cloud monitoring and control traffic in the presence of compromised nodes. The implementation is publicly available in the Spines messaging toolkit at www.spines.org.

VIII. Acknowledgement

This work was supported in part by DARPA grant N660001-1-2-4014. Its contents are solely the responsibility of the authors and do not represent the official view of DARPA or the Department of Defense.

References

[1] A. Toonk, "Chinese ISP hijacks the Internet," bgpmon.net/blog/?p=282, 2010, access: 2015-12-15.

[2] "YouTube hijacking: A RIPE NCC RIS case study," www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study, 2008, access: 2015-12-15.

[3] A. Studer and A. Perrig, "The coremelt attack," in *14th European Symp. Research in Comput. Security (ESORICS)*, 2009, pp. 37–52.

[4] M. S. Kang, S. B. Lee, and V. Gligor, "The crossfire attack," in *IEEE Symp. Security and Privacy (SP)*, May 2013, pp. 127–141.

[5] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, Jul. 1982.

[6] G. Finn, "Reducing the vulnerability of dynamic computer networks," USC/Information Sciences Inst., Tech. Rep. ISI-RR-88-201, June 1988.

[7] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: a defense against wormhole attacks in wireless networks," in *22nd Annu. Joint Conf. IEEE Comput. and Commun. Soc. (INFOCOM)*, vol. 3. IEEE, 2003, pp. 1976–1986.

[8] "The spines messaging system," www.spines.org, access: 2015-12-15.

[9] P. Papadimitratos and Z. J. Haas, "Securing the internet routing infrastructure," *IEEE Communications Magazine*, vol. 40, no. 10, pp. 60–68, Oct. 2002.

[10] B. Kumar and J. Crowcroft, "Integrating security in inter-domain routing protocols," *ACM SIGCOMM Computer Communication Review*, vol. 23, no. 5, pp. 36–51, 1993.

[11] S. L. Murphy and M. Badger, "Digital signature protection of the OSPF routing protocol," in *Proc. Symp. Network and Distributed Syst. Security*. IEEE, 1996, pp. 93–102.

[12] H. Johansen, A. Allavena, and R. van Renesse, "Fireflies: Scalable support for intrusion-tolerant network overlays," in *Proc. 1st ACM SIGOPS European Conf. Comput. Syst. (EuroSys)*, 2006, pp. 3–13.

[13] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach, "Secure routing for structured peer-to-peer overlay networks," *SIGOPS Operating Syst. Review*, vol. 36, no. SI, pp. 299–314, Dec. 2002.

[14] H. C. Li, A. Clement, E. L. Wong, J. Napper, I. Roy, L. Alvisi, and M. Dahlin, "BAR gossip," in *Proc. 7th Symp. on Operating Syst. Design and Implementation*, 2006, pp. 191–204.

[15] K. A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R. A. Olsson, "Detecting disruptive routers: A distributed network monitoring approach," *IEEE Network*, vol. 12, no. 5, pp. 50–60, 1998.

[16] S. Cheung and K. N. Levitt, "Protecting routing infrastructures from denial of service using cooperative intrusion detection," in *Proc. Workshop on New Security Paradigms*, 1997, pp. 94–106.

[17] J. Deng, R. Han, and S. Mishra, "INSENS: Intrusion-tolerant routing for wireless sensor networks," *Computer Communications*, vol. 29, no. 2, pp. 216–230, 2006.

[18] R. Obelheiro and J. Fraga, "A lightweight intrusion-tolerant overlay network," in *9th IEEE Int. Symp. Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, April 2006, pp. 8–15.

[19] B. Awerbuch, R. Curtmola, D. Holmer, C. Nita-Rotaru, and H. Rubens, "ODSBR: An on-demand secure byzantine resilient routing protocol for wireless ad hoc networks," *ACM Trans. Information and Syst. Security*, vol. 10, no. 4, pp. 6:1–6:35, Jan. 2008.

[20] Y. Amir, P. Bunn, and R. Ostrovsky, "Authenticated adversarial routing," in *Proc. 6th Theory of Cryptography Conf (TCC)*, 2009, pp. 163–182.

[21] X. Zhang, H.-C. Hsiao, G. Hasker, H. Chan, A. Perrig, and D. Andersen, "SCION: Scalability, control, and isolation on next-generation networks," in *IEEE Symp. Security and Privacy (SP)*, May 2011, pp. 212–227.

[22] R. Perlman, "Network layer protocols with Byzantine robustness," Ph.D. dissertation, Massachusetts Institute of Technology, 1989.

[23] ——, "Routing with Byzantine robustness," Sun Microsystems, Inc., Mountain View, CA, USA, Tech. Rep. SMLI TR-2005-146, 2005.

[24] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. Symp. Operating Syst. Principles*, 2001, pp. 131–145.

[25] Y. Amir and C. Danilov, "Reliable communication in overlay networks," in *Proc. IEEE/IFIP Int. Conf. Dependable Syst. and Networks*, June 2003, pp. 511–520.

[26] A. Newell, D. Obenshain, T. Tantillo, C. Nita-Rotaru, and Y. Amir, "Increasing network resiliency by optimally assigning diverse variants to routing nodes," in *Proc. 43rd IEEE/IFIP Int. Conf. Dependable Systems and Networks*, 2013, pp. 1–12.

[27] J. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, no. 2, pp. 125–145, 1974.

[28] D. Sidhu, R. Nair, and S. Abdallah, "Finding disjoint paths in networks," in *Proc. Conf. on Commun. Architecture & Protocols*, 1991, pp. 43–51.

[29] J. C. Corbett *et al.*, "Spanner: Google's globally distributed database," *ACM Trans. Comput. Syst.*, vol. 31, no. 3, pp. 8:1–8:22, Aug. 2013.

[30] D. Obenshain, "Practical intrusion-tolerant networking," Ph.D. dissertation, Johns Hopkins University, 2015.

[31] E. Rescorla and N. Modadugu, "Datagram transport layer security," Internet Requests for Comments, RFC Editor, RFC 4347, April 2006. [Online]. Available: www.rfc-editor.org/rfc/rfc4347.txt

[32] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[33] S. Savage, N. Cardwell, D. Wetherall, and T. Anderson, "TCP congestion control with a misbehaving receiver," *SIGCOMM Computer Communication Review*, vol. 29, no. 5, pp. 71–78, Oct. 1999.

[34] A. Homescu, S. Neisius, P. Larsen, S. Brunthaler, and M. Franz, "Profile-guided automated software diversity," in *IEEE/ACM Int. Symp. Code Generation and Optimization (CGO)*, Feb 2013, pp. 1–11.

[35] M. Castro and B. Liskov, "Practical Byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, Nov. 2002.

[36] M. Platania, D. Obenshain, T. Tantillo, R. Sharma, and Y. Amir, "Towards a practical survivable intrusion tolerant replication system," in *Proc. IEEE Int. Symp. Reliable Distributed Syst. (SRDS)*, 2014, pp. 242–252.

[37] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[38] H. Krawczyk, R. Canetti, and M. Bellare, "HMAC: Keyed-hashing for message authentication," Internet Requests for Comments, RFC Editor, RFC 2104, February 1997. [Online]. Available: www.rfc-editor.org/rfc/rfc2104.txt

[39] "FIPS PUB 180-4: Secure hash standard," *US Department of Commerce, National Institute of Standards and Technology*, 2012.

[40] "OpenSSL project," www.openssl.org, access: 2015-12-15.

[41] H. Lee, J. Seibert, E. Hoque, C. Killian, and C. Nita-Rotaru, "Turret: A platform for automated attack finding in unmodified distributed system implementations," in *IEEE Int. Conf. Distributed Computing Syst.*, June 2014, pp. 660–669.

[42] "LTN global communications," www.ltnglobal.com, access: 2015-12-15.