iSAVE: Incrementally Deployable Source Address Validation*

Jelena Mirković, Zhiguo Xu, Jun Li, Matthew Schnaider, Peter Reiher, and Lixia Zhang

Abstract

Reliable information concerning the reverse path to a particular source address space would be useful for a number of applications, most notably for the filtering of packets with spoofed source addresses. The SAVE protocol makes this information available at every router, although, in the absence of full deployment, it is difficult for SAVE to maintain either correct or complete incoming tables.

We propose iSAVE, a modified version of the SAVE protocol, designed to address the case of incremental deployment. In our design, a fraction of the core routers are delegated as filtering routers. Autonomous systems participate in iSAVE by providing authoritative information that validates the appropriate incoming interface for their source address space at each interested filtering router. We present simulation results that demonstrate the effectiveness of the filtering capabilities provided by iSAVE and measurements of the performance costs associated with iSAVE control messages.

1 Introduction

Today's Internet does not maintain reliable information that allows a router to determine the proper incoming interface for a packet with a particular IP source address. As a result, it is difficult to perform any operation that could benefit from that knowledge, such as building reverse-path forwarding (RPF) structures [7] or detecting attempts to spoof source addresses in IP packets.

The SAVE protocol [11] is able to build router tables that properly describe incoming interfaces despite asymmetries in routing. These tables can be used for many purposes, including detection of IP spoofing. The tables are built using methods similar to the creation of routing tables, though there are important differences. The SAVE protocol only works properly when ubiquitously deployed at all routers in the network. Such a requirement is not realistic.

Park and Lee [14] demonstrated that for purposes of filtering spoofed IP packets, deployment of filtering capabilities at approximately 18% of Internet autonomous domains can catch almost all spoofed packets with few or no false positives. This work, however, did not describe how to create the incoming tables required for such filtering. It concentrated on finding the proper deployment pattern. Unfortunately, the SAVE protocol cannot build the necessary tables for the deployment patterns suggested in [14]. This paper presents a different version of the SAVE protocol that works well in partial deployment situations. The iSAVE protocol is actually fundamentally different than the SAVE protocol in most ways, aside from the common goal of building proper incoming tables. Rather than exchanging complete information on potential routing patterns with all other routers, the iSAVE protocol only disseminates incoming interface information to places where it is actually needed.

This paper describes the iSAVE protocol in detail. We discuss a simulation implementation of iSAVE and present data that shows iSAVE can be highly effective even if deployed at a modest percentage of all Internet routers.

2 Related Work

Source address validity has been addressed through both preventive approaches and reactive approaches. Filtering and cryptography-based authentication are preventive approaches. Tracing is primarily reactive.

*This work is funded by NSF under contract number ANI-9980501. The authors can be reached at {sunshine, zhiguo, lijun, matt, reiher, lixia}@cs.ucla.edu.

Much research has focused on applying cryptographic operations in order to guarantee authenticity of packet information, including the source address field (IPsec is one representative at the IP layer [10]). A digital signature on a packet can prove the authenticity of the its purported origin. Packet encryption also works, since the recipient will only be able to successfully decrypt a packet if it was encrypted at its true origin. Unfortunately, the high computational overhead of cryptographic operations prevents such approaches from being widely employed on a per-packet basis. This also usually involves complex key exchanges at the Internet scale.

Also widely studied is packet tracing [4] [5] [15] [16]. While tracing is a very useful capability, a disadvantage is that it is typically performed after an attack is detected, frequently too late to avoid damage. Moreover it requires fairly complex operations. These difficulties are compounded when the volume of attack traffic is small or the attack is distributed [13].

In contrast, filtering is a preventive approach and does not necessarily need cryptographic operations. Ideally, a router should be able to determine the validity of the source address of any packet, as long as that packet is from a protected domain. Filtering based on routing tables is one possible approach; however, it assumes symmetric routing, i.e., the outgoing interface that a router uses to reach a given address, as specified by its routing table, is also the valid incoming interface for packets originating from that address. However, since routing asymmetry on the Internet is common, this approach is unrealistic.

Source-address-based filtering has been supported by router vendors (such as Cisco and 3COM) using static, manual configuration (such as access control lists for Cisco routers) [17]. Martian addresses (loopback address, broadcast address, non-unicast addresses, etc.) can be easily filtered [3]. Particular rules can also be applied: for instance, ingress filtering [8] can be enforced at a periphery router by ensuring that packets leaving the domain of a periphery router have a source address from inside the domain, and egress filtering can ensure that packets entering have an outside source address. This kind of filtering, however, does not offer sufficient protection. For instance, research has shown that unless ingress filtering is deployed almost everywhere, nearly arbitrary forgery is still possible [14].

3 Route-Based Distributed Packet Filtering

Route-based distributed packet filtering uses routing information to determine if the incoming interface of a packet matches the expected incoming interface of the source address inscribed in the packet's IP header. This information—the proper incoming interface for any given source address—is stored in an *incoming table*. In [14], the authors suggested a deployment pattern of filtering routers that achieves a high degree of filtering effectiveness with a small number of deployment points. The filtering is most efficiently performed by routers in core domains. The high degree of connectivity of core routers increases their ability to discriminate between spoofed and legitimate traffic flows.[1] Core routers forward large volumes of traffic and thus any policing undertaken by those routers will severely impact an attempted attack.

3.1 The SAVE Protocol

A distributed protocol called SAVE [11] was proposed to build the incoming table at every filtering router. SAVE is similar to routing protocols; all participating routers play an equal role by providing valid interface information about their associated *source address space*[2] through *SAVE updates*, and by processing and forwarding updates received from other routers. The goal of a SAVE router is to validate routes from itself to all reachable hosts. This is performed by sending periodic updates to all possible hosts and triggered updates only to those hosts for which the route has recently changed. To avoid the immense overhead of issuing a SAVE update per IP address, SAVE issues one update per *destination address space*,[3] using the

[1] A router that has a single incoming interface cannot discriminate between spoofed and legitimate packets. Its incoming table is trivial—all packets must come on that single interface. Spoofed packets arriving at a router with two incoming interfaces have a good chance (50% if we assume that incoming addresses are uniformly distributed across interfaces) of arriving on a valid interface for a given source address and will thus avoid filtering. A larger number of router interfaces reduces this probability of error.

[2] The source address space of a SAVE router is a set of IP addresses that use this router as an exit router to reach the Internet. This space can be configured through administrative means.

[3] A destination address space is a set of addresses that might share a route.

address prefixes from a forwarding table to define destination address spaces. A random IP address from the destination address space is used as the destination address in the IP header of the update.

Fields in the SAVE update specify both the source address space and the destination address space; valid interface information is implicitly specified by the interface on which the update arrives. Additionally downstream routers that process and forward the update may append their source address space information.

A SAVE router that receives an update organizes the specified source address space information in an intermediate structure—an *incoming tree*. The incoming tree captures the routes from all source address spaces to that router: if a route from address space A crosses address space B, node A will be a child of node B in the incoming tree. If an update indicates that some node in the tree should be moved, all its children are moved with it. The router further processes the tree and updates its incoming table. It then decides whether to forward or drop the update. If the router has a local route to the entire destination address space, the update is dropped. Otherwise, it is forwarded in accordance with the forwarding table entries. If the entire destination address space can be reached via a single entry, the update is forwarded on that entry. If, however, the forwarding table contains several entries for different portions of the destination address space, the update is cloned and each instance is forwarded on one of the entries, with the destination address space adjusted to reflect the address range from that entry.

In [11], the authors show that SAVE operates correctly when fully deployed; it builds correct incoming tables, successfully filters all spoofed packets and reacts to routing changes by adjusting the incoming table information promptly as a result of triggered SAVE updates. Additionally, its cost is comparable to that of the routing protocols. However, it is unreasonable to expect that any protocol can be fully deployed on the Internet. Under partial deployment, the cooperation of downstream routers cannot be guaranteed. This introduces the possibility of incorrect protocol operation. If a legacy router has several different routes for portions of the destination address space, only the one corresponding to the destination IP address in update header will be used to forward the update. This leads to invalid and incomplete incoming entries in the downstream SAVE routers. Additionally routing changes will be handled through triggered SAVE updates only if they affect SAVE routers. The straightforward solution is to increase the granularity of the destination address space and send a SAVE update to every IP address. The approach solves the problem, but incurs enormous cost, especially since periodic updates are necessary to deal with routing changes.

4 iSAVE

This paper proposes an alternative protocol called iSAVE (incremental SAVE) that operates correctly in the case of incremental deployment. The key difference is that the valid incoming interface information is communicated only to filtering routers on recently traversed routes (by legitimate or spoofed traffic). iSAVE uses proactive and reactive updates to deliver this information. Proactive updates are similar to periodic SAVE updates: they are sent to validate routes for legitimate traffic. Instead of attempting to validate all routes, a router generating proactive updates only performs best-effort validation. Proactive updates will validate routes to destinations to which the I-router is forwarding traffic. It also considers the overhead incurred in generating the updates to define its acceptable proactive update rate. Reactive updates are generated on demand when a downstream filtering router encounters a packet that matches an incorrect or non-existent entry in its incoming table.

iSAVE makes a distinction between *informational routers* that validate routes for their source address space through iSAVE updates and *filtering routers* that use these updates to build their incoming tables and perform spoofed packet filtering. This distinction is purely functional, and both operations can be performed by any given router. However, due to their high connectivity, core routers are better positioned to provide this type of filtering, while edge routers are the ingress point for the majority of Internet traffic and are the in a better position to host informational routers. This deployment pattern indicates that filtering iSAVE routers can be offered as an infrastructural service, while informational iSAVE routers can be established by interested clients who want to protect their source address space. Measurements and analysis in this paper assume this deployment pattern.

4.1 Description of iSAVE Protocol Operation

An iSAVE informational router, hereafter called an I-router, generates proactive updates to validate recently used routes and reactive updates as a response to *iSAVE requests* by filtering routers. Proactive updates are sent to a chosen set of active destinations. In order to detect which destinations are active, an I-router keeps a *Validation Hash Table* (VHT) of destinations toward which an iSAVE update (proactive or reactive) has been sent. Entries in this table expire after interval T_{VH}, and can expire sooner if the table overflows. Each outgoing packet's destination is matched to entries in the VHT. New destinations that trigger generation of proactive iSAVE updates are inserted in the VHT. The lifetime of existing destinations is prolonged with each matching packet.

Since a source address space will potentially communicate with a large number of destinations at any given time, eager generation of proactive messages can overwhelm its resources. On the other hand, routes to many destinations are likely to overlap, and the first message sent to a set of destinations carries the same information as the later ones, making them redundant. An I-router can adjust the generation frequency of proactive iSAVE updates to preserve its resources.[4] However, an I-router cannot infer which destinations have overlapping routes, and thus must guess the likely destinations of its proactive updates. It is therefore possible that some filtering routers can have incomplete or incorrect information in their routing tables, since they do not receive timely proactive updates. Filtering routers attempt to compensate for this effect by issuing iSAVE requests when they suspect that they have incorrect or incomplete information. I-routers then generate reactive updates in response to iSAVE requests. Those updates do not affect the VHT, for reasons explained in Section 5.3.1.

An iSAVE filtering router, hereafter called an F-router, uses three structures to validate the source address of incoming packets: the *Incoming Table*, the *Pending Cache* and the *Legacy Cache*.

The Incoming Table stores the information on proper incoming interfaces for certain source address spaces, learned through iSAVE updates. Each iSAVE update instantiates an entry and starts the associated timer T_{fresh}. Until this timer expires, all conflicting packets for this entry will be dropped and no iSAVE requests will be issued. After the T_{fresh} timer expires, conflicting packets will trigger iSAVE requests. An iSAVE request specifies the destination address of the conflicting packet and is sent to the source address indicated in the packet header. Upon entering the domain that contains the given source address, the iSAVE request will be intercepted by either the authoritative I-router for that source address space or by a border router that recognizes iSAVE requests and redirects them to the appropriate I-router. The I-router then generates a reactive update and sends it toward the specified destination address.

The F-router starts the $T_{IT-pend}$ timer upon sending the request. iSAVE allows the authoritative I-router to specify the preferred policy to be used while waiting for the requested update. For example, when an I-router is fully operational and is eagerly generating proactive messages, it can request that all conflicting packets be dropped in the intervening period. If, however, the I-router is approaching scheduled downtime, it can request that all conflicting packets be forwarded to minimize harm to legitimate packets in the case of a routing change. If the F-router has no information about the I-router's preferences, the default behavior is to allow packets through.

If the $T_{IT-pend}$ expires and no iSAVE update has been received, this can indicate that one of the following conditions exists:

1. The source address belongs to a legacy space.
2. The I-router that protects the space is incapable of responding.
3. The iSAVE request never reached the authoritative I-router.
4. The I-router responded with an update, but the update was dropped by the network.
5. The I-router responded, but the path to the destination does not cross the requesting F-router.

F-router determines which of these is the case by issuing several more iSAVE requests and restarting the timer with an exponentially increasing value. These requests specify the IP address of the F-router in addition to the destination address of the conflicting packet. Thus, if the space is iSAVE protected and the

[4]For instance, an I-router can send updates to new destinations with a certain probability.

authoritative I-router is operational, the requesting F-router should receive an update, even if it is not on the route from the given source address space to the destination. This retransmission mechanism is much like TCP's retransmission mechanism with exponential backoff. After $N_{IT-trial}$ unsuccessful iSAVE requests, the entry is deleted from the Incoming Table and inserted into the Legacy Cache. The entry can also be expunged from the Incoming Table if no packets (iSAVE updates or data packets) have been received by the time T_{active} expires.

Entries in the Legacy Cache represent the address spaces that did not reply to repeated iSAVE requests and are therefore considered legacy spaces. Packets from these spaces are forwarded to the destination and no requests are issued. Legacy Cache entries are deleted after T_{legacy} expires.

If the packet arrives from a source address space that does not exist either in the Incoming Table or in the Legacy Cache, the address space is inserted into the Pending Cache, the $T_{PC-pend}$ timer is started and an iSAVE request is sent. If the iSAVE update arrives, the entry is inserted into the Incoming Table. Otherwise, a retransmission process similar to the one described above takes place. Upon $N_{PC-trial}$ unsuccessful requests, the entry is inserted into the Legacy Cache.

5 Theoretical Analysis of iSAVE

While we offer simulation results in Section 6 to evaluate the performance of the iSAVE protocol, limited computational resources prevent us from simulating iSAVE in a network larger than several hundred nodes. Theoretical analysis can show the effectiveness and worst-case cost each participant would endure if iSAVE were deployed at Internet scale.

5.1 Cost Analysis of iSAVE

Let us observe a network with N nodes arranged in arbitrary topology. Each node can act as a router, a sender and a receiver simultaneously. Out of these N nodes, F nodes are F-routers and I nodes are I-routers. A node can be both an F-router and an I-router.

The cost of running iSAVE has two components:

1. Bandwidth cost reflected in number of messages/bytes sent per second in iSAVE requests and iSAVE updates
2. Storage cost for iSAVE-related data structures

5.1.1 Bandwidth Cost

We discuss bandwidth cost per informational iSAVE router, per filtering iSAVE router, and per network link to discover potential sources of network congestion in iSAVE protocol design.

Bandwidth Cost of an Informational Router

Informational routers send iSAVE updates proactively to validate the path to each new destination, and reactively in response to iSAVE requests. A proactive iSAVE update is sent for each new destination, preceding data packets to that destination. Let us assume that the size of an iSAVE update is s bytes and an informational router detects, on the average, D new destinations per second. Then the bandwidth cost of proactive iSAVE updates is:

$$BW_{pro} = D \times s \tag{1}$$

per second.

Using the observation period of 30 seconds and profiling several packet header traces [1] [2] [9] [12], we estimate the size of D to be approximately 100 for a medium-size network. Letting $D = 100$ and $s = 40\,\mathrm{B}$, the proactive cost of running iSAVE for the informational router is approximately 100 messages or 4 KB per second.

A reactive iSAVE update is sent in response to an iSAVE request issued by an F-router to validate or invalidate an incoming interface change at this router. An F-router checks for an incoming interface change

when it receives a data packet on a different interface than expected for its source address. One reason for this can be that a routing change occurred on the path from the source address to the given filtering router; in this case the incoming interface should be changed and packets forwarded further. The other possible reason is a flood of spoofed data packets that should be filtered out.

Let p be the probability of link failure per second and L_{if} be the number of links in the route between a particular informational router R_i and particular filtering router R_f. Assuming that link failures are independent, the probability of a routing change between R_i and R_f is

$$p_{rc} = 1 - (1-p)^{L_{if}} \tag{2}$$

Let us apply the conservative assumption that any routing change leads to a change of the incoming interface at R_f associated with $SAS(R_i)$, the source address space advertised by R_i. Then R_f should issue, on the average, p_{rc} messages toward R_i per second. Ignoring the beneficial effect of incidental updates (the iSAVE update, issued by I-router A on request from F-router B, passes on its way through F-router C, updating C), an I-router R_i will generate, on the average,

$$numMess_{rc} = \sum_{f=1}^{F} 1 - (1-p)^{L_{if}}, \quad BW_{rc} = s \times numMess_{rc} \tag{3}$$

bytes per second.

The flood of spoofed packets arriving on an unexpected interface of an F-router R_f will lead to issuing an iSAVE request toward the spoofed source address space. As described in Section 4.1, the F-router uses the $T_{IT-pend}$ timer to control the issuing rate of iSAVE requests. $T_{IT-pend}$ is initially set to a value IT_{start} and exponentially increases if it expires before an iSAVE update arrives. Let us again disregard the beneficial effect of incidental updates. We can further assume that the attacker sends a vast number of packets spoofing the source address from the particular source address space to every F-router in the network. Then the cost incurred by the I-router for that source address space will be, at most:

$$numMess_{att} = \frac{F}{IT_{start}}, \quad BW_{att} = s \times \frac{F}{IT_{start}} \tag{4}$$

bytes per second.

Assuming $F = 1000$ (the estimated number of F-routers from [14]), $IT_{start} = 1\,\text{sec}$, and $s = 40\,\text{B}$, we estimate that as a consequence of malicious attack, an I-router will generate at most 40 KB per second.

There are two other possible attacks on an I-router:

1. An attacker could launch a *distributed denial-of-service* (DDoS) attack against the I-router (or somehow disable its ability to answer iSAVE requests, or intercept and drop iSAVE requests) while triggering iSAVE requests from F-routers. Since no update would arrive for their requests, F-routers would generate subsequent requests with interval $n \times IT_{start}$. Let us assume again $F = 1000$, $IT_{start} = 1\,\text{sec}$ and $s = 40\,\text{B}$. In the first second, the I-router would receive the highest load of iSAVE requests and would attempt to generate 40 KB of iSAVE updates. Since IT_{start} is exponentially increasing, the same number of later iSAVE requests would be divided over longer and longer time intervals, thus relieving the load.

2. An attacker could forge an arbitrary number of iSAVE requests, thus inducing the generation of numerous redundant iSAVE updates. With regard to the I-router's incoming bandwidth consumption, this is no different than any other flooding attack on the I-router. However, the I-router could decide to preserve its outgoing bandwidth by refusing to answer more than a certain number of iSAVE requests per second.

Bandwidth Cost of a Filtering Router

The incoming bandwidth of an F-router is consumed by the proactive messages from various I-routers that use this router to reach certain destinations. F-routers that lead to popular destinations could thus be overwhelmed by proactive iSAVE updates. The optimization of the protocol could include the ability of an

F-router to request a lesser degree of generation of proactive messages from I-routers, but in the extreme case the F-router should be able to handle all proactive messages received.

Let D_{fi} be the number of popular destinations that an I-router R_i reaches through a given F-router R_f. Then the cost the F-router experiences for handling proactive messages is:

$$Cost_{pro} = \sum_{i=1}^{I} D_{fi}, \quad BW_{pro} = s \times Cost_{pro} \tag{5}$$

bytes per second.

Outgoing bandwidth of an F-router is consumed by iSAVE requests sent in response to data packets that arrive on unexpected interfaces due to routing changes or spoofed packets. Let the probability of a route change for a particular source address space to a given F-router be as described in Equation 2. We assume that there is no benefit from incidental SAVE updates. Let r be the size of an iSAVE request. Then the F-router will generate:

$$numMess_{rc} = \sum_{i=1}^{I} 1 - (1-p)^{L_{fi}}, \quad BW_{rc} = r \times numMess_{rc} \tag{6}$$

bytes per second in response to routing changes. It will receive the same or a lesser number of replies.

Each spoofed packet has the potential of triggering an iSAVE request when it encounters an F-router. Thus, by spoofing addresses from a wide variety of source address spaces, an attacker can trigger an arbitrary number of requests per second. While, regarding the incoming bandwidth, this is no different than any other flooding attack on the router, the F-router could preserve its outgoing bandwidth and refuse to act as a reflector by limiting the number of iSAVE requests per second.

Bandwidth Cost to the Network

The bandwidth cost to the network depends on the network topology. Some links may be common for routes from several I-routers to F-routers and thus carry a higher load of iSAVE requests and updates. However, the analysis above shows that since each specific router does not experiences a high load during normal operation, and since iSAVE updates and requests take the same path as data packets, then iSAVE control messages should not a represent significant additional load for the network as a whole. During DDoS attacks on the iSAVE infrastructure, some links may naturally experience congestion. However, since iSAVE does not multiply messages, it cannot be used to amplify DDoS attacks.

5.1.2 Storage Cost

I-routers store data about recently active destinations in a VHT. The allowed length of the inactivity interval defines the size of this table: longer intervals will produce a larger table but will result in a lower number of proactive iSAVE updates.

F-routers store the data about protected source address spaces in the Incoming Table and data about legacy domains in the Legacy Cache. The size of these structures is configurable by the F-router's administrator. Larger structures will preserve data longer and thus reduce need for iSAVE requests due to nonexistent entries.

5.2 Filtering effectiveness

As described in Section 4.1, F-routers do not filter conflicting packets all of the time. When an iSAVE update is received there is a period T_{fresh} during which all conflicting packets are dropped. After this period, conflicts will invoke iSAVE requests and, during the wait for iSAVE updates, an F-router will drop or forward packets, based on the policy specified by the I-router responsible for a given source address space. Assuming that the I-router has specified a drop policy, the filtering effectiveness depends on several factors:

- *Choice of* T_{fresh}. If T_{fresh} is small, almost no filtering is performed, but during routing changes legitimate packets suffer almost no drops. With a large T_{fresh}, the F-router drops conflicting packets most of the time, thus potentially harming legitimate traffic during routing changes.

- *Frequency of proactive updates.* Even with a small T_{fresh}, if proactive updates are sent very often, the F-router will be able to drop most of the spoofed packets. At the same time, legitimate packets are less likely to be dropped when routes change, since frequent proactive updates will likely validate the new interface quickly.

Let us assume that no proactive updates are currently sent from informational router R_i to a filtering router R_f, because the source address space associated with R_i sends no data traffic through R_f. Let RTT be the round-trip time from R_f to R_i, T_{isr} the time to issue an iSAVE request, T_{isu} the time to process an iSAVE request and issue a reactive update, and T_{fresh} the time that the entry should be considered valid for filtering. Assume that the attack uses packets spoofing the source address space of R_i and lasts A seconds, with the average rate of n_A packets per second. Assume further that there are no filtering routers on the route from R_i to R_f. Then the attacker can manage to pass:

$$p_A = \frac{T_{isr} + RTT + T_{isu}}{T_{fresh} + T_{isr} + RTT + T_{isu}} \times A \times n_A \tag{7}$$

packets through R_f. Proactive updates reduce this amount, since they enlarge the validity interval of the associated entry. Note that if the attacker can prevent I-router from answering iSAVE requests and no proactive updates arrive, the measure p_A can be infinitely large.

When a routing change occurs, legitimate packets may start arriving on invalid interfaces. The worst-case behavior occurs if the entry was just updated. In that case, the F-router will blindly drop all conflicting packets during T_{fresh}, issue an iSAVE request, and apply the default policy until it receives the reactive update. The interval during which legitimate packets are being dropped is then:

$$T_{drop} = \begin{cases} T_{fresh} & \text{if } policy = forward \\ T_{fresh} + T_{isr} + RTT + T_{isu} & \text{if } policy = drop \end{cases}$$

If the attacker can prevent the I-router from answering iSAVE requests and there are no proactive updates arriving from that router, legitimate packets can be dropped as long as the entry stays in the Incoming Table. In the worst case, this interval is T_{active}.

5.3 Security of iSAVE

Special care must be taken to secure the operation of iSAVE against malicious attempts to compromise, misuse or disable the protocol. We next discuss several possible attacks on iSAVE and offer suggestions on how to secure operation of I-routers and F-routers.

5.3.1 Security of I-router operation

I-routers can be attacked in multiple ways. One such attack can can be mounted by instigating a flood of iSAVE requests. An attacker can spoof the I-router's associated source address space in a flood of packets sent to many F-routers. Each F-router would then request an update, thereby overwhelming the I-router. An attacker can also simply generate bogus iSAVE requests to provoke generation of iSAVE updates. Since updates are not significantly larger than requests, this attack would not have a magnifying effect. Note that both types of iSAVE request flood attacks generate, on average, one reflected packet for every packet in the initial flood, thus making it unattractive to attackers.

Simultaneous issue of iSAVE requests is also undesirable in normal operation. If a default forwarding policy is used while waiting for a reactive update, a spoofed packet is likely to invoke multiple iSAVE requests as it travels through F-routers and in turn cause multiple reactive updates. This, obviously, is a waste of resources. We propose two optimizations that can avoid the simultaneous issuance of iSAVE requests:

1. If an F-router forwards a conflicting packet and issues an iSAVE request, it can mark the packet so that downstream F-routers will not issue additional requests. This marking must be done in a secure fashion so that the mark cannot be forged by the attacker.

2. The F-router that encounters a conflicting packet can delay the issuance of iSAVE requests proportionally to its distance from the I-router associated with the source address of the packet. This distance

can be estimated from the routing cost for that source address stored in the routing table or from the TTL of the packet. Thus, if the upstream router has issued the request, the reactive update would manage to preempt issuance of additional requests by downstream routers.

Another security optimization is the choice to bypass the VHT with reactive updates prevents attackers from controlling the contents of the VHT. If reactive iSAVE updates affected the VHT, an attack with bogus iSAVE requests could overflow the table, thus potentially triggering deletion of legitimate traffic destinations from the table and causing a continual generation of proactive iSAVE updates for this traffic (in addition to reactive iSAVE updates generated in reply to bogus requests).

5.3.2 Security of F-router operation

Currently, Internet routers drop packets very rarely. Thus, the greatest harm attackers can do through attacks that forge routing updates is to lead routers to misroute packets or drop them if they conclude that the destination cannot be reached. iSAVE enhances routers with the ability to drop spoofed packets, but it introduces additional dangers. If attackers can inject arbitrary iSAVE updates, many legitimate packets can be dropped. It is therefore crucial that F-router operation be properly secured. Since iSAVE updates are exchanged between routers, we expect that many approaches used to secure routing update exchange can also be used to secure iSAVE operation. We suggest that:

- iSAVE updates should be exchanged only between routers, excluding regular hosts. Thus, in order to mount an attack via iSAVE updates, an attacker would need to compromise some router, a challenging task in itself.
- Routers should establish trust relationships prior to exchanging iSAVE updates.
- Each F-router must have the means to establish that the update it receives is valid, fresh and issued by a router that has the authority to validate the source address space inscribed in the update. This implies that each iSAVE update should be signed (or encrypted) by its sending router and be verified by its receiving router in order to guarantee its integrity. Replay of SAVE updates must also be prevented, using standard cryptographic methods.
- The processing (including the authentication) of iSAVE updates should be lightweight to prevent a denial-of-service attack on the F-router.
- Although securing iSAVE requests would enable I-routers to recognize bogus requests and avoid replying, the authentication cost is likely to be larger than the cost of issuing an iSAVE update. Thus we recommend iSAVE requests should not be authenticated.

The iSAVE protocol also has a correctness issue similar to that of routing protocols—a compromised router, if undetected, can severely damage the proper functioning of the network by sending bogus, but correctly signed and authentic, iSAVE updates. Some kind of intrusion detection implemented in routers might help counter this problem.

The attacker can mount a flooding attack on an F-router in an attempt to control the size of the Pending Cache. The attacker generates a large number of bogus packets, spoofing in a random or sequential manner addresses from the whole Internet address space. There is a great probability that most of these addresses will not be in either the Incoming Table or the Legacy Cache. Thus an iSAVE request will be sent toward them, and they will be added to the Pending Cache while awaiting a reply. Since the attacker can probably cover all addresses in the IPv4 address space before the iSAVE update reaches the requesting F-router, the size of the Pending Cache is obviously under the attacker's control. An optimization of iSAVE that avoids deletion of address spaces from Legacy Cache but instead triggers iSAVE requests can help mitigate this effect. Additionally, the F-router could generate only one iSAVE request per source address space instead of per source address. The router can assume that size of the source address space is 2^8. Since many address spaces will be either in the Incoming Table or in the Legacy Cache, this leaves a very small range of addresses that the attacker can spoof to fill the Pending Cache.

6 Simulation

In this section, we use simulation to evaluate the performance of i-SAVE. Two important performance measures we investigate through simulation are the effectiveness under partial deployment and adaptivity to routing change.

Incremental deployment imposes two basic requirements on iSAVE: iSAVE routers must be capable of interoperating with legacy routers in the Internet, and iSAVE must be able to work effectively without full deployment. undetected. In our simulation, we also investigate the relationship between the deployment strategy and effectiveness of iSAVE.

When encountering a routing change, iSAVE should be able to synchronize corresponding incoming entries in light of new topological information in a timely manner. An important feature of iSAVE is the resiliency to the loss of control messages. Even if the network suffers from data loss, we expect that iSAVE's convergence time for corresponding entries will not significantly degrade.

6.1 Simulation Design

iSAVE has been implemented and tested in Javasim [18]. We use the transit-stub topology generator from GT-ITM [6] to generate the simulation topologies corresponding to the two-level routing infrastructure of the Internet. Approximately 75% nodes represent the exit border router of a stub domain, which has one or two connections to the rest of the network. We designed three experiments to test the performance of iSAVE.

The first experiment measures the percentage of spoofable routes with partial deployment of F-routers at transit domains. We assume that only stub domains are capable of generating the traffic, whereas transit domains only relay packets between the source and the destination. All combinations of spoofable packets are then generated by letting each stub domain spoof all possible source address spaces in the network by sending packets to every other stub domain. Let N_s be the number of stub domains in the test topology. The total number of spoofable routes will be $N_s \times (N_s - 1) \times (N_s - 1)$. With the deployment of iSAVE, the number of spoofable routes should be greatly reduced. In this experiment, 100% of stub routers act as I-routers for their respective source address spaces. The final data is obtained from experiments run on ten different topologies.

The second experiment is designed to measure the performance of iSAVE in the face of a spoofing attack. With partial deployment of both iSAVE I-routers and F-routers, an attacker A sends malicious data traffic toward one destination D using a spoofed IP address belonging to a source address space SAS_i protected by an I-router I. We want to find out the increase in the number of control messages generated by the attack, as well as the percentage of spoofed data packets that successfully reach the victim. We consider four different scenarios:

1. There is data traffic from SAS_i to D.
2. There is no data traffic from SAS_i to any destination.
3. There is random data traffic from SAS_i to any other destination.
4. There is random data traffic from SAS_i to any other destination except D.

In the cases when there exists data traffic between SAS_i and some destination, we expect that F-routers can benefit from proactive iSAVE updates, as discussed in Section 5.1.1. In our simulation, the duration of each attack is exponential with an expected value of 5 seconds, and the interval between attacks is exponential with an expected value of 10 seconds. Background traffic for cases 1, 3, and 4 is generated by having router I initiate data connections with a fixed rate, but with the exponentially distributed interval between connections. This simulation has been performed on ten different topologies with ten trials for each topology. During each run I, D and A are chosen randomly among stub domains.

Our third experiment is designed to measure the adaptivity of iSAVE during routing changes. iSAVE is essentially an event-driven protocol. Routing changes do not trigger iSAVE updates immediately when new route is established. F-routers will take action only if the routing changes result in a conflict between real data traffic and their incoming entries. To better model an actual network environment, we consider

the case in which control messages may be lost. This experiment tests the settling time in the face of routing changes, both with and without control message loss. Because a single iSAVE validation message can update the corresponding entries at each F-router along its path, we expect that the impact of control message loss will be minimal. In the experiment, we use a physical link delay of 1 millisecond. We obtain this measurement using ten different topologies.

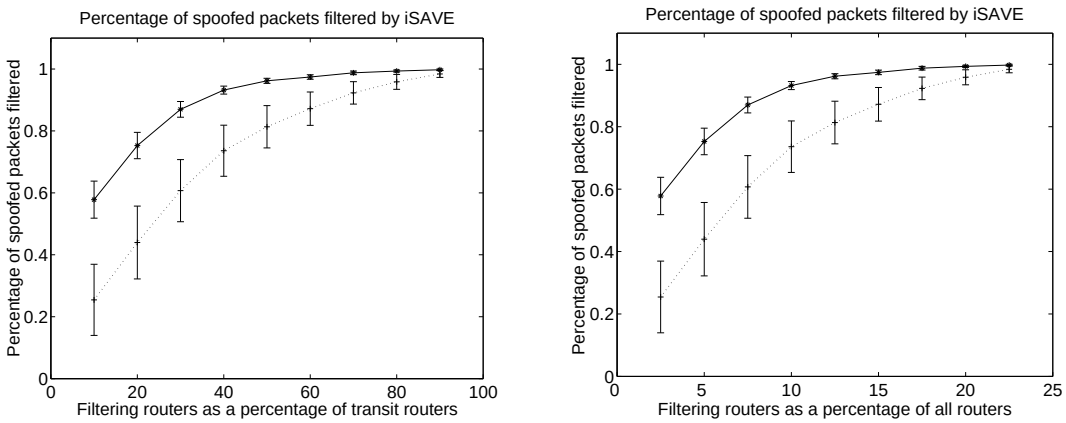

Figure 1: Percentage of all possible spoofed packets filterable by iSAVE

6.2 Results and Analysis

Park and Lee [14] proposed that deploying perfect incoming tables at approximately 18% of all autonomous domains on the Internet would provide the ability to filter nearly all spoofed traffic. This work also showed that a vertex covering of transit routers is the optimal deployment strategy necessary to achieve this result. As seen in Figure 1, the vertex covering of transit routers consistently outperforms random sets of filtering nodes of the same size. The vertex cover deployment achieves better than 90% filtering efficiency using less than 40% of the transit routers, or equivalently, 10% of all autonomous domains in our simulated topology.

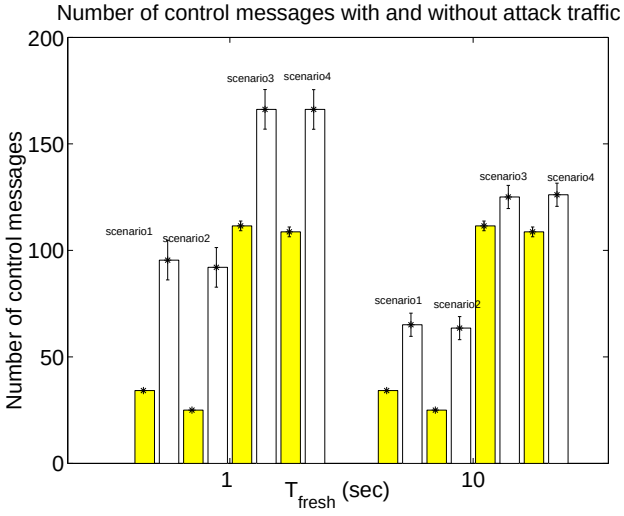

Figure 2: Number of control messages (with and without attack traffic)

The results from our second experiment demonstrate the minimal additional overhead incurred by control

messages under spoofing conditions as compared to the control messages required in the absence of spoofed traffic. Figures 2 and 3 show a comparison of the varying values of our freshness timer and the effects the choice of timer has on control overhead. The gray bars show the baseline control messages generated by iSAVE in the absence of spoofed packets. The white bars illustrate the total number of control messages issued in response to spoofed traffic.

In scenario one, we observed a sizable increase in control messages when spoofed packets were introduced. This is reasonable since only F-routers along the path from I to D are expected to maintain active entries for $SAS(I)$. Therefore, F-routers that encounter spoofed traffic must request an iSAVE update. In scenario two, this effect is amplified by the fact that no F-routers have active entries for $SAS(I)$, since no data traffic is originating from $SAS(I)$.

Scenario three demonstrated the benefits gained from proactive updates sent by I during normal communication with various other domains. While we do see an increase in the total number of control messages, proportionally fewer additional control messages are induced by the introduction of spoofed traffic. In this scenario, we also see a significant benefit from using a T_{fresh} of 10 seconds.

Scenario four reveals that even if $SAS(I)$ is not sending actual traffic to destination D, spoofed traffic addressed to D is not likely to induce any additional control traffic in the network. The result is particularly encouraging since it implies that an attacker will not be easily able to significantly affect the number of control messages required for iSAVE operation.

In general, data from all four scenarios indicates that choosing a larger value of T_{fresh} substantially reduces the additional control message overhead induced by spoofed packets in the network.

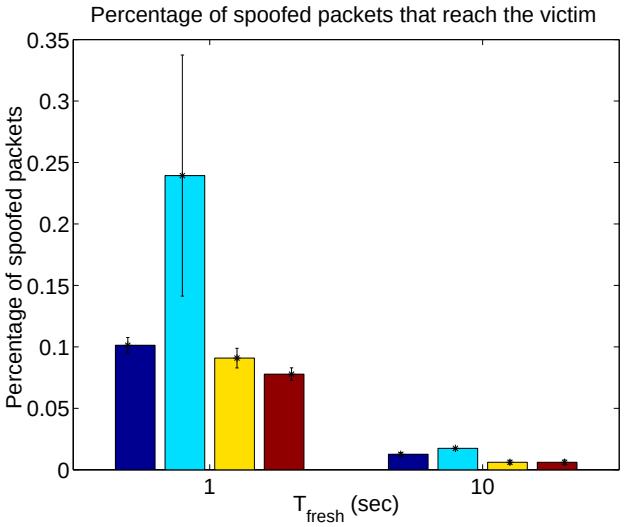

Figure 3: Percentage of spoofed packets passed in each of four scenarios

In addition to measuring the control overhead under varying conditions, we also collected data on the number of spoofed packets that were successfully able to escape filtering in each of the four scenarios. Figure 3 shows that far more packets are passed when $SAS(I)$ is silent or only communicating with $SAS(D)$. As data from scenarios 3 and 4 shows, wide distribution of proactive iSAVE updates sent out during communication with a diverse set of destinations provides many F-routers in the network with up-to-date information that will allow them to filter spoofed traffic with high confidence. Again, as one might suppose, choosing a larger value of T_{fresh} improves filtering efficacy. It is important to note, however, that this benefit comes at the cost of greater inertia in the face of routing changes. The larger the value of T_{fresh}, the longer incorrect entries may be honored, thereby dropping valid data traffic on new routes that have not yet been validated.

Our third measurement showed very promising results. When a routing change is introduced, F-routers are able to correct the entries affected by the change in a very short period of time. Figure 4 illustrates the distribution of settling times for entries affected by a routing change. When no control packets are subject to loss, 50% of the disrupted entries take less than 50 milliseconds to be corrected. Even with 15% control

message loss, 95% of the entries take less than 77 milliseconds to be corrected, with 100% of the entires taking less than 80 milliseconds.

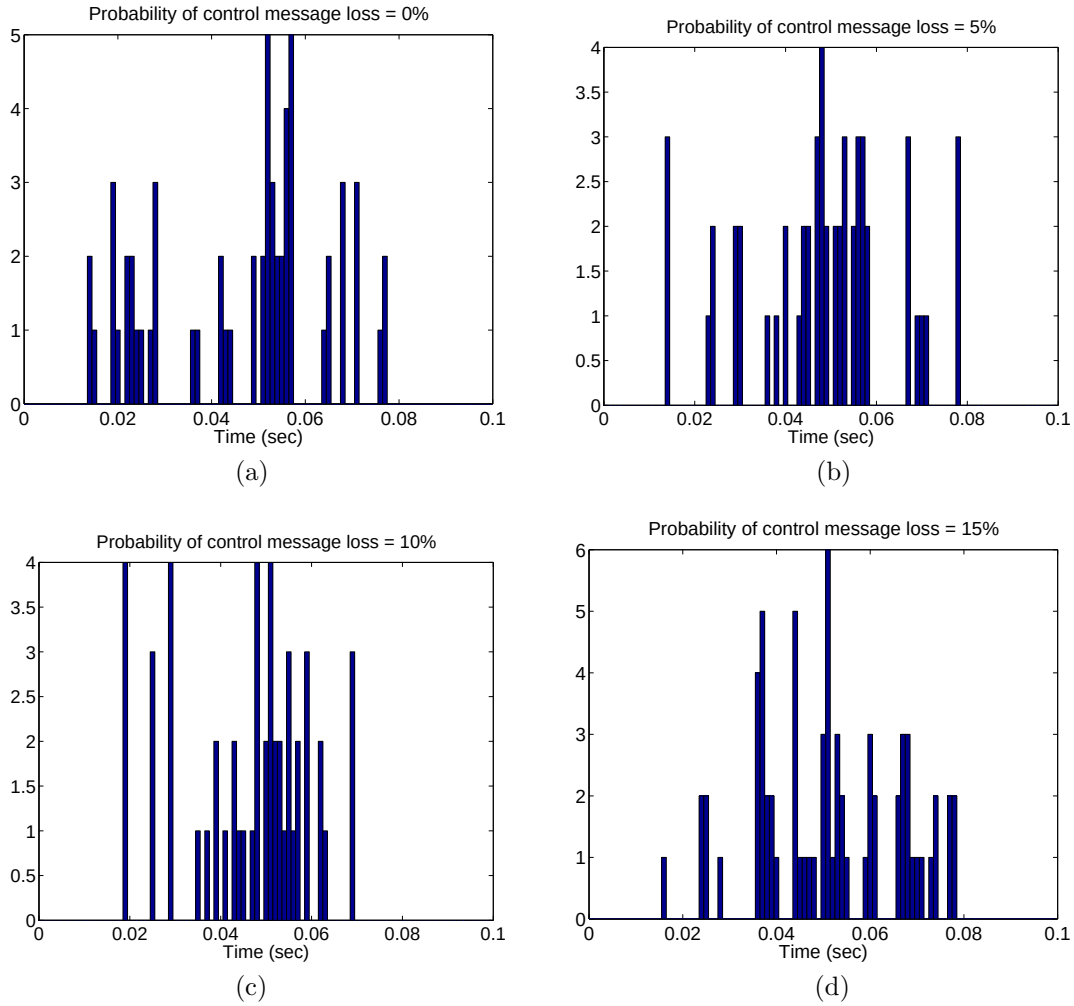

Figure 4: Settling time distributions for routing change recovery

7 Conclusion

Over its thirty years of operation, the Internet has been delivering data packets based solely on their destination addresses. As the Internet continues to evolve, however, there is an increasing need for the routers to maintain incoming interface tables in addition to the forwarding tables. These incoming tables can be used to facilitate the RPF checking used in all multicast routing protocols, and to validate the source address carried in each packet. The ability to enforce source address validity, in turn, can help prevent source address spoofing in malicious attacks that have been occurring frequently in the operational Internet, as well as help facilitate network diagnostics because network operators can now easily trace the origin of packets that cause erroneous effects.

The SAVE protocol [11] was developed to meet the above need. However, successful deployment of any new protocol requires that the protocol be designed with an incremental deployment path. This paper presents iSAVE, the incrementally deployable SAVE. iSAVE can successfully deliver source address validation updates to all iSAVE-speaking routers, even when the updates must cross legacy routers. In addition, iSAVE has also made a few improvements to the original SAVE design. While SAVE proactively delivers validating

updates, iSAVE supports both proactive (triggered by traffic) and reactive (triggered by interface conflicts) validation. Compared to SAVE validation updates that are typically addressed to address spaces, iSAVE validation messages only need to reach single addresses, which can be easily encapsulated in regular IP packets.

We have evaluated iSAVE's effectiveness in terms of the coverage of protected address spaces, the convergence time after detecting incoming interface discrepancy, and the bandwidth and storage cost. The behavior of iSAVE in the presence of spoofing attacks is also examined. Our study shows that even when only partially deployed, iSAVE is effective in protecting a high percentage of source address spaces. Once an ISP or an individual administrative domain deploys iSAVE, it can be assured of protection from most spoofing efforts by third parties.

References

[1] The Internet Traffic Archive. LBL-PKT Traces. http://ita.ee.lbl.gov/html/contrib/LBL-PKT.html, 1994.

[2] The Internet Traffic Archive. DEC-PKT Traces. http://ita.ee.lbl.gov/html/contrib/DEC-PKT.html, 1995.

[3] F. Baker. Requirements for IP Version 4 Routers. RFC 1812, June 1995.

[4] S. Bellovin, M. Leech, and T. Taylor. ICMP Traceback Messages. http://search.ietf.org/internet-drafts/draft-ietf-itrace-01.txt, October 2001.

[5] H. Burch and W. Cheswick. Tracing Anonymous Packets to Their Approximate Source. In *Proceedings of 2000 Systems Administration Conference*, December 2000.

[6] K. L. Calvert, M. B. Doar, and E. W. Zegura. Modeling Internet Topology. *IEEE Communications Magazine*, 35(6), June 1997.

[7] Cisco. Unicast Reverse Path Forwarding. http://www.cisco.com/univercd/cc/td/doc/product/software/ios111/cc111/uni_rpf.htm, 2000.

[8] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2827, May 2000.

[9] National Laboratory for Applied Network Research. NLANR Packet Header Traces. http://pma.nlanr.net/Traces/Traces/long/auck/2/, 2000.

[10] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. RFC 2401, November 1998.

[11] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang. SAVE: Source Address Validity Enforcement Protocol. In *Proceedings of INFOCOM 2002*, June 2002. To appear.

[12] University of California, Los Angeles, Computer Science Department. UCLA CSD Packet Header Traces. http://lasr.cs.ucla.edu/ddos/traces/, August 2001.

[13] K. Park and H. Lee. On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack. In *Proceedings of Infocom 2001*, April 2001.

[14] K. Park and H. Lee. On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets. In *Proceedings of ACM SIGCOMM 2001*, August 2001.

[15] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical Network Support for IP Traceback. In *Proceedings of ACM SIGCOMM 2000*, August 2000.

[16] R. Stone. CenterTrack: An IP Overlay Network for Tracking DoS Floods. In *Proceedings of 9th USENIX Security Symposium*, August 2000.

[17] Computer Emergency Response Team. CERT Advisory CA-2000-01 Denial-of-Service Developments. http://www.cert.org/advisories.

[18] H. Tyan. *Design, Realization and Evaluation of a Component-based Compositional Software Architecture For Network Simulation*. PhD thesis, Ohio State University, 2001.