Efficient and Secure Source Authentication with Packet Passports

Xin Liu and Xiaowei Yang
Department of Computer Science
University of California, Irvine
{*xinl,xwy*}@*ics.uci.edu*

David Wetherall and Thomas Anderson
Department of Computer Science & Engineering
University of Washington
{*djw,tom*}@*cs.washington.edu*

Abstract

A key challenge in combating Denial of Service (DoS) attacks is to reliably identify attack sources from packet contents. If a source can be reliably identified, routers can stop an attack by filtering packets from the attack sources without causing collateral damage to legitimate traffic. This task is difficult because attackers may spoof arbitrary packet contents to hide their identities.

This paper proposes a packet passport system to address this challenge. A packet passport efficiently and securely authenticates the source of a packet. A packet with a valid passport must have originated from the claimed source. The packet passport system can be incrementally deployed without introducing extra control messages. It also provides incentives for early adoption: a domain that deploys packet passport system can prevent other domains from spoofing its source identifiers. Our preliminary analysis suggests that the packet passport system can be implemented at high-speed routers with today's technologies.

1 Introduction

DoS flooding attacks have posed an increasing threat to the reliability of the Internet. An early study showed that DoS attacks occurred at a rate of nearly 4000 attacks per week [23]. Servers, ranging from critical infrastructure such as the root DNS servers [5], to commercial web portals such as yahoo [20], to personal web sites [14], may all fall victim to those attacks. More recently, DoS attacks have been used for online extortion [8]. If this trend continues, it will significantly undermine the potential of the Internet to support mission critical applications, online commerce, and real-time communications.

A key challenge in combating DoS attacks is to reliably identify attack sources from packet contents. This ability can enable a number of DoS defense and prevention mechanisms. Routers can stop an ongoing attack by filtering packets from the attack sources without causing collateral damage to legitimate traffic. They can enforce source-based resource allocation mechanisms such as weighted fair queueing to prevent attack sources from overwhelming legitimate sources. It's also possible to limit network-layer reflector attacks if attack sources that spoof their addresses are reliably identified. More importantly, the ability to identify the sources of an attack alone may deter future DoS attacks.[1]

It is difficult to reliably identify the source of a packet because attackers may spoof arbitrary packet contents to hide their identities. A number of proposals, such as ingress and egress filtering [13], path-based source address validation [21, 22, 24], path marking schemes [4, 30], and SPM [3] all intend to address this challenge. These proposals can effectively limit source address or path spoofing if they are deployed globally and the routing system is trustworthy. However, a single weak link, e.g., a subverted router, or an undeployed region, allows address or path spoofing.

The goal of our work is to provide an efficient and secure mechanism to authenticate the source of a packet. By efficient we mean that routers can authenticate the source of a packet at line speed; by secure we mean that the identify of a source cannot be forged even if part of the routing system is compromised or has not deployed the authentication mechanism.

Our solution is the packet passport system. A packet passport authenticates the origin of a packet as the packet travels from the source to the destination. Routers at passport checking points can independently validate a passport without trusting the rest of the Internet. A packet passport is cryptographically unforgeable, and can be used by a destination as the proof-of-victim to accuse an attack source, and by routers as a reliable source identifier to enforce policies like precisely filtering packets from an attack source.

Our design only requires light-weight Message Authentication Code (MAC) computation at packet forwarding time and is suitable for high-speed routers. It introduces no extra control messages, and provides incentives for early adoption. ISPs that deploy the pass-

[1] According to [15], today's attacks rarely use spoofed addresses because they are already effective without spoofing. However, this situation may change in the future if source-based automated filtering mechanisms are deployed.

port system can immediately prevent other domains from spoofing the passports of their own hosts. It is in contrast to ingress filtering, with which early adopters cannot prevent other domains from spoofing their addresses.

The rest of this paper is organized as follows. We state our design assumptions and threat model in Section 2. Section 3 describes the design of the passport system. We present a preliminary feasibility evaluation in Section 4. Section 5 compares our work with related work. We conclude in Section 6.

2 Threat Model and Assumptions

We assume that both hosts and routers can be compromised. When hosts or routers are compromised, they may modify, duplicate, or attempt to forge the passports of other sources. The goal of our design is to defend against such attacks. Compromised routers may drop packets or modify packet contents, but we do not address this type of attack.

In this paper, we present an architectural design and assume we can change both routers and hosts software. We use the terms *domain* and *AS* (Autonomous System) interchangeably. For clarity, we describe the key aspects of the design at the domain-level, and omit the detailed operations at the router-level. We also assume that a path-vector routing protocol such as BGP is used for inter-domain routing and each border router has the AS path information for each reachable address prefix.

3 Design

The purpose of a passport is to provide an authentic source identifier that can be used by the network to enforce policies such as filtering or rate limiting, and by destinations as proof-of-victim. An ideal packet passport should satisfy the following requirements:

1. A packet passport is unspoofable. Only the real source can produce passports that claim the source as the origin.
2. Domains should be able to verify packet passports without trusting other domains.
3. A packet passport can be efficiently generated and verified at packet forwarding time. Routers should be able to discard packets with invalid passports as close to the actual source as possible.
4. A valid passport cannot be replayed to send multiple packets. Otherwise, an attacker that intercepts a packet with a valid passport may replay the packet to flood a destination, causing the destination to blame an innocent source for the damage.
5. The passport system can be bootstrapped without much manual configuration or out-of-band communication.
6. The passport system itself is robust against DoS attacks.

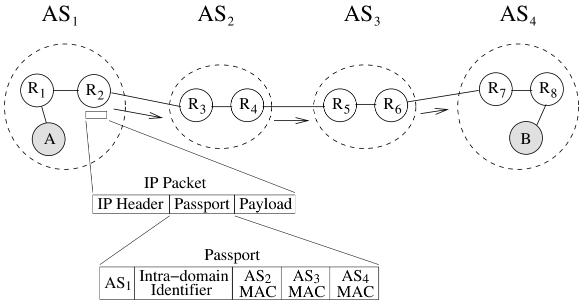

Figure 1: **An example of a packet passport.**

We describe how we gear our design to meet these requirements.

3.1 Unforgeable and Efficient Passports

Conceptually, a packet passport can be implemented via digital signatures. A source signs its packets, and routers validate the digital signatures with the source's public key. We discard this approach as digital signatures are computationally expensive to generate and verify. Instead, our design uses a light-weight MAC (such as UMAC [19][2] or HMAC [18]) to implement a packet passport. A packet passport consists of a sequence of AS numbers and corresponding MACs, each computed using a secret key known only by the source domain and a passport-checking domain en route to the destination. These secret keys are distributed as described in Section 3.2. A host sends a packet without a passport. When the border router of the source domain receives the packet, it verifies that the packet is originated from a host within the domain and stamps a passport. When a transit domain receives the packet, it validates the corresponding MAC using the secret key it shares with the source domain. As the key is only known by the source domain and the transit domain, if the MAC matches, it suffices to show that the packet is originated from the source domain. With this design, even if the border routers of a domain are compromised, they cannot forge the passports of other domains.

A packet passport may be implemented as an IP option or as a shim layer. Domains that request passport checking examine the option header or the shim layer, and other domains ignore the passports. Figure 1 shows an example, in which we assume all domains have deployed packet passport system and request passport checking. Let $K(AS_i, AS_j)$ denote the secret key shared between AS_i and AS_j. Suppose the host A in AS_1 sends a packet

[2] The packet ID field in the passport, as shown in Figure 2, can serve as the nonce required by UMAC.

to the host B in AS_4. The border router R_2 stamps a passport that has three MACs. Each MAC is computed using a key $K(AS_1, AS_j), j = 2, 3, 4$, and covers part of the packet as described in Section 3.4. When the packet arrives at AS_2, the border router R_3 uses the secret key $K(AS_1, AS_2)$ to verify that the packet comes from AS_1. Similarly, R_5 and R_7 each uses the corresponding key shared with AS_1 to verify the origin of the packet.

Our design uses a two-level hierarchy for host identification. A source identifier (carried in a passport) consists of a domain identifier (e.g., AS number) and an intra-domain host identifier. The passport issued by a border router only allows a remote domain to verify the origin domain of a packet, not the source host of a packet. We assume that the origin domain uses a separate intra-domain identification mechanism to verify that the host identifier is not spoofed. Our design does not restrict how a domain implements intra-domain host identification. A domain may use any general mechanism to implement the function. It is outside the scope of this paper to discuss these mechanisms, but we sketch a few possibilities.

In one extreme, if a domain can completely prevent source address spoofing inside itself with mechanisms such as fine-grained ingress filtering [13] or SAVE [21], it can use the source address as the intra-domain host identifier. Universal prevention of source address spoofing inside a domain is achievable as a domain is under a single administration. It's also possible that a domain uses the hardware address of a network interface card to identify a host, as commonly used by wireless access points for access control. In another extreme, a domain may use a cryptography-based approach similar to the inter-domain packet passport to identify a host. When a host is connected to the network, it is configured with a secret key shared with the routers of its domain. A host uses this key to authenticate itself inside its domain.

The hierarchical structure of our design improves scalability, but sacrifices security to some extent. A rogue or compromised domain may fake host identifiers in its passports. However, we do not think this is a problem in practice. Routers may allocate resources on a per-domain basis, such as using per-domain weighted fair queueing to allocate its bandwidth. This will discourage a domain from faking host identifiers to obtain more bandwidth share. Similarly, routers on the path may filter or rate-limit all packets from a source domain if the number of hosts that send malicious traffic from the domain exceeds a threshold. This will prevent a domain from launching attacks using spoofed host identifiers.

In our design, each transit domain can independently verify packet passports. This allows routers to drop packets with invalid passports as early as possible and prevents these packets from congesting downstream links. However, it has the side effect of binding a passport to an AS path. In the process of routing convergence, the actual AS path a packet traverses may differ from the one in its passport that is obtained from the source domain's BGP table, and then the packet may be dropped by a transit domain. However, this limitation may not be a serious problem in practice, because even without the passport system there is no guarantee that a packet can reach its destination in case of routing inconsistency. Alternatively, routers may demote packets with invalid passports as low priority packets. When there is no congestion, such packets can still arrive at their destinations.

3.2 Automated Key Distribution

A domain needs to share a secret key with a source domain in order to validate the packet passports from the source domain. The key distribution process must be automated in order to be feasible. In our design, keys are distributed within the inter-domain routing system using a Diffie-Hellman key exchange protocol [9]. We assume each domain has a public/private key pair, and all domains agree on two system-wide Diffie-Hellman parameters p and g.

The key distribution process works as follows. A domain AS_i announces its public key PK_{AS_i} and a public value $d_{AS_i} = g^{r_{AS_i}} \ mod \ p$ to all other domains. r_{AS_i} is a private value chosen by AS_i. The public key and the public value can be piggybacked in a domain's address prefix announcements. With BGP, both PK_{AS_i} and d_{AS_i} can be embedded into one or more prefixes announced by AS_i as optional and transitive path attributes. This design makes the passport system incrementally deployable. To prevent an attacker from modifying the public value d_{AS_i}, AS_i signs d_{AS_i} with its private key.

An attacker may attempt to falsify the public key PK_{AS_i} attached in prefix announcements, i.e. replace it with a different key and use the corresponding private key to sign a fake d_{AS_i} of its choice. This will compromise the Diffie-Hellman key exchange between AS_i and other domains. To prevent this type of attack, our design uses the same mechanism that the routing system uses to prevent the falsification of prefix announcements. For instance, PK_{AS_i} can be certified by a trusted certificate authority (CA) such as ICANN or regional Internet registries as in sBGP [17]. Other approaches such as web-of-trust [29, 33] can also be used to bind PK_{AS_i} to its origin AS AS_i.

The distribution of public keys may be eliminated with Identity Based Encryption infrastructure [6] (IBE). However, we discard the IBE approach because of its rigid key revocation mechanism and its requirement for a single public key generator.

When a domain AS_i receives a prefix announcement from a domain AS_j, it receives d_{AS_j} and vice versa. The two domains then share a Diffie-Hellman secret key:

$K(AS_i, AS_j) = d_{AS_i}^{r_{AS_j}} \ mod \ p = d_{AS_j}^{r_{AS_i}} \ mod \ p$. It is computationally infeasible for an attacker to obtain $K(AS_i, AS_j)$ even if he can eavesdrop the public values d_{AS_i} and d_{AS_j}.

A domain can use a dedicated server such as a route reflector or RCP server [7] to generate r_{AS_i} and d_{AS_i} and to synchronize all the keys on its border routers. If a router reboots and loses its keys, it may obtain them from this dedicated server or other routers in the same domain. It's possible that an entire domain AS_i loses all keys (other than its public/private key pair) due to failures. In this case, after it is back on line, it may acquire the public keys and the public values of other domains from its neighbors, similar to a BGP table transfer after a router reboots. At the same time, it starts announcing prefixes with the latest d_{AS_i} so that the secret key $K(AS_i, AS_j)$ stored in AS_j can be updated.

Distributing keys within the routing system has two primary advantages. First, it allows the passport system to bootstrap. Before keys are properly distributed, a domain cannot send packets with valid passports to cross domains. Distributing keys within the routing system solves this problem. As routing packets are exchanged between adjacent domains, adjacent routers can be configured to accept each other's routing packets without requiring a passport. Second, distributing keys within the routing system allows us to protect the key distribution channel from DoS attacks, i.e. preventing key distribution messages from being dropped due to attacker flooding. As the routing system is a "closed" system, i.e. routers only accept routing messages from known peers, it is possible to configure the routing system to prevent malicious hosts from injecting routing packets. For instance, two adjacent routers may be configured to only accept routing packets from the port connecting to the peer with a TTL equal to 255; non-directly connected routers (such as iBGP peers) may be configured to communicate using MPLS tunnels. Routers can then forward and process routing packets with the highest priority. Normal data traffic cannot congest the routing channel, and therefore cannot DoS the key distribution process. And if the routing channel is congested, a router can determine the misbehaving peer and cut off that peer.

3.3 Prevent Replay Attacks

The packet passport system we described so far prevents forgery, but does not prevent replay attacks. An attacker that intercepts a packet with a valid passport can duplicate the packet multiple times to launch flooding attacks. This may cause a router to block an innocent source whose packets are replayed by attackers.

Our design prevents replay attacks with a combination of rapid rekeying and bloom filters [12]. We discard timestamps because they require global clock synchronization, and we discard sequence numbers because they must be synchronized among border routers of a domain.

Our design uses a hash chain to provide rapid rekeying without excessive key distribution messages. A source domain AS_i does not use the secret key $K(AS_i, AS_j)$ to generate and validate MACs directly. Instead, both AS_i and AS_j use this key to seed a hash chain $\{K_m(AS_i, AS_j) | K_m(AS_i, AS_j) = Hash^m(K(AS_i, AS_j))\}$ that is used in a descreasing order. Each $K_m(AS_i, AS_j)$ is used to generate and validate MACs for a short interval such as a few seconds. When stamping a passport, the source domain attaches m to specify which keys are used. For instance, in the example shown in Figure 1, m being 2 indicates that the keys used to generate the MACs are $\{K_2(AS_1, AS_j), j = 2, 3, 4\}$. A transit or destination domain uses the index m in the passport to locate the proper key and validates the passport. A lower index $m - 1$ invalidates all keys with a higher index, thereby preventing replay attacks. A transit or destination domain AS_j may keep a window of two keys $K_{m-1}(AS_i, AS_j)$ and $K_m(AS_i, AS_j)$ to deal with out of order packet delivery or out of sync key indexes, i.e. a border router of the source domain AS_i may advance to $K_{m-1}(AS_i, AS_j)$ while other routers of AS_i are still using $K_m(AS_i, AS_j)$.

Domains also need to periodically change the secret keys $\{K(AS_i, AS_j)\}$ for better security. This requires periodic distribution of new $\{d_{AS_i}\}$, which can happen at a large time scale. For instance, if each key $K(AS_i, AS_j)$ lasts for a day, then a hash chain of a length 17280 will allow m to change every five seconds.

Rapid rekeying prevents an attacker from replaying a packet passport a few seconds old. However, an attacker may still replay sniffed traffic on the fly, e.g. duplicating every sniffed packet right after it receives the packet. To prevent this type of attack, we use a bloom filter to track packet passports generated with the same hash key, similar to the technique used in [27]. We add a 64-bit packet ID field in a packet passport. A border router of a source domain will stamp a unique ID into the packet passport and include the ID in its MAC computations. (To ensure uniqueness in the same domain, each border router may start with a different ID space.) When a router in a transit domain receives the packet and validates its passport, it hashes the packet ID and the origin AS number into a bloom filter, and discards the packet if the bloom filter indicates the passport is a duplicate.

A bloom filter has a false positive rate, but this rate can be reduced to a very small value by increasing the filter size. Moreover, it is even possible to accommodate the false positives by allowing a small number of "duplicate" passports to pass. For instance, a router can keep a counter of how many duplicates its bloom filter

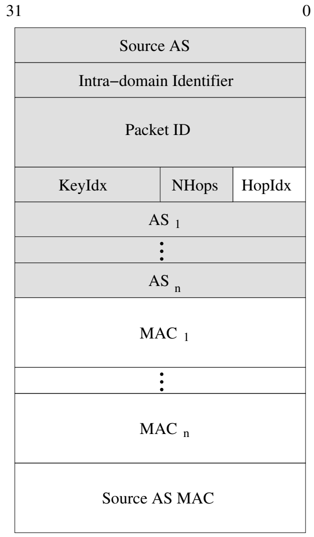

Figure 2: **The format of a packet passport.**

has reported. The counter is reset when the bloom filter is refreshed. If the counter is smaller than a threshold, the router assumes the duplicates are actually false positives and would not discard them; otherwise, the router knows that passport replay attacks are going on and discard any duplicate. The false positive rate of a bloom filter can be computed given its parameters [12], and the threshold can be set such that without attacks all false positives can pass the router. Allowing some duplicate passports to pass gives attackers the possibility of more advanced passport replay attack; the impact of such mechanisms is still under investigation.

3.4 Passport Format

Figure 2 summarizes the format of a packet passport. The source AS field and the intra-domain identifier field together constitutes a source identifier.[3] The packet ID field is used for bloom filter hashing. The KeyIdx is the current hash key index m as described in 3.3. The NHops and HopIdx fields specify the total number of domain-level hops and the current hop. The rest of the passport is the AS path the packet takes, followed by a sequence of MACs. The MACs are computed in a descreasing order, i.e. MAC_n is computed first and MAC_1 is computed last. MAC_j will be validated by AS_j when the packet is forwarded to its destination. MAC_j is computed from the secret key shared between the source AS and AS_j, the source and destination addresses in the IP header, the shaded fields in Figure 2, and all the existing MACs $\{MAC_i, i > j\}$[4]. The last MAC, the source AS MAC, is computed by the source AS using a key known only to itself. This MAC is for the source AS to verify that a passport is generated by itself. We note that if we include more fields and the payload in the packet into the MACs, we can also protect the integrity of the packet. But this requires more computation at packet forwarding time. It's our future work to evaluate the cost and benefit of protecting packet integrity at the network-layer.

Our design uses a 64-bit MAC for each AS hop. The length of the MAC is a tradeoff between security and the packet header overhead. As the hash key used to compute MACs changes rapidly, we think that a 64-bit MAC offers acceptable security.

3.5 Applications

The immediate application of a packet passport is to filter unwanted traffic, because a passport shows the real source of a packet. Suppose host B in Figure 1 classifies packets from host A as attack traffic and wants to block traffic from A. B could send filtering requests to AS_1 as described in [4] to block A. The packet passports stamped by AS_1 would serve as a proof-of-victim, because B cannot possibly forge the passports.

Packet passports may also be used to limit reflector attacks. If each transit domain checks whether the source address of a packet belongs to its origin domain, cross-domain source address spoofing is effectively eliminated and network-layer reflector attacks are limited such that the attackers and the victim must be in the same domain. If a domain can further eliminate source address spoofing inside itself, attackers in this domain cannot perform network-layer reflector attacks at all.

Packet passports may enable new types of resource allocation schemes. For instance, ISPs can queue packets based on their domain identifiers. This scheme would allow ISPs to assign different weights to different domains based on how much they pay. In addition, this also enforces fate sharing among packets from the same domain, and encourages local security. If a compromised host launches DoS attacks, it will only congest the queue for its domain and cause damage to other hosts in the same domain.

3.6 Incremental Deployment

Our design supports incremental deployment and provides incentives for early adoption. Domains can embed their keys in BGP prefix announcements as optional and transitive path attributes. Domains with passport system would extract keys from these attributes, while domains without passport system would pass on these attributes to

[3] The current AS number in the Internet is 16-bit, but IETF is working on the transition to 32-bit AS numbers. Our design assumes a 32-bit AS number.

[4] This way, if a malicious router corrupts MACs for down-stream domains in a packet, the packet would be dropped at the next passsport-checking router and could not consume more bandwidth.

other domains. In case of a partial deployment, the AS path in a passport generated by the source domain only includes the domains that support passports. Passport-enabled domains will validate the passports, and treat legacy packets and packets with invalid passports with low priority. Domains without passport system would ignore the passports and forward the packets.

Early adopters of the passport system can gain the following advantages. First, a domain that deploys the passport system can prevent other domains from spoofing its source identifiers. Second, it can identify any attack source from another domain that also supports the passport system. Third, it can locate any attack source in itself when a victim presents the passports of the attack traffic. Fourth, it can avoid liability for attacks that claim to originate from it but are actually not. Lastly, its packets will treated with higher priority at domains that also support the packet passport system.

3.7 Packet Fragmentation

Unfortunately, packet passports are incompatible with fragmentations at intermediate routers. If an intermediate router fragments a packet, a fragment either has a duplicate passport, which will be caught by the bloom filters as a replay, or does not have a valid passport. Therefore, end hosts should use path MTU discovery to avoid fragmentations. We do not think this incompatibility a significant obstacle to deployment, because fragments are discouraged due to various performance and security issues. IPv6 has already disabled the use of fragmentations at intermediate routers.

4 Feasibility Analysis

It requires a comprehensive experimental study to evaluate the performance and feasibility of our design. As a first step, we provide a preliminary analysis to assess the feasibility of the design. Our design has three primary sources of cost: 1) the computational cost for passport generation and validation; 2) communication and computational cost for key distributions; 3) computational and memory cost introduced by bloom filters.

First, we estimate the computational cost for passport generation and validation. With our design, the source domain of a packet needs to compute L MACs to generate a passport, where L is the length of an AS path. A transit domain or the destination domain needs to compute one MAC to validate a passport. MAC functions such as UMAC [19] can be computed efficiently on modern processors. Speed test with UMAC implementation from [1] shows that a Pentium(R) 4 2.80GHz processor is able to compute 3.9 million HMACs per second over 64-byte blocks. If we assume the average AS path length is four [10], the input used to compute a packet passport is less than 64 bytes. A commodity PC can generate 975K passports and validate 3.9 million passports per second. In a router we expect that special hardware can perform MAC generation at a much higher speed.

Second, we analyze the communication and computational cost for key distributions. With BGP, the key distribution process requires that one public key (probably with certificate) and one digitally signed message with AS_i and d_{AS_i} as the content be appended to each UPDATE message, and d_{AS_i} be refreshed every key distribution interval. With hash chains, the key distribution interval can be relatively long, typically 24 hours, and therefore we think both the communication and computation cost is affordable.

Finally, we look at the memory cost of bloom filters. Our design uses bloom filters to prevent passport replay within the short life span of a hash key, which is on the order of seconds. Bloom filters have a false positive rate $P = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - e^{-kn/m}\right)^k$, in which k is the number of hash functions, m is the size of the filter in unit of bits, and n is the total number of key values that are already in the filter [12]. When $k = 8$, and the memory efficient factor m/n is 32, we can achieve a false positive rate of 5.7×10^{-6}, which we think is acceptable in the Internet. With these parameters, if we use a filter of 16MB SDRAM and assume an average packet size of 400 bytes [2], a bloom filter can “remember” 2.5Gbps traffic for 5 seconds. For a window of two overlapping keys as described in Section 3.3, two bloom filters with a total memory of 32MB can prevent replay attacks from a 2.5Gbps incoming link.

5 Related Work

Most related work falls into two categories. The first category includes work on preventing source address spoofing, i.e., ingress filtering [13], SAVE [21], route-based filtering [24], and reverse path filtering [22]. In contrast to packet passports, source addresses are not self-verifiable. A destination ISP cannot trust a source address unless it assumes the rest of the network has eliminated source address spoofing.

There is also work on detecting source address spoofing near the destination, such as Hop-count Filtering [16]. The routers near the destination can simply drop packets with spoofed source addresses. However, this type of approaches cannot identify the attacking source and can only protect the last-hop link.

Another category of work focuses on providing unspoofable identifiers with cryptographic primitives. Perlman’s work on sabotage-proof routing uses heavy-weight public-key digital signatures [26], while our work uses light-weight MACs. PI [30] and AITF [4] use a path identifier to approximate an unspoofable source identifier. Routers insert secure tokens into a packet before they forward the packet. Although PI and AITF have lower packet header overhead, a notable challenge for them is that a source can prepend a path identifier with arbitrary values, and a compromised router can also forge the path identifiers. In contrast, packet passports are cryptographically unforgeable. Source identifiers protected by packet passports can be directly used in filter expressions to block attack sources.

Authenticated Marking Scheme [28] also uses MACs to protect router-inserted path identifiers, but the keys are revealed to verifiers after generating MACs. A verifier has to store received packets for a period of time before being able to verify the MACs. In contrast, packet passports can be verified on the fly, and do not require additional protocol messages to reveal MAC keys.

SPM [3] uses secrets shared between domains as proof-of-source, but the secrets are put into packet headers as plain text and therefore vulnerable to eavesdroppers. Moreover, the secret distribution of SPM is not DoS resilient: if links used to distribute the secrets are flooded, the secret distribution itself would fail. Our packet passport system can accommodate eavesdroppers, and our key distribution is fully protected against DoS attacks.

The Visa protocol [11] shares similarity with our work. A packet carries an exit visa to leave its source domain, and an entry visa to enter a destination domain. The Visa protocol requires per-connection Visa request, but our work does not. The visa request channel still needs to be protected from DoS attacks and address spoofing. The Visa protocol can use our passport system to protect its request channel. In a similar vein, capability-based systems such as SIFF [31] and TVA [32] and ticket system [25] can also use packet passports to protect their request channels.

6 Conclusion

This paper proposes a packet passport system that securely and efficiently authenticate the source of a packet. A packet passport is crytographically unforgeable. Routers at passport checking points can independently validate the authenticity of a packet passport without trusting the rest of the network. Destinations can use the passports from attack packets to accuse the attack sources, and routers can use them to validate source identities and block attack sources.

We present the design of a packet passport system that uses a light-weight MAC, introduces low message overhead, and is incrementally deployable. A preliminary study suggests that the design is feasible with today's hardware technology.

References

[1] UMAC implementation. `http://www.cs.ucdavis.edu/~rogaway/umac/`.

[2] Caida report. `http://www.caida.org/analysis/AIX/plen_hist/`, February 2000.

[3] B.-B. A. and L. H. Spoofing prevention method. In *Proc. IEEE Infocom*, 2005.

[4] K. Argyraki and D. Cheriton. Active Internet Traffic Filtering: Real-Time Response to Denial-of-Service Attacks. In *USENIX 2005*, 2005.

[5] DDoS attacks still pose threat to Internet. BizReport, 11/4/03.

[6] D. Boneh and M. Franklin. Identity based encryption from the weil pairing. *SIAM J. of Computing*, 32(3):586–615, 2003.

[7] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, and A. S. andJacobus van der Merwe. Design and implementation of a routing control platform. In *Proc. of NSDI*, 2005.

[8] Extortion via DDoS on the rise. Network World, 5/16/05.

[9] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[10] D.Magoni and J.J.Pansiot. Analysis of the autonomous system network topology. *Computer Communications Review*, 31(3):26–37, July 2001.

[11] D. Estrin, J. C. Mogul, G. Tsudik., and K.Anand. Visa protocols for controlling inter-organization datagram flow. *IEEE Journal on Selected Areas in Communication*, 7(4):486–498, May 1989.

[12] L. Fan, P. Cao, J. Almeida, and A. Z. Broder. Summary cache: A scalable wide-area web cache sharing protocol. Technical Report 1361, Department of Computer Science, University of Wisconsin-Madison, 1998.

[13] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks that Employ IP Source Address Spoofing. Internet RFC 2827, 2000.

[14] S. Gibson. The strange tale of the attacks against grc.com. `http://grc.com/dos/grcdos.htm`, 2005.

[15] A. Greenhalgh, M. Handley, and F. Huici. Using routing and tunneling to combat DoS attacks. In *Proc. of USENIX SRUTI*, 2005.

[16] C. Jin, H. Wang, and K. G. Shin. Hop-count filtering: an effective defense against spoofed ddos traffic. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 30–41, 2003.

[17] S. Kent, C. Lynn, and K. Seo. Secure border gateway protocol (s-bgp). *IEEE Journal on Selected Areas in Communications*, 2000.

[18] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-hashing for message authentication. Internet RFC 2104, 1997.

[19] T. Krovetz. UMAC: Message authentication code using universal hashing. Internet RFC 4418, 2006.

[20] R. Lemos. A year later, DDoS attacks still a major web threat. `http://news.com.com/A+year+later,+DDoS+attacks+still+a+major+Web+threat/2009-1001_3-252187.html`, Feb. 2001.

[21] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang. SAVE: Source address validity enforcement. In *Proc. of INFOCOM*, 2002.

[22] K. Miller. Three practical ways to improve your network. In *Proc. of USENIX LISA*, 2003.

[23] D. Moore, G. Voelker, and S. Savage. Inferring Internet Denial of Service Activity. In *Usenix Security Symposium*, Aug. 2001.

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

[25] B. Patel and J. Crowcroft. Ticket based service access for the mobile user. In *Proceedings of MobiCom '97*, pages 223–233, 1997.

[26] R. Perlman. Network Layer Protocols with Byzantine Robustness. MIT Ph.D. Thesis, 1988.

[27] A. Snoeren, C. Partridge, L. Sanchez, C. Jones, F. Tchakountio, S. Kent, and W. Strayer. Hash-Based IP Traceback. In *ACM SIGCOMM*, 2001.

[28] D. Song and A. Perrig. Advance and Authenticated Marking Schemes for IP Traceback. In *Proc. IEEE Infocom*, 2001.

[29] R. White. Securing bgp through secure origin bgp. *The Internet Protocol Journal*, 2003.

[30] A. Yaar, A. Perrig, and D. Song. Pi: A Path Identification Mechanism to Defend Against DDoS Attacks. In *IEEE Symposium on Security and Privacy*, 2003.

[31] A. Yaar, A. Perrig, and D. Song. SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks. In *IEEE Symposium on Security and Privacy*, 2004.

[32] X. Yang, D. Wetherall, and T. Anderson. A DoS-Limiting Network Architecture. In *ACM SIGCOMM*, Philadelphia, PA, Aug. 2005.

[33] H. Yu, J. Rexford, and E. Felten. A distributed reputation approach to cooperative internet routing protection. In *Proceedings of Workshop on Secure Network Protocols*, 2005.