Unspoofable Source Identifiers without Global Trust

Xin Liu (student) 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

1. INTRODUCTION

Unspoofable source identifiers in packets are a basic building block in combating Denial of Service (DoS) attacks. Routers may rely on these identifiers to precisely block attack traffic or enforce fair resource allocation. It is also possible to restrict or even eliminate reflector attacks with these identifiers. More importantly, the ability to identify the source of a packet alone may deter future DoS attacks.

The Internet community has long realized the importance of unspoofable source identifiers. A number of proposals [2, 5, 4, 6], all intend to prevent source address spoofing so that a source address can serve as a weak source identifier. Unfortunately, source addresses are not verifiable: given a packet, the destination cannot trust the source address unless it assumes that the entire network has eliminated source address spoofing. However, in a heterogeneous environment such as the Internet, we think it is too strong an assumption to assume global trust, especially considering that attackers may compromise routers to inject packets with spoofed source addresses. Other cryptography-based approaches either use expensive public-key digital signatures [7] or do not prevent spoofing when there are eavesdroppers or compromised routers [1, 8].

This paper addresses the problem of providing an unspoofable source identifier without assuming global trust. We design a packet passport system, with which every packet carries a "passport" that shows its source and can be verified by routers at packet forwarding time. A key contribution of our design is its feasibility. First, our design only requires light-weight Message Authentication Code (MAC) computation at packet forwarding time and is suitable for high-speed routers. Second, a router can verify a packet passport independently, i.e. without trusting the rest of the Internet. Third, our design prevents passport replay attacks and DoS attacks to the passport system itself. Lastly, our design supports incremental deployment and provides incentives to early adoption. ISPs that deploy the passport system can immediately prevent other domains from spoofing the passports of their own hosts. This approach is in contrast to ingress/egress filtering, with which early adopters cannot prevent other domains from spoofing their addresses.

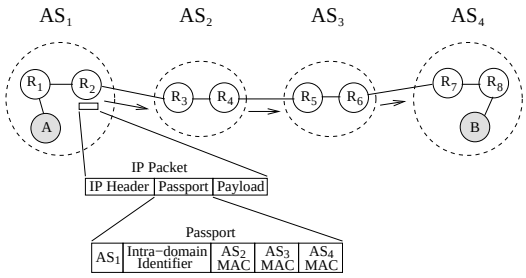

Figure 1: An example of a packet passport

2. DESIGN OVERVIEW

Conceptually, packet passports can be implemented via digital signatures. A source signs its packets, and routers verify 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 makes use of light-weight Message Authentication Code (such as HMAC [3]). A packet passport consists of the AS path that the packet is going to traverse and a sequence of MACs. Each MAC corresponds to one domain in the AS path and is computed using a secret key known only by the source domain and the corresponding domain. Each MAC covers the passport itself as well as some packet header fields such as the destination address. The secret keys are established as described in Section 2.1. A host sends a packet without a passport. When a 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 verifies the corresponding MAC in the passport 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 the source domain are compromised, they cannot forge the passports of other domains. In addition, each domain can independently authenticate the origin of a packet without trusting other domains.

Figure 1 shows an example of a packet passport. 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 to the host B in AS_4. The border router R_2 stamps a passport that

has three MACs, each computed using a key $K(AS_1, AS_j)$, $j = 2, 3, 4$. 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 passport.

Our design uses a two-level hierarchy for host identification. A source identifier in a passport consists of a domain identifier, i.e. the AS number of the source domain, and a host identifier, i.e. the intra-domain identifier. Our design does not restrict how a domain implements its intra-domain identifier, like an inter-domain routing protocol never specifies how intra-domain routing should be done. For instance, if a domain can completely prevent source address spoofing inside itself using mechanisms such as SAVE [4], it can use the source address as the intra-domain identifier.

The hierarchical structure of our design improves scalability, but sacrifices security to some extent. The problem is that an intra-domain identifier can only be verified within its source domain. A malicious or compromised domain may forge arbitrary intra-domain identifiers in the passports it generates. In this case, other domains may choose to ignore its intra-domain identifiers and do fair resource allocation or filtering only based on the source domain identifier.

2.1 Key Distribution

A domain needs to share a secret key with a source domain in order to verify the packet passports from the source domain. In our design, keys are distributed within the inter-domain routing system such as BGP. This allows the passport system to bootstrap, because key distribution messages cross domain boundaries in eBGP traffic, which does not need to carry passports. Moreover, as the routing system is a "closed" system, i.e. routers only accept routing messages from known peers, we can make the routing system resistant to DoS attacks, which implies the key distribution be resistant to DoS attacks. For example, routing traffic may be forwarded with highest priority so that normal data traffic cannot congest the routing channel. If the routing channel is congested by routing traffic, misbehaving routers can be easily located and then repaired or disconnected.

Our key distribution uses Diffie-Hellman key exchange protocol. Domain AS_i generates a private value r_{AS_i}, calculates the corresponding public value $d_{AS_i} = g^{r_{AS_i}} \bmod p$, and piggybacks d_{AS_i} into its address prefix announcements. When domain AS_j receives a prefix originated from AS_i, it can get d_{AS_i}; similarly, AS_i can get d_{AS_j} in the same way. Then AS_i and AS_j have established a secret key $K(AS_i, AS_j) = d_{AS_i}^{r_{AS_j}} \bmod p = d_{AS_j}^{r_{AS_i}} \bmod p$.

d_{AS_i} has to be bound to AS_i so that compromised BGP routers cannot inject fake d_{AS_i}. This can be done by requiring every domain have a public/private key pair, signing d_{AS_i} with AS_i's private key, and distributing AS_i's public key in the same way as distributing d_{AS_i}. To be secure AS_i's public key also has to be bound to AS_i; this can be achieved using the same mechanism that binds an address prefix to a domain. For instance, if there is already a PKI to secure inter-domain routing, this PKI can be reused to certify all the public keys in the passport system.

2.2 Prevent Replay Attacks

The packet passport system must be able to prevent passport replay attacks, because compromised routers may replay packets to launch DoS flooding attacks. If the passport system is combined with a filtering mechanism, a victim cannot locate the compromised routers and therefore cannot effectively block the replayed packets. Our design prevents replay attacks with a combination of bloom filters and fast re-keying. We discard timestamps because they require global clock synchronization, and we discard sequence numbers because they must be synchronized among multiple border routers of a domain.

Bloom filter is a space-efficient probabilistic data structure that is used to test whether or not an element is a member of a set. Ideally a router can "remember" every passport it has seen with a bloom filter and easily catch replayed passports. However, a bloom filter has limited storage space. On a high-speed router, in order to keep a low false positive rate, it has to be flushed every few seconds and therefore can only "remember" a particular passport for a few seconds. To solve this problem, our design requires the secret keys used to generate and verify passports change rapidly so that when a passport is forgotten by a bloom filter, it has already become invalid due to the key changes. This is called fast re-keying.

We achieve fast re-keying without frequent key distribution by using hash chains. The key $K(AS_i, AS_j)$ is only used to seed a hash chain $\{K_m | K_m = Hash^m(K(AS_i, AS_j))\}$, and the new hashed keys are used sequentially for passports, each only lasting a few seconds. Every passport contains a key index m showing what keys are used to generate it, and transit domains can locate the proper keys to verify the passport using the key index. Every domain also records the latest key indexes it has seen from other domains, so that when a passport with an old key index is received, it is considered a replayed passport and will be discarded or demoted.

A more detailed design and a feasibility analysis can be found at:

http://www.ics.uci.edu/~xinl/pktpassport-sruti06.pdf

3. REFERENCES

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

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

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

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

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

[6] 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.

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

[8] 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.