Amendment to: Highly Secure and Efficient Routing

Ioannis Avramopoulos and Hisashi Kobayashi
Dept. of Electrical Engineering
School of Engineering and Applied Science
Princeton University, Princeton, NJ 08544
{iavramop, hisashi}@ee.princeton.edu

Randolph Wang
Dept. of Computer Science
School of Engineering and Applied Science
Princeton University, Princeton, NJ 08544
rywang@cs.princeton.edu

Arvind Krishnamurthy
Dept. of Computer Science
Yale University
New Haven, CT 06520
arvind@cs.yale.edu

I. Introduction

This is an addendum to paper "Highly Secure and Efficient Routing" that appears in the Proceedings of IEEE Infocom 2004 [1]. The purpose of the addendum is to amend two security vulnerabilities that we found in the protocols that are proposed therein. We would like to kindly ask the readers of "Highly Secure and Efficient Routing" to reference this addendum in addition to the published paper.

In particular, we will modify Sections III.B, III.C, and V.

II. Regarding Section III.B of [1]

In Section III.B of [1] we state that the authentication structure, consisting of message of authentication codes (MACs), that is used for packets, should be used for ACKs and FAs as well. However, if this structure is used for ACKs and FAs, then it gives the adversary the advantage to discredit any link in the path between the source and the adversarial router. We will illustrate this with an example. Consider a path $< s, n_1, n_2, n_3, x, \ldots >$. In this path, s, n_1, n_2, and n_3 are non-faulty, whereas x is faulty. On receipt of a packet from s, router x generates an FA with valid MACs for routers n_3 and n_2, and an invalid MAC for router n_1. Routers n_2 and n_3 will forward the FA, whereas n_1 will drop it and, thus, the source will account a fault to non-faulty link (s, n_1).

In the following we amend this vulnerability by providing a novel way to construct the authenticators of ACKs and FAs. ACKs and FAs must satify two properties. First, they must be impractical to forge. Otherwise the adversary will be able to deny packet delivery without the detection of the location of the packet delivery failure even if the source and destination are non-faulty (by forging ACKs) and will be able to discredit non-faulty elements (by forging FAs). In both cases Byzantine robustness would be violated. Second, if an ACK or FA verifies at one non-faulty router in the path, then it must verify at all other non-faulty routers in the path. Otherwise, the adversary would gain the opportunity to discredit non-faulty elements as shown above.

We assume that the source s shares secret keys with all the routers $n_1, \ldots, n_i, \ldots, n_m$ in the path that communication will be carried out. We denote $K_s^{n_i}$ the secret key shared between the source s and router n_i. We also assume a secret key is shared for every pair of neighboring routers in the path. We will use a one-way hash function $h(\cdot)$. Given y and $h(\cdot)$ it is impossible to derive any x such that $h(x) = y$.

Date: February 24, 2004

Suppose that the sequence number of a packet is k and its source route is $< s, n_1, \ldots, n_i, \ldots, n_m >$, where s is the source and n_m is the destination. The source constructs m hash chains each of length three. The first element $r_i^0(k)$ of the hash chain for node i, $i = 1, \ldots, m$, is constructed by concatenating the sequence number k and the key $K_s^{n_i}$. The second and third elements $r_i^1(k)$ and $r_i^2(k)$ are constructed by applying a one-way hash function $h(\cdot)$ to the previous element. We will call k the *authenticator seed* or *seed*, $r_i^1(k)$ the *authenticator*, and $r_i^2(k)$ the *authenticator anchor* or *anchor*.

Subsequently the source announces with the packet the anchors, i.e., elements $r_i^2(k)$, which are protected by the authentication tag of the packet. The authentication tag of the packet is as in [1]: a MAC is computed for each downstream router with the requirement that the MAC for router i is computed on both the packet and the MACs for routers $j, j > i$. Each recipient n_i is able to construct $r_i^0(k)$ by concatenating the seed with the secret key shared with the destination and then $r_i^1(k)$ by applying $h(\cdot)$. The latter element is used as the authenticator for the FA, if n_i is an intermediate router, or the ACK.

A. Security

The security of the protocol relies on the secrecy of the keys and the aforementioned one-way property of hash function $h(\cdot)$. I.e., a router j cannot derive the authenticator $r_i^1(k)$ for any other router i since j cannot infer $r_i^0(k)$ because of the secrecy of $K_s^{n_i}$ and since it is impossible to derive any x such that $h(x) = r_i^2(k)$. We should point out that a faulty source can cause ACKs or FAs to be dropped at non-faulty routers by providing wrong authenticators (i.e., an incorrect value of $r_i^2(k)$). Requiring from downstream routers to sanity check that $h(r_i^1(k)) = r_i^2(k)$ would add unnecessary cryptographic overhead since the source has other means to create the same effect [1].

III. Regarding Section III.D of [1]

The aforementioned authentication mechanism does not impose *prefix span* restrictions on the path calculation algorithm.

IV. Regarding Section V of [1]

In Section V.A of [1] two optimizations are proposed for fault-free paths. Packets are distinguished in two types: *normal* and *query*. FAs do not need to be generated for normal packets but, in the event of drops, enough state is maintained for query packets to identify the locations of packet failures. Normal packets still require a destination ACK which, for correctness, must now be authenticated as in above (using a single hash element). Modifying the fault location identification mechanism using the correct structure is straightforward.

In the second optimization of the same section the source authenticates control parameters only of the packet to intermediate routers but provides a MAC of the full packet to the destination. The destination can, therefore, detect a modification and, in such event, reflect in the ACK the hash of the (modified) packet, which intermediate routers compare with the hash of their stored packet in order to detect the location of the modification. This protocol requires the destination to authenticate an arbitrary message to upstream routers and, therefore, a single hash element is not sufficient. Even with an ACK of the structure of [1] the protocol is correct provided that the source associates faults with the destination only (as the destination can cause the ACK to be dropped at any non-fault router, and, thus, discredit any non-faulty link).

Section V.B of [1] proposes a mechanism to mitigate the delay that malicious router can introduce to packets without being identified. This mechanism requires from downstream routers to communicate to their correspoding upstream routers, in a secure way, information on the delay that packets experience in them. Two methods to communicate this information are suggested. In the first, each router authenticates to every upstream router the delay information using the authentication structure of data packets. This method is vulnerable to the aforementioned attack (i.e., it gives adversarial routers the possibility to discredit any of their upstream links by forging MACs). The second method requires a single destination ACK. This method is secure provided that

- faults are associated with the given destination only and
- the authentication tag of the ACK uses the structure of data packets in Awerbuch et al.

We next provide the reason that the authentication tag of ACKs in [1] is not sufficient. Consider the following path: $< s, \dots, x_1, u, v, x_2, \dots, t >$. In this path routers x_1 and x_2 are malicious. Suppose that the packet from the source s contains a query for the delay. Router x_1 instead of forwarding the packet to u it detours the packet to x_2. Router x_2 before forwarding the packet appends bogus query replies that appear to originate from u and v. Destination t will reply with an ACK that will arrive at x_2 which will detour the packet to x_1. The source will accept bogus query replies from u and v.

This vulnerability can be amended if the authentication tag of the ACK has the same structure as the source authentication tag of Awerbuch et al., i.e., with an encryption step in addition to the MAC computation, that enforces the ACK to follow the complete list of routers in the path. Therefore, the approval of the ACK by routers u and v in the above example cannot be bypassed.

References

[1] I. Avramopoulos, H. Kobayashi, R. Wang, and A. Krishnamurthy, "Highly secure and efficient routing," in *Proc. IEEE Infocom 2004*, Hong Kong, Mar. 2004.