Active Networks Authentication

Lawrence Cheng, Chris Todd, Alex Galis

Electrical Engineering Department, University College London, Torrington Place, London, WC1E 7JE, UK.

Abstract: This paper suggests that due to the dynamic nature of active networking, active network authentication should be both end-to-end and hop-to-hop based. This paper outlines the drawbacks suffered by existing active network authentication technique such as Secure Active Node Transfer System (SANTS) [2] in terms of scalability, efficiency and security. This paper then describes a generic solution in which Safe and Nimble Active Packets (SNAP) [8] is used as the underlying active packet language.

1. Introduction to Active Network Authentication Challenges

Authentication in active networks faces a different set of challenges from authentication in conventional networks. In conventional networks, authentication is usually *end-to-end* based: an end client wishing to be authenticated by a server located at the other end of the network (and vies versa). In active networks, end-to-end authentication is needed as well as *hop-to-hop* authentication [1][2]. The latter refers to: *each recipient (of an active packet) should authenticate the received active packet based on where the packet has been modified.* The need to enforce hop-to-hop authentication is due to the dynamic nature of active networks.

Unlike conventional passive networks, active networks are dynamic. Active packets that carry executable code are injected by active service providers (ASP) on behalf of their clients. The embedded code in active packets is executed in the execution environments (EEs) on the desired active nodes for the purpose of service control. The desired active nodes are known as the *intermediate active nodes*: these nodes are managed by the ASPs and are located in-between the end client and the service provider (SP), the latter offers privileged services to the end clients. It should be note that the results of code execution are push back onto the active packets before the packets are forwarded to their next hop. Thus, unlike conventional packets, the contents of active packets may vary as they traverse through the network. In order to ensure that intermediate active nodes do not process contaminated active packets, active packets must therefore be authenticated in an end-to-end and hop-to-hop manner.

2. Existing Solutions

Hicks developed a Packet Language for Active Networks (PLAN) [3]. He addressed the need of enforcing hop-to-hop authentication in active networks and made a proposal on using cryptographic techniques to protect the authenticity of PLAN packets. However, asymmetric authentication would require each intermediate active node to be equipped with a distinctive private key for digitally signing active packets. Each intermediate node then signs the modified part on the packet as the packet traverses. Digital signing each of the modifications on each traversing packet at each intermediate node would generate an undesirable performance overhead. Also, signing a packet at an intermediate node may overwrite previous signatures. In contrast, symmetric authentication requires a pre-shared key among intermediate nodes. The shared private key is used at each intermediate node to protect the authenticity, confidentiality and integrity of inter-node communication. We will discuss shortly how we may solve our authentication problem with symmetric authentication.

Murphy had proposed Secure Active Node Transfer System (SANTS) [2][4]. ANTS (Active Node Transfer System) packets are split into static and dynamic parts. Static parts will be digitally signed by the source whereas the dynamic parts of an ANTS packet will be protected by hop protection e.g. HMAC-SHA1. The split ANTS packet is encapsulated in a modified version of Active Network Encapsulation Protocol (ANEP)[1] prior transmission. At each modifying node, the codes embedded on a packet will be executed; the respective credential reference (of that modifying node) will be kept on the packet. A SANTS packet therefore keeps a list of credential references as it traverses through the

[1] The Payload field of ANEP is split into Static and Dynamic Payload in SANTS.

active network. On arrival, these references are extracted and are compared to the corresponding certificates that are stored in DNSSEC [5][6].

Active packet splitting is a known solution to the depicted active network authentication problem in this paper. However, SANTS may suffer from: 1) *scalability*: SANTS uses a modified version of ANEP for packet encapsulation. Special ANEP-SANTS packet dispatcher must be equipped at each intermediate active node in order to intercept ANEP-SANTS packets. 2) *Efficiency:* ANTS packets must be split before ANEP encapsulation may be applied. However, ANTS packets have a data-code inter-mixed structure. The lack of a clear distinction between data and code in ANTS packets implies that at each intermediate node, the contents of an ANTS packet must be individually analysed, and then the node must decide which parts of the packet is static and which part is dynamic. This may generate an undesirable performance overhead on the network. It should note that this efficiency problem is related to the choice of the underlying active packet language rather than of the design of SANTS. 3) *Security*: SANTS enforces hop-to-hop authentication by using DNSSEC for keeping certificates. This paper suggests that using DNSSEC to distribute public keys may expose the active networking system to certain risks, namely: *security* and *efficiency*. Massey outlined the security risks when using DNSSEC to distribute public keys in [7]. Under the DNSSEC approach, a DNS zone administrator should generate a unique set of key pair. The administrator uses his private key to sign authoritative DNS data as well as the certificates that it holds on behalf of the intermediate active nodes; whereas his public key is kept as a KEY record and is distributed to all resolvers upon request. When making a DNS query, a resolver will receive a) the requested DNS information, b) a digital signature (e.g. a *SIG record*). The resolver will then use the KEY record (distributed from the zone administrator) to validate the received SIG record. But how can the resolver validate the KEY record at the first place?

In general, public key (and certificate) validation can be enforced through a third-party e.g. a certificate authority (CA). Massey suggests that this would *not* be a solution for DNSSEC as DNS is the most fundamental service on the Internet, using external public key database would either 1) recreate the functionality of DNS or, 2) relying on (insecure) DNS to provide fundamental (information to reach the desire CA) [7]. SANTS may also suffer further in terms of efficiency if DNSSEC is in place. Under the approach of SANTS, when an active packet reaches its destination, each of the credential references on the packet will be extracted, and then the intermediate node will have to make out-of-band certificate queries for each of these credential references in order to validate the packet. Such arrangement will generate an undesirable performance overhead on the network as an out-of-band mechanism is in place to enforce credential check for each credential reference on each packet at its destination.

Given these problems, this paper suggests that an active network authentication system should contain: 1) a *standardised* encapsulation protocol that would eliminate the scalability problem in SANTS; 2) an active packet language that has a *clear* distinction between its data and code structure for efficient packet splitting; 3) an efficient *in-band* validation mechanism for enforcing hop-to-hop authentication.

3. Authentication in FAIN Active SNMP EE

In the Future Active IP Networks (FAIN) project [9], Safe and Nimble Active Packets (SNAP) was used as the underlying active packet protocol in the Active SNMP EE. The developers of SNAP claims SNAP to be an efficient programming language (SNAP latencies are no more 10% slower than IP's [8]) that provides active packets at high level of safety [8]. The discussion on SNAP in FAIN is beyond the scope of this paper, relevant information can be found at [9][15]. Essentially, SNAP active packet programs carry a series of byte code instructions, a stack and a heap. The embedded instructions are executed in the Active SNMP EE for the purpose of service control on FAIN active nodes. Example instructions are: *forw* (move on if not at destination), *here* (push current node address onto the stack), and *getsrc* (get source field) etc. These instructions are static: once they are generated by the source (i.e. the ASP), they will not be modified whilst the packet is in-transit. On the other hand, SNAP packet programs keep variable data on its stack and its heap. This paper proposes to use SNAP as the underlying active packet language as it has a distinctive packet structure that separates its static and dynamic data. Noted that, however, SNAP is designed to be a light weight active packet

language, thus there is no authentication facility provided for within SNAP at all: the SNAP developers claim that the first line of defence of SNAP is that: *SNAP must not be used to exert control over a node* [8]. This is certainly a drawback to an active packet that meant to be practical. We will show in later section how we may overcome this drawback of SNAP in our approach.

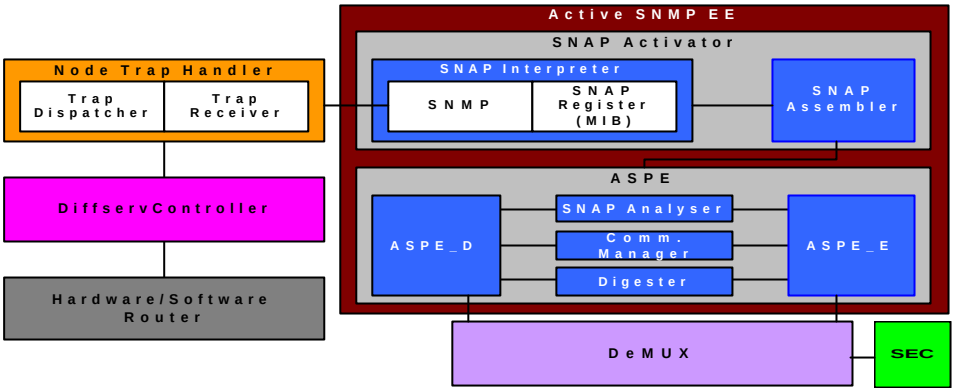

Figure 1 – The FAIN active node architecture

Figure 1 shows various components of the FAIN active node. It should be note that node integrity protection is enforced through the Security Manager (SEC). Node integrity is protected by strong policy-based authorisation. Relevant details can be found at [1]. We will discuss later on how we may determine a contaminated node should this strong authorisation fails. A SNAP active packet is injected by SNAP Activator. To avoid the scalability problem in SANTS, we used the *standard* ANEP format. The DeMultiplexer (DeMUX) on FAIN active nodes is able to process any types of active packets that are encapsulated in the standard ANEP format. In this way we can achieve an *architectural-independent* solution. Different contents of SNAP active packets are then encapsulated into respective field of ANEP by ANEP-SNAP Packet Engine (ASPE). The static SNAP contents will be encapsulated in ANEP Payload, the entire of the SNAP packet (which includes the dynamic data) will be encapsulated into ANEP Option. The result is known as an *ANEP-SNAP packet.* Note that we eliminate the scalability problem in SANTS by using the standard ANEP format.

N	4N+0 Byte	4N+1 Byte	4N+2 Byte	4N+3 Byte
0	Version	Flags	Type ID	
1	Header length		Packet length	
2	Option's flag		Option length	
m	SNAP packet(~bytes)			
M+1	Payload length			
n	SNAP static content(~bytes)			

Figure 2 – ANEP-SNAP packet format[2] [3]

At the point of injection (i.e. at the ASP), the static contents of the ANEP-SNAP packet (the Payload) will be digitally signed by the source. Whilst the packet is traversing through the network, hop-to-hop protection is enforced. Intermediate nodes must establish a trusted form of relationship among themselves through the creation of negotiated security associations (SA). We propose that symmetric cryptography techniques such as IPSEC or SSL are ideal for identification between neighbouring active nodes. Within a SA, neighbouring nodes can achieve peer authentication plus inter-node integrity and confidentiality protection of active packets [1]. As mentioned earlier, we propose to use an in-band mechanism for hop-to-hop authentication. A SNAP packet will be marked with the network identifier e.g. IP address of the current node as it traverses through the network. Since some network location identifiers e.g. datalink identifiers, Media Access Control (MAC) addresses are more

[2] Several ANEP Options are used for keeping virtual environment (VE) ID, EE ID, SNAP packet… etc. Not all options are shown in Figure 2 for simplicity.

[3] The Payload Length field keeps the length of the SNAP static content, the latter is needed by SNAP Activator in order to process SNAP packets. This is because the ANEP Payload field may contain dummy bytes (to fulfil the 4 byte boundary).

unique, they may form better candidates than the others (e.g. IP addresses) as network-location identifiers. The network-location-authentication approach discussed in this paper can make use of either of them. To simplify the discussion, IP addresses are used as the network-location identifier in this paper. This is essentially a form of the *IP traceback* technique [10][11][12][13][14]. Upon the arrival of the packet at a recipient (i.e. an intermediate node / the final destination), the recipient will be able to determine through which active nodes that the received packet has traversed. Network location identifiers are generally insufficient for strong authentication. However, as SA is enforced for inter-node communication, the network location identifiers on the packet are protected by the pre-shared private key at each node.

SNAP provides an efficient way of marking packets with network location identifiers. SNAP provides the *push* byte code instruction that pushes the current node address onto the stack of SNAP. We avoid stack values being overwritten by subsequent nodes by keeping a record of the network resource bound value on the packet. For example should the network resource bound value is set to be 6, the zero to the third byte of the stack would be used to keep the IP address of the first intermediate node. The next IP address will be marked on the fourth to the seventh byte etc. Note that this IP traceback method does not protect node integrity, but provides an efficient way of determining on which active nodes that an active packet has traversed. With the list of traversed nodes we can limit our scope when determining spoofed nodes should the authorisation technique described in [1] fails to operate.

4. Conclusion

Due to the dynamic nature of active networking, this paper identified that it is necessary to enforce both end-to-end and hop-to-hop authentication in active networks. Hop-to-hop authentication was defined as: *each recipient to authenticate the received active packets based on where the packets has been modified*, and should be applied to the dynamic data of active packets. End-to-end authentication was defined as*: the recipient to authenticate the received active packets based on the source's digital signature on the packets*, and should be applied to the static data of active packets. This paper identified the several drawbacks that were experienced in existing solutions, namely: *scalability*, *efficiency* and *security*. This paper described a solution that: a) used the standard ANEP packet format to avoid the scalability problem; b) used SNAP as the underlying active packet language to eliminate the efficiency problem. SNAP clearly distinguished its static and dynamic contents to be its byte codes and its stack respectively, which enabled us to perform a clean separation of active packets; c) enforced hop protection to protect the integrity and confidentiality of the entire ANEP-SNAP packet, hop authentication was achieved via an in-band mechanism that follows the concept of IP traceback.

Acknowledgments

Part of this paper describes the work undertaken by the authors in the context of the FAIN–IST 10561 project, a 3 years project during 2000-2003. The IST programme is partially funded by the Commission of the European Union (EU). The authors would like to acknowledge all FAIN partners.

References

[1] FAIN Internal Report R25.2 "SNAP, ANEP, and Security", Nov 2002 **http://www.ist-fain.org**
[2] S. Murphy, "Strong Security for Active Networks", IEEE OpenArch 2001
[3] M. Hicks, "A Secure PLAN", IWAN 1999, July 1999, vol.1653
[4] D. J. Wetherall, "ANTS: a toolkit for building and dynamically deploying network protocols", OpenArch 1998, San Francisco, CA, April 1998, pp.117-129, IEEE.
[5] D. Eastlake, "RFC 2535 – Domain Name System Security Extensions", March 1999.
[6] D. Eastlake, "RFC 2538 – Storing Certificates in the Domain Name System", March 1999.
[7] D. Massey, Ed. Lewis, O. Gudmundson, R. Mundy, A. Mankin, "*Public Key Validation for the DNS Security Extension*", IEEE 2001.
[8] SNAP (Safe and Nimble Active Packets), **http://www.cis.upenn.edu/~dsl/SNAP/**
[9] FAIN (Future Active IP Networks), **http://www.ist-fain.org**
[10] S. Savage, D. Wetherall, "Network Support for IP Traceback", IEEE/ACM Transactions on Volume 9 Issue 3, 2001, pp.226 – 237.
[11] T. Baba, S. Matsuda, "Tracing Network Attacks to their Sources", IEEE Internet Computing, March/April 2002 (Vol. 6, No. 2), pp.20-26.
[12] R. Stone, "CenterTrack: An IP Overlay Network for Tracking DoS Floods", 2000 USENIX Security Symp., 2000, pp. 199-212.
[13] S. M. Bellovin, "ICMP Traceback Messages", Internet Draft: draft-bellovin-itrace-00.txt, 2000.
[14] D. X. Song, A. Perrig, "Advanced and Authenticated Marking Schemes for IP Traceback", INFOCOM 2001, Vol. 2, pp. 878-886.
[15] W. Eaves, L. Cheng, A. Galis, "SNAP Based Resource Control for Active Networks", GLOBECOM 2002.