Message Authentication Systems - History and Standards

Pushkar Joshi
ppj@eecs.berkeley.edu

2-25-2004 - 1^{st} half

1 Introduction

This lecture is about various crytographic message authentication systems in practice. Message authentication emerged as a significant field only about ten years ago, which is around the time that people started thinking of methods to ensure message integrity.

The first task was to standardize MAC (the IP-SEC) standard. Most standard MAC's were based on cryptographic hash functions (such as SHA1 or MD5) which are one way, collision free hash functions that reduce the number of bits from input to output and behaved like a random function.

2 Ad-Hoc Heuristics

Using cryptographic hash functions, several ad-hoc heuristics for message authentication were constructed.

Suppose we have a message m; one can encrypt the message by using a cryptographic hash function as:

1. $F_k(m) = \text{SHA1}(k \| m)$
2. $F_k(m) = \text{SHA1}(m \| k)$

3. $F_k(m) = \mathrm{SHA1}(k\|m\|k)$

Several such heuristics were formulated. However, none of them were theoretically proven to be secure. Once an attacker understood the underlying structure of the hash function, it was possible to break these heuristics.

For instance, if given SHA1(x) and y, it was possible to compute SHA1($x\|y$) *without* knowing x. This allowed one to easily break the first heuristic.

It was also proven that for a relatively small key size, one could create internal collisions within the hash function structure. That could be used to break the second and third heuristic and made the use of impractically long keys necessary.

3 HMAC

HMAC was proposed as a provably secure message authentication heuristic and is now the de-facto standard in message authentication systems. It is formulated as:

$$F_k(m) = SHA1(k \oplus C_1 \| SHA1(k \oplus C_2 \| m))$$

where C_1 and C_2 are some constants.

Given some explicit assumptions on the hash function, the above was shown to be secure.

3.1 NMAC

The security of HMAC was proven by using the security of a smaller construct, the NMAC, which used two keys k and k'.

$$NMAC_{k,k'}(m) = SHA1(k \| SHA1(k' \| m))$$

The term $SHA1(k'\|m)$ can be considered a 2-universal hash, and $SHA1(k\|SHA1(k'\|m))$ can be thought of as a PRF. The assumption is that if the 2-universal hash is at least ϵ-universal and the PRF is a good PRF, then NMAC is a secure MAC.

With SHA1, it is plausible to assume that it is hard to find two messages that collide without given access to the output. This proves that the 2-universal hash function is ϵ-universal.

To prove that the second part acts as a good PRF, re-write it as:

$$E_k(x) = SHA1(k \| x), x \in \{0,1\}^{160}$$

(where 160 is the length of SHA1 output).

Given the definition of SHA1, the above behaves like a good PRF, hence we can deduce that NMAC is a good MAC.

The assumptions necessary for NMAC to work as a secure MAC are rather weak, making it feasible to use for message authentication.

3.2 NMAC to HMAC

Given the security of NMAC, we can deduce security of HMAC. However, unlike NMAC, we have only one key in HMAC. To construct an NMAC-like structure, one can send $k \oplus C_1$ along with the initialization vector to the hash function block to get an output that is sufficiently random from $k \oplus C_2$. That is, we can use $k \oplus C_1$ as one key and $k \oplus C_2$ as the other key. This allows us to use HMAC with only one key.

4 Integrity Authentication Systems in Practice

Now we shall describe some of the message authentication systems in practice.

4.1 SSH ver. 1

SSH is used in client-server communication to ensure message integrity and authentication. Each message m is appended with a CRC checksum and then encrypted before being sent. That is, the packet sent over the network is

$$E_k(m \| CRC_{32}(m))$$

The same key is used in both directions (client-to-server and server-to-client).

4.1.1 Attacks against SSH ver. 1

- Reflection Attack

An attacker can intercept a message from a server and send it back to the server. Since both the client and server use the same key, the message from the attacker will be accepted as valid, thereby compromising message integrity.

- Forge CRC

  If one is using a stream cipher (like RC4) for encryption, an attacker can modify bits in the CRC checksum to send a forged message that will be accepted by the server as valid.

4.2 WEP

WEP is the message authentication system used in 802.11 wireless protocols. It is implemented in the link layer and is the method used to encrypt/decrypt TCP packets exchanged between clients with wireless cards and a base station. The base station decrypts messages sent by clients and forwards the decrypted packet to the internet, and encrypts messages sent from the internet before forwarding them on the wireless web.

As in SSH ver. 1, all clients and base stations use the same key. Since this is a link layer, a stream cipher is used. The data sent over the wireless network is of the form

$$RC4(k \| ctr) \oplus (m \| CRC_{32}(m))$$

where m is usually a TCP packet.

4.2.1 Attacks against WEP

There are several methods to compromise the security of WEP. Here are a couple interesting examples:

- IP Redirection Attack

  An attacker can flip bits in the message corresponding to the IP header of the TCP packet (and patch the appropriate CRC and TCP checksums) to redirect the message to a malicious or unintended server on the internet *after* decryption by the base station. This is a good example where failure of integrity leads to failure of confidentiality.

- Retrieving Message using TCP Checksum

  Assume the attacker knows what messages are TCP packets. TCP has its own checksum (a weak, 16 bit checksum). Suppose we flipped two bits in message m, say bits

	IND-CPA	INT-CTXT
SSH ver. 1	Y	N
WEP	N	N
old IPSEC	Y	N

Table 1: Security Comparison of Integrity Authentication Systems

15 and 31. That is, we XOR the message with the bit string 0x0001000100... Then, the checksum of the modified message TCPCheckSum($m \oplus$0x00010001000....) is equal to that of the unmodified message TCPCheckSum(m) when $m_{15} \neq m_{31}$.

Then, the attacker can intercept the message from a client, and modify the message by XOR-ing it with the desired bit string (say, as above, 0x00010001000...)and send it to the base station. The base station will then decrypt the message and forward it to the server, which will return either an ACK (if the TCP checksum was successful) or a NACK (if the TCP checksum failed). The encrypted response from the server will be obtained by the attacker, and depending on the size of the response, the attacker will be able to tell whether the server responded with an ACK or a NACK. Based on that information, the attacker is able to figure out whether m_{15} and m_{31} are equal, that is, gain some information about the message. The same method can be used to obtain information about the rest of the message, thereby compromising confidentiality.

4.3 Security Comparison of Integrity Authentication Systems

Table 1 compares the security of three systems we have seen in class: SSH ver. 1, WEP and old IPSEC (from second half of lecture).