Message authentication and authenticated encryption

Security Protocols (bmevihim132)

Dr. Levente Buttyán
associate professor
BME Híradástechnikai Tanszék
Lab of Cryptography and System Security (CrySyS)
butyan@hit.bme.hu, butyan@crysys.hu

Outline
- message authentication
  - naïve constructions
  - HMAC
  - CBC-MAC
  - CMAC
- authenticated encryption
  - generic combinations
  - CCM
  - OCB
  - GCM
Secret prefix method

\[ \text{MAC}_k(x) = H(k|x) \]

- insecure!
  - assume an attacker knows the MAC on \( x \): \( M = H(k|x) \)
  - he can produce the MAC on \( x'|y \) as \( M' = f(M,y) \), where \( x' \) is \( x \) with padding and \( f \) is the compression function of \( H \)

\[
\begin{align*}
&k \mid x_1 & & x_2 & & x_L \mid \text{padding} & & y \mid \text{padding'} \\
&f & & f & & f & & f & & f \\
&CV_0 & & M & & M' = \text{MAC}_k(x'|y) \\
\end{align*}
\]

A similar mistake

\[ \text{MAC}_k(x) = H_k(x) \]

where \( H_k(.) \) is \( H(.) \) with \( CV_0 = k \)

\[
\begin{align*}
&x_1 & & x_2 & & x_L \mid \text{padding} & & y \mid \text{padding'} \\
&f & & f & & f & & f & & f \\
&k & & f & & M & & M' = \text{MAC}_0(x'|y) \\
\end{align*}
\]
Secret suffix method

\[ \text{MAC}_k(x) = H(x|k) \]

- insecure if \( H \) is not collision resistant
  - using a birthday attack, the attacker finds two inputs \( x \) and \( x' \) such that \( H(x) = H(x') \) (can be done off-line without the knowledge of \( k \))
  - then obtaining the MAC \( M \) on one of the inputs, say \( x \), allows the attacker to forge a text-MAC pair \((x', M)\)
- weakness
  - key is involved only in the last step

Envelop method

\[ \text{MAC}_K(x) = H(k|x|k) \]

- a key recovery attack has been discovered on this scheme (requiring \( 2^{64} \) text-MAC pairs for MD5 with 128-bit key)
- although, not really practical, the attack still represents an architectural flaw
HMAC

\[ \text{HMAC}_k(x) = \text{H}((k^* \oplus \text{ipad}) \mid \text{H}((k^* \oplus \text{ipad}) \mid x)) \]

where

- \( h \) is a hash function with input block size \( b \) and output size \( n \)
- \( k^* \) is \( k \) padded with 0s to obtain a length of \( b \) bits
- \( \text{ipad} \) is 00110110 repeated \( b/8 \) times
- \( \text{opad} \) is 01011100 repeated \( b/8 \) times

\[ k^* \oplus \text{ipad} \]

\[ CV_0 \]

\[ CV_{\text{inner}} \]

\[ f \]

\[ CV_1 \]

\[ f \]

\[ f \]

\[ \text{HMAC}_k(x) \]

Encrypted hash

\[ \text{MAC}_K(x) = E_K(\text{H}(x)) \]

- off-line search for messages with colliding MAC values is possible here without the knowledge of \( k \) \( \rightarrow \) \( H \) must be collision resistant!
- collision resistant hash functions usually have larger output size than the block size of the block cipher \( \rightarrow \) which mode to use to encrypt the hash?
- two messages having the same hash value will have the same MAC value under all keys
CBC-MAC

- CBC MAC is secure for messages of a fixed number of blocks
- forgery is possible if variable length messages are allowed (see countermeasures later)

\[
X_1 \oplus \ldots \oplus X_N | 100\ldots
\]

A known-text forgery

- given two text-MAC pairs (X, M) and (X', M'), a third valid text-MAC pair can be computed as follows:
  \((X | 100\ldots | X'_1 \oplus M | X'_2 | \ldots | X'_L, M')\)
A chosen-text forgery

- given a known text-MAC pair \((X_1, M_1)\)
- request MAC for \(M_1\), receive \(M_2 = E_K(M_1 \oplus 0) = E_K(M_1)\)
- \(M_2\) is the MAC of the message \((X_1|0)\)

Another chosen-text forgery

- given two known text-MAC pairs: \((X_1, M_1), (X_2, M_2)\)
- request a MAC for message \(X_1|M_1 \oplus M_2 \oplus Z\), where \(Z\) is an arbitrary block
- receive \(M_3 = E_K(M_1 \oplus M_2 \oplus Z \oplus M_1) = E_K(M_2 \oplus Z)\)
- \(M_3\) is also the MAC for message \(X_2|Z\)
How to use CBC-MAC in practice?

- use the optional final encryption
  - reduces the threat of exhaustive key search (key is \((K, K')\) \(\rightarrow\) key length is doubled)
  - prevents the previously presented existential forgeries
  - has marginal overhead (only last block is encrypted multiple times)

- start the message with an extra block containing the length of the message

- use \(K\) to encrypt the length and obtain \(K' = E_K(\text{length})\), and use \(K'\) as the MAC key (i.e., use message dependent MAC keys)

CMAC

- proposed to fix problems with CBC-MAC

\[
\begin{align*}
X_1 & \quad E \quad K \quad Y_1 \\
X_2 & \quad E \quad K \quad Y_2 \\
& \quad \vdots \quad Y_{N-1} \\
X_N & \quad 100... \\
K' & \quad \text{computed from } E_K(0) \\
K & \quad E \\
& \quad \text{CMAC}_K(X)
\end{align*}
\]
**Authenticated encryption schemes**

- simultaneously protect the confidentiality and the integrity of a message

**motivations:**
- to prevent chosen-ciphertext attacks (such as the Vaudenay attack)
  - the decryption oracle immediately recognizes improperly constructed ciphertexts and refuses to decrypt them
  - attacker can construct a correct ciphertext only if he already knows the plaintext → decryption oracle becomes useless
- efficiency (in some cases)
  - needs fewer operations if the message is encrypted and the authentication tag is computed in a single pass

**approaches:**
- specialized schemes (e.g., CCM, OCB, GCM)
- generic combination of encryption and MAC (but be careful!)
  - $E_{k_1}(X|MAC_{k_2}(X))$ (check for padding oracle attack!)
  - $E_{k_1}(X)|MAC_{k_2}(X)$ (check for padding oracle attack!)
  - $E_{k_1}(X)|MAC_{k_2}(E_{k_1}(X))$ (considered to be the most secure approach)

**CCM – CTR mode with CBC-MAC**

- input: message $X$, key $K$, associated data $A$, nonce $N$
- output: encrypted message $Y$, authentication tag $T$ (encrypted CBC MAC value)
- single key
- two passes ($2n+m+2$ invocation of the block cipher)
OCB – Offset Code Book

- single key, single pass
- no associated data
- preserves plaintext length
- provably secure
- patented

\[
\text{checksum} = X_1 \oplus \cdots \oplus X_{n-1} \oplus Y_n^{0*} \oplus C
\]

\[
E^K Y_1 + \cdots + E^K Y_{n-1} + X_1 \cdots + X_{n-1} + Y_n^{0*} + C
\]

GCM – Galois/Counter Mode

- \(xH\) is multiplication with \(H\) over \(GF(2^{128})\) defined by the polynomial \(x^{128} + x^7 + x^2 + x + 1\)
- very efficient (~OCB)
- provably secure and not patented

\[
xH \quad \cdots \quad xH \quad \cdots \quad xH \quad \cdots \quad xH
\]

\[
E^K \quad \cdots \quad E^K \quad \cdots \quad E^K \quad \cdots \quad E^K
\]

\[
N|\text{ctr}_0 \quad \text{inc} \quad \text{ctr}_1 \quad \text{inc} \quad \cdots \quad \text{ctr}_n \quad 0^{128}
\]

\[
E^K \quad \cdots \quad E^K \quad \cdots \quad E^K \quad \text{len}(A_i Y)
\]

\[
A_1 \quad A_m \quad Y_1 \quad Y_n
\]

\[
xH \quad xH \quad xH \quad \text{trunc}\quad \text{trunc'}
\]

\[
\text{len}(A_i Y) \quad T
\]
Summary

- Naïve hash based MAC constructions are usually not secure
- Better to use standard, well-studied constructions, e.g., HMAC
- CBC-MAC
  - Interesting, because it does not need a hash function, but it can use the same block cipher that is used for encryption, anyway
  - Existential forgeries against CBC-MAC exist, but there are countermeasures (e.g., adding context data such as message length to the message, multiple encryption of the last block, etc.)
- Be careful with generic constructions for authenticated encryption
- Specific authenticated encryption modes have some advantages
  - Efficiency: the two goals may be achieved in a single pass
  - Security: no information is leaked through a padding oracle
- CCM
  - Single key, two pass, associated data, not patented
- OCB
  - Single key, single pass, no associated data, patented
- GCM
  - Single key, single pass, associated data, not patented
  - Performance is comparable to OCB