Cryptographic hash functions

\[ H : \{0, 1^*\} \rightarrow \{0, 1\}^L \]

SHA256  \( L = 256 \)

deterministic

\[ H(x) = \text{hash of } x \]

digest

fingerprint

Correctness: deterministic

Efficiency: computing \( H \) should be easy

Security:
1) One-way "preimage resistance"

\[ \forall \text{Adv}, \Pr[ x \leftarrow \text{rand}; y = H(x) : \text{Adv}(y) \rightarrow x ] \]
\[ = \text{negl} \left( \frac{1}{2^{\text{size of output of } H/L}} \right) \]

\[ \forall \text{Adv}, \forall x, \Pr[ \text{Adv}(H(x)) \rightarrow x ] = \text{negl} \]

2) Collision-resistance (CR)

It is infeasible to find \((x, x')\) s.t. \((x \neq x')\) and \(H(x) = H(x')\)

SHA256 is assumed currently to be CR
Also

\[ \text{hash(sourcecode)} \neq H \]

If an unknown code \( H \) is added to a website

\[ \text{source code} \]

\[ \star \text{Malice} \]

downloaded file

\[ \text{source code} \]
<table>
<thead>
<tr>
<th>Confidentiality</th>
<th>Symmetric-Key</th>
<th>Asym-Key</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>symmetric-key encryption (AES-CBC)</td>
<td>public-key enc (ElGamal)</td>
</tr>
<tr>
<td>MAC (AES-EMAC)</td>
<td></td>
<td>Digital signatures (RSA)</td>
</tr>
</tbody>
</table>
Alice \( \rightarrow \) M \( \rightarrow \) Bob (Bank)

"Transfer 1000 to Chris"

Authenticity: M comes from the expected user

Integrity: M was not modified

\[ \text{El Gamal} \quad (g^M \mod p) \]

Encryption does not provide these properties

\[ \frac{H \cdot g^K \mod p}{x^2} \]

\[ = 2 \cdot H \cdot g^K \mod p \]

Easy to modify
MAC (Message authentication code)

Alice \[ M, T \] \rightarrow Bob
K
\{ \}
MAC(K, M) = T

Correctness: determinism
Efficiency: compute MAC should be poly time
Security: EU-CPA existentially-unforgeable under chosen plaintext attack

Adv cannot forge a MAC for a new message
\[ \text{Challenger} \]

\[ K \]

\[ \array{ r c l }
\text{M}_i & \xrightarrow{\text{MAC}(K, M_i)} & \text{Adv} \\
\text{M} = \{ \text{M}_i \} & \leftarrow & \text{T} \\
\endarray \]

\[ \forall \text{Adv}, \quad \Pr[K \leftarrow \text{Keygen}(); \text{Adv}() \rightarrow (M, T) \]
\[ \text{s.t.} \quad M = \{ \text{M}_i \}; \quad \text{MAC}(K, M) = T \]

\[ = \text{negl} \]
AES-EMAC

Keygen() \rightarrow
Choose 2 $\$1$ keys $(K_1, K_2) = K$

MAC$(K, M)$:

Split $M$ into $P_1 \| P_2 \ldots \| P_n$

Propose $MAC'(K, M) = (S_1, \ldots, S_n)$ insecure

Cannot forge $MAC'$; Adv asks MAC $M = (P_1 \| P_2 \ldots \| S_1, S_2)$

$M^{\infty} = P_1$; MAC is $S_1$
Propose $\text{MAC}''(K, M) = S_n$ (insecure MAC for exercise)

Adv asks for $P_1 \rightarrow S_i$

$P_1' \rightarrow S_i'$

$P_1 || P_2 \rightarrow T$  \[\Rightarrow \text{points to needing } K_2 \text{ at the end}\]

Adv forges MAC for $(P_1; S_i \oplus P_2 \oplus S_i') \rightarrow T$
Alice

\[ C = \text{Enc}(K_E, M); \]

Bob

\[ K = (K_E, K_M) \]

\[ \text{MAC}(K_M, C) \]

both confidentiality

2 integrity/auth

1. Checks MAC

\[ \text{for } C \text{ using } K_M \]

\[ \text{MAC}(K_M, C) = T \]

2. Decrypt to get \( M \)

\[ \text{Dec}(K_E, C) = M \]