Cryptographic hash functions

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

SHA256, \( L = 256 \) are deterministic

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

Correctness: deterministic

Efficiency: computing \( H \) should be easy

Security:
4) One-way "preimage resistance"

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

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

```
"2" \rightarrow H(2)
```

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
Alice

Assume H is trusted hash H

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

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

Website

Source code

Source code

download site

Malice
<table>
<thead>
<tr>
<th>Confidentiality</th>
<th>Symmetric-Key</th>
<th>Asym-Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symmetric-Key Encryption</td>
<td>(AES-CBC)</td>
<td>Public-Key Enc (ElGamal)</td>
</tr>
<tr>
<td>MAC</td>
<td>(AES-EMAC)</td>
<td>Digital Signatures (RSA)</td>
</tr>
</tbody>
</table>
Alice $\xrightarrow{M} \text{Bob (Bank)}$

"Transfer $1000 to Chris"

Authenticity: $M$ comes from the expected user

Integrity: $M$ was not modified

Encryption does not provide these properties

$\uparrow$ El Gamal $(g^m \mod p)$

$$\frac{H \cdot g^k \mod p}{g^2}$$

$= 2H \cdot g^k \mod p$

$\downarrow$

Easy to modify
MAC (Message authentication code)

Alice \[\begin{array}{c}
M, T \\
K
\end{array}\] \rightarrow Bob

\[\begin{array}{c}
K \\
\text{MAC}(K, M) = T
\end{array}\]

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
Challenger

K

\[ \text{Adv} \]

\[ \begin{array}{c}
  \text{Mi} \\
  \text{MAC}(K, M_i) \\
  M = \{ M_i \} \\
  T
\end{array} \]

\[ \text{Adv, Pr} \left[ K \leftarrow \text{Keygen}(); \text{Adv()} \rightarrow (M, T) \right] \]

s.t. \( M = \{ M_i \} \), \( \text{MAC}(K, M) = T \)

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

Keygen() \rightarrow
Choose 2 $\Delta$ Keys \((K_1, K_2) = K\)

\(\text{MAC}(K, M) = \frac{128}{32}\)

Split \(M\) into \(P_1 || P_2 - \cdots - || P_n\)

Propose \(\text{MAC}(K, M) = (S_1, \ldots, S_n)\) **insecure**

Can forge MAC: Adv asks MAC \(M = (P_1 || P_2) - || S_1, S_2)\)
\(M^* = P_1 \wedge \text{MAC is } S_1\)
Propose $\text{MAC}''(K, M) = S_n$ (insecure MAC for exercise)

$\text{Adv}$ asks for $P_1 \rightarrow S_i$
$P_1' \rightarrow S_i'$
$P_1 || P_2 \rightarrow T$

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

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

Bob

\[ K = (K_E, KM) \]

both confidentiality and integrity/auth

1. Checks MAC for C using KM

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

2. Decrypt to get M

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