Lecture 15
Outline

Digital Signatures

Zero Knowledge

Commitments

Scribe: Kartik
Authenticated
Key Exchange
Authenticated Key Exchange

• Recall key exchange

• Adversary eavesdrops on the network
Authenticated Key Exchange

- Adversary has complete control of the network
  - Modify/inject/delete packets

\[ (g^{b'})^a = g^{a \cdot b'} \]
\[ (g^a)^{b'} = (g^{b})^a' \]
\[ (g^{a'})^b = g^{a' \cdot b} \]
Trusted Third Parties

- Adversary has complete control of the network
  - Modify/inject/delete packets
Authenticated Key Exchange
Important Component: Signature Scheme

Public-key MAC

Verification key is public
Signing key is hidden

\[ \text{Verifying: } \text{Verify}(m, \text{tag}) \]

\[ \text{Signing: } \text{Sign}_{sk}(m) = \text{tag} \]

\[ \text{Public Key: } vk \]

\[ \text{Secret Key: sk} \]
Digital Signatures

\[ \text{Gen}(1^n; r) \rightarrow (vk, sk) \]

\[ \text{Sign}(m, sk, r) \rightarrow \sigma \]

\[ \text{Ver}(m, \sigma, vk) \rightarrow \begin{cases} 1 & \text{accept} \\ 0 & \text{reject} \end{cases} \]

Correctness: \( \forall m, \ \text{Verify}(m, \sigma, vk) = 1 \)

whenever \( \sigma = \text{Sign}(m, sk, r) \) and \((vk, sk) \leftarrow \text{Gen}\)
Digital Signatures

Public-key version of “MACs”
One-Time Signatures

- Definition of security

For a sig. scheme \((\text{Gen}, S, V)\) and adv. A define a game as:

Adversary wins if \(\mathcal{V}(pk, m, \sigma) = \text{accept} \) and

\(\mathcal{V}(vk, m', \sigma') = \text{accept} \) AND \(m' \neq m\).
Lamport One-Time Signatures (from OWF)

- **vk =**
  - $f(x_1)
  - f(x_2)
  - f(x_3)
  - ...
  - f(x_n)

- **sk =**
  - $x_1$
  - $x_2$
  - ...
  - $x_n$

- **Sign** ($m = b, sk$) \[ \rightarrow \sigma_b = \oplus x_b \]

- **Verify** ($m, \sigma, vk$) \[ \rightarrow \text{check if } y_b = f(\sigma) \]
Lamport One-Time Signatures

- \( \text{vk} = \begin{bmatrix} y_1^0 & y_2^0 & \cdots & y_n^0 \\ j_1^i & y_2^i & \cdots & y_n^i \end{bmatrix} \)

- One-time security (definition):

\[ \text{Ch} \quad \text{vk} \quad \left\{ m_i, x_i \mid i \in [n] \right\} \quad (m', \sigma') \quad \text{s.t.} \quad m' \neq m \]
Lamport One-Time Signatures

• \( \text{vk} = \) 

• One-time security (proof):

\[
\sigma' = \frac{\sum}{2} x_i m'_i 2 ^{i \cdot c[i]} \text{ for } m'_i \neq m_i \text{ for some } i
\]

\( \leq \)

\[
\text{Xi} \text{ is the inverse of } y_i \text{ for } m'_i \neq m_i
\]

Attacker is inventing DUF

\( \Downarrow \) contradiction.
Lamport One-Time Signatures

• What about two-time security?

0...0

(s_i^0_{i∈C})

{x_i'}_{i∈[n]}
Many-Time Signatures

For a sig. scheme \((\text{Gen}, S, V)\) and adv. \(A\) define a game as:

Chal. \((pk, sk) \leftarrow \text{Gen}\)

\[ m_1 \in M \quad m_2, \ldots, m_q \]

\( \sigma_1 \leftarrow S(sk, m_1) \quad \sigma_2, \ldots, \sigma_q \)

Adv. \((m, \sigma)\)

Adv. wins if \(V(pk, m, \sigma) = \text{`accept'\} \text{ and } m \not\in \{m_1, \ldots, m_q\}\)

\(\text{STRONG:} \quad (m, \sigma) \in \xi(m_1, \sigma_1, \ldots, m_q, \sigma_q)\)
Applications?

- Authenticating software updates
One-time => Many-time authenticated channel

AUTHENTICATED CHANNEL: “vk, Alice”

“m1, sign1”

“m2, sign2”
Signing email: DKIM (domain key identified mail)

Problem: bad email claiming to be from someuser@gmail.com but in reality, mail is coming from domain baguy.com
Rightarrow Incorrectly makes gmail.com look like a bad source of email

Solution: gmail.com (and other sites) sign every outgoing mail
Important application: Certificates

Problem: browser needs server’s public-key to setup a session key
Solution: server asks trusted 3rd party (CA) to sign its public-key pk

**Diagram:***

- **browser**
  - $pk_{CA}$
  - verify cert
  - verification key

- **Gmail.com**
  - choose (pk, sk)
  - $pk_{gmail}$
  - pk and proof “I am Gmail”

- **Certificate Authority (CA)**
  - check proof
  - $sk_{CA}$
  - signing key

- **Certificate**
  - pk is key for Gmail
  - CA sig

---

Server uses Cert for an extended period (e.g. one year)
MACs versus signatures

• Use MAC when
  1. signer & verifier are same
  2. single signer & single verifier

• Use signature when
  1. many verifiers for same signer
  2. public verification
Can you use MACs here?

• Authenticating software updates