18734 Recitation

Cryptography

Tools: Black Box Auditing, The QII tool
Classical model of encryption

- Goal of the adversary:
  - to systematically recover plaintexts from ciphertexts
  - to deduce the (decryption) key

Slide: Jean-Pierre Hubaux
Basic Cryptographic Concepts

- Encryption scheme (symmetric and public key)
- Signature scheme
- Message authentication code
- Hash function

- A network protocol like SSL is built using these primitives
Symmetric Encryption Scheme

- **Key generation algorithm**
  - Produces a key that is used for encryption and decryption
- **Algorithm to encrypt a message**
- **Algorithm to decrypt a ciphertext**
- **Correctness:**
  - Decrypting a ciphertext obtained by encrypting message $m$ with the corresponding key $k$ returns $m$
    \[
    \text{dec}(\text{enc}(m,k),k) = m
    \]
- **(Symbolic) Security:**
  - A ciphertext cannot be decrypted without access to the key
Example Symmetric Encryption Scheme: One time pad (OTP)

- **Key generation algorithm**
  - generate random bits

- **Algorithm to encrypt a message**
  - \( \text{enc}(m, k) = m \oplus k \)

- **Algorithm to decrypt a ciphertext**
  - \( \text{dec}(c, k) = c \oplus k \)

- **Correctness:**
  \[
  \text{dec(\text{enc}(m, k), k)} = \text{dec}(m \oplus k, k) = (m \oplus k) \oplus k = m
  \]
Example Symmetric Encryption Scheme: One time pad (OTP)

• *Key generation algorithm*
  – generate random bits

• *Algorithm to encrypt a message*
  – \( \text{enc}(m,k) = m \oplus k \)

• *Algorithm to decrypt a ciphertext*
  – \( \text{dec}(c,k) = c \oplus k \)

• (Symbolic) security: A ciphertext cannot be decrypted without access to the private decryption key
  • Entropy\(_m\) = Entropy\(_{m \text{ given } c}\)
Public-Key Encryption Scheme

• *Key generation* algorithm
  – Produces private decryption & public encryption key pair

• Algorithm to *encrypt* a message

• Algorithm to *decrypt* a ciphertext

• Correctness:
  – Decrypting a ciphertext obtained by encrypting message $m$ with the corresponding encryption key returns $m$

\[
\text{dec(} \text{enc}(m, pk(A)), sk(A)) = m
\]

• *(Symbolic) Security:* 
  – A ciphertext cannot be decrypted without access to the private decryption key
Example Public-Key Encryption Scheme

• **Key generation** algorithm
  – Generate public key: e, secret key: d
  – s.t. \( ed = 1 \pmod{n} \)

• **Algorithm to encrypt** a message
  – \( 	ext{enc}(m, e) = m^e \pmod{n} \)

• **Algorithm to decrypt** a ciphertext
  – \( 	ext{dec}(c, d) = c^d \pmod{n} \)

• **Correctness**:
  – \( 	ext{dec}(	ext{enc}(m, e), d) = 	ext{dec}(m^e \pmod{n}, d) = m^{ed} \pmod{n} = m \).

• **(Symbolic) Security**:
  – A ciphertext cannot be decrypted without access to the private decryption key.
RSA Signatures

• Key Generation
  – Generate primes p, q; N =pq
  – Public key = e; private key = d s.t.
    \[ ed = 1 \mod (p-1)(q-1) \]

• Sign
  – \[ C = M^d \mod N \]

• Verify
  – Check \[ M \mod N = C^e \mod N \]
  – Note \[ C^e \mod N = M^{ed} \mod N = M \mod N \]
Signature Scheme

• *Key generation algorithm*
  – Produces private signing & public verification key pair

• Algorithm to *sign* data

• Algorithm to *verify* signature

• *Correctness:*
  – Message signed with a signing key verifies with the corresponding verification key
    \[ \text{verify}(m, \text{sign}(m, d), e) = \text{ok} \]

• *Security:*
  – A signature cannot be produced without access to the private signing key
Signature Scheme

- **Key generation algorithm**
  - private signing & public verification key pair \((e, d=1/e)\)
- **Algorithm to sign data**
  - \(\text{sign}(m, e) = m^e\)
- **Algorithm to verify signature**
  - \(\text{verify}(m, c, d) = \text{return } \text{ok } \text{iff } m == c^d\)
- **Correctness:**
  - \(\text{verify}(m, \text{sign}(m,d), e) = \text{ok}. \text{Satisfied?}\)
- **Security:**
  - A signature cannot be produced without access to the private signing key. \text{Satisfied?}
Message Authentication Code (MAC)

- **Key generation algorithm**
  - Produces a key
- Algorithm to *mac* a message
- Algorithm to *verify* a mac on a message
- Correctness:
  - Message mac-ed with key verifies with the same key
    \[ \text{verify}(k, m, \text{mac}(k, m)) = \text{ok} \]
- Security:
  - A MAC cannot be produced without access to the key

Similar to signature, but uses symmetric key

*What property does a signature have, but a MAC does not?*
Hash Functions

• Algorithm to *hash* a message $m$ to a fixed length output $\text{hash}(m)$

• Security (Collision resistance)

• Given hash function $\text{hash}: X \rightarrow Y$, cannot find a collision, i.e. $x, x' \in X$ s.t. $x \neq x'$ and $\text{hash}(x) = \text{hash}(x')$
  
  – $\text{Hash(password)} \neq \text{Hash(pa$sword$)}$
Hash Functions

• Algorithm to hash a message $m$ to a fixed length output $\text{hash}(m)$
  $\quad \text{– } \text{hash}(m) = m \% 10$, where $m$ is an integer

• Security (Collision resistance)

Given hash function $\text{hash}: X \to Y$, cannot find a collision, i.e. $x, x' \in X$ s.t. $x \neq x'$ and $\text{hash}(x) = \text{hash}(x')$. Satisfied?

SHA1, SHA3, MD5, etc.