Cryptography

Lecture 14: Zero-Knowledge Proofs

January 28, 2025

Overview

Contents

1. Classical cryptography
   (Shift & Vigenère cipher, one-time pad, perfect secrecy)
2. Security definitions & threat models
   (Computational security, CPA & CCA)
3. Private-key cryptography
   (Message authentication, hash functions, primitives, relevant ciphers)
4. Public-key cryptography
   (Assumptions, key management, digital signatures, relevant ciphers)

Overview

	Private-key (symmetric)	Public-key (asymmetric)
Secrecy	Private-key encryption	Public-key encryption
Integrity	Message authentication codes	Digital signature schemes

Digital Signatures

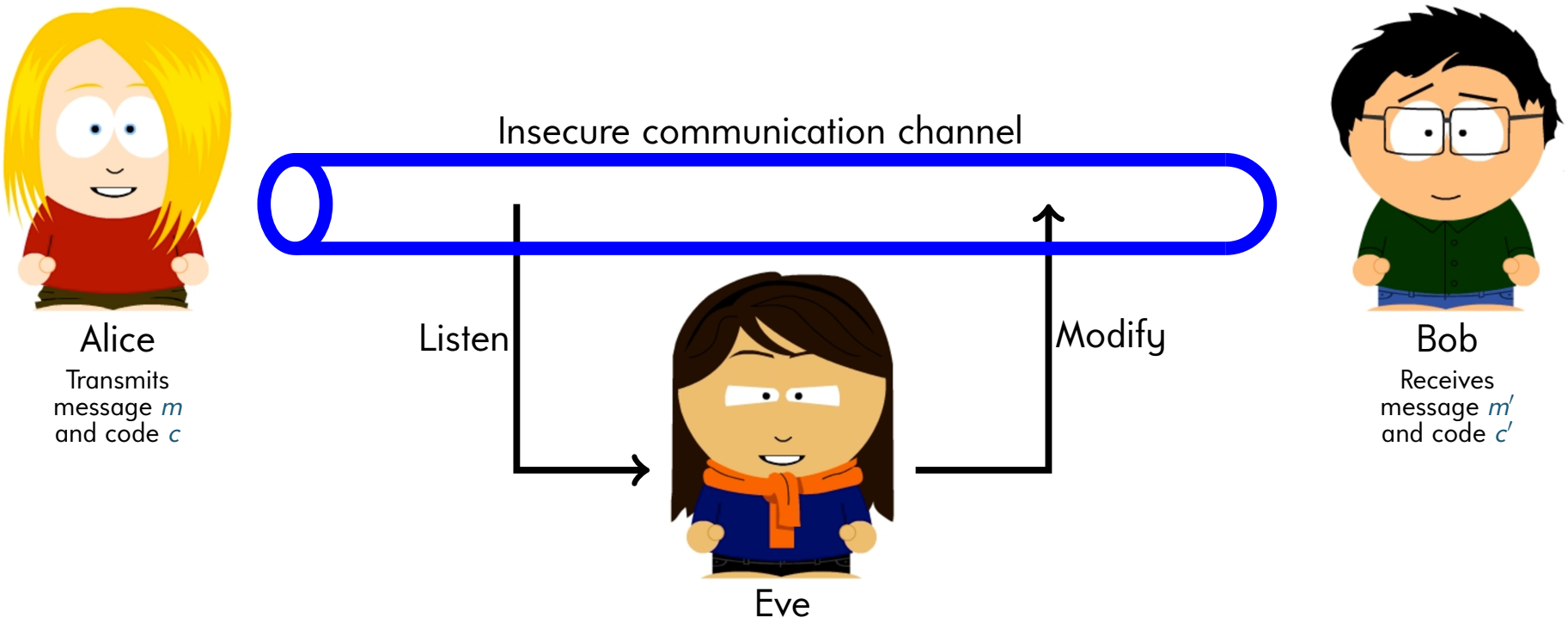
Motivation

- Ensure integrity (instead of secrecy) of transmitted messages (detect changes in message & authenticate origin)
- Authentication of origin much stronger than MAC
  - Everybody can validate (only public key required)
  - Signature transferable (same signed message works for all parties)
  - Only secret key owner can sign $\rightarrow$ Non-repudiation

Applications

- Identification
- Contract verification

Digital Signatures



Notes

- Alice sends message together with signature (tag) using secret key
- Eve might intercept & modify message & code using public key
- Bob receives potentially different message & code using public key (verifies whether they are consistent)

Digital Signatures

§12.7 Definition (cf. §7.2; Efficient digital signature scheme)

(Efficient) digital signature scheme is triple $(\mathrm{gen}, \mathrm{sign}, \mathrm{val})$ of probabilistic polynomial-time computable functions

- $\mathrm{gen}\colon R \times \{1\}^* \to K^2$
- $\mathrm{sign}\colon R \times K \times M \to C$ and $\mathrm{val}\colon K \times M \times C \to \{0,1\}$
- $|k_e| \geq n$ and $|k_d| \geq n$ for all $r \in R$, $n \in \mathbb{N}$, and $\langle k_e, k_d \rangle = \mathrm{gen}(r, 1^n)$
- $\mathrm{val}_{k_e}(m, c) = 1$ for all $\langle k_e, k_d \rangle \in \mathrm{ran}(\mathrm{gen})$, $m \in M$, and $c \in C$ with $\mathrm{P}\left[c \leftarrow \mathrm{sign}_{k_d}(m)\right] \neq 0$ (perfect correctness)

Notes

- Public key k_e used to validate
- Secret key k_d used to sign

Digital Signatures

Alice

Eve

Digital Signatures

Alice

$\langle k_e, k_d \rangle \leftarrow \text{gen}(1^n)$

Public key k_e

Eve

§12.8 Definition (Existential signature forging game)

Let $n \in \mathbb{N}$, $\mathcal{E} = (\text{gen}, \text{sign}, \text{val})$ efficient digital signature scheme, and $\mathcal{A}$ stateful algorithm. Adversarial forging game $\text{Forge}^{\text{Sign}}_{\mathcal{E},\mathcal{A}}(n)$ is

1. $\langle k_e, k_d \rangle \leftarrow \text{gen}(1^n)$ and public key k_e sent to adversary $\mathcal{A}$

Digital Signatures

Alice

$\langle k_e, k_d \rangle \leftarrow \text{gen}(1^n)$

Public key k_e

Message m & code c

Eve

§12.8 Definition (Existential signature forging game)

Let $n \in \mathbb{N}$, $\mathcal{E} = (\text{gen}, \text{sign}, \text{val})$ efficient digital signature scheme, and $\mathcal{A}$ stateful algorithm. Adversarial forging game $\text{Forge}^{\text{Sign}}_{\mathcal{E},\mathcal{A}}(n)$ is

1. $\langle k_e, k_d \rangle \leftarrow \text{gen}(1^n)$ and public key k_e sent to adversary $\mathcal{A}$
2. $\mathcal{A}$ selects message $m \in M$ and $c \in C$

Digital Signatures

Alice

$\langle k_e, k_d \rangle \leftarrow \text{gen}(1^n)$

$b = \text{val}_{k_e}(m, c)$

Public key k_e

Message m & code c

Eve

$b' = \left((m, c) \overset{?}{\notin} Q\right)$

§12.8 Definition (Existential signature forging game)

Let $n \in \mathbb{N}$, $\mathcal{E} = (\text{gen}, \text{sign}, \text{val})$ efficient digital signature scheme, and $\mathcal{A}$ stateful algorithm. Adversarial forging game $\text{Forge}^{\text{Sign}}_{\mathcal{E},\mathcal{A}}(n)$ is

1. $\langle k_e, k_d \rangle \leftarrow \text{gen}(1^n)$ and public key k_e sent to adversary $\mathcal{A}$
2. $\mathcal{A}$ selects message $m \in M$ and $c \in C$
3. Return $b \wedge b'$ ($\mathcal{A}$ wins) with $b = \text{val}_{k_e}(m, c)$ and $b' = \left((m, c) \overset{?}{\notin} Q\right)$ where Q set of message & code pairs obtained via oracle

Digital Signatures

Alice

$\langle k_e, k_d \rangle \leftarrow \text{gen}(1^n)$

$b = \text{val}_{k_e}(m, c)$

Public key k_e

Message m & code c

Eve

$b' = \left((m, c) \stackrel{?}{\notin} Q\right)$

Return $b \wedge b'$

§12.8 Definition (Existential signature forging game)

Let $n \in \mathbb{N}$, $\mathcal{E} = (\text{gen}, \text{sign}, \text{val})$ efficient digital signature scheme, and $\mathcal{A}$ stateful algorithm. Adversarial forging game $\text{Forge}^{\text{Sign}}_{\mathcal{E},\mathcal{A}}(n)$ is

1. $\langle k_e, k_d \rangle \leftarrow \text{gen}(1^n)$ and public key k_e sent to adversary $\mathcal{A}$
2. $\mathcal{A}$ selects message $m \in M$ and $c \in C$
3. Return $b \wedge b'$ ($\mathcal{A}$ wins) with $b = \text{val}_{k_e}(m, c)$ and $b' = \left((m, c) \stackrel{?}{\notin} Q\right)$ where Q set of message & code pairs obtained via oracle

Digital Signatures

§12.9 Definition (Secure signature scheme)

Digital signature scheme $\mathcal{E} = (\mathsf{gen}, \mathsf{sign}, \mathsf{val})$ is secure
if for every PPT algorithm $\mathcal{A}$ with access to sign-oracle

$$\mathrm{P}\left[\mathrm{Forge}^{\mathrm{Sign}}_{\mathcal{E},\mathcal{A}}(n)\right] \simeq 0$$

§12.10 Principle

Secure digital signature schemes permit replay attacks

Digital Signatures

§12.11 Definition (Plain RSA signatures)

Let f be RSA key generator, for which RSA problem is hard.
Digital signature scheme $\mathsf{RSA}' = (\mathsf{gen}, \mathsf{sign}, \mathsf{val})$ given by

- $\mathsf{gen}(1^n)$ runs $(\ell, e, d) \leftarrow f(1^n)$ and returns $\big(\langle \ell, e\rangle, \langle \ell, d\rangle\big)$
- $\mathsf{sign}_{\langle \ell, d\rangle}(m) = m^d \bmod \ell$ for all $m \in \mathbb{Z}_\ell^*$
- $\mathsf{val}_{\langle \ell, e\rangle}(m, c) = \big(m \stackrel{?}{=} c^e \bmod \ell\big)$ for all $m, c \in \mathbb{Z}_\ell^*$

Is RSA' secure?

Digital Signatures

RSA signatures are trivially not secure

1. Can forge valid signature $1 = m^d \bmod \ell$ on message $m = 1$
2. If standard RSA key generation with $e = 3$ is used,
   then signatures for "small" messages can be forged
   (e.g. $\sqrt[3]{8} = 2$, so 2 valid signature for message 8)
3. Take any $a \in \mathbb{Z}_\ell^*$ and compute $m = a^e \bmod \ell$
   (trivially a is valid signature for message m)
4. Given valid signature & message pairs (c_1, m_1) and (c_2, m_2)
   forge signature $c = c_1 \cdot c_2 \bmod \ell$ for $m = m_1 \cdot m_2 \bmod \ell$
   $(c^e = (c_1 \cdot c_2)^e = (m_1^d \cdot m_2^d)^e = m_1^{ed} \cdot m_2^{ed} = m_1 \cdot m_2 = m \bmod \ell)$

Digital Signatures

Practical approach

- Hash-and-Sign (or Hash-and-MAC)
- Utilize hash function to reduce size of message
- Sign hash (or MAC) instead of message

§12.12 Theorem (Theorems 5.6 & 12.4 of [Katz & Lindell])

Given secure digital signature scheme (secure MAC) and collision-resistant hash function, Hash-and-Sign (Hash-and-MAC) yields secure digital signature scheme (secure MAC)

Note

- Secure signature scheme constructible from one-way permutations (via one-time secure, input extension, removal one-time)

Zero-Knowledge Proofs

Motivation

- Proof of knowledge without revealing that knowledge
  (e.g. root of polynomial without naming it)
- Used for identification
  (if that knowledge is intimately tied to person)
- Demonstrate knowledge & skills without working for free
  (e.g. programming skills without completing next project for free)
- Cryptocurrencies
  (prove validity of transaction and paid fees without revealing any detail of transaction [sender, receiver, amount] except that it occurred)

Zero-Knowledge Proofs

Principle

- Ask you to calculate sufficiently many products $x \cdot y$
- If all products correct, then you proved that you know how to multiply
  (recall all those school tests in 3rd and 4th grade)
- But no knowledge about your multiplication method is obtained

Zero-Knowledge Proofs

Suppose CDH problem is hard for cyclic group generator $\mathcal{G}$

Small example

- Protocol between Alice and Bob with agreed upon security parameter n
- Alice convinces Bob that she knows $x = \log_g(a)$ in $\mathcal{G}(1^n) = (\mathbb{G}, q, g)$
- Repeatedly execute following steps

  a. Alice selects uniform $e \in \mathbb{Z}_q$ and sends $z = g^e$ to Bob
  b. Bob selects $b \leftarrow U_1$, queries Alice for $e + bx \bmod q$, and verifies received value r

$$g^r \stackrel{?}{=} z \cdot a^b$$

- Alice knowing x can trivially correctly answer all queries
- Bob learns either e or $e + x \bmod q$ (never both) in each iteration, which are just uniformly chosen elements of $\mathbb{Z}_q$
  (if he knows $e + x \bmod q$, then he would need to obtain $e = \log_g(z)$)

Zero-Knowledge Proofs

$x = \log_g(a)$ in $\mathcal{G}(1^n) = (\mathbb{G}, q, g)$

(a) Alice selects uniform $e \in \mathbb{Z}_q$ and sends $z = g^e$ to Bob

(b) Bob selects $b \leftarrow U_1$, queries Alice for $e + bx \bmod q$, and verifies received value r

$$g^r \stackrel{?}{=} z \cdot a^b$$

Can PPT Alice successfully fool Bob?

- Suppose Alice does not know x and is restricted to PPT
- If she faithfully executes (a), then she can correctly answer query for e but $\log_g(za) = e + x \bmod q$ hard to compute and guess likely wrong
- Alternatively, Alice can send e.g. $z = g^e \cdot a^{-1}$ in step (a)
  Then query for $e + x$ answered by e, which verifies $g^e = (g^e \cdot a^{-1}) \cdot a$
  But query for e requires $r = e - \log_g a \bmod q$
- Overall $\approx 50\%$ success probability per iteration
  (negligible if n iterations are executed)

Summary

- Digital signature schemes
- Zero-knowledge

All the best for the exam!