Lecture 1. Introduction

Introduction to cryptology

Bruno Grenet

M1 INFO, MOSIG & AM

Université Grenoble Alpes – IM²AG

https://membres-ljk.imag.fr/Bruno.Grenet/IntroCrypto.html

What is cryptography?

Protecting secret data from adversaries

- Communications (email, web, credit card payment, …)
- Storage (encrypted hard drive, …)
- Computations (electronic voting, …)
- …

Used with various hardware

- High-end CPUs, mobile phones, microcontrollers, dedicated hardware
- Varying speed (throughput & latency), code/circuit size, energy consumption, …

"Doing crypto"

- Designing new primitives, constructions, protocols, …
- Analysing existing primitives, …
- Deploying crypto in products — incl. implementation

What is this course about?

- Cryptographic constructions
  - What is a block cipher?
  - What is a key exchange?
  - …
- Some standard attacks
  - *Birthday* attack
  - …
- Real-life usage
  - What's inside TLS?

But not (really) about

- Implementation
- Usage of existing standard cryptographic softwares, libraries, …

Example of a protocol: TLS

Goals

- Confidentiality — *no adversary can read the data*
- Authenticity — *no adversary can impersonate the sender*
- Integrity — *no adversary can modify the data*

Some ingredients

- Key exchange — *e.g.* Diffie-Hellman
  - *public-key (a.k.a.* ***a****symmetric) cryptography*
- Authenticated encryption — *e.g* using AES
  - *symmetric cryptography*
- Signatures — *e.g.* ECDSA
  - *public-key* + *symmetric cryptography*

Contents (tentative)

1. Introduction — *One-time pad*
2. Block ciphers — *AES, DES*
3. Symmetric encryption — *CBC & CTR* modes of operation
4. Hash functions — *SHA-2, SHA-3*
5. Messages authentication codes & authenticated encryption — *CBC-MAC, HMAC, GCM*
6. Key exchange — *Diffie-Hellman*
7. Asymmetric encryption & key encapsulation — *ElGamal*
8. Signatures — *Schnorr, DSA*
9. RSA
10. Putting it all together — *TLS*

- Definitions and security notions
- Proofs of security
- Examples

Historical ciphers

- Shift ciphers *Caesar (50 BC);* `rot13`
- Substitution ciphers *Atbash (600-500 BC)*
- Transposition ciphers *Scytale (400 BC)*
- Polyalphabetic cipher *Vigenère (1553); Enigma (1920s)*
- ...

Historical ciphers

- Shift ciphers *Caesar (50 BC);* `rot13`
- Substitution ciphers *Atbash (600-500 BC)*
- Transposition ciphers *Scytale (400 BC)*
- Polyalphabetic cipher *Vigenère (1553); Enigma (1920s)*
- ...

- None is safe: brute force, frequency analysis (1863), ...
- Some lessons drawn:
  - You need a large enough *key space*.
  - Designing an encryption system is *difficult*.

1. A first example: the one-time pad

2. Computational security

The one-time pad

Input: *Plaintext* $m \in \{0,1\}^\ell$ (or *message*)
Secret: Key $k \in \{0,1\}^\ell$
Output: *Ciphertext* $c \in \{0,1\}^\ell$

Encryption: $\mathsf{Enc}_k(m) =$
Decryption: $\mathsf{Dec}_k(c) =$

The one-time pad

Input: *Plaintext* $m \in \{0,1\}^\ell$ (or *message*)
Secret: Key $k \in \{0,1\}^\ell$
Output: *Ciphertext* $c \in \{0,1\}^\ell$

Encryption: $\mathsf{Enc}_k(m) = m \oplus k$
Decryption: $\mathsf{Dec}_k(c) =$

The one-time pad

Input: *Plaintext* $m \in \{0,1\}^\ell$ (or *message*)
Secret: Key $k \in \{0,1\}^\ell$
Output: *Ciphertext* $c \in \{0,1\}^\ell$

Encryption: $\mathsf{Enc}_k(m) = m \oplus k$
Decryption: $\mathsf{Dec}_k(c) = c \oplus k$

The one-time pad

Input: *Plaintext* $m \in \{0,1\}^\ell$ (or *message*)
Secret: Key $k \in \{0,1\}^\ell$
Output: *Ciphertext* $c \in \{0,1\}^\ell$

Encryption: $\mathsf{Enc}_k(m) = m \oplus k$
Decryption: $\mathsf{Dec}_k(c) = c \oplus k$

Correctness: $\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = (m \oplus k) \oplus k = m$

The one-time pad

Input: *Plaintext* $m \in \{0,1\}^\ell$ (or *message*)
Secret: Key $k \in \{0,1\}^\ell$
Output: *Ciphertext* $c \in \{0,1\}^\ell$

Encryption: $\mathrm{Enc}_k(m) = m \oplus k$
Decryption: $\mathrm{Dec}_k(c) = c \oplus k$

Correctness: $\mathrm{Dec}_k(\mathrm{Enc}_k(m)) = (m \oplus k) \oplus k = m$

Pros
- Used during the cold war
- Used for small plaintexts/secrets
- *Perfectly secret*

Cons & caveats
- Key as long as the message
- Can be used only once
- The key must be uniformly sampled

Perfect secrecy

a.k.a information-theoretic security, *a.k.a.* unconditional security

> No matter what an attacker knows about the message, the ciphertext will not give them any extra information.

Formalisation

Knowledge: probability distributions over messages / ciphertexts / keys

Message: random variable M over $\mathcal{M}$ *space of messages*

Ciphertext: random variable C over $\mathcal{C}$ *space of ciphertexts*

Definition *Shannon (1949)*

An encryption scheme (Enc, Dec) is perfectly secret if for every *probability distribution* for M, every message $m \in \mathcal{M}$ and every $c \in C$ (s.t. $\Pr[C = c] > 0$),

$$\Pr[M = m | C = c] = \Pr[M = m].$$

Security proof for the one-time pad

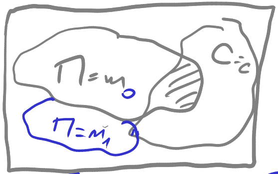
Theorem *Shannon (1949)*

The one-time pad is perfectly secret.

Idea of the proof

Since the key is uniform in $\{0,1\}^\ell$, C is uniform no matter what (the distribution of) M is

Proof $\Pr[M=m \mid C=c] = \Pr[M=m]$



- $\Pr[M=m \mid C=c] = \dfrac{\Pr[M=m \wedge C=c]}{\Pr[C=c]}$
- $\Pr[M=m \wedge C=c] = \Pr[M=m \wedge K=m\oplus c] = \Pr[M=m]\,\underbrace{\Pr[K=m\oplus c]}_{1/2^\ell}$
- $\Pr[C=c] = \Pr[K=m\oplus c] = 1/2^\ell$ ∎

Limitations of perfect secrecy

Theorem *Shannon (1949)*

For a *perfectly secret* encryption scheme with message space $\mathcal{M}$ and key space $\mathcal{K}$,

(i) $|\mathcal{K}| \geq |\mathcal{M}|$

(ii) if $|\mathcal{K}| = |\mathcal{M}|$, k must be uniformly sampled from $\mathcal{K}$

Proof of (i) Assume $|K| < |M|$

We have to prove $\Pr[M=m \mid C=c] \neq \Pr[M=m]$ for <u>some</u> distribution, <u>some</u> m and c

We use the uniform distribution on M.

Define $M(c) = \{m \in M : \exists\, k \ \mathrm{Dec}_k(c) = m\}$

Then $|M(c)| \leq |K| < |M|$

Take $m \in M \setminus M(c)$: $\Pr[M=m \mid C=c] = 0 \neq \Pr[M=m] = 1/|M|$ ☒

Conclusion

- One-time pad: perfectly secret but…
- … perfect secrecy impossible with *small* keys

Relaxation of the security notion

- Allow to recover *some* (very little!) information — *statistical secrecy*
- Put a limit on the computational power of an attacker — *computational security*

Other problems

- An attacker can modify any message — no integrity
  - $c = m \oplus k \implies c \oplus m' = (m \oplus m') \oplus k$

$\rightarrow$ Need definitions!

Principles of modern cryptography

Formal definitions

- Example: what does *secure encryption* mean?
  - An attacker cannot recover the key
  - An attacker cannot recover the message from the ciphertext
  - An attacker cannot retrieve any character of the message from the ciphertext
  - …

Principles of modern cryptography

Formal definitions

- Example: what does *secure encryption* mean?
  - (good definition) Whatever information an attacker has about the message, the ciphertext only provides them with *very little* additional information

Principles of modern cryptography

Formal definitions

- Example: what does *secure encryption* mean?
  - (good definition) Whatever information an attacker has about the message, the ciphertext only provides them with *very little* additional information
- Example: what is an *attacker* ?
  - Ciphertext only attack COA
  - Known plaintext attack KPA
  - Chosen plaintext attack CPA
  - Chosen ciphertext attack CCA

Principles of modern cryptography

Formal definitions

- Example: what does *secure encryption* mean?
  - (good definition) Whatever information an attacker has about the message, the ciphertext only provides them with *very little* additional information
- Example: what is an *attacker*?
  - Ciphertext only attack — COA
  - Known plaintext attack — KPA
  - Chosen plaintext attack — CPA
  - Chosen ciphertext attack — CCA

Principles of modern cryptography

Formal definitions

- Example: what does *secure encryption* mean?
  - (good definition) Whatever information an attacker has about the message, the ciphertext only provides them with *very little* additional information
- Example: what is an *attacker*?
  - Ciphertext only attack — COA
  - Known plaintext attack — KPA
  - Chosen plaintext attack — CPA
  - Chosen ciphertext attack — CCA

Specific assumptions

- Computational power of an attacker (complexity theory)
- Validity of assumptions, comparison between them and *necessary* assumptions

Principles of modern cryptography

Formal definitions

- Example: what does *secure encryption* mean?
  - (good definition) Whatever information an attacker has about the message, the ciphertext only provides them with *very little* additional information
- Example: what is an *attacker* ?
  - Ciphertext only attack — COA
  - Known plaintext attack — KPA
  - Chosen plaintext attack — CPA
  - Chosen ciphertext attack — CCA

Specific assumptions

- Computational power of an attacker (complexity theory)
- Validity of assumptions, comparison between them and *necessary* assumptions

Provable security

Proving that a protocol satisfies a *security definition*, assuming *assumptions*.

Indistinguishability

Alternative definition for (perfect / statistical) secrecy

Indistinguishability **experiment** for Enc : $\mathrm{Exp}^{\mathrm{IND}}_{\mathrm{Enc}}(A)$

Adversary	chooses two messages $m_0, m_1 \in \mathcal{M}$
Challenger	draws $k \twoheadleftarrow \mathcal{K}$, $b \twoheadleftarrow \{0,1\}$ and computes $c = \mathrm{Enc}_k(m_b)$
Adversary	receives c, tries to guess b and outputs a bit $\hat{b}$
Output	TRUE if $\hat{b} = b$

Indistinguishability

Alternative definition for (perfect / statistical) secrecy

Indistinguishability **experiment** for Enc : $\mathrm{Exp}^{\mathrm{IND}}_{\mathrm{Enc}}(A)$

Adversary	chooses two messages $m_0, m_1 \in \mathcal{M}$
Challenger	draws $k \twoheadleftarrow \mathcal{K}$, $b \twoheadleftarrow \{0,1\}$ and computes $c = \mathrm{Enc}_k(m_b)$
Adversary	receives c, tries to guess b and outputs a bit $\hat{b}$
Output	True if $\hat{b} = b$

Indistinguishability **advantage** and ε-indistinguishability

- Advantage of adversary A:

$$\mathrm{Adv}^{\mathrm{IND}}_{\mathrm{Enc}}(A) = \Pr\left[\mathrm{Exp}^{\mathrm{IND}}_{\mathrm{Enc}}(A) = \textsc{true}\right] - \frac{1}{2}$$

- Enc is ε-indistinguishable if

$$\max_A \mathrm{Adv}^{\mathrm{IND}}_{\mathrm{Enc}}(A) \leq \varepsilon$$

Indistinguishability and secrecy

- 0-indistinguishable $\iff$ perfectly secret
- ε-indistinguishable $\iff$ *ε-secret* not defined here

Shortcomings

- Perfect secrecy: requires key length $\geq$ message length
- ε-secrecy: requires key length *close to* message length (if ε to be small)

Information-theoretic guarantee usually unachievable in practice

Solution

- Do not consider *any* adversary…
- … but *computationally bounded* adversaries only
- Remark: adversary = randomized algorithm

From information theory to complexity theory

Computational security

- Maximal advantage for *resource-bounded* adversaries: $\max_{A:\dots} \mathrm{Adv}^{\mathrm{IND}}_{\mathrm{Enc}}(A)$
- Concrete security: *chosen in this course*
  - Consider adversaries that perform $\leq t$ *elementary operations*
  - Express the advantage with respect to t
- Asymptotic security: *complexity theory*
  - Consider (randomized) *polynomial-time* adversaries (in a *security parameter* n)
  - Prove that the advantage is *negligible* ($\ll \frac{1}{\mathrm{poly}(n)}$)

Provable security

- Design a security experiment
  - choose the adversary's means (CPA / CCA) & goals (IND / NM)
- Bound the advantage of an adversary for this experiment *probability of success*

Orders of magnitude

Computational time

- $t \simeq 2^{40}$: ~ 1 day on my laptop
- $t \simeq 2^{60}$: possible on a large CPU/GPU cluster — done in academia
- $t \simeq 2^{80}$: possible with an ASIC cluster — Bitcoin mining
- $t \simeq 2^{128}$: seems hard enough

Example: perform 2^{128} operations within 34 years ($\simeq 2^{30}$ seconds)

- Hypotheses:
  - Hardware at 2^{50} op/s — *quite fast*
  - Hugely parallelizable — *not always true*
  - 1000 W per device — *quite good*
- Results:
  - Require $\simeq 2^{128}/(2^{50} \cdot 2^{30}) = 2^{48}$ machines — $> 280 \cdot 10^{12}$
  - Require $\simeq 280\,000$ TW — $> 1.7 \cdot 10^{9}$ EPR

Conclusion

One-time pad

- First example of encryption scheme
- Strong security… in a very weak model!
- Vastly insufficient in practice

Computational security

- Experiment + advantage $\rightarrow$ security notion
- Various security models, depending on the experiment
  - Fix goals & means

What's next?

- Symmetric and public-key encryption
- Authentication and integrity
- Each time:
  - What is the suitable security notion?
  - How to achieve this security notion?