AMERICAN UNIVERSITY OF BEIRUT

Applied Cryptography

CMPS 297AD/396AI

Fall 2025

Part 1: Provable Security

1.2: The Provable Security Mindset

Nadim Kobeissi

https://appliedcryptography.page

Section 1

Math Review

Logs & Exponents

- Key identities to remember:

$$(x^a)(x^b) = x^{a+b}$$
$$(x^a)^b = x^{ab}$$
$$\log_x(ab) = \log_x a + \log_x b$$

- Never write $(x^a)(x^b) = x^{ab}$!

Modular Arithmetic

Key definitions

- **Integers**: $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$
- **Natural numbers**: $\mathbb{N} = \{0, 1, 2, \ldots\}$
- **Divides**: $n|x$ means $x = kn$ for some integer k
- **Example**: $7|84$ because $84 = 12 \cdot 7$

The Modulo Operation

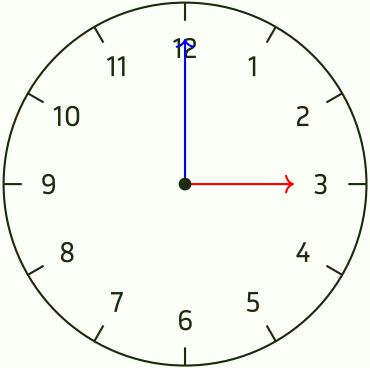
- $a \bmod n$ gives the remainder when dividing a by n
- Result is always in $\{0, 1, \ldots, n-1\}$
- **Even for negative numbers!**

$$\begin{aligned} 21 \bmod 7 &= 0 \\ 20 \bmod 7 &= 6 \\ -20 \bmod 7 &= 1 \quad \text{(not -6!)} \end{aligned}$$

- Think: "a is ($a \bmod n$) more than a multiple of n"

Understanding Modulo with clocks

- Clock arithmetic is modulo 12
- 15:00 = 3:00 PM because $15 \bmod 12 = 3$
- 7 hours after 9:00?
  - $(9 + 7) \bmod 12 = 16 \bmod 12 = 4$
- 5 hours before 2:00?
  - $(2 - 5) \bmod 12 = -3 \bmod 12 = 9$
- Wrapping around: when we go past 12, we start over from 1



Example: 15:00 = 3:00pm

Binary Strings

- $\{0, 1\}^n$ = set of n-bit binary strings
- $|x|$ = length of string x
- 0^n = string of n zeros, 1^n = string of n ones

Probability basics

- **Distribution**: assigns probability to each outcome
- For each outcome x: $0 \le \Pr[x] \le 1$
- Sum of all probabilities = 1
- **Uniform distribution**: $\Pr[x] = \frac{1}{|\mathcal{X}|}$ for all $x \in \mathcal{X}$
  - $\frac{1}{|\mathcal{X}|}$ means “one divided by the size of set $\mathcal{X}$”
  - Example: Rolling a fair die has uniform distribution with $\Pr[x] = \frac{1}{6}$ for each outcome
- Basic fact: $\Pr[A] = 1 - \Pr[\neg A]$
  - $\neg A$ means “not A”
  - All outcomes where A does not occur

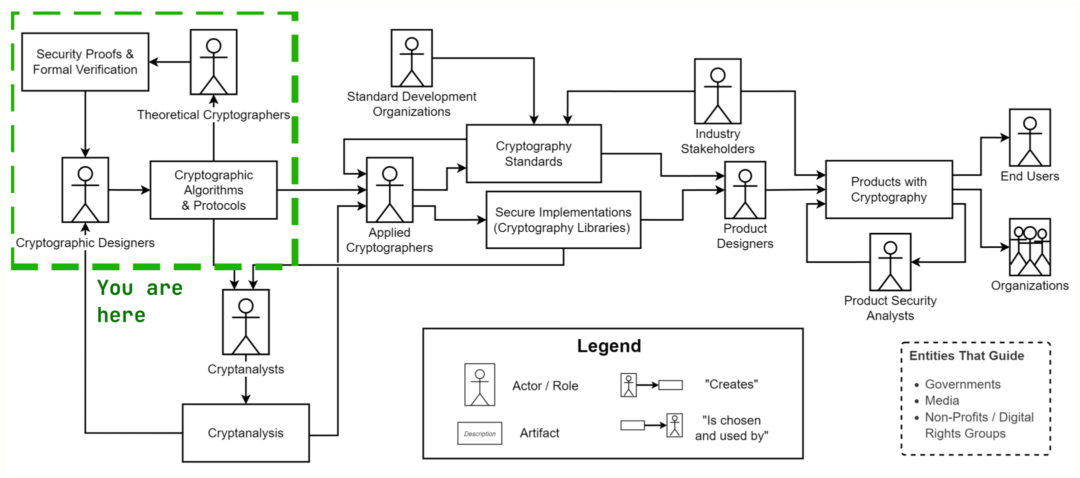
Notation in pseudocode

- $x \twoheadleftarrow \mathcal{X}$: sample x uniformly from set $\mathcal{X}$
- $x := 5$: assign value 5 to variable x
- $x \overset{?}{=} y$: check if x equals y (returns true/false)

Section 2

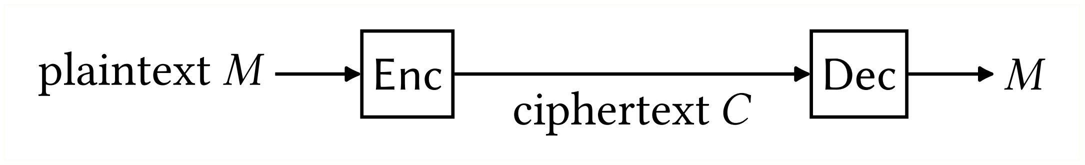
The Provable Security Mindset

How it’s made



Fischer et al., The Challenges of Bringing Cryptography from Research Papers to Products: Results from an Interview Study with Experts, USENIX Security 2024

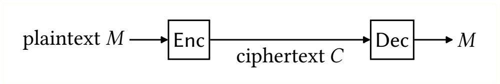
Thinking about secrecy



Source: The Joy of Cryptography

Thinking about secrecy

- Keep the whole design secret?
- **“Advantages”**:
    - Attacker doesn’t know how our cipher (or system, more generally,) works.
- **Disadvantages**:
    - Figuring out how the thing works might mean a break.
    - Can’t expose cipher to scrutiny.
    - Everyone needs to invent a cipher.

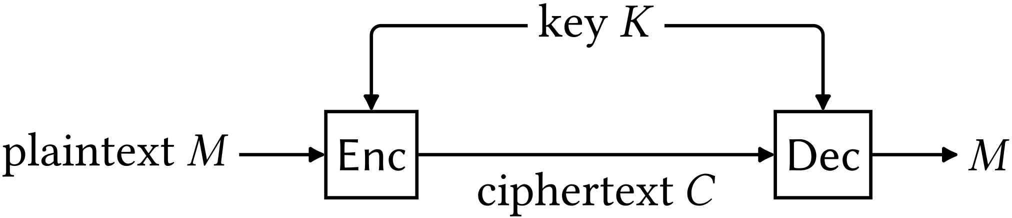


Source: The Joy of Cryptography

Kerckhoff’s principle

- *“A cryptosystem should be secure even if everything about the system, except the key, is public knowledge.”* — Auguste Kerckhoffs, 1883
- **Why it matters**:
  - No “security through obscurity”
  - The key is the only secret: the rest can be audited, tested, trusted
  - Encourages open standards and peer review
  - If your system’s security depends on nobody knowing how it works, it’s not secure.

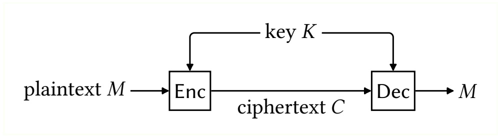
Thinking about secrecy



Concentrate all the need for secrecy in the key!

Thinking about secrecy

- Cipher can be scrutinized, used by anyone.
- Design can be shown to hold so long as the key is secret.
- This is how virtually all cryptography is designed today.



Source: The Joy of Cryptography

One-time pad

First look at a symmetric cipher

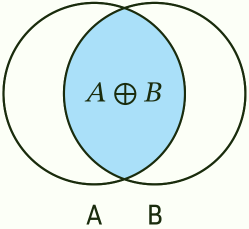
```
$\underline{\textsc{Enc}(K, M):}$
$C := K \oplus M$
return $C$
```

```
$\underline{\textsc{Dec}(K, C):}$
$M := K \oplus C$
return $M$
```

XOR (Exclusive OR) operation

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

Table: Truth table for XOR operation



- XOR returns 1 when inputs differ
- XOR returns 0 when inputs are the same
- Key property: $x \oplus x = 0$ and $x \oplus 0 = x$
- Self-inverse: $(M \oplus K) \oplus K = M$

One-time pad

First look at a symmetric cipher

$$\begin{array}{cll} & \texttt{11101111101111100011} & M \\ \oplus & \texttt{00011001110000111101} & K \\ \hline = & \texttt{11110110011111011110} & C = \mathsf{Enc}(K, M) \end{array}$$

(We're encoding the message and key as bits)

One-time pad

First look at a symmetric cipher

$$\begin{array}{cll} & \texttt{11110110011111011110} & C \\ \oplus & \texttt{00011001110000011101} & K \\ \hline = & \texttt{11101111101111100011} & M = \text{Dec}(K, C) \end{array}$$

(We’re encoding the message and key as bits)

Key derivation

Uniform distribution

- How to derive K?
- K is ideally random.
- True randomness isn't practical, so K is in practice pseudo-random.
- We need a pseudo-random uniform distribution:
- If $\mathcal{S}$ is a set of m items, then the uniform distribution over $\mathcal{S}$ assigns probability $\frac{1}{m}$ to each item $x \in \mathcal{S}$
- In practice, this just means we need the bits to be random, unpredictable, uniformly distributed in terms of probability
- Sampling a K from a pseudo-random uniform distribution is written as $K \twoheadleftarrow \{0, 1\}^n$

One-time pad

Correctness proof

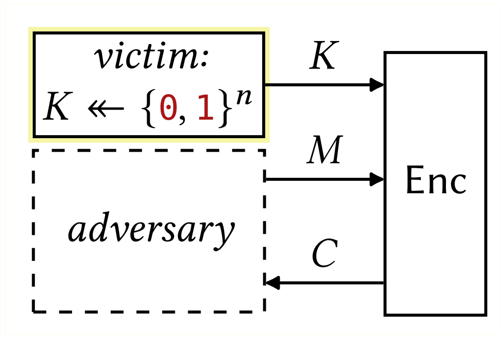
- $\forall(n > 0,\ K \in \{0,1\}^n,\ M \in \{0,1\}^n),\ \mathsf{Dec}(K, \mathsf{Enc}(K, M)) = M$
- For all positive n, any key of n bits and message of n bits will decrypt back to the same plaintext if encrypted into a ciphertext.
- **Proof**:

$$\begin{aligned}
\mathsf{Dec}(K, \mathsf{Enc}(K, M)) &= \mathsf{Dec}(K, K \oplus M) \\
&= K \oplus (K \oplus M) \\
&= (K \oplus K) \oplus M \\
&= 0^n \oplus M \\
&= M \qquad \square
\end{aligned}$$

One-time pad

How do we prove security?

- When we prove security, we prove what is or isn't possible by the attacker calling Attack(M).

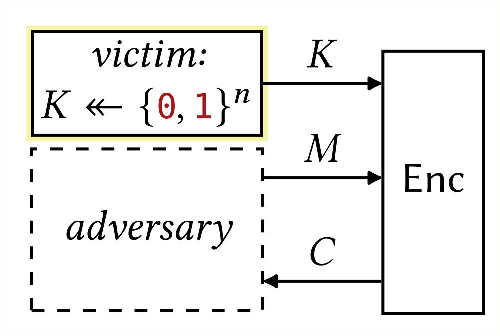


Source: The Joy of Cryptography

One-time pad

How do we prove security?

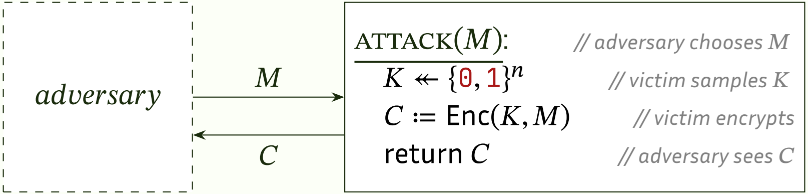
- "Victim" chooses their key.
  - Fresh key for each message (each key used only once)
  - This means output will differ even if same plaintext is input twice by adversary
- Adversary chooses the message and receives the ciphertext.
- We say that **the adversary has access to an encryption oracle**.



Source: The Joy of Cryptography

One-time pad

How do we prove security?

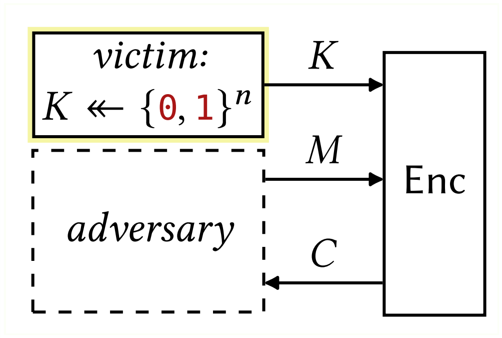


```
ATTACK($M$):                          // adversary chooses $M$
    $K \twoheadleftarrow \{0, 1\}^n$            // victim samples $K$
    $C := \mathsf{Enc}(K, M)$          // victim encrypts
    return $C$                         // adversary sees $C$
```

One-time pad

How do we prove security?

- Adversary can query oracle an unbounded number of times.
- Two queries with same M return different (C, C'), since victim uses different K.
- K is always chosen correctly (pseudo-random uniform sampling)
- “Randomized oracle”
- Attacker cannot see K.



Source: The Joy of Cryptography

“If I use OTP according to the attack scenario (I sample keys uniformly and use each key to encrypt just one ciphertext), then no matter how the plaintexts are chosen, and no matter how the ciphertext is subsequently used, I can enjoy a certain security guarantee.”

One-time pad

How do we prove security?

- **Generally**: a cipher is secure if the adversary can't distinguish the output of calls to $ATTACK$ from random junk.
- **Formally**: For all positive integers n and all choices of plaintext $M \in \{0, 1\}^n$, the output of the following subroutine is uniformly distributed:

```
$\textsc{Attack}(M)$:
$K \twoheadleftarrow \{0, 1\}^n$
$C := K \oplus M$
return $C$
```

One-time pad

How do we prove security?

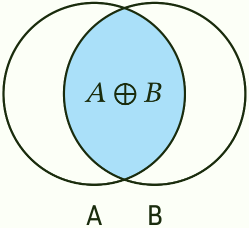
- If the key is random, the output will be uniformly distributed!
- Suppose $M = \texttt{01}$:
    - $K = \texttt{00}$ is chosen with probability $1/4$: $C = K \oplus M = \texttt{00} \oplus \texttt{01} = \texttt{01}$.
    - $K = \texttt{01}$ is chosen with probability $1/4$: $C = K \oplus M = \texttt{01} \oplus \texttt{01} = \texttt{00}$.
    - $K = \texttt{10}$ is chosen with probability $1/4$: $C = K \oplus M = \texttt{10} \oplus \texttt{01} = \texttt{11}$.
    - $K = \texttt{11}$ is chosen with probability $1/4$: $C = K \oplus M = \texttt{11} \oplus \texttt{01} = \texttt{10}$.

```
$\textsc{Attack}(M)$:
$K \twoheadleftarrow \{\texttt{0}, \texttt{1}\}^n$
$C := K \oplus M$
return $C$
```

XOR (Exclusive OR) operation

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

Table: Truth table for XOR operation



- XOR returns 1 when inputs differ
- XOR returns 0 when inputs are the same
- Key property: $x \oplus x = 0$ and $x \oplus 0 = x$
- Self-inverse: $(M \oplus K) \oplus K = M$

One-time pad

What's so special about XOR?

- Let's replace $\oplus$ with $\wedge$. What would happen?
- Output no longer uniform!

A	B	A ∧ B
0	0	0
0	1	0
1	0	0
1	1	1

Table: Truth table for AND operation

```
$\textsc{Attack}(M)$:
$K \twoheadleftarrow \{0, 1\}^n$
$C := K \wedge M$
return $C$
```

One-time pad

How do we prove security?

- What if this is true only for $M = \texttt{01}$?
- Fine, let's pick any $M, C \in \{\texttt{0}, \texttt{1}\}^n$.
- What is $\Pr[\texttt{Attack(M) = C}]$?
- Answer: Exactly when $C = \text{Enc}(K, M) = K \oplus M$.
- …which occurs for exactly one K.
- Since K is chosen uniformly from $\{\texttt{0}, \texttt{1}\}^n$, the probability of choosing that K is $\frac{1}{2^n}$. □

```
$\textsc{Attack}(M)$:
$K \twoheadleftarrow \{\texttt{0}, \texttt{1}\}^n$
$C := K \oplus M$
return $C$
```

One-time pad

From the adversary's perspective...

```
$\textsc{Attack}(M)$:
    $K \twoheadleftarrow \{0,1\}^n$
    $C := K \oplus M$
    return $C$
```

$$\approxeq$$

(indistinguishable from)

```
$\textsc{Junk}(M)$:
    $C \twoheadleftarrow \{0,1\}^n$
    return $C$
```

“Real or random?”

One-time pad

What about $\pmod{n}$?

- Let's replace $\oplus$ with $\pmod{n}$. What would happen?
- Still good!
- Can you prove correctness and security?

```
$\underline{\textsc{Attack}(M)\text{:}}$
$K \twoheadleftarrow \mathbb{Z}_n$
$C := (K + M) \pmod{n}$
return $C$
```

Section 3

Breaking One-Time Pads

Two-time pads

What happens when we reuse a key?

- Alice sends two messages with the same key:
- $C_1 = M_1 \oplus K$
- $C_2 = M_2 \oplus K$
- What can Eve learn from C_1 and C_2?
- $C_1 \oplus C_2 = (M_1 \oplus K) \oplus (M_2 \oplus K)$
- $C_1 \oplus C_2 = M_1 \oplus M_2 \oplus (K \oplus K)$
- $C_1 \oplus C_2 = M_1 \oplus M_2$
- The key cancels out completely!

Exploiting $M_1 \oplus M_2$

Why is this dangerous?

- Knowing $M_1 \oplus M_2$ allows statistical attacks:
- **Language patterns**: Natural language has predictable structure
  - Common words: “the”, “and”, “of”, “to”
  - Letter frequencies: E is most common in English (12.7%)
  - Spaces are very frequent (about 18% of characters)
- **Crib dragging**: Try XORing common words at different positions
- **Example**: If we guess “the” at position i in M_1:
  - Extract: $(M_1 \oplus M_2)[i : i + 3] \oplus$ “the”
  - If result looks like valid text, we may have found part of M_2!

Malleability of one-time pads

Another practical weakness

- OTP provides confidentiality but not integrity
- **Malleability**: Attacker can modify ciphertext predictably
- Given $C = M \oplus K$
- Attacker creates $C' = C \oplus \Delta$
- Decryption: $C' \oplus K = (M \oplus K \oplus \Delta) \oplus K$
- Result: $M' = M \oplus \Delta$
- Attacker controls the change without knowing M or K!

Practical attack

Bit flipping

- **Scenario**: Bank transfer message
- Original: “Transfer $0100.00 to Account 12345”
- Attacker knows position of amount field
- Flips bit to change ‘1’ (ASCII 49 = `00110001`) to ‘9’ (ASCII 57 = `00111001`)
- XOR difference: `00110001` $\oplus$ `00111001` = `00001000`
- Apply Δ = `00001000` at correct position in ciphertext
- Result: “Transfer $0900.00 to Account 12345”
- No need to break encryption or know the key!

Section 4

Limitations of Security Proofs

Limitations of security proofs

Part 1

- Rigor and the real world famously don't mix.
- Security proofs are good for rigor but address very little regarding real-world concerns:
    - How can Alice & Bob obtain a secret key, which only they know?
    - How can they keep K secret?
    - How can a computer sample from the uniform distribution?
    - How can Alice ensure that C is sent reliably to Bob?

Limitations of security proofs

Part 2

- More questions proofs don't address:
  - How can Alice hide the fact that she is talking to Bob (rather than hide only the content)?
  - How can Alice be sure that she is communicating with Bob, not an impostor?
  - How can we incentivize Alice and Bob to use encryption?
  - Should the government be allowed to obtain a warrant to read encrypted communications?
- Security proofs are about specific properties within specific models.
- Real-world security depends on many factors beyond what our models capture.
- Having a security proof is necessary but not sufficient for real-world security.

The value of security proofs

- Despite limitations, security proofs provide important benefits:
  - **Precise guarantees**: Clearly define what security properties are achieved.
  - **Confidence**: When properly structured, proofs ensure no obvious attacks exist.
  - **Foundation for composition**: Proven components can be securely combined.
  - **Precise terminology**: Forces us to clearly define our terms and assumptions.
- Security proofs help identify the *boundaries* of security:
  - What assumptions are necessary?
  - What threats are addressed vs. unaddressed?
  - What conditions must hold for security to be maintained?

The provable security mindset

- Building systems with provable security in mind:
  - Start with clear security goals and adversary model.
  - Design systems whose security can be formally analyzed.
  - Identify and document necessary assumptions.
  - Distinguish between proven properties and conjectures.
- Good practical security requires both:
  - Rigorous proofs for core mechanisms.
  - Practical engineering to address real-world constraints.

OTP: security assumptions & constraints

Part 1

- Our security proofs rely on specific assumptions about the adversary:
  - **Key reuse**: Keys are never intentionally reused (though may repeat by chance).
  - **Observation only**: Adversary passively observes ciphertext but doesn't tamper with it
  - **Message independence**: Choice of message M is independent of key K.
  - **Key secrecy**: Adversary learns nothing about the key.
  - **No sampling influence**: Adversary cannot influence how the key is sampled.
- These constraints are *necessary* for the security proofs to hold!

OTP: security assumptions & constraints

Part 2

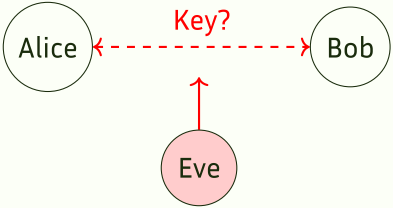
- Side-channel attacks violate our model:
    - We assume adversary cannot coerce victim to run a different algorithm.
    - Cannot observe execution details:
        - CPU timing information (clock cycles).
        - Memory access patterns.
        - Cache hits/misses.
        - Power consumption during encryption.
- Real-world security requires considering these additional attack vectors.
- Our security proofs address a *specific threat model* that may not capture all real-world threats.

Key Distribution Problem

The elephant in the room

- OTP requires key as long as message
- How do Alice and Bob share this key?
- **The paradox**:
  - If you can securely send the key...
  - ...why not just send the message?
- **Historical solutions**:
  - Diplomatic pouches
  - Trusted couriers
  - Pre-shared codebooks
- **Modern issue**: Impractical for internet scale

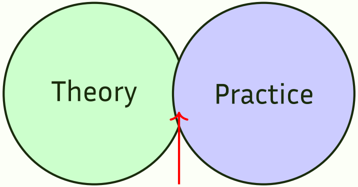
Problem: How to share key securely?



If channel is secure for key, why not use it for message?

From theory to practice

- **Theory provides foundation**:
  - Precise security definitions
  - Mathematical guarantees
  - Clear assumptions
- **Practice reveals challenges**:
  - Implementation vulnerabilities
  - Usability constraints
  - Side-channel attacks
  - Human factors



Sweet spot:
Provably secure &
practically usable

AMERICAN UNIVERSITY OF BEIRUT

Applied Cryptography

CMPS 297AD/396AI

Fall 2025

Part 1: Provable Security

1.2: The Provable Security Mindset

Nadim Kobeissi

https://appliedcryptography.page