Classical Cryptography and Perfect Secrecy

CS 171: Cryptography

Sanjam Garg

September 1, 2026

Overview

Recap from Previous Lecture

- **Private-key encryption** $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$: Alice sends $c \leftarrow \mathsf{Enc}_k(m)$, and Bob recovers $m := \mathsf{Dec}_k(c)$ using the shared secret key k.

Recap from Previous Lecture

- **Private-key encryption** $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$: Alice sends $c \leftarrow \mathsf{Enc}_k(m)$, and Bob recovers $m := \mathsf{Dec}_k(c)$ using the shared secret key k.
- **Kerckhoffs's principle**: only the key is secret — use open, well-studied tools.

Recap from Previous Lecture

- **Private-key encryption** $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$: Alice sends $c \leftarrow \mathsf{Enc}_k(m)$, and Bob recovers $m := \mathsf{Dec}_k(c)$ using the shared secret key k.
- **Kerckhoffs's principle**: only the key is secret — use open, well-studied tools.
- The **shift cipher**: shift every letter by $k \in \{0, \ldots, 25\}$ positions (mod 26) — insecure, since a **brute-force** attack tries all 26 keys.

Recap from Previous Lecture

- **Private-key encryption** $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$: Alice sends $c \leftarrow \mathsf{Enc}_k(m)$, and Bob recovers $m := \mathsf{Dec}_k(c)$ using the shared secret key k.
- **Kerckhoffs's principle**: only the key is secret — use open, well-studied tools.
- The **shift cipher**: shift every letter by $k \in \{0, \ldots, 25\}$ positions (mod 26) — insecure, since a **brute-force** attack tries all 26 keys.
- Today: more classical ciphers and their lessons, then the principles of modern cryptography and our first security definition.

From an Art to a Science

Last time

Cryptography has developed from an **art** to a **science**.

Is the Shift Cipher Secure?

The key was $k = 2$ (the letter 'c')

ciphertext:	`etarvqogcpuetarvqitcrja`
key:	`ccccccccccccccccccccccc`
plaintext:	`cryptomeanscryptography`

Is the Shift Cipher Secure?

The key was $k = 2$ (the letter 'c')

ciphertext:	`etarvqogcpuetarvqitcrja`
key:	`ccccccccccccccccccccccc`
plaintext:	`cryptomeanscryptography`

Exhaustive search over k

c:	`etarvqogcpuetarvqitcrja`	
$k = 1$:	`dszqupnfbotdszquphsbqiz`	
$k = 2$:	`cryptomeanscryptography`	(✓ makes sense)
$k = 3$:	`bqxosnldzmrbqxosnfqzogx`	
⋮	⋮	

Lesson: Large Key Space

Lesson

Any secure encryption scheme must have a sufficiently **large key space** (so as to make a brute-force attack computationally infeasible).

Lesson: Large Key Space

Lesson

Any secure encryption scheme must have a sufficiently **large key space** (so as to make a brute-force attack computationally infeasible).

- **Not sufficient**: the next cipher has a huge key space, yet is easily broken.
- Also, a brute-force attack identifies the key only once the ciphertext is long enough (more details later).

The Mono-alphabetic Substitution Cipher

- $\mathcal{K} = \{$all bijections (or permutations) from $\{a, \ldots, z\}$ to $\{a, \ldots, z\}\}$.
- An example permutation/key π would be:

The Mono-alphabetic Substitution Cipher

- $\mathcal{K} = \{$all bijections (or permutations) from $\{a, \ldots, z\}$ to $\{a, \ldots, z\}\}$.
- An example permutation/key π would be:
- How large is the key space now?

$$26! \approx 2^{88}$$

CIPHER ALPHABET

A = B	H = A	O = O	V = L
B = V	I = D	P = Y	W = P
C = G	J = Z	Q = F	X = U
D = Q	K = C	R = J	Y = I
E = K	L = W	S = X	Z = R
F = M	M = S	T = H	
G = N	N = E	U = T	

The Mono-alphabetic Substitution Cipher, Formally

Definition 1 (Mono-alphabetic Substitution Cipher)

- Gen: choose uniform $k = \pi \in \mathcal{K}$.
- $\mathrm{Enc}_k(m_1 \cdots m_t)$: output $c_1 \cdots c_t$, where $c_i := \pi(m_i)$.
- $\mathrm{Dec}_k(c_1 \cdots c_t)$: output $m_1 \cdots m_t$, where $m_i := \pi^{-1}(c_i)$.

The Mono-alphabetic Substitution Cipher, Formally

Definition 1 (Mono-alphabetic Substitution Cipher)

- Gen: choose uniform $k = \pi \in \mathcal{K}$.
- $\mathrm{Enc}_k(m_1 \cdots m_t)$: output $c_1 \cdots c_t$, where $c_i := \pi(m_i)$.
- $\mathrm{Dec}_k(c_1 \cdots c_t)$: output $m_1 \cdots m_t$, where $m_i := \pi^{-1}(c_i)$.

Correctness

For each i: $\pi^{-1}(c_i) = \pi^{-1}(\pi(m_i)) = m_i$.

Is the Mono-alphabetic Substitution Cipher Secure?

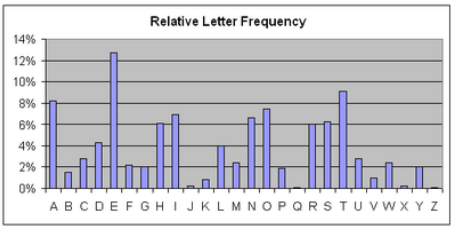
Is the Mono-alphabetic Substitution Cipher Secure?

No — frequency analysis

The substitution is fixed, so the frequency of each ciphertext letter equals the frequency of the corresponding plaintext letter. Match observed frequencies against known English frequencies.

- Sample ciphertext:

  `jgrmqoyghmvbjwrwqfpwhgf...`



Relative letter frequency in English.

Lesson: Smooth the Frequency Distribution

Lesson

Need to "smooth out" the frequency distribution of characters
(rule out statistical tests).

- This needs a larger ciphertext than was needed to break the shift cipher.

Vigenère Cipher (Multiple Shift Cipher)

$\mathcal{K} = \{a, \ldots, z\}^t$ for a period t (description of the scheme by example)

Example (from Katz and Lindell)

plaintext:	`tellhimaboutme`
key:	`cafecafecafeca`
ciphertext:	`veqpjiredozxoe`

- How large is the key space? 26^t ($\approx 2^{70}$ for $t = 15$).

Vigenère Cipher (Multiple Shift Cipher)

$\mathcal{K} = \{a, \ldots, z\}^t$ for a period t (description of the scheme by example)

Example (from Katz and Lindell)

plaintext:	`tellhimaboutme`
key:	`cafecafecafeca`
ciphertext:	`veqpjiredozxoe`

- How large is the key space? 26^t ($\approx 2^{70}$ for $t = 15$).
- **Smooth out the distribution**: the two '`l`'s in '`tell`' are encrypted differently.

Vigenère Cipher (Multiple Shift Cipher)

$\mathcal{K} = \{a, \ldots, z\}^t$ for a period t (description of the scheme by example)

Example (from Katz and Lindell)

plaintext:	`tellhimaboutme`
key:	`cafecafecafeca`
ciphertext:	`veqpjiredozxoe`

- How large is the key space? 26^t ($\approx 2^{70}$ for $t = 15$).
- **Smooth out the distribution**: the two '`l`'s in '`tell`' are encrypted differently.
- Remained unbroken for centuries.

Is the Vigenère Cipher Secure?

Attack (when t is known)

- For each $j \in \{1, \ldots, t\}$, consider the stream $c_j, c_{j+t}, c_{j+2t}, \ldots$
- Every letter in this stream is shifted by the *same* value k_j — it is a shift-cipher encryption.
- Can do frequency analysis: guess that its most frequent letter corresponds to 'e' and recover k_j.

- Next: a more methodical attack (removing the guesswork).

Better Attack on the Shift Cipher

- Let p_i $(0 \leq i \leq 25)$ denote the frequency of the ith English letter in normal English plaintext.
  - One can compute $\sum_i p_i^2 \approx 0.065$.
- Let q_i $(0 \leq i \leq 25)$ denote the frequency of the ith English letter in the given ciphertext.
  - If the key is k, then $q_{i+k} \approx p_i$ for every i.

Better Attack on the Shift Cipher

- Let p_i $(0 \leq i \leq 25)$ denote the frequency of the ith English letter in normal English plaintext.
  - One can compute $\sum_i p_i^2 \approx 0.065$.
- Let q_i $(0 \leq i \leq 25)$ denote the frequency of the ith English letter in the given ciphertext.
  - If the key is k, then $q_{i+k} \approx p_i$ for every i.
- Output the $j \in \{0, \ldots, 25\}$ for which $I_j = \sum_i p_i \, q_{i+j}$ (subscripts mod 26) is closest to 0.065: this j is (very likely) the key.
  - $I_k \approx \sum_i p_i^2 \approx 0.065$, while a wrong shift misaligns the frequencies, giving $I_j \approx 0.038$.

Better Attack on the Shift Cipher

- Let p_i $(0 \le i \le 25)$ denote the frequency of the ith English letter in normal English plaintext.
  - One can compute $\sum_i p_i^2 \approx 0.065$.
- Let q_i $(0 \le i \le 25)$ denote the frequency of the ith English letter in the given ciphertext.
  - If the key is k, then $q_{i+k} \approx p_i$ for every i.
- Output the $j \in \{0, \ldots, 25\}$ for which $I_j = \sum_i p_i \, q_{i+j}$ (subscripts mod 26) is closest to 0.065: this j is (very likely) the key.
  - $I_k \approx \sum_i p_i^2 \approx 0.065$, while a wrong shift misaligns the frequencies, giving $I_j \approx 0.038$.

Vigenère cipher with known period t: apply the same attack separately to each stream $c_\beta, c_{\beta+t}, c_{\beta+2t}, \ldots$ for $\beta \in \{1, \ldots, t\}$.

What if the Period t is Unknown?

- **Simple strategy**: try all possible choices of t!
  (Only a few possibilities, given an upper bound on t.)

What if the Period t is Unknown?

- **Simple strategy**: try all possible choices of t!
  (Only a few possibilities, given an upper bound on t.)
- **More efficient**: for a given τ, let r_i $(0 \leq i \leq 25)$ denote the frequency of the ith English letter in the stream $c_1, c_{1+\tau}, c_{1+2\tau}, \ldots$ and compute

$$S_\tau = \sum_i r_i^2.$$

What if the Period t is Unknown?

- **Simple strategy**: try all possible choices of t!
  (Only a few possibilities, given an upper bound on t.)
- **More efficient**: for a given τ, let r_i $(0 \leq i \leq 25)$ denote the frequency of the ith English letter in the stream $c_1, c_{1+\tau}, c_{1+2\tau}, \ldots$ and compute

$$S_\tau = \sum_i r_i^2.$$

- If $\tau = t$: the stream is a single shift cipher, so $S_\tau \approx \sum_i p_i^2 \approx 0.065$.
- Otherwise: the stream mixes letters encrypted under different shifts, so $r_i \approx \frac{1}{26}$ and $S_\tau \approx 26 \times \frac{1}{26^2} = 0.038$ (heuristically).
- Output the smallest τ with $S_\tau \approx 0.065$ (multiples of t pass the test too).

Lesson: Ad Hoc Fixes Break

Lesson

Ad hoc fixes are likely to break.

Design Principles

Three pillars of modern cryptography

1. **Definitions** — formally specify what security means.
2. **Assumptions** — state precisely what is assumed to be hard.
3. **Proofs of security** — show a construction meets the definition under the assumptions.

- We study each in turn, then use them to define **secure encryption**.

Importance of Definitions

Why bother defining security?

If you don't understand what you want to achieve, how can you possibly know when (or if) you have achieved it?

Importance of Definitions

Why bother defining security?

If you don't understand what you want to achieve, how can you possibly know when (or if) you have achieved it?

Definitions let us ask

What is the "right" definition for a setting? Does a scheme satisfy this definition?

Importance of Definitions

Why bother defining security?

If you don't understand what you want to achieve, how can you possibly know when (or if) you have achieved it?

Definitions let us ask

What is the "right" definition for a setting? Does a scheme satisfy this definition?

Definitions also

Communicate security properties — useful when the scheme is used inside a larger system.

Assumptions

- Unconditional security, howsoever desirable, is not always achievable.

Assumptions

- Unconditional security, howsoever desirable, is not always achievable.
- We need assumptions — e.g. $P \neq NP$, and in fact more!

Assumptions

- Unconditional security, howsoever desirable, is not always achievable.
- We need assumptions — e.g. $P \neq NP$, and in fact more!
- An assumption should be **clearly stated**.
  - We can then validate / invalidate it.
  - We can compare schemes based on different assumptions.

Proofs of Security

What a proof gives us

A construction satisfies the considered security definition under the specified assumptions.

Proofs of Security

What a proof gives us

A construction satisfies the considered security definition under the specified assumptions.

Very strong guarantee!

No attack of the considered construction can succeed, as long as the assumptions hold.

Proofs of Security

What a proof gives us

A construction satisfies the considered security definition under the specified assumptions.

Very strong guarantee!

No attack of the considered construction can succeed, as long as the assumptions hold.

Limitations

- The **definition** may not capture the real-world attack space.

Proofs of Security

What a proof gives us

A construction satisfies the considered security definition under the specified assumptions.

Very strong guarantee!

No attack of the considered construction can succeed, as long as the assumptions hold.

Limitations

- The **definition** may not capture the real-world attack space.
- The **assumption** might be invalid.
- The **implementation** might be buggy.

Crypto Is Still a Bit of an Art

- Crypto remains a bit of an **art**!
- Yet, definitions and security proofs are immensely valuable — they **reduce the attack space**.

Defining Secure Encryption

Goal

Define **perfect** secrecy.

Defining Secure Encryption

Goal

Define **perfect** secrecy.

- This setting illustrates **definitions** and **proofs**...
- ...but **not assumptions** for now (perfect secrecy needs none).

Private-Key Encryption (Syntax)

Definition 2 (Private-Key Encryption Scheme)

A **private-key encryption scheme** is a message space $\mathcal{M}$, a key space $\mathcal{K}$, and algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$:

- Gen (key generation): outputs $k \in \mathcal{K}$.
- Enc (encryption): on key k and message $m \in \mathcal{M}$, outputs ciphertext $c \leftarrow \mathsf{Enc}_k(m)$.
- Dec (decryption): on key k and ciphertext c, outputs $m := \mathsf{Dec}_k(c)$ (or "error").

Private-Key Encryption (Syntax)

Definition 2 (Private-Key Encryption Scheme)

A **private-key encryption scheme** is a message space $\mathcal{M}$, a key space $\mathcal{K}$, and algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$:

- Gen (key generation): outputs $k \in \mathcal{K}$.
- Enc (encryption): on key k and message $m \in \mathcal{M}$, outputs ciphertext $c \leftarrow \mathsf{Enc}_k(m)$.
- Dec (decryption): on key k and ciphertext c, outputs $m := \mathsf{Dec}_k(c)$ (or "error").

Correctness

For all $m \in \mathcal{M}$ and k output by Gen:
$\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = m$.

Private-Key Encryption (Syntax)

Definition 2 (Private-Key Encryption Scheme)

A **private-key encryption scheme** is a message space $\mathcal{M}$, a key space $\mathcal{K}$, and algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$:

- Gen (key generation): outputs $k \in \mathcal{K}$.
- Enc (encryption): on key k and message $m \in \mathcal{M}$, outputs ciphertext $c \leftarrow \mathsf{Enc}_k(m)$.
- Dec (decryption): on key k and ciphertext c, outputs $m := \mathsf{Dec}_k(c)$ (or "error").

Correctness

For all $m \in \mathcal{M}$ and k output by Gen:
$\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = m$.

Secret

k must be kept secret.

Attempt at Security?

Is it enough to keep your key secret?

Attempt at Security?

Is it enough to keep your key secret?

No — key secrecy is not itself a security goal: $\mathsf{Enc}_k(m) = m$ keeps k perfectly secret, yet reveals everything.

Attempt at Security?

Is it enough to keep your key secret?

No — key secrecy is not itself a security goal: $\mathrm{Enc}_k(m) = m$ keeps k perfectly secret, yet reveals everything.

Is it enough if the attacker cannot recover the entire message?

Attempt at Security?

Is it enough to keep your key secret?

No — key secrecy is not itself a security goal: $\mathrm{Enc}_k(m) = m$ keeps k perfectly secret, yet reveals everything.

Is it enough if the attacker cannot recover the entire message?

No — leaking "most" of the message is already a disaster.

Attempt at Security?

Is it enough to keep your key secret?

No — key secrecy is not itself a security goal: $\mathrm{Enc}_k(m) = m$ keeps k perfectly secret, yet reveals everything.

Is it enough if the attacker cannot recover the entire message?

No — leaking "most" of the message is already a disaster.

Is it enough if every character / bit of the message is hidden?

Attempt at Security?

Is it enough to keep your key secret?

No — key secrecy is not itself a security goal: $\mathrm{Enc}_k(m) = m$ keeps k perfectly secret, yet reveals everything.

Is it enough if the attacker cannot recover the entire message?

No — leaking "most" of the message is already a disaster.

Is it enough if every character / bit of the message is hidden?

No — the attacker could still learn relations among bits.

Attempt at Security?

Is it enough to keep your key secret?

No — key secrecy is not itself a security goal: $\mathrm{Enc}_k(m) = m$ keeps k perfectly secret, yet reveals everything.

Is it enough if the attacker cannot recover the entire message?

No — leaking "most" of the message is already a disaster.

Is it enough if every character / bit of the message is hidden?

No — the attacker could still learn relations among bits.

Can we require that the attacker doesn't learn *anything* about the message?

Attempt at Security?

Is it enough to keep your key secret?

No — key secrecy is not itself a security goal: $\mathrm{Enc}_k(m) = m$ keeps k perfectly secret, yet reveals everything.

Is it enough if the attacker cannot recover the entire message?

No — leaking "most" of the message is already a disaster.

Is it enough if every character / bit of the message is hidden?

No — the attacker could still learn relations among bits.

Can we require that the attacker doesn't learn *anything* about the message?

This is the right idea — let us make it precise.

The Right Definition

Informal definition

Regardless of any information an attacker already has, a ciphertext should leak **no additional information** about the plaintext.

The Right Definition

Informal definition

Regardless of any information an attacker already has, a ciphertext should leak **no additional information** about the plaintext.

Caveat

What about the **length** of the message? The ciphertext may reveal the message length; perfect secrecy is over a *fixed-length* message space.

Perfect Security: Formally

No assumptions!

Perfect secrecy is an **information-theoretic** notion: it holds against a computationally *unbounded* adversary and rests on **no computational assumptions**.

- We first fix notation, then give three equivalent definitions.

Notation

Fix $\mathcal{M}$, $\mathcal{K}$ and $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Treat the key, message, and ciphertext as **random variables**.

Random variable

A variable that takes on (discrete) values with certain probabilities.

- K: random variable for the output of Gen(), so $\Pr[K = k] = \Pr[\mathsf{Gen}() \text{ outputs } k]$.
- M: random variable for the message, ranging over $\mathcal{M}$. If $\Pr[M = m] > 0$ then $m \in \mathcal{M}$. *(Application specific, e.g.* $\Pr[M =$ *"Attack!"*$] = .6$, $\Pr[M =$ *"Retreat"*$] = .4$.*)*
- C: random variable for the ciphertext, $\Pr[C = c] = \Pr[\mathsf{Enc}_K(M) = c]$ (over the randomness of Enc as well).

Notation

Fix $\mathcal{M}$, $\mathcal{K}$ and $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Treat the key, message, and ciphertext as **random variables**.

Random variable

A variable that takes on (discrete) values with certain probabilities.

- K: random variable for the output of Gen(), so $\Pr[K = k] = \Pr[\mathsf{Gen}() \text{ outputs } k]$.
- M: random variable for the message, ranging over $\mathcal{M}$. If $\Pr[M = m] > 0$ then $m \in \mathcal{M}$. *(Application specific, e.g.* $\Pr[M = \text{“Attack!”}] = .6$, $\Pr[M = \text{“Retreat”}] = .4$.*)*
- C: random variable for the ciphertext, $\Pr[C = c] = \Pr[\mathsf{Enc}_K(M) = c]$ (over the randomness of Enc as well).

Key independence

M and K are **independent**!

Shift Cipher: Example 1

Setup

Shift cipher, $k \in \{0, \ldots, 25\}$, single-character messages with $\Pr[M = \text{'a'}] = 0.6$, $\Pr[M = \text{'b'}] = 0.4$.

- What is $\Pr[K = k]$?

Shift Cipher: Example 1

Setup

Shift cipher, $k \in \{0, \ldots, 25\}$, single-character messages with $\Pr[M = \text{'a'}] = 0.6$, $\Pr[M = \text{'b'}] = 0.4$.

- What is $\Pr[K = k]$? $= 1/26$.

Shift Cipher: Example 1

Setup

Shift cipher, $k \in \{0, \ldots, 25\}$, single-character messages with $\Pr[M = \text{`a'}] = 0.6$, $\Pr[M = \text{`b'}] = 0.4$.

- What is $\Pr[K = k]$? $= 1/26$.
- What is $\Pr[C = \text{`z'}]$?

Shift Cipher: Example 1

Setup

Shift cipher, $k \in \{0, \ldots, 25\}$, single-character messages with $\Pr[M = \text{`a'}] = 0.6$, $\Pr[M = \text{`b'}] = 0.4$.

- What is $\Pr[K = k]$? $= 1/26$.
- What is $\Pr[C = \text{`z'}]$?

$$\begin{aligned}\Pr[C = \text{`z'}] &= \Pr[\mathsf{Enc}_K(M) = \text{`z'}] \\ &= .6 \times \Pr[\mathsf{Enc}_K(\text{`a'}) = \text{`z'}] + .4 \times \Pr[\mathsf{Enc}_K(\text{`b'}) = \text{`z'}] \\ &= .6 \times \tfrac{1}{26} + .4 \times \tfrac{1}{26} = \tfrac{1}{26}.\end{aligned}$$

Shift Cipher: Example 2

Setup

Two-character messages: $\Pr[M = \text{'aa'}] = 0.6$, $\Pr[M = \text{'ab'}] = 0.4$. Observe $C = \text{'zz'}$.

- Can you guess what m is?

Shift Cipher: Example 2

Setup

Two-character messages: $\Pr[M = \text{'aa'}] = 0.6$, $\Pr[M = \text{'ab'}] = 0.4$. Observe $C = \text{'zz'}$.

- Can you guess what m is?
- Only $m = \text{'aa'}$ can encrypt to 'zz' (both characters shift by the same k), so $m = \text{'aa'}$ with certainty.

Shift Cipher: Example 2

Setup

Two-character messages: $\Pr[M = \text{'aa'}] = 0.6$, $\Pr[M = \text{'ab'}] = 0.4$. Observe $C = \text{'zz'}$.

- Can you guess what m is?
- Only $m = \text{'aa'}$ can encrypt to 'zz' (both characters shift by the same k), so $m = \text{'aa'}$ with certainty.

Conclusion

The shift cipher is **insecure** even for messages of length two.

The Right Definition: Formally

Informal

Regardless of any information an attacker already has, a ciphertext should leak **no** additional information about the plaintext.

The Right Definition: Formally

Informal

Regardless of any information an attacker already has, a ciphertext should leak **no** additional information about the plaintext.

Definition 3 (Definition 1 — Perfect Secrecy)

An encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ is **perfectly secret** if for every probability distribution over $\mathcal{M}$, every message $m \in \mathcal{M}$, and every ciphertext c with $\Pr[C = c] > 0$:

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

Example

Setup

Shift cipher, $k \in \{0, \ldots, 25\}$, $\Pr[M = \text{`a'}] = 0.6$, $\Pr[M = \text{`b'}] = 0.4$.

We have $\Pr[K = k] = 1/26$ and $\Pr[C = \text{`z'}] = 1/26$. By Bayes:

$$\begin{aligned}\Pr[M = \text{`a'} \mid C = \text{`z'}] &= \frac{\Pr[C = \text{`z'} \mid M = \text{`a'}] \cdot \Pr[M = \text{`a'}]}{\Pr[C = \text{`z'}]} \\ &= \frac{\frac{1}{26} \cdot 0.6}{\frac{1}{26}} = 0.6 = \Pr[M = \text{`a'}].\end{aligned}$$

Bayes' Theorem

$$\Pr[A \mid B] = \frac{\Pr[B \mid A] \cdot \Pr[A]}{\Pr[B]}.$$

Definition 2

Definition 4 (Definition 2 — Perfect Secrecy)

An encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ is **perfectly secret** if for every two messages $m, m' \in \mathcal{M}$ and every ciphertext c (in the ciphertext space):

$$\Pr[\mathsf{Enc}_K(m) = c] = \Pr[\mathsf{Enc}_K(m') = c],$$

where the probability is only over K and the random coins of Enc.

Definition 2

Definition 4 (Definition 2 — Perfect Secrecy)

An encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ is **perfectly secret** if for every two messages $m, m' \in \mathcal{M}$ and every ciphertext c (in the ciphertext space):

$$\Pr[\mathsf{Enc}_K(m) = c] = \Pr[\mathsf{Enc}_K(m') = c],$$

where the probability is only over K and the random coins of Enc.

Claim

Definition 1 is equivalent to Definition 2.

Definition 2 $\Rightarrow$ Definition 1

Given / To prove

Given: $\forall m, m', c:\ \Pr[\mathrm{Enc}_K(m) = c] = \Pr[\mathrm{Enc}_K(m') = c]$.
To prove: for every distribution on $\mathcal{M}$, every m, and every c with $\Pr[C = c] > 0$: $\Pr[M = m \mid C = c] = \Pr[M = m]$.

By Bayes' theorem and the law of total probability,

$$\Pr[M = m \mid C = c] = \frac{\Pr[C = c \mid M = m] \cdot \Pr[M = m]}{\Pr[C = c]} = \frac{\Pr[\mathrm{Enc}_K(m) = c] \cdot \Pr[M = m]}{\sum_{m'} \Pr[C = c \mid M = m'] \cdot \Pr[M = m']}$$
$$= \frac{\Pr[\mathrm{Enc}_K(m) = c] \cdot \Pr[M = m]}{\sum_{m'} \Pr[\mathrm{Enc}_K(m') = c] \cdot \Pr[M = m']} = \frac{\Pr[M = m]}{\sum_{m'} \Pr[M = m']} = \Pr[M = m].$$

Definition 2 ⇒ Definition 1

Given / To prove

Given: $\forall m, m', c: \ \Pr[\mathrm{Enc}_K(m) = c] = \Pr[\mathrm{Enc}_K(m') = c]$.
To prove: for every distribution on $\mathcal{M}$, every m, and every c with $\Pr[C = c] > 0$: $\Pr[M = m \mid C = c] = \Pr[M = m]$.

By Bayes' theorem and the law of total probability,

$$\Pr[M = m \mid C = c] = \frac{\Pr[C = c \mid M = m] \cdot \Pr[M = m]}{\Pr[C = c]} = \frac{\Pr[\mathrm{Enc}_K(m) = c] \cdot \Pr[M = m]}{\sum_{m'} \Pr[C = c \mid M = m'] \cdot \Pr[M = m']}$$
$$= \frac{\Pr[\mathrm{Enc}_K(m) = c] \cdot \Pr[M = m]}{\sum_{m'} \Pr[\mathrm{Enc}_K(m') = c] \cdot \Pr[M = m']} = \frac{\Pr[M = m]}{\sum_{m'} \Pr[M = m']} = \Pr[M = m].$$

Try on your own

Definition 1 ⇒ Definition 2.

Definition 3 (Game Style)

Game $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}$ (eav = eavesdropper)

1. $\mathcal{A}$ outputs $m_0, m_1 \in \mathcal{M}$.
2. Challenger samples $b \leftarrow \{0,1\}$, $k \leftarrow \mathsf{Gen}()$, and the **challenge ciphertext** $c \leftarrow \mathsf{Enc}_k(m_b)$ is given to $\mathcal{A}$.
3. $\mathcal{A}$ outputs b'.
4. **Output** 1 if $b = b'$, and 0 otherwise.

Definition 3 (Game Style)

Game $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}$ (eav = eavesdropper)

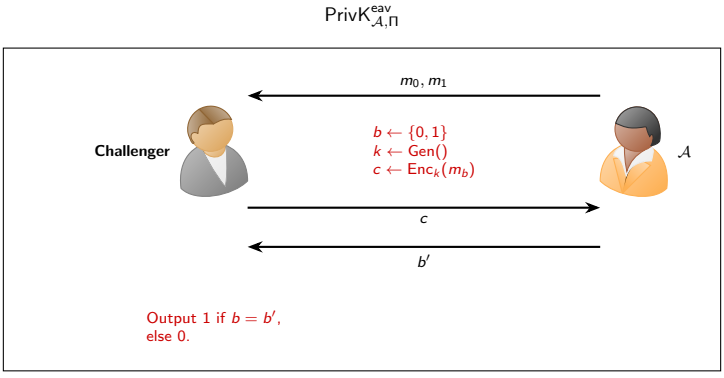
1. $\mathcal{A}$ outputs $m_0, m_1 \in \mathcal{M}$.
2. Challenger samples $b \leftarrow \{0,1\}$, $k \leftarrow \mathsf{Gen}()$, and the **challenge ciphertext** $c \leftarrow \mathsf{Enc}_k(m_b)$ is given to $\mathcal{A}$.
3. $\mathcal{A}$ outputs b'.
4. **Output** 1 if $b = b'$, and 0 otherwise.

Definition 3 — Perfect Indistinguishability

$\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ is **perfectly indistinguishable** if for all (even unbounded) $\mathcal{A}$:

$$\Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1\right] = \frac{1}{2}.$$

$\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}$: Challenger vs Adversary



Remark

$\mathcal{A}$ can always succeed with probability $\frac{1}{2}$. How? Just guess b' at random — so $\frac{1}{2}$ is the best achievable, hence the definition.

Equivalence of the Definitions

Lemma 5

An encryption scheme is **perfectly secret** *if and only if it is* **perfectly indistinguishable**.

- Together with the earlier equivalence, all three definitions coincide.
- **Prove on your own.**

Summary

- Classical ciphers and how they break:
  - Shift cipher — tiny key space (brute force).
  - Mono-alphabetic substitution — frequency analysis.
  - Vigenère — reduces to several shift ciphers once the period is found.
- **Lessons**: large key space (necessary), smooth statistics, and ad hoc fixes break.
- Modern crypto rests on three pillars: **definitions**, **assumptions**, **proofs**.
- **Perfect secrecy** is information-theoretic — three equivalent definitions:
  - **Def. 1**: $\Pr[M = m \mid C = c] = \Pr[M = m]$.
  - **Def. 2**: $\Pr[\mathrm{Enc}_K(m) = c] = \Pr[\mathrm{Enc}_K(m') = c]$.
  - **Def. 3**: perfect indistinguishability, $\Pr[\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi} = 1] = \frac{1}{2}$.

Reading

Today: KL 1.3–1.4, 2.1. Next time: perfect secrecy continued, the one-time pad, and computational security (KL 2.2–2.3, 3.1).