https://brown-csci1660.github.io

CS1660: Intro to Computer Systems Security
Spring 2026

Lecture 3: Confidentiality I

Instructor: **Nikos Triandopoulos**

January 29, 2026

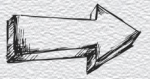
BROWN

CS1660: Announcements

- Course updates
  - Please make sure you complete Homework 0 and Project 0
  - Please make sure you have access to Ed Discussion and Gradescope
  - Project 1 “Cryptography” is going out today; due in 3 weeks

Last class

- Introduction to Computer Security
  - Motivation
  - Basic security concepts
- Cryptography
  - Secret communication
    - Symmetric-key encryption & classical ciphers
    - Perfect secrecy & the One-Time Pad



Completed

Current

Upcoming

Today

- Cryptography
  - Secret communication
    - Symmetric-key encryption & classical ciphers
    - Perfect secrecy & the One-Time Pad
  - Symmetric-key encryption in practice
    - Computational security, pseudo-randomness
    - Stream & block ciphers, modes of operations for encryption, DES & AES
    - Introduction to modern cryptography

Confidentiality

Intro to Crypto

3.0 Symmetric-key encryption

Problem setting: Secret communication

Two parties wish to communicate over a channel

- Alice (sender/source) wants to send a message m to Bob (recipient/destination)

Underlying channel is unprotected

- Eve (attacker/adversary) can eavesdrop any sent messages
- e.g., packet sniffing over networked or wireless communications

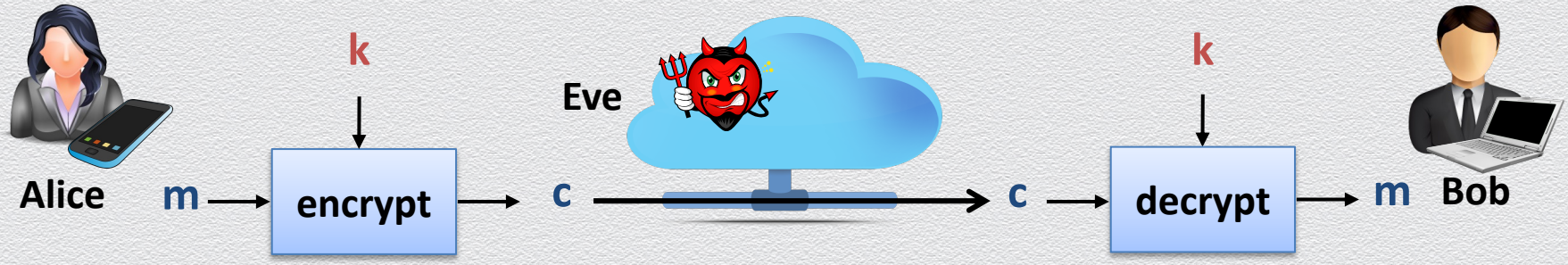
Alice m

m Bob

Solution concept: Symmetric-key encryption

Main idea

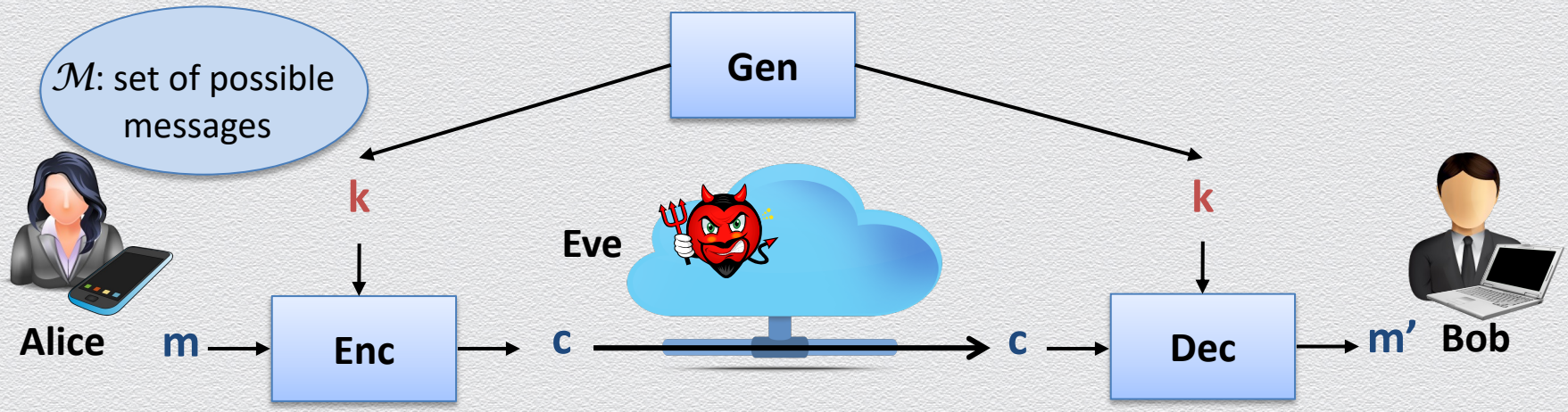
- secretly transform message so that it is **unintelligible** while in transit
  - Alice **encrypts** her message m to **ciphertext c**, which is sent instead of **plaintext m**
  - Bob **decrypts** received message c to original message m
  - Eve can intercept c but "**cannot learn**" m from c
  - Alice and Bob share a **secret key k** that is used for both message transformations



Security tool: Symmetric-key encryption scheme

Abstract cryptographic primitive, **a.k.a. cipher**, defined by

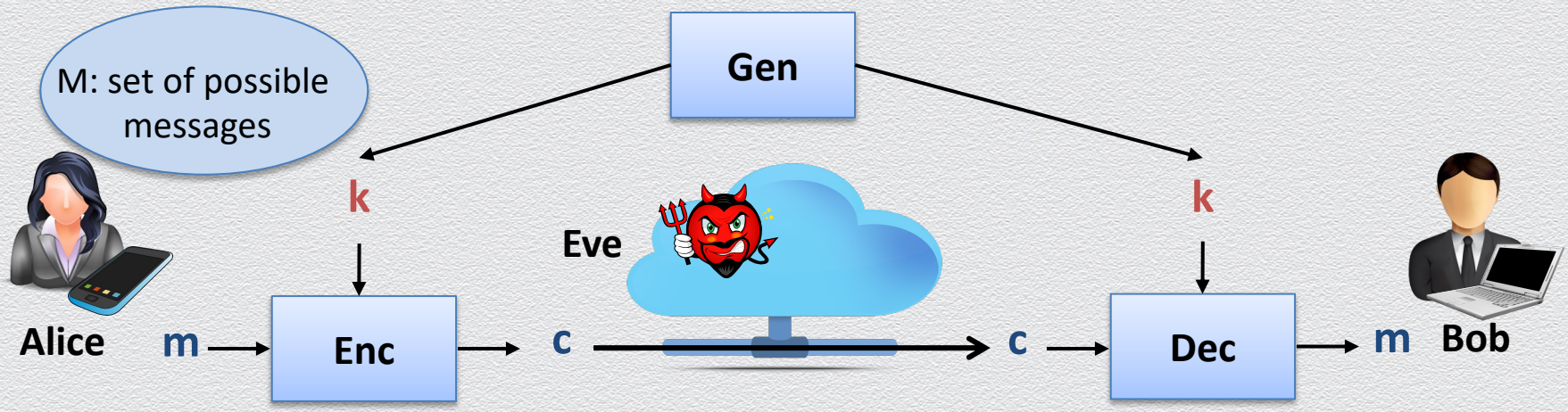
- a **message space $\mathcal{M}$**; and
- a triplet of algorithms **(Gen, Enc, Dec)**
  - Gen is randomized algorithm, Enc may be raldomized, whereas Dec is deterministic
  - Gen outputs a uniformly random key k (from some key space $\mathcal{K}$)



Desired properties for symmetric-key encryption scheme

By design, any symmetric-key encryption scheme should satisfy the following

- **efficiency**: key generation & message transformations "are fast"
- **correctness**: for all m and k, it holds that Dec(Enc(m, k) , k) = m
- **security**: one "cannot learn" plaintext m from ciphertext c



(Auguste) Kerckhoff's principle (1883)

"The cipher method must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience."

Reasoning

- due to security & correctness, Alice & Bob must share some secret info
- if no shared key captures this secret info, it must be captured by Enc, Dec
- but keeping Enc, Dec secret is problematic
  - harder to keep secret an algorithm than a short key (e.g., after user revocation)
  - harder to change an algorithm than a short key (e.g., after secret info is exposed)
  - riskier to rely on custom/ad-hoc schemes than publicly scrutinized/standardized ones

(Auguste) Kerckhoff's principle (1883)

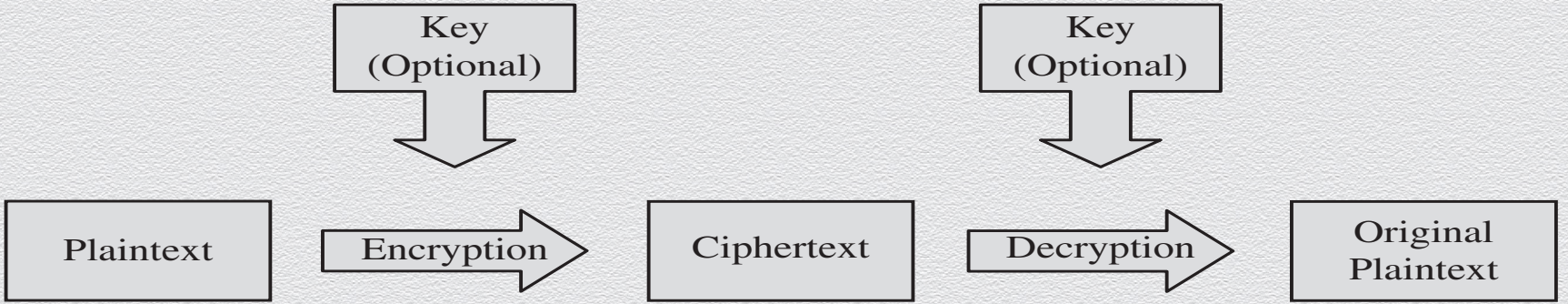
"The cipher method must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience."

General good-hygiene principle (beyond encryption)

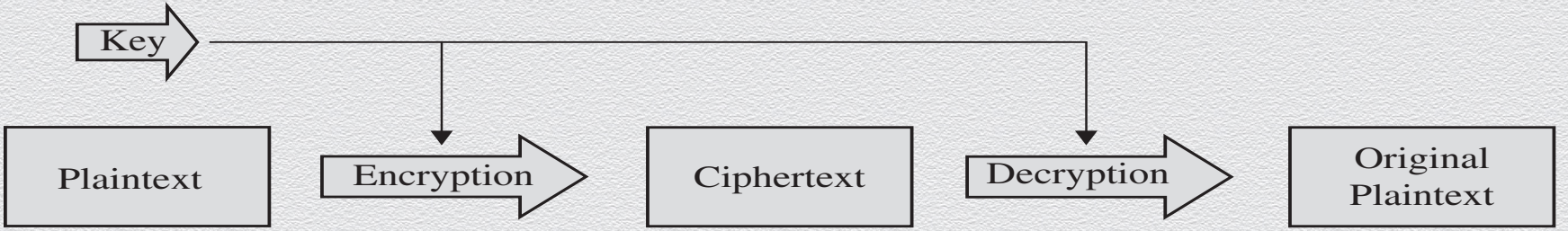
- Security relies solely on keeping secret keys
- System architecture and algorithms are publicly available
- Claude Shannon (1949): *"one ought to design systems under the assumption that the enemy will immediately gain full familiarity with them"*
- Opposite of "security by obscurity" practice

Symmetric-key encryption

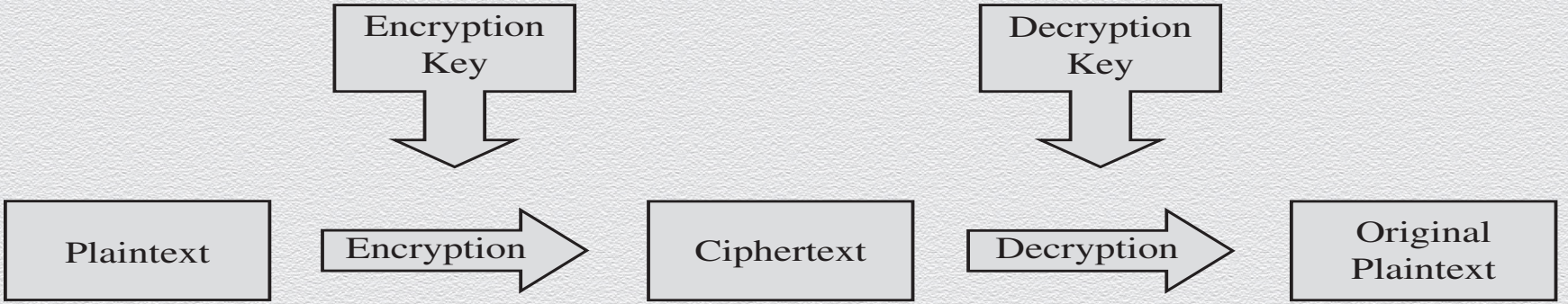
- Also referred to as simply “symmetric encryption”



Symmetric Vs. Asymmetric encryption



(a) Symmetric Cryptosystem



(b) Asymmetric Cryptosystem

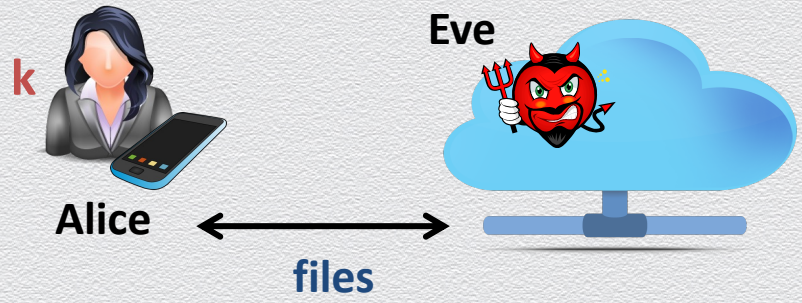
Main application areas

Secure communication

- **encrypt messages** sent among parties
- assumption
  - Alice and Bob **securely generate, distribute & store shared key k**
  - attacker does not learn key k

Secure storage

- **encrypt files** outsourced to the cloud
- assumption
  - Alice **securely generates & stores key k**
  - attacker does not learn key k



Brute-force attack

Generic attack

- given a captured ciphertext c and known key space $\mathcal{K}$, Dec
- strategy is an **exhaustive search**
  - for all possible keys k in $\mathcal{K}$
    - determine if Dec (c,k) is a likely plaintext m
- **requires some knowledge on the message space $\mathcal{M}$**
  - i.e., structure of the plaintext (e.g., PDF file or email message)

Countermeasure

- key should be a **random** value from a **sufficiently large** key space $\mathcal{K}$ to make exhaustive search attacks **infeasible**

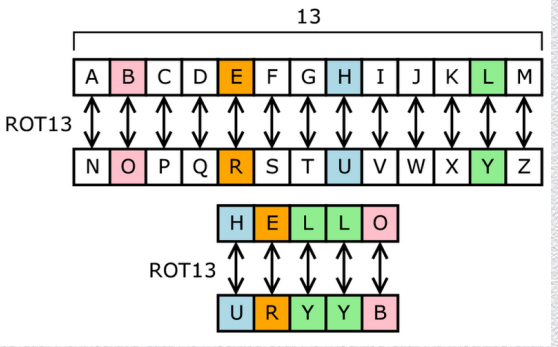
3.1 Classical ciphers

Substitution ciphers

Large class of ciphers: each letter is **uniquely** replaced by another

- key is a (random) permutation over the alphabet characters
- there are 26! ≈ 4×10^{26} possible substitution ciphers
- huge key space (larger than the # of starts in universe)
- e.g., one popular substitution “cipher” for some Internet posts is ROT13

- historically
  - all classical ciphers are of this type



Classical ciphers – general structure

Class of ciphers based on letter substitution

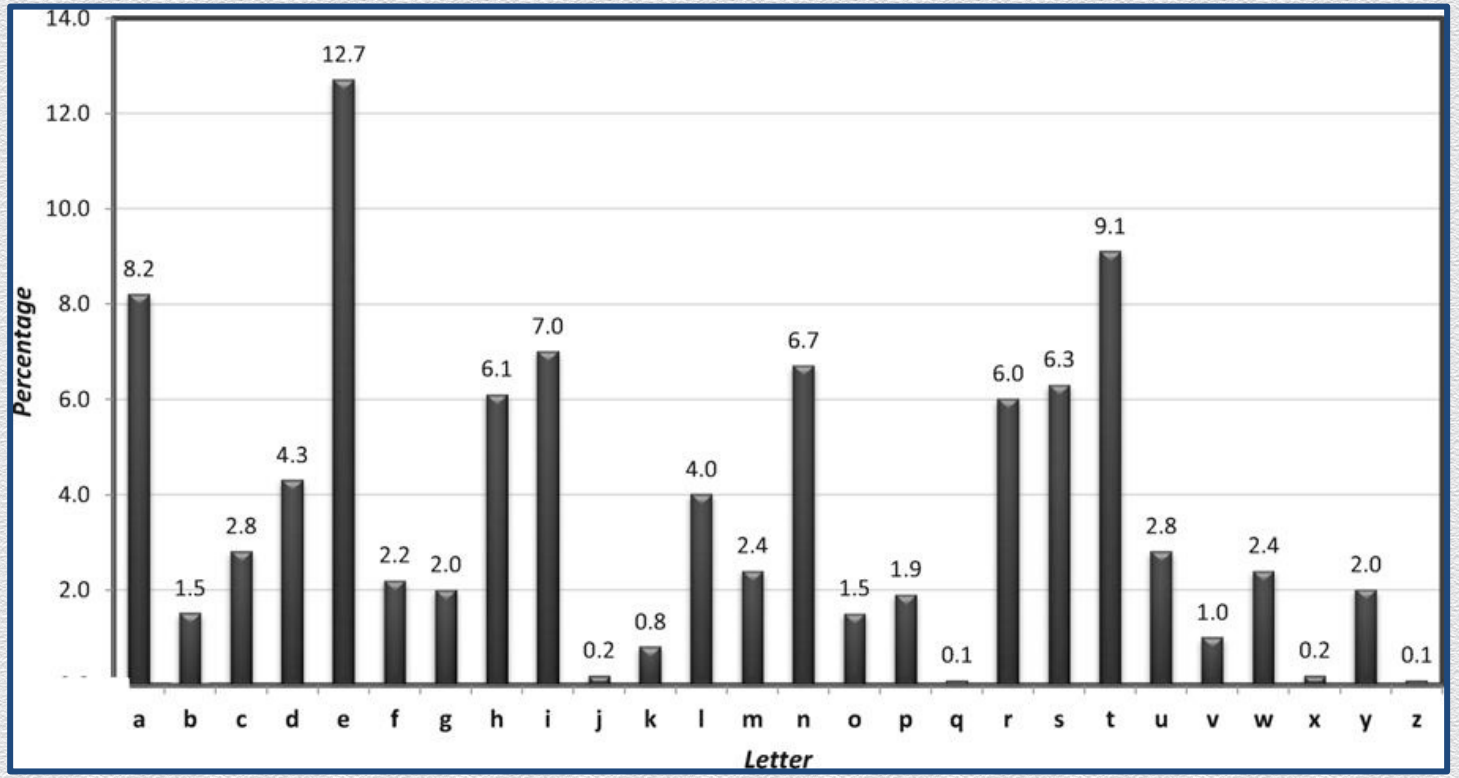
- message space $\mathcal{M}$ is "**valid words**" from a given alphabet
  - e.g., English text without spaces, punctuation or numerals
  - characters can be represented as numbers in [0:25]
- based on a predetermined **1-1** character mapping
  - map each (plaintext) character into another **unique** (ciphertext) character
  - typically defined as a "**shift**" of each plaintext character by a **fixed** per alphabet character number of positions in a canonical ordering of the characters in the alphabet
- encryption: character shifting occurs with "**wrap-around**" (using mod 26 addition)
- decryption: **undo shifting** of characters with "wrap-around" (using mod 26 subtraction)

Limitations of substitution ciphers

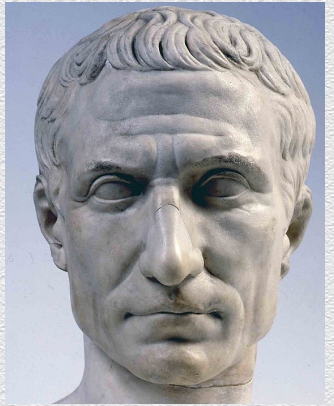
Generally, susceptible to **frequency (and other statistical) analysis**

- letters in a natural language, like English, are not uniformly distributed
- cryptographic attacks against substitution ciphers are possible
  - e.g., by exploiting knowledge of letter frequencies, including pairs and triples
    - most frequent letters in English: e, t, o, a, n, i, ...
    - most frequent digrams: th, in, er, re, an, ...
    - most frequent trigrams: the, ing, and, ion, ...
  - Attack framework first described in a 9th century book by al-Kindi

Letter frequency in (sufficiently large) English text



Classical ciphers – examples



(Julius) Caesar's cipher

- shift each character in the message by 3 positions
  - I.e., 3 instead of 13 positions as in ROT-13
- cryptanalysis
  - **no secret key is used** – based on “security by obscurity”
  - thus the code is trivially insecure once knows Enc (or Dec)

Classical ciphers – examples (II)

Shift cipher

- **keyed extension** of Caesar's cipher
- randomly set key k in [0:25]
  - shift each character in the message by k positions
- cryptanalysis
  - **brute-force attacks** are effective given that
    - **key space is small** (26 possibilities or, actually, 25 as 0 should be avoided)
    - message space M is **restricted to "valid words"**
      - e.g., corresponding to valid English text

Alternative attack against “shift cipher”

- brute-force attack + inspection if English “make sense” is quite **manual**
- a better **automated** attack is based on statistics
  - if character i (in [0:25]) in the alphabet has frequency p_i (in [0..1]), then
    - from known statistics, we know that $\Sigma_i\ p_i^2 \approx 0.065$, so
    - since character i (in plaintext) is mapped to character i + k (in ciphertext)
      - if $L_j = \Sigma_i\ p_i\ q_{i+j}$, then we expect that $L_k \approx 0.065$ (q_i: frequency of character i in ciphertext)
- thus, a brute-force attack can **test** all possible keys w.r.t. the **above criterion**
  - the search space **remains the same**
  - yet, the condition to finish the search **becomes much simpler**: Choose j so that $L_j \approx 0.065$

Classical ciphers – examples (III)

Mono-alphabetic substitution cipher

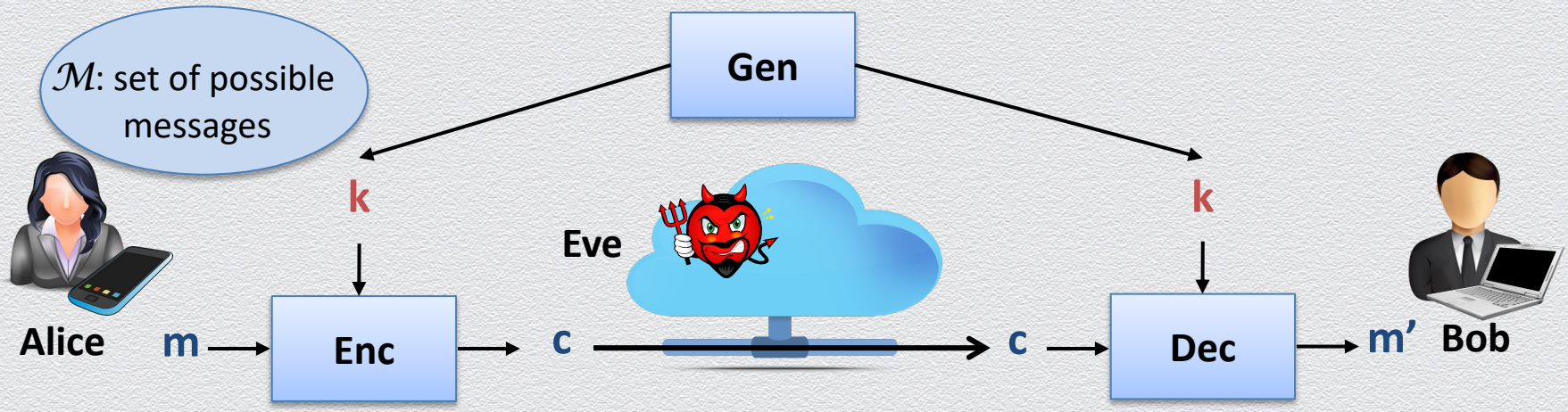
- **generalization** of shift cipher
- key space defines **permutation** on alphabet
  - use a **1-1 mapping between characters** in the alphabet to produce ciphertext
  - i.e., shift each **distinct** character in the plaintext (by some appropriate number of positions defined by the key) to get a **distinct** character in the ciphertext
- cryptanalysis
  - key space is large (of the order of 26! or ~2^{88}) but cipher is vulnerable to attacks
  - character mapping is **fixed** by key so **plaintext & ciphertext exhibit same statistics**

3.2 Perfect secrecy

Security tool: Symmetric-key encryption scheme

Abstract cryptographic primitive, **a.k.a. cipher**, defined by

- a **message space $\mathcal{M}$**; and
- a triplet of algorithms **(Gen, Enc, Dec)**
  - Gen is randomized algorithm, Enc may be raldomized, whereas Dec is deterministic
  - Gen outputs a uniformly random key k (from some key space $\mathcal{K}$)



Probabilistic formulation

Desired properties

- Efficiency
- Correctness
- Security

Our setting so far is a random experiment

- a message m is chosen according to $\mathcal{D}_{\mathcal{M}}$
- a key k is chosen according to $\mathcal{D}_{\mathcal{K}}$
- $\text{Enc}_k(m) \rightarrow c$ is given to the adversary

Perfect correctness

For any $k \in \mathcal{K}$, $m \in \mathcal{M}$ and any ciphertext c output of $Enc_k(m)$,

it holds that

$$\Pr[\, Dec_k(c) = m \,] = 1$$

Perfect security

Defining security for an encryption scheme is not trivial

- what we mean by "Eve "cannot learn" m (from c)" ?

Attempt 1: Protect the key k!

- Security means that

the adversary should **not** be able to **compute the key k**

- Intuition
  - it'd better be the case that the key is protected!...
- Problem
  - this definition fails to exclude clearly insecure schemes
  - e.g., the key is never used, such as when $Enc_k(m) := m$

necessary condition

but not
sufficient condition!

Attempt 2: Don’t learn m!

- Security means that

  the adversary should **not** be able to **compute the message m**

- Intuition
  - it’d better be the case that the message m is not learned...
- Problem
  - this definition fails to exclude clearly undesirable schemes
  - e.g., those that protect m partially, i.e., they reveal the least significant bit of m

Attempt 3: Learn nothing!

- Security means that

  the adversary should **not** be able to **learn any information about m**

- Intuition
  - it seems close to what we should aim for perfect secrecy…
- Problem
  - this definition ignores the adversary's prior knowledge on $\mathcal{M}$
  - e.g., distribution $\mathcal{D}_{\mathcal{M}}$ may be known or estimated
    - m is a valid text message, or one of "`attack`", "`no attack`" is to be sent

Attempt 4: Learn nothing more!

- Security means that

  the adversary should **not** be able to **learn any <u>additional</u> information on m**

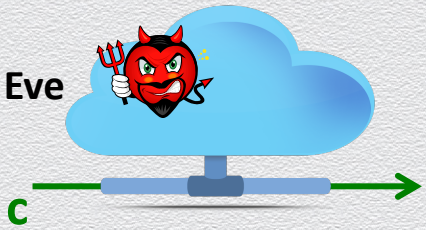
- How can we formalize this?

Alice **m**

$\mathbf{Enc_k(m) \rightarrow c}$

m = { `attack` w/ prob. 0.8 ; `no attack` w/ prob. 0.2 }

Eve's view remains the same!



m = { `attack` w/ prob. 0.8 ; `no attack` w/ prob. 0.2 }

Two equivalent views of perfect secrecy

a posteriori = a priori ~ **C is independent of M**

For every $\mathcal{D}_{\mathcal{M}}$, m ∈ $\mathcal{M}$ and c ∈ $\mathcal{C}$, for which Pr [C = c] > 0, it holds that

Pr[M = m | C = c] = Pr[M = m]

For every m, m' ∈ $\mathcal{M}$ and c ∈ $\mathcal{C}$, it holds that

Pr[Enc_K(m) = c] = Pr[Enc_K(m') = c]

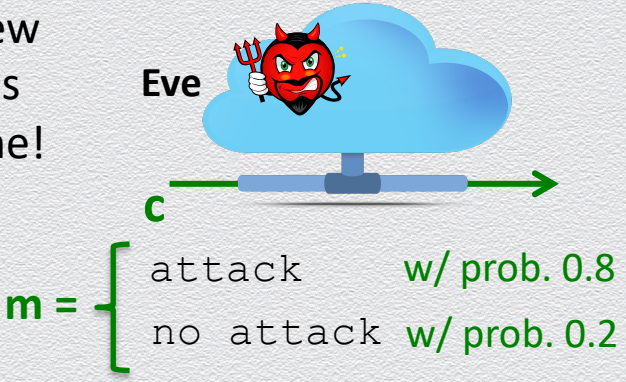
random experiment

$\mathcal{D}_{\mathcal{M}} \rightarrow$ **m = M**

$\mathcal{D}_{\mathcal{K}} \rightarrow$ k = K

Enc_k(m) → c = C

Eve's view remains the same!



Perfect secrecy (or information-theoretic security)

Definition 1

A symmetric-key encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$, is **perfectly secret** if for every $\mathcal{D}_{\mathcal{M}}$, every message m ∈ $\mathcal{M}$ and every ciphertext c ∈ C for which Pr [C = c] > 0, it holds that

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

- Intuitively
  - the *a posteriori* probability that any given message m was actually sent is the **same** as the *a priori* probability that m would have been sent
  - observing the ciphertext reveals **nothing (new)** about the underlying plaintext

Alternative view of perfect secrecy

Definition 2

A symmetric-key encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$, is **perfectly secret** if for every messages m, m' $\in \mathcal{M}$ and every c $\in C$, it holds that

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

- Intuitively
  - the probability distribution $\mathcal{D}_C$ **does not depend** on the plaintext
  - i.e., M and C are **independent** random variables
  - the ciphertext contains "**no information**" about the plaintext
  - "**impossible to distinguish**" an encryption of **m** from an encryption of **m'**

3.3 The One-Time Pad

The one-time pad: A perfect cipher

A type of "substitution" cipher that is "absolutely unbreakable"

- invented in 1917 Gilbert Vernam and Joseph Mauborgne
- "substitution" cipher
  - **individually** replace plaintext characters with **shifted** ciphertext characters
  - **independently** shift each message character in a **random** manner
    - to encrypt a plaintext of length n, use n uniformly random keys $k_1, \ldots, k_n$
- "absolutely unbreakable"
  - **perfectly secure** (when used correctly)
  - based on message-symbol specific **independently random** shifts

The one-time pad (OTP) cipher

Fix n to be any positive integer; set $\mathcal{M} = C = \mathcal{K} = \{0,1\}^n$

- **Gen**: choose n bits uniformly at random (each bit independently w/ prob. .5)
  - Gen $\rightarrow \{0,1\}^n$
- **Enc**: given a key and a message of equal lengths, compute the bit-wise XOR
  - $\text{Enc}(k, m) = \text{Enc}_k(m) \rightarrow k \oplus m$ (i.e., mask the message with the key)
- **Dec**: compute the bit-wise XOR of the key and the ciphertext
  - $\text{Dec}(k, c) = \text{Dec}_k(c) := k \oplus c$
- Correctness
  - trivially, $k \oplus c = k \oplus k \oplus m = 0 \oplus m = m$

OTP is perfectly secure (using Definition 2)

For all n-bit long messages m_1 and m_2 and ciphertexts c, it holds that

$$\Pr[\, E_K(m_1) = c \,] \quad = \quad \Pr[\, E_K(m_2) = c],$$

where probabilities are measured over the possible keys chosen by Gen.

Proof

- events "$Enc_K(m_1) = c$", "$m_1 \oplus K = c$" and "$K = m_1 \oplus c$" are equal-probable
- K is chosen at random, irrespectively of m_1 and m_2, with probability 2^{-n}
- thus, the ciphertext does not reveal anything about the plaintext

OTP characteristics

A "substitution" cipher

- encrypt an n-symbol m using n uniformly random "shift keys" $k_1, k_2, \ldots, k_n$

2 equivalent views

- $\mathcal{K} = \mathcal{M} = C$
- "shift" method

view 1 $\{0,1\}^n$

bit-wise XOR ($m \oplus k$)

or

view 2 G, (G,+) is a group

addition/subtraction (m +/- k)

Perfect secrecy

- since each shift is random, every ciphertext is equally likely for any plaintext

Limitations (on efficiency)

- "shift keys" (1) are **as long as messages** & (2) **can be used only once**

Perfect, but impractical

Despite its perfect security, OTP has 2 notable weaknesses

- the key has to be **as long as** the plaintext
  - limited applicability
  - key-management problem
- the key **cannot be reused** (thus, the “one-time” pad)
  - if reused, perfect security is not satisfied
    - e.g., reusing a key once, leaks the XOR of two plaintext messages
    - this type of leakage can be devastating against secrecy

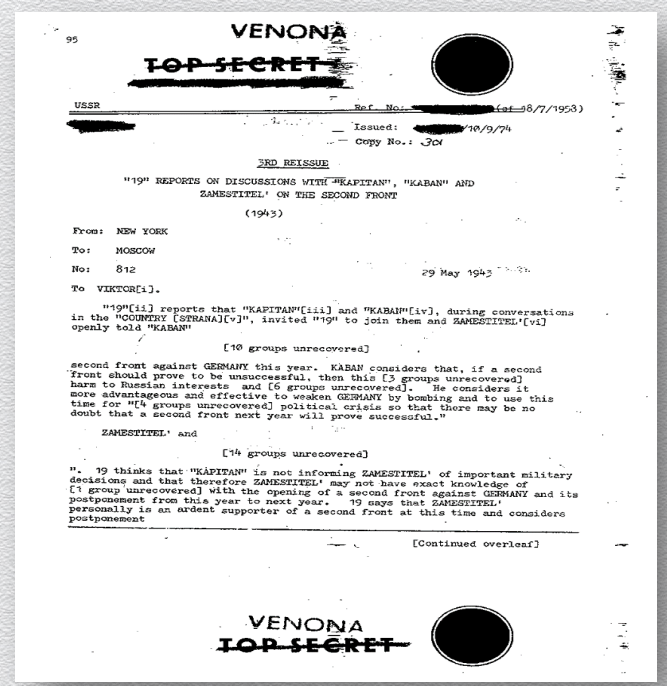
These weakness are detrimental to secure communication

- securely distributing fresh long keys is as hard as securely exchanging messages…

Importance of OTP weaknesses

Inherent trade-off between efficiency/practicality Vs. perfect secrecy

- historically, OTP has been used efficiently & insecurely
  - repeated use of one-time pads compromised communications during the cold war
    - NSA decrypted Soviet messages that were transmitted in the 1940s
    - that was possible because the Soviets reused the keys in the one-time pad scheme
- modern approaches resemble OTP encryption
  - efficiency via use of pseudorandom OTP keys
  - “almost perfect” secrecy



95

VENONA

~~TOP SECRET~~ [redacted]

USSR

Ref. No.: [redacted] (of 8/7/1953)

[redacted] Issued: [redacted]10/9/74

Copy No.: 304

3RD REISSUE

"19" REPORTS ON DISCUSSIONS WITH "KAPITAN", "KABAN" AND ZAMESTITEL' ON THE SECOND FRONT

(1943)

From: NEW YORK

To: MOSCOW

No: 812

29 May 1943

To VIKTOR[i].

"19"[ii] reports that "KAPITAN"[iii] and "KABAN"[iv], during conversations in the "COUNTRY [STRANA][v]", invited "19" to join them and ZAMESTITEL'[vi] openly told "KABAN"

[10 groups unrecovered]

second front against GERMANY this year. KABAN considers that, if a second front should prove to be unsuccessful, then this [3 groups unrecovered] harm to Russian interests and [6 groups unrecovered]. He considers it more advantageous and effective to weaken GERMANY by bombing and to use this time for "[4 groups unrecovered] political crisis so that there may be no doubt that a second front next year will prove successful."

ZAMESTITEL' and

[14 groups unrecovered]

". 19 thinks that "KAPITAN" is not informing ZAMESTITEL' of important military decisions and that therefore ZAMESTITEL' may not have exact knowledge of [1 group unrecovered] with the opening of a second front against GERMANY and its postponement from this year to next year. 19 says that ZAMESTITEL' personally is an ardent supporter of a second front at this time and considers postponement

[Continued overleaf]

VENONA

~~TOP SECRET~~ [redacted]