https://brown-csci1660.github.io

CS1660: Intro to Computer Systems Security
Spring 2025

Lecture 3: Cryptography II

Co-Instructor: **Nikos Triandopoulos**

January 30, 2025

BROWN

CS1660: Announcements

- Override requests
  - Status update
- Course updates
  - Homework 0, Project 0 past due
  - Ed Discussion, Top Hat (code: 084705), Gradescope (to become available soon)
  - Lectures, online reading resources, in-class demos

Today

- Cryptography
  - Symmetric-key ciphers
  - Classical ciphers
  - Perfect secrecy
  - The One Time Pad
  - Ciphers in practice

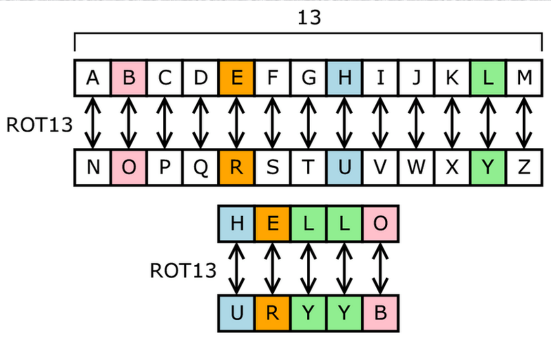
3.0 Classical ciphers

Substitution ciphers

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

- key is a (random) permutation over the alphabet characters
- there are $26! \approx 4 \times 10^{26}$ 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

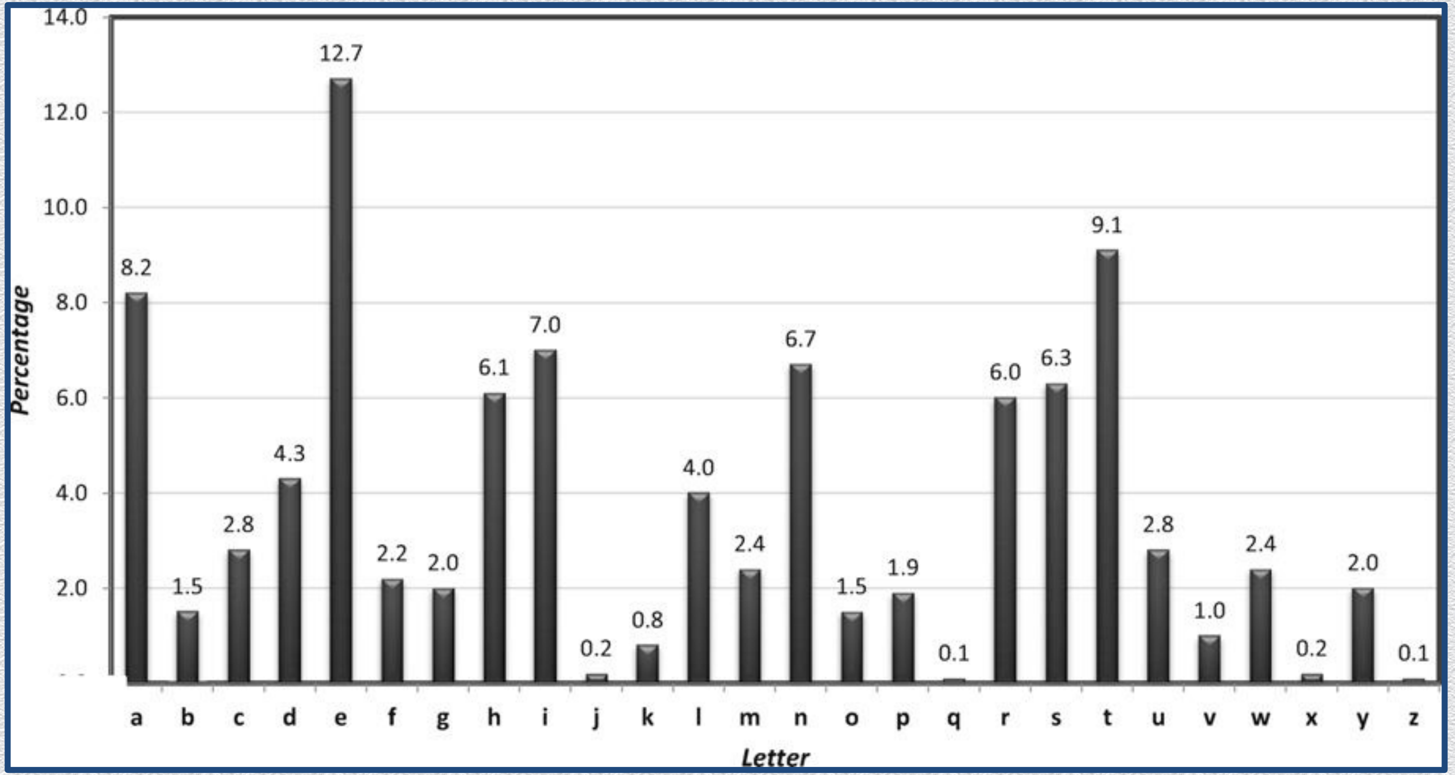
- message space $\mathcal{M}$ is "**valid words**" from a given alphabet
  - e.g., English text without spaces, punctuation or numerals
  - characters can be represented as numbers in [0:25]
- based on a predetermined **1-1** character mapping
  - map each (plaintext) character into another **unique** (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

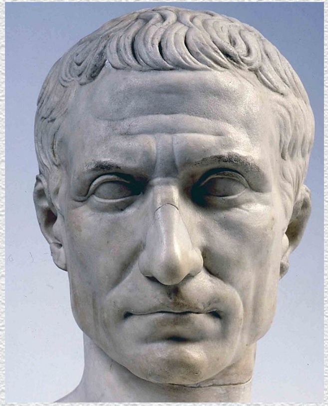
Generally, susceptible to **frequency (and other statistical) analysis**

- 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)
      - $\boxed{\text{if } L_j = \Sigma_i \, p_i \, q_{i+j}\text{, then we expect that } L_k \approx 0.065}$ (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

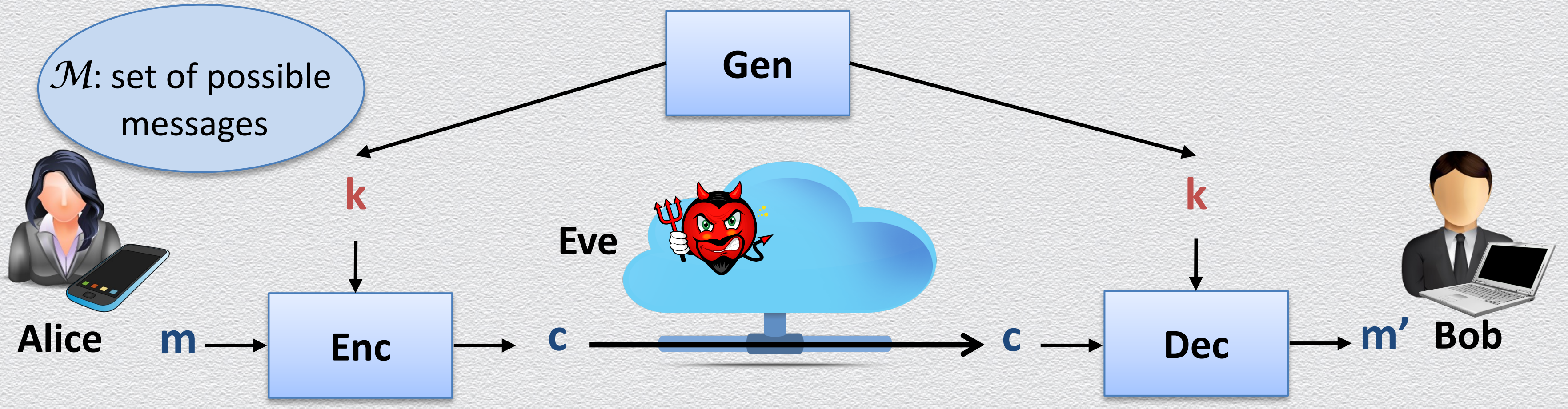
- **generalization** of shift cipher
- key space defines **permutation** on alphabet
  - use a **1-1 mapping between characters** in the alphabet to produce ciphertext
  - i.e., shift each **distinct** character in the plaintext (by some appropriate number of positions defined by the key) to get a **distinct** 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**

3.1 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 $\mathcal{D}_{\mathcal{K}}$
- $\mathrm{Enc}_k(m) \rightarrow c$ 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!...

necessary condition

- Problem
  - this definition fails to exclude clearly insecure schemes
  - e.g., the key is never used, such as when $Enc_k(m) := m$

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

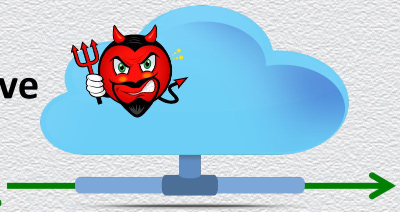
$Enc_k(m) \rightarrow c$

Eve

$$m = \begin{cases} \texttt{attack} & \text{w/ prob. } 0.8 \\ \texttt{no attack} & \text{w/ prob. } 0.2 \end{cases}$$

Eve's view remains the same!

Eve



c

$$m = \begin{cases} \texttt{attack} & \text{w/ prob. } 0.8 \\ \texttt{no attack} & \text{w/ prob. } 0.2 \end{cases}$$

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

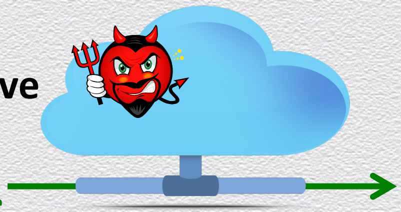
Enc_k(m) $\rightarrow$ c = C

Eve

m = $\begin{cases} \texttt{attack} & \text{w/ prob. 0.8} \\ \texttt{no attack} & \text{w/ prob. 0.2} \end{cases}$

Eve's view remains the same!

Eve



c

m = $\begin{cases} \texttt{attack} & \text{w/ prob. 0.8} \\ \texttt{no attack} & \text{w/ prob. 0.2} \end{cases}$

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 ∈ $\mathcal{C}$ 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' ∈ $\mathcal{M}$ and every c ∈ $\mathcal{C}$, 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'**

3.2 The one-time pad

The one-time pad: A perfect cipher

A type of "<u>substitution</u>" cipher that is "<u>absolutely unbreakable</u>"

- 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 $\mathcal{M} = \mathcal{C} = \mathcal{K} = \{0,1\}^n$

- **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
  - $\mathrm{Enc}(k, m) = \mathrm{Enc}_k(m) \rightarrow k \oplus m$ (i.e., mask the message with the key)
- **Dec**: compute the bit-wise XOR of the key and the ciphertext
  - $\mathrm{Dec}(k, c) = \mathrm{Dec}_k(c) := k \oplus c$
- 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

	view 1	or	view 2
$\mathcal{K} = \mathcal{M} = \mathcal{C}$	$\{0,1\}^n$		G, (G,+) is a group
"shift" method	bit-wise XOR ($m \oplus k$)		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

In spite of its perfect security, OTP has two 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

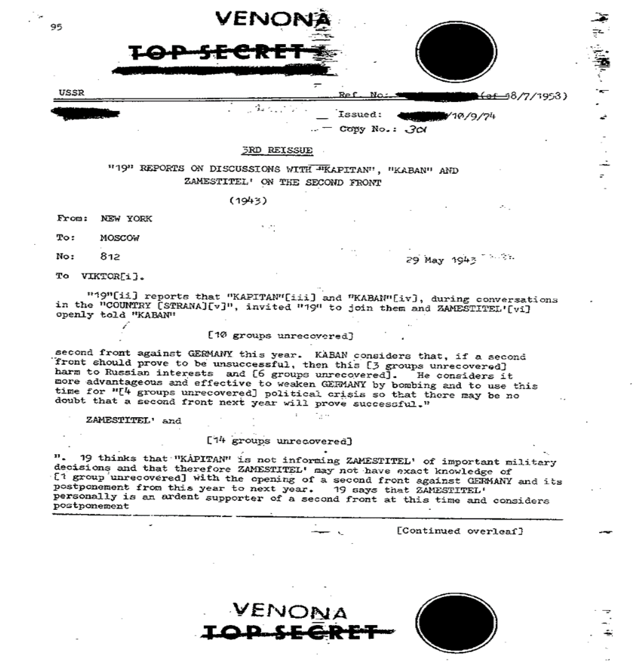
These weakness are detrimental to secure communication

- 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]

[redacted]

USSR

Ref. No: [redacted] (of 18/7/1953)

[redacted]

Issued: [redacted]/10/9/74

Copy No.: 301

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]

3.3 Computational security

The big picture: OPT is perfect but impractical!

We formally defined and constructed the perfectly secure OTP cipher

- This scheme has some major drawbacks
  - it employs a very large key which can be used only once!
- Such limitations are unavoidable and make OTP not practical
  - why?

Now, what?

Our approach: Relax “perfectness”

Initial model

- the **perfect secrecy** (or security) requires that
  - the ciphertext **leaks absolutely no extra information** about the plaintext
  - to adversaries of **unlimited computational power**

Refined model

- a relaxed notion of security, called **computational security**, requires that
  - the ciphertext leaks **a tiny amount of extra information** about the plaintext
  - to adversaries with **bounded computational power**

Security relaxation for encryption

Perfect security: $|k| = 128$ bits, M, $Enc_K(M)$ are independent, **unconditionally**

- no extra information is leaked to any attacker

Computational security: M, $Enc_K(M)$ are independent, **for all practical purposes**

- no extra information is leaked but a tiny amount
  - e.g., with prob. 2^{-128} (or much less than the likelihood of being hit by lighting)
- to computationally bounded attackers
  - e.g., who cannot count to 2^{128} (or invest work of more than one century)
- attacker's best strategy remains **ineffective**
  - random guess a secret key or exhaustive search over key space (brute-force attack)

3.4 Symmetric encryption, revisited: OTP with pseudorandomness

Perfect secrecy & randomness

Role of randomness in encryption is **integral**

- in a perfectly secret cipher, the ciphertext **doesn't depend** on the message
  - the ciphertext appears to be **truly random**
  - the uniform key-selection distribution **is imposed also onto** produced ciphertexts
    - e.g., c = k XOR m (for uniform k and any distribution over m)

When security is computational, randomness is **relaxed** to "pseudorandomness"

- the ciphertext appears to be "**pseudorandom**"
  - it **cannot be efficiently distinguished** from truly random

Symmetric encryption as "OPT with pseudorandomness"

Stream cipher

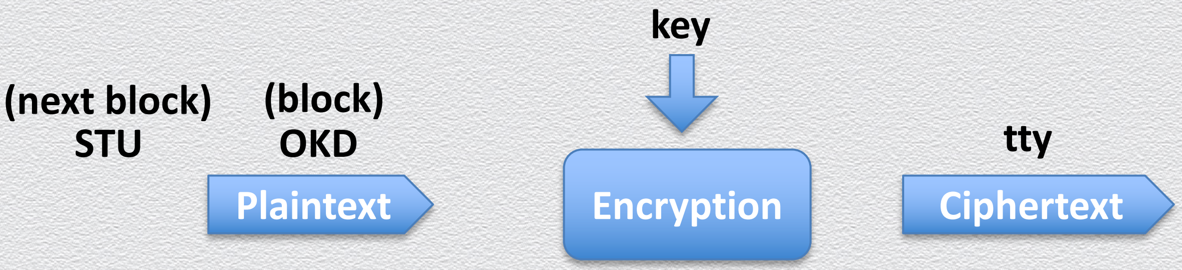
Uses a **short** key to encrypt **long** symbol **streams** into a **pseudorandom** ciphertext

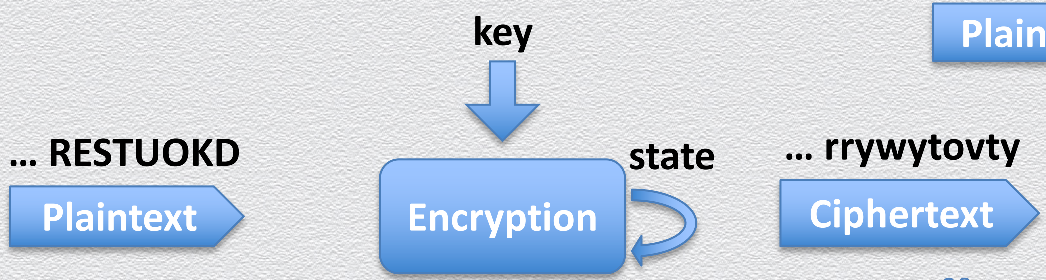
- based on abstract crypto primitive of **pseudorandom generator (PRG)**

Block cipher

Uses a **short** key to encrypt **blocks** of symbols into **pseudorandom** ciphertext blocks

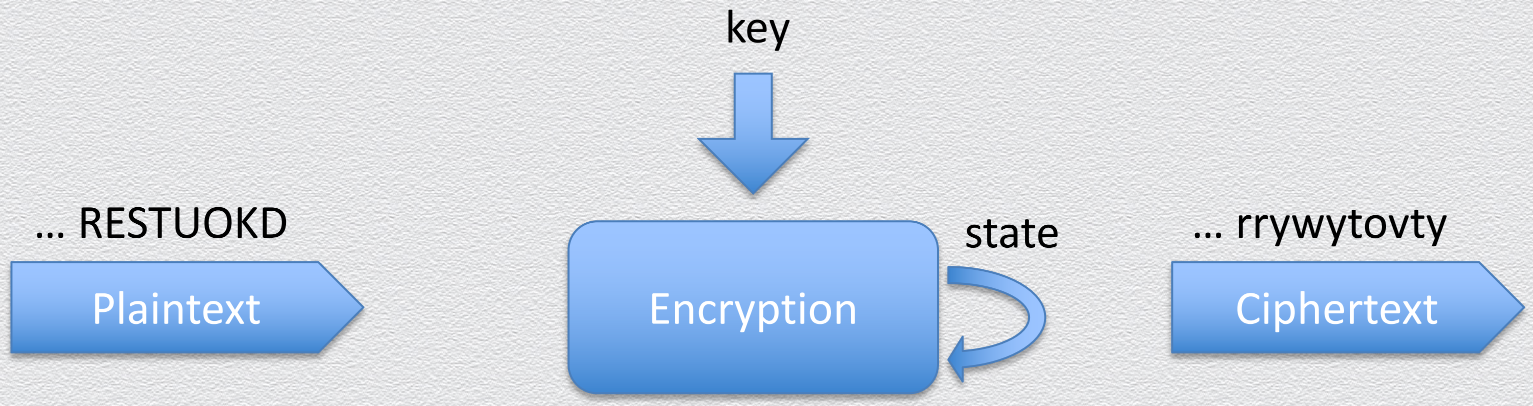
- based on abstract crypto primitive of **pseudorandom function (PRF)**



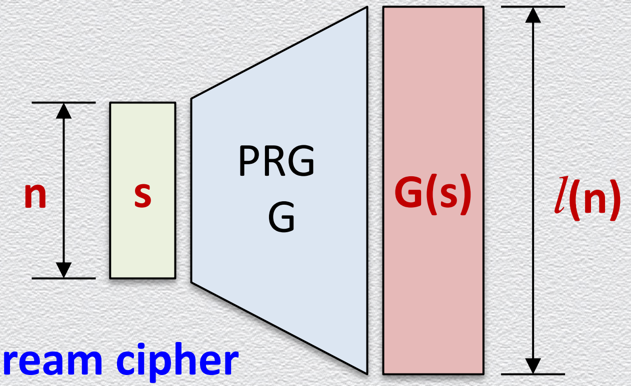


3.4.1 Pseudorandom generators

Stream ciphers



Pseudorandom generators (PRGs)



Deterministic algorithm G that
on input a **<u>seed</u>** $s \in \{0,1\}^t$, outputs $G(s) \in \{0,1\}^{l(t)}$

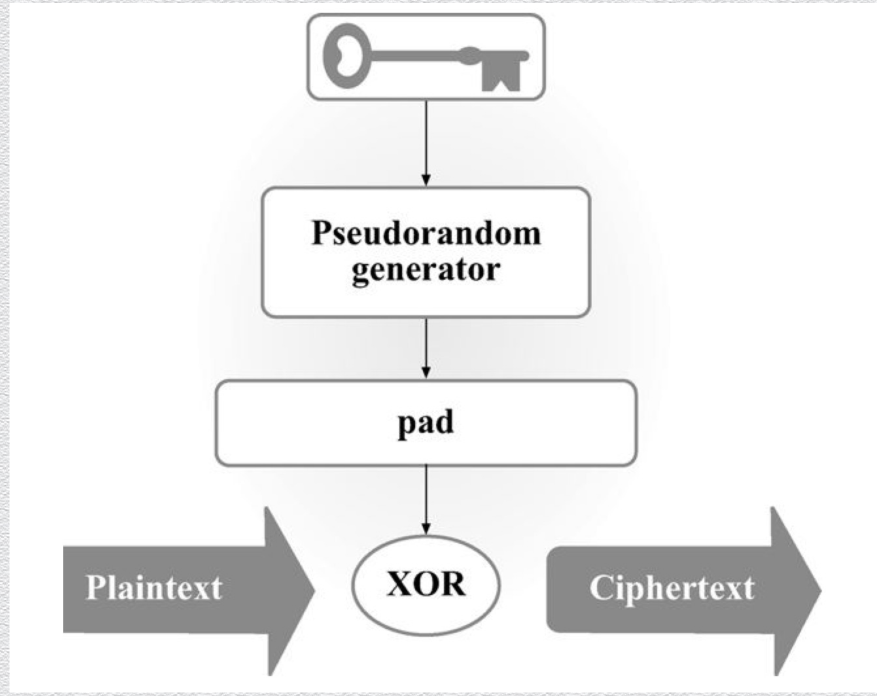
a.k.a. **stream cipher**

G is a PRG if:

- **expansion**
  - for polynomial *l*, it holds that for any n, *l*(n) > n
  - models the process of **<u>extracting</u>** randomness from a short random string
- **pseudorandomness**
  - **<u>no efficient statistical test can tell apart</u>** G(s) from a **<u>truly random</u>** string

Generic PRG-based symmetric encryption

- **Fixed-length** message encryption



encryption scheme is plain-secure
as long as the underlying PRG is secure

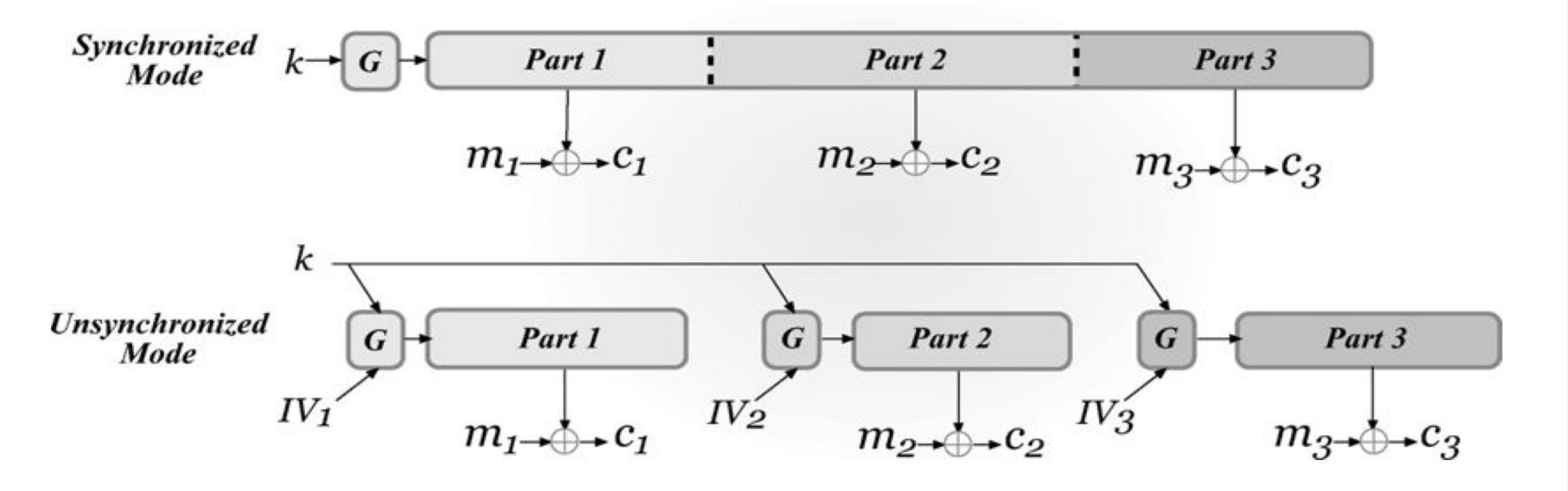
Generic PRG-based symmetric encryption (cont.)

- **Bounded- or arbitrary-length** message encryption
  - specified by a mode of operation for using an underlying stateful stream cipher, repeatedly, to encrypt/decrypt a stream of symbols

Stream ciphers: Modes of operations

- **Bounded- or arbitrary-length** message encryption

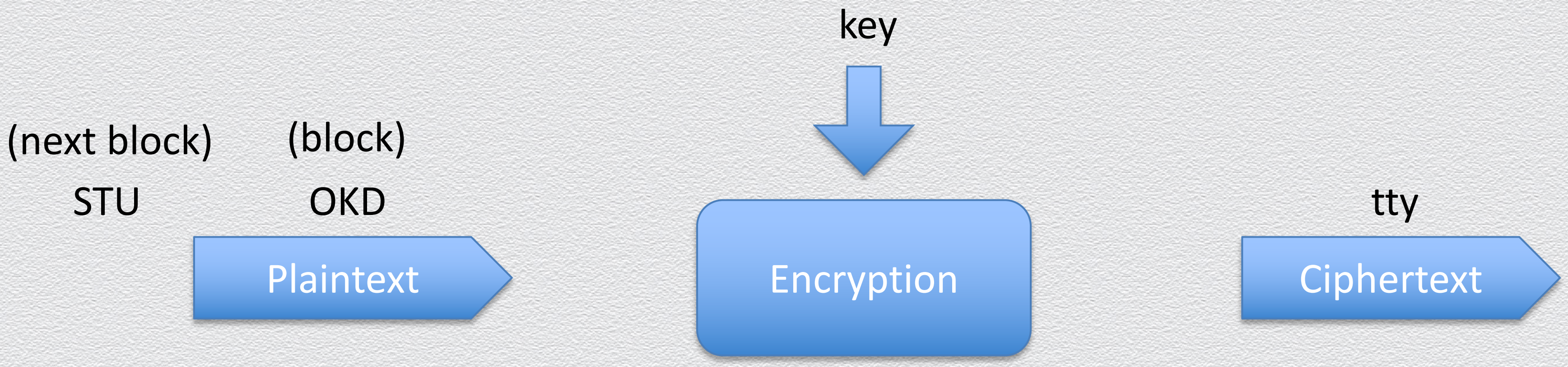
on-the-fly computation of new pseudorandom bits, no IV needed, plain-secure



random IV used for every new message is sent along with ciphertext, advanced-secure

3.4.2 Pseudorandom functions

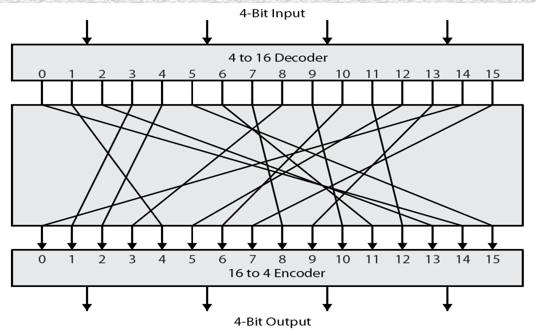
Block ciphers



Realizing ideal block ciphers in practice

We want a **random** mapping of n-bit inputs to n-bit outputs

- there are ~2^(n2^n) possible such mappings
- none of the above can be implemented in practice



Instead, we use a keyed function $F_k : \{0,1\}^n \rightarrow \{0,1\}^n$

- indexed by a t-bit key k
- there are only 2^t such keyed functions
- a random key selects a "random-enough" mapping or a **pseudorandom function**

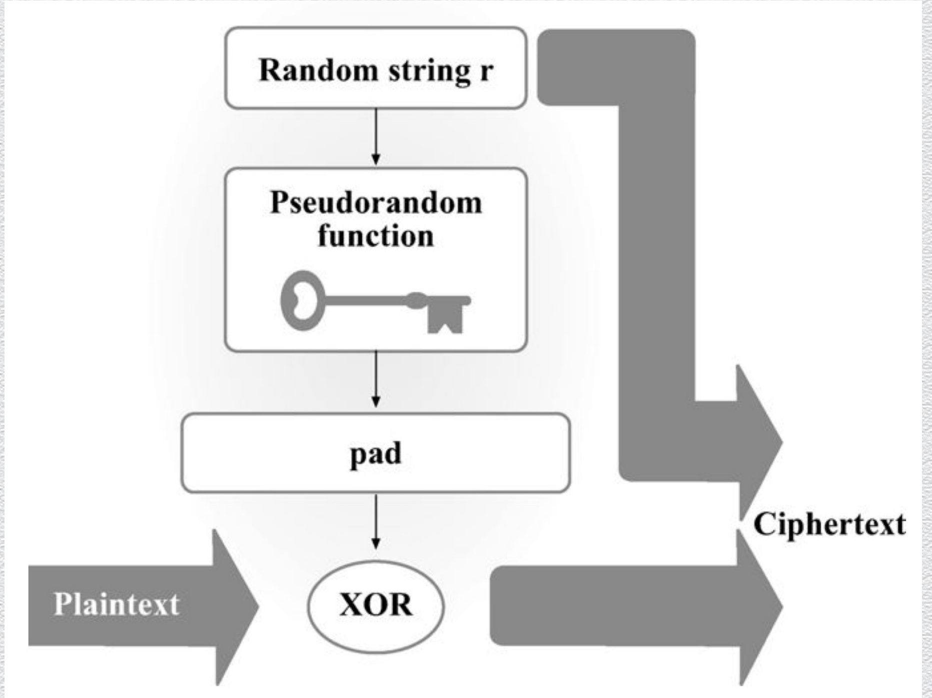
x

F_k

$y = F_k(x)$

Generic PRF-based symmetric encryption

- **Fixed-length** message encryption



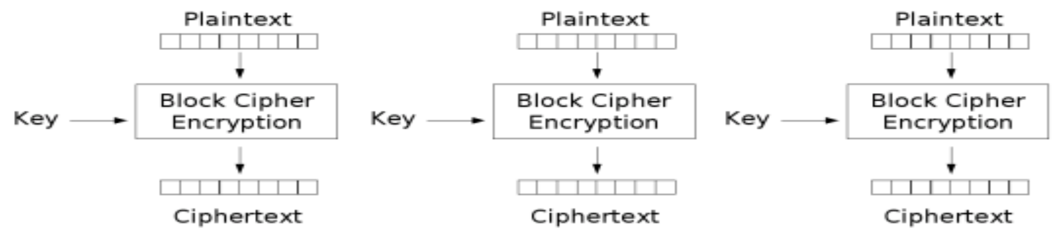
encryption scheme is advanced-secure as long as the underlying PRF is secure

Generic PRF-based symmetric encryption (cont.)

- **Arbitrary-length** message encryption
  - specified by a mode of operation for using an underlying stateless block cipher, repeatedly, to encrypt/decrypt a sequence of message blocks

Electronic Code Book (ECB)

- The simplest mode of operation
  - block P[i] encrypted into ciphertext block $C[i] = Enc_k(P[i])$
  - block C[i] decrypted into plaintext block $M[i] = Dec_k(C[i])$



Electronic Codebook (ECB) mode encryption

Strengths & weaknesses of ECB

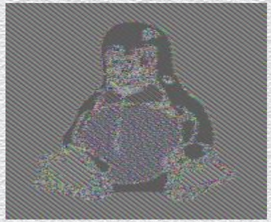
Strengths

- very simple
- allows for parallel encryptions of the blocks of a plaintext
- can tolerate the loss or damage of a block

Weaknesses

- poor security
- produces the same ciphertext on the same plaintext (under the same key)
- documents and images are not suitable for ECB encryption, since patterns in the plaintext are repeated in the ciphertext
- e.g.,

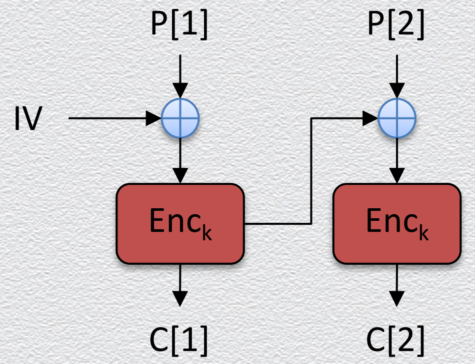
ECB

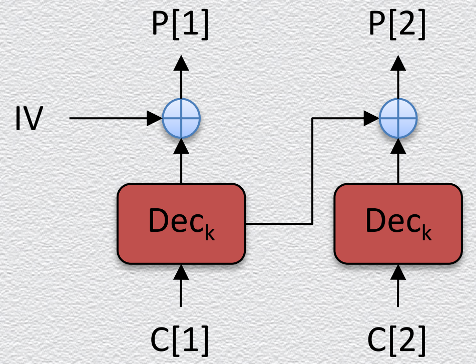


Cipher Block Chaining (CBC) [or chaining]

Alternatively, the previous-block ciphertext is “mixed” with the current-block plaintext

- e.g., using XOR
  - each block is encrypted as $C[i] = Enc_k\,(C[i-1] \oplus P[i])$,
  - each ciphertext is decrypted as $P[i] = C[i-1] \oplus Dec_k\,(C[i])$
  - here, $C[0] = IV$ is a uniformly random initialization vector that is transmitted separately





CBC