UNIVERSITÄT PASSAU

Fakultät für Informatik und Mathematik

# 6090: Security of Computer and Embedded Systems

## *Week 6:* Cryptographic Foundations Part 1

***Elif Bilge Kavun***

elif.kavun@uni-passau.de

# This Week's Outline

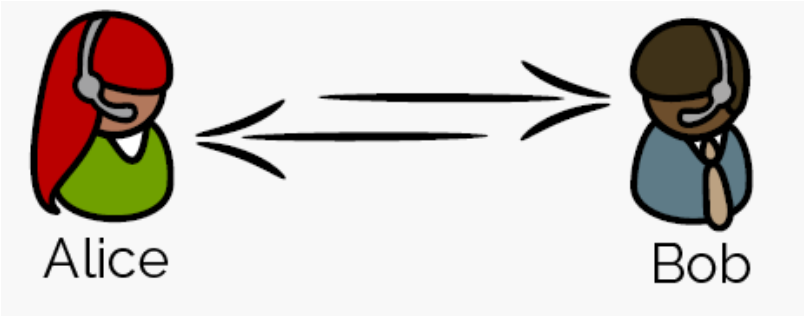
- Introduction to Cryptography, Motivation
- Mathematical Foundations
- Symmetric Encryption

  Other terms:

  Single-key, One-key, Private-key, Conventional Encryption
  - Early examples
  - Modern algorithms

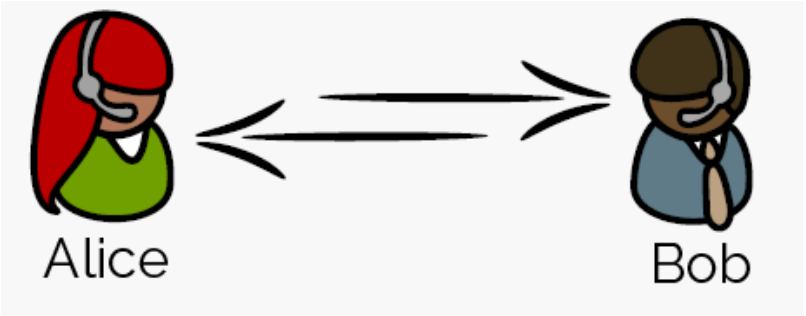
# Motivation

- How can we turn an untrustworthy channel into a trustworthy one?



# Motivation

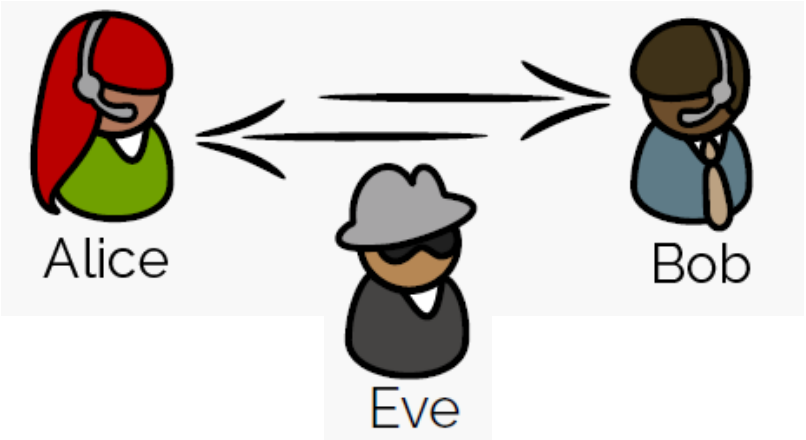
- How can we turn an untrustworthy channel into a trustworthy one?




- Recall our fundamental information security goals
  - **Confidentiality:** Transmitted information remains secret
  - **Integrity:** Information not corrupted (or alterations detected)
  - **Authentication:** Principals know with whom they are speaking

# Motivation

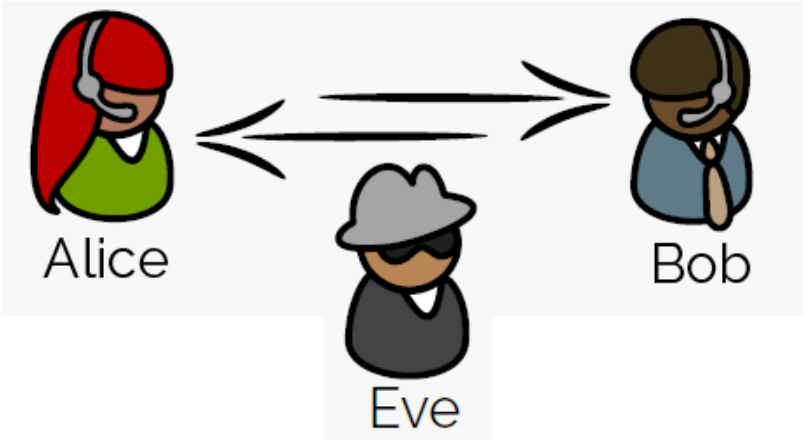
- How can we turn an untrustworthy channel into a trustworthy one?




- Recall our fundamental information security goals
  - **Confidentiality:** Transmitted information remains secret
  - **Integrity:** Information not corrupted (or alterations detected)
  - **Authentication:** Principals know with whom they are speaking

# Motivation

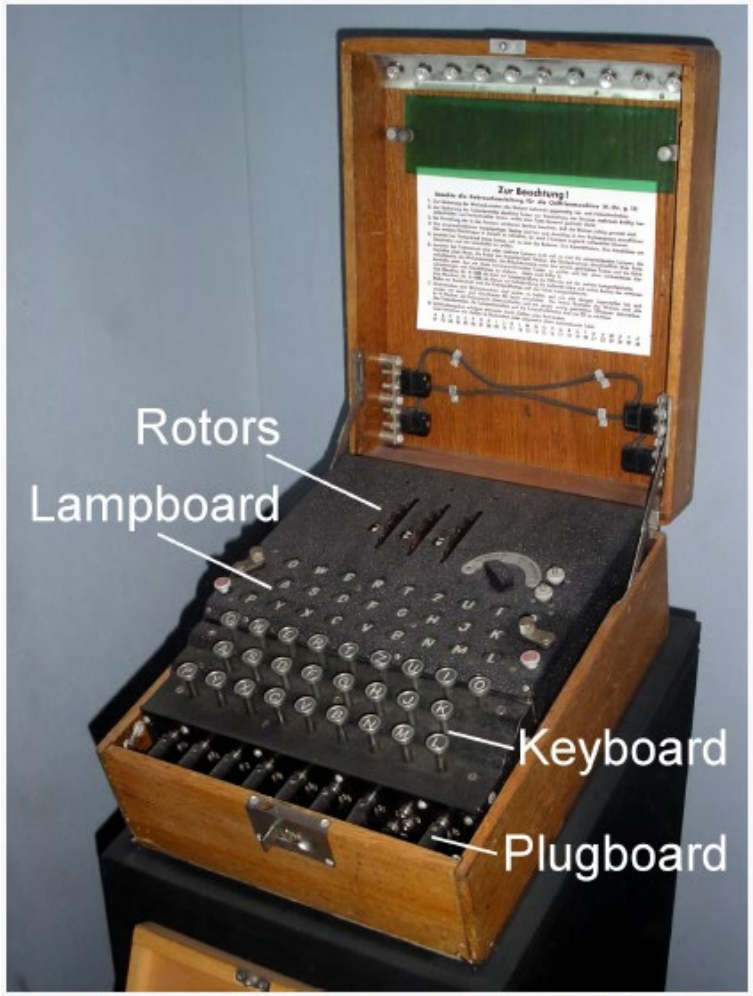
- How can we turn an untrustworthy channel into a trustworthy one?




- Recall our fundamental information security goals
  - **Confidentiality:** Transmitted information remains secret
  - **Integrity:** Information not corrupted (or alterations detected)
  - **Authentication:** Principals know with whom they are speaking
- *Cryptography is an enabling technology*

# Clarifying Notation

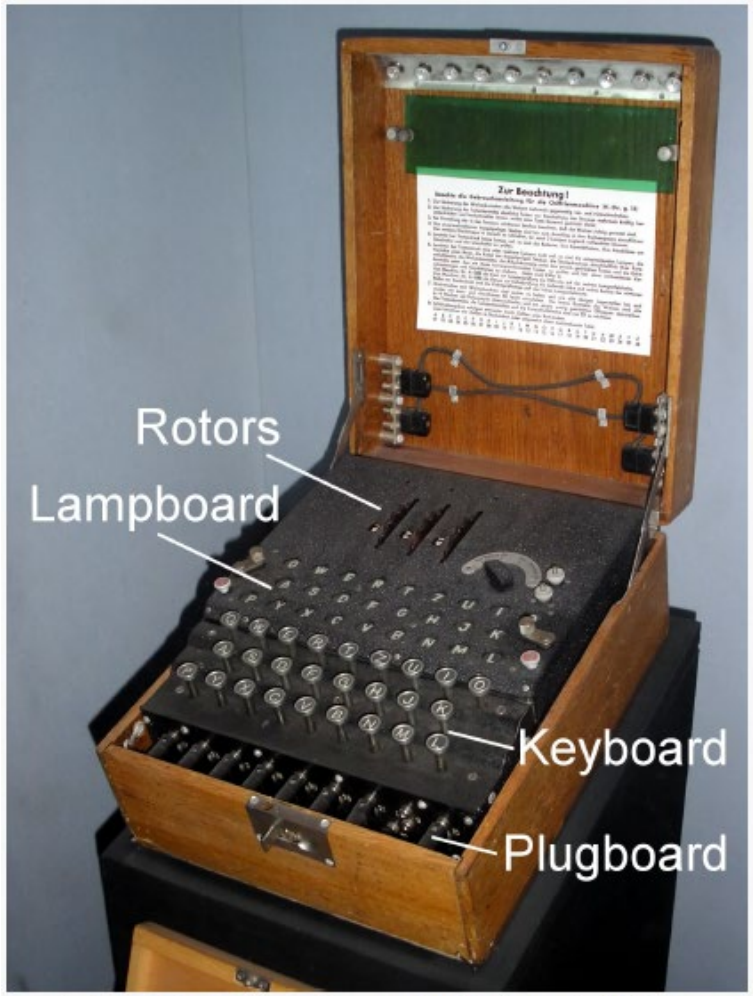
- **Steganography**
- **Cryptography**
- **Cryptanalysis**




https://en.wikipedia.org/w/index.php?title=Enigma_machine&oldid=764760662

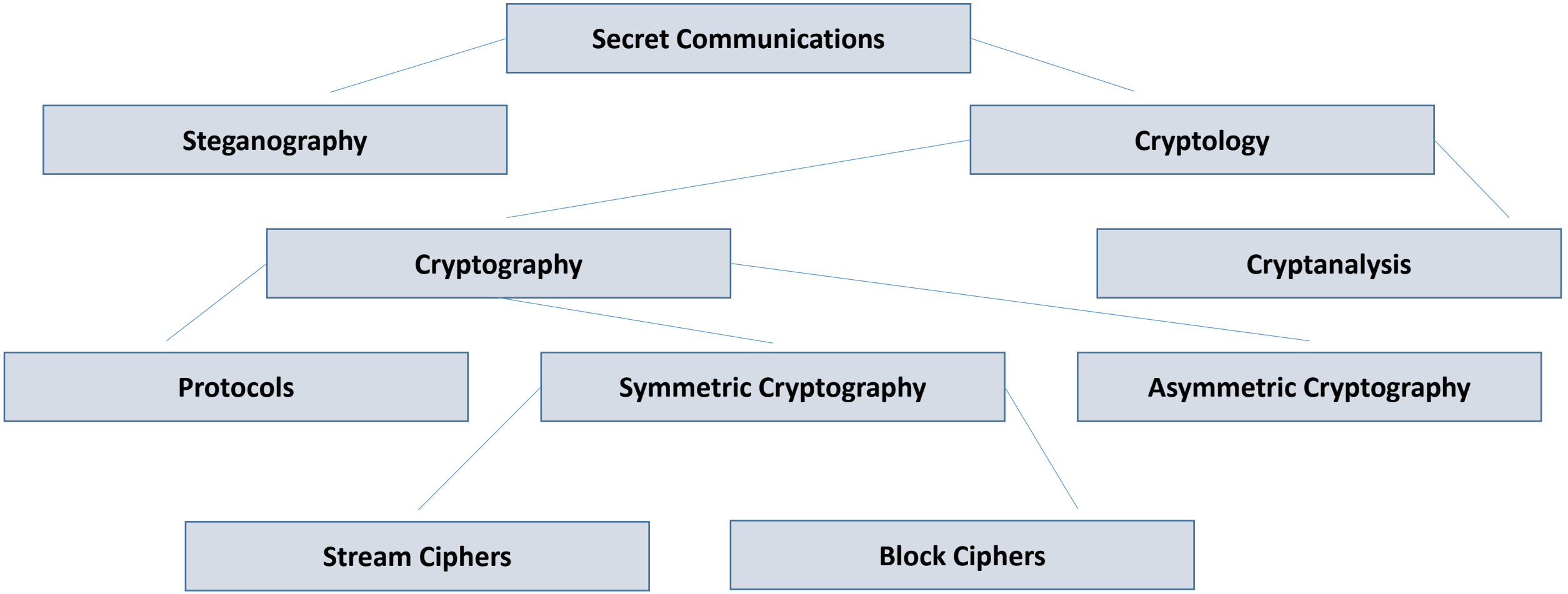
# Clarifying Notation

- **Steganography:** The science of hiding messages in other messages or images
- **Cryptography:** The science of secret writing
- **Cryptanalysis:** The science of analyzing a cryptographic system to break/circumvent its protection

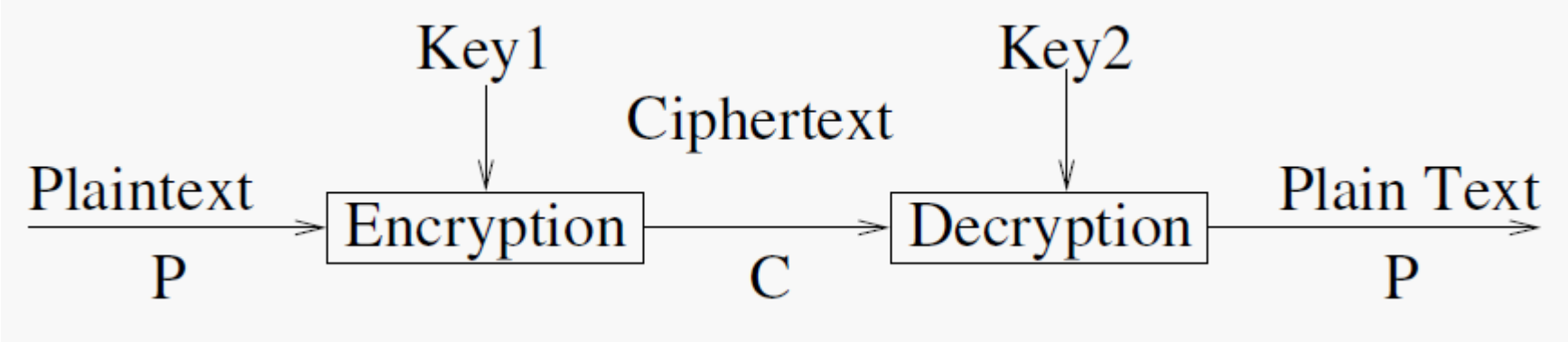



https://en.wikipedia.org/w/index.php?title=Enigma_machine&oldid=764760662

# Terminology

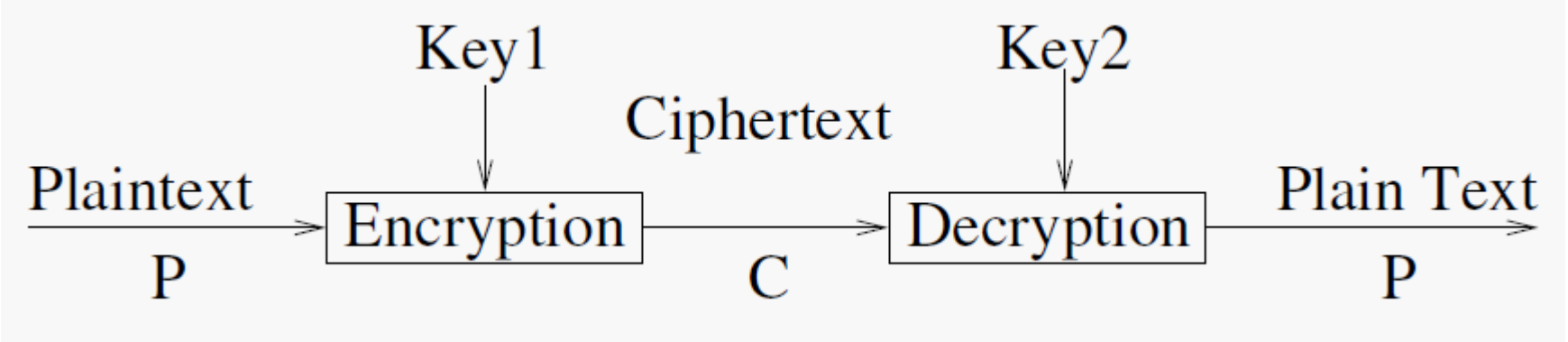


# A General Cryptographic Schema




where $E_{Key1}(P)=C$ and $D_{Key2}(C)=P$, hence: $D_{Key2}(E_{Key1}(P))=P$

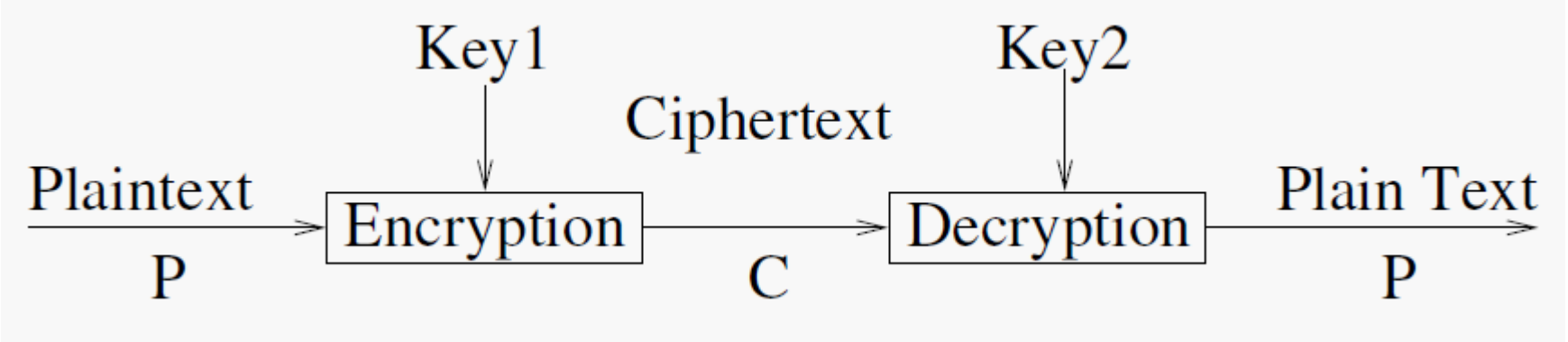
# A General Cryptographic Schema




where $E_{Key1}(P)=C$ and $D_{Key2}(C)=P$, hence: $D_{Key2}(E_{Key1}(P))=P$

- Symmetric encryption
  - **Key1** = **Key2** (or can be easily derived from each other)

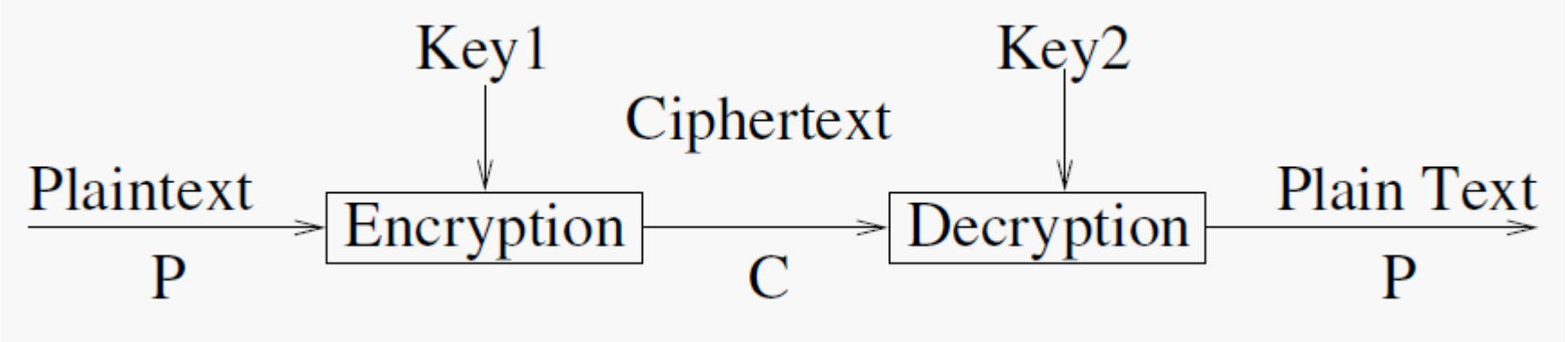
# A General Cryptographic Schema




where $E_{Key1}(P)=C$ and $D_{Key2}(C)=P$, hence: $D_{Key2}(E_{Key1}(P))=P$

- Symmetric encryption
  - **Key1** = **Key2** (or can be easily derived from each other)
- Asymmetric encryption (public key)
  - **Key1** ≠ **Key2** (cannot be easily derived from each other)
  - The *public* key (**Key1**) can be published without compromising the *private* key (**Key2**)

# A General Cryptographic Schema




where $E_{Key1}(P)=C$ and $D_{Key2}(C)=P$, hence: $D_{Key2}(E_{Key1}(P))=P$

- Symmetric encryption
  - **Key1** = **Key2** (or can be easily derived from each other)
- Asymmetric encryption (public key)
  - **Key1** ≠ **Key2** (cannot be easily derived from each other)
  - The *public* key (**Key1**) can be published without compromising the *private* key (**Key2**)
- Encryption and decryption should be easy, if keys are known
- Security depends on secrecy of the key, not the encryption/decryption algorithms

# Encryption & Decryption

- We introduce
  - A finite set $A$, called the *alphabet*
  - The *message space* $M \subseteq A^*$ and $\mathrm{M} \in M$ is a *plaintext (message)*
  - The *ciphertext space* $C$, whose alphabet may differ from $M$
  - $K$ denoting the *key space* of *keys*
- Moreover
  - Each $e \in K$ determines a bijective function from $M$ to $C$, denoted by $E_e$
    - $E_e$ is the *encryption function*
  - For each $d \in K$, $D_d$ denotes a bijection from $C$ to $M$
    - $D_d$ is the *decryption function*
- Applying $E_e$ (or $D_d$) is called *encryption* (or *decryption*)

# Encryption (and Decryption) Schemes

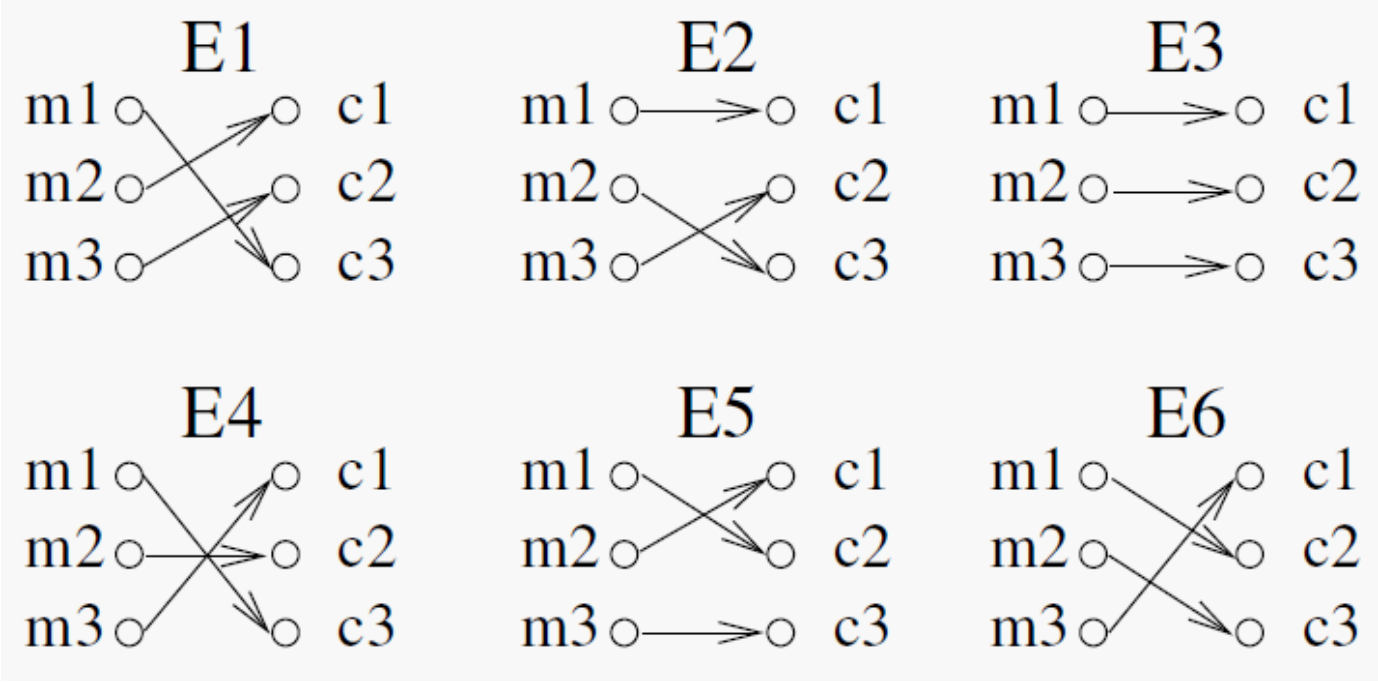
- An *encryption scheme* (or *cipher*) consists of a set $\{E_e | e \in K\}$ and a corresponding set $\{D_d | d \in K\}$ such that for each $e \in K$ there is a unique $d \in K$ with $D_d = E_e^{-1}$; i.e.,

$$D_d(E_e(m)) = m, \text{ for all } m \in M$$

  - The keys $e$ and $d$ form a *key pair*, sometimes denoted by $(e, d)$
    - They can be identical (i.e., the symmetric key) of a symmetric encryption scheme
  - Constructing an encryption scheme requires fixing a message space $M$, a ciphertext space $C$, and a key space $K$, as well as encryption transformations $\{E_e | e \in K\}$ and corresponding decryption transformations $\{D_d | d \in K\}$

# An Example

- Let $M = \{m_1, m_2, m_3\}$ and $C = \{c_1, c_2, c_3\}$
- There are $3! = 6$ bijections from $M$ to $C$
- The key space $K = \{E_1, E_2, E_3, E_4, E_5, E_6\}$ specifies transformations
- Assume Alice and Bob agree on $E_1$
- To encrypt $m_1$, Alice computes $E_1(m_1) = c_3$
- Bob decrypts $c_3$ by reversing the arrows on the diagram for $E_1$ and observing that $c_3$ points to $m_1$



# Codes

- String of symbols standing for a complete message
- One of the simplest and earliest forms of cryptography
- Translation given by a "*code-book*"
- Still used today

  - *“This year’s trial of the embassy bombings revealed that Bin Laden associates began to use encryption before 1998. Sometimes members of the Al-Qaida confederation have alternatively resorted to simple code words. For instance, "working" is said to mean Jihad, "tools" meant weapons, "potatoes" meant grenades and "the director" was an alias for Bin Laden.”*

    Lisa Krieger; Mercury News; Oct 1, 2001

# Mono-Alphabetic Substitution Ciphers

- Simplest kind of cipher (idea over 2000 years old)
- Let $K$ be the set of all permutations on the alphabet $A$
- For each $e \in K$, we define an encryption transformation $E_e$ on strings $m = m_1 m_2 \cdots m_n \in M$ as

$$E_e(m) = e(m_1)e(m_2) \cdots e(m_n) = c_1 c_2 \cdots c_n = c$$

- To decrypt $c$, compute the inverse permutation $d = e^{-1}$ and

$$D_d(c) = d(c_1)d(c_2) \cdots d(c_n) = m$$

- $E_e$ is a *simple substitution cipher* or a *mono-alphabetic substitution cipher*

# Examples of Substitution Cipher

- *D*(KHOOR ZRUOG) =
  - *Caesar cipher*
    - Each plaintext character is replaced by the character three to the right modulo 26 (e.g., *E*(A)=*D*)

# Examples of Substitution Cipher

- *D*(KHOOR ZRUOG) = HELLO WORLD
  - *Caesar cipher*
    - Each plaintext character is replaced by the character three to the right modulo 26 (e.g., *E*(A)=*D*)

# Examples of Substitution Cipher

- *D*(`KHOOR ZRUOG`) = `HELLO WORLD`
  - *Caesar cipher*
    - Each plaintext character is replaced by the character three to the right modulo 26 (e.g., *E*(A)=*D*)
- *D*(`Zl anzr vf Nqnz`) =
  - *ROT13* (also a *Caesar cipher*)
    - Shift each letter by 13 places

# Examples of Substitution Cipher

- *D*(`KHOOR ZRUOG`) = `HELLO WORLD`
  - *Caesar cipher*
    - Each plaintext character is replaced by the character three to the right modulo 26 (e.g., *E*(A)=*D*)
- *D*(`Zl anzr vf Nqnz`) = `My name is Adam`
  - *ROT13* (also a *Caesar cipher*)
    - Shift each letter by 13 places

# Examples of Substitution Cipher

- *D*(`KHOOR ZRUOG`) = `HELLO WORLD`
  - *Caesar cipher*
    - Each plaintext character is replaced by the character three to the right modulo 26 (e.g., *E*(A)=*D*)
- *D*(`Zl anzr vf Nqnz`) = `My name is Adam`
  - *ROT13* (also a *Caesar cipher*)
    - Shift each letter by 13 places
- *D*(2-25-5 2-25-5) =
  - *Alphanumeric*
    - Substitute numbers for letters

# Examples of Substitution Cipher

- *D*(`KHOOR ZRUOG`) = `HELLO WORLD`
  - *Caesar cipher*
    - Each plaintext character is replaced by the character three to the right modulo 26 (e.g., *E*(A)=*D*)
- *D*(`Zl anzr vf Nqnz`) = `My name is Adam`
  - *ROT13* (also a *Caesar cipher*)
    - Shift each letter by 13 places
- *D*(2-25-5 2-25-5) = `BYE BYE`
  - *Alphanumeric*
    - Substitute numbers for letters

# Examples of Substitution Cipher

- *D*(`KHOOR ZRUOG`) = `HELLO WORLD`
  - *Caesar cipher*
    - Each plaintext character is replaced by the character three to the right modulo 26 (e.g., *E*(A)=*D*)
- *D*(`Zl anzr vf Nqnz`) = `My name is Adam`
  - *ROT13* (also a *Caesar cipher*)
    - Shift each letter by 13 places
- *D*(2-25-5 2-25-5) = `BYE BYE`
  - *Alphanumeric*
    - Substitute numbers for letters
- How hard are these to break?

# (In)security of Substitution Ciphers

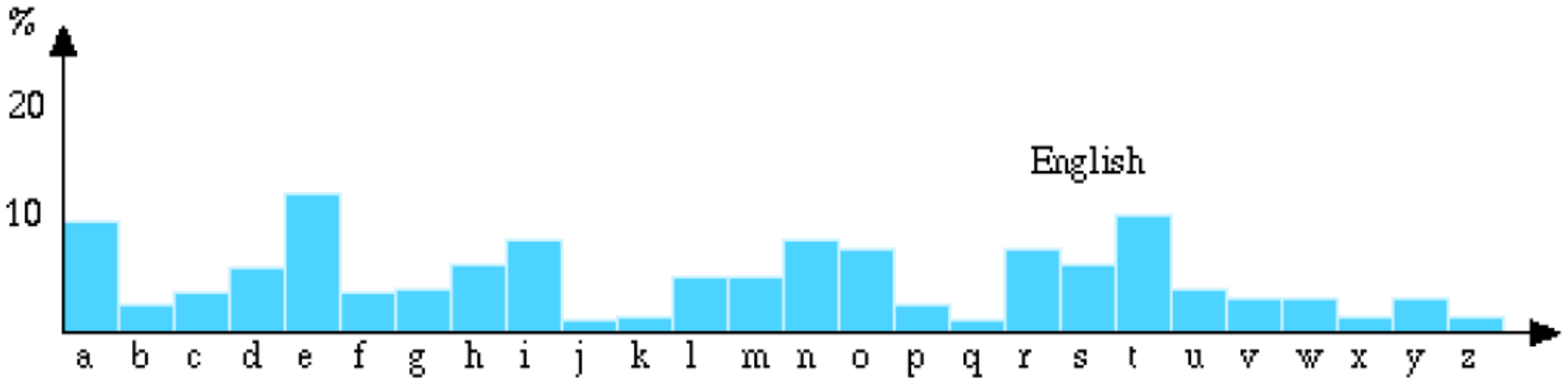
- Key spaces are typically huge
  - 26 letters → 26! possible keys

# (In)security of Substitution Ciphers

- Key spaces are typically huge
  - 26 letters → 26! possible keys
- Trivial to break using frequency analysis (letters, digraphs, etc.)

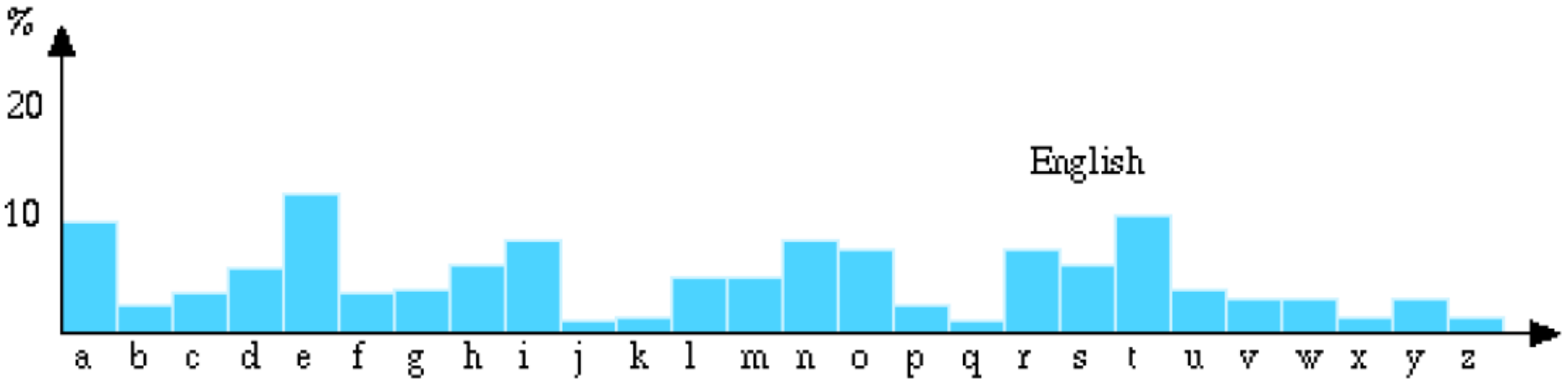
# (In)security of Substitution Ciphers

- Key spaces are typically huge
  - 26 letters → 26! possible keys
- Trivial to break using frequency analysis (letters, digraphs, etc.)
- Frequencies for English are based on data-mining books/articles



# (In)security of Substitution Ciphers

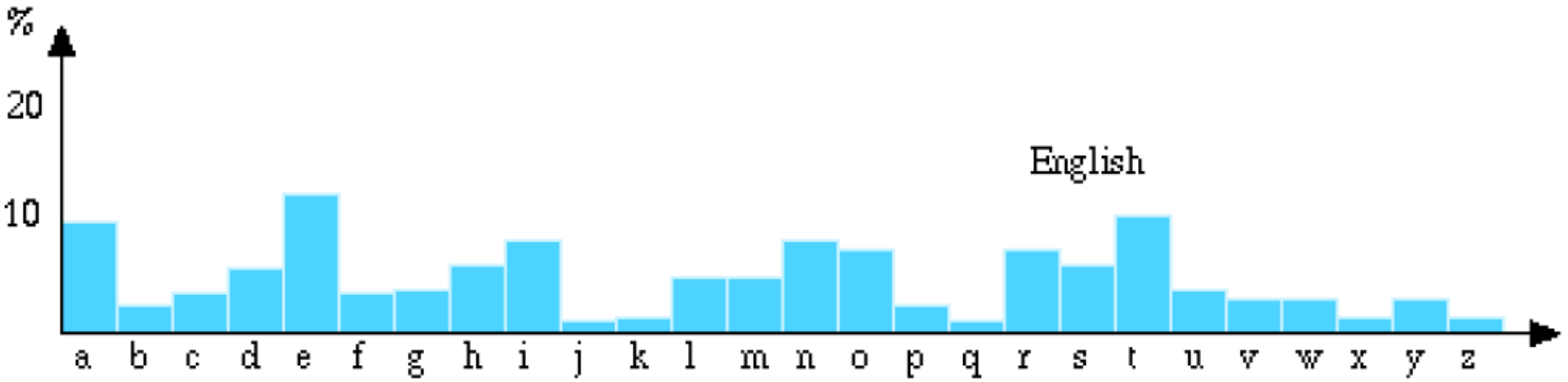
- Key spaces are typically huge
  - 26 letters → 26! possible keys
- Trivial to break using frequency analysis (letters, digraphs, etc.)
- Frequencies for English are based on data-mining books/articles




- Easy to apply, except for short, atypical texts

# (In)security of Substitution Ciphers

- Key spaces are typically huge
  - 26 letters → 26! possible keys
- Trivial to break using frequency analysis (letters, digraphs, etc.)
- Frequencies for English are based on data-mining books/articles




- Easy to apply, except for short, atypical texts
- More sophistication required to mask statistical regularities

# Polyalphabetic Substitution Ciphers

- Idea
  - Conceal distribution using family of mappings

- A *polyalphabetic substitution cipher* is a block cipher with block length $t$ over alphabet $A$ where:
  - The key space $K$ consists of all ordered sets of $t$ permutations over $A$, $(p_1 p_2 \cdots p_t)$
  - Encryption of $m = m_1 \cdots m_t$ under key $e = (p_1, \cdots, p_t)$ is $E_e(m) = p_1(m_1) \cdots p_t(m_t)$
  - Decryption key for $e$ is $d = (p_1^{-1}, \cdots, p_t^{-1})$

# Polyalphabetic Substitution Ciphers

- Vigenère Cipher
  - Key given by sequence of numbers $e = e_1, \cdots, e_t$, where
    $$p_i(a) = (a + e_i) \bmod n$$
    defining a permutation on an alphabet of size $n$

  - Example: English ($n = 26$), with $k = 3, 7, 10$

    $m =$ **THI SCI PHE RIS CER TAI NLY NOT SEC URE**

# Polyalphabetic Substitution Ciphers

- Vigenère Cipher
  - Key given by sequence of numbers $e = e_1, \cdots, e_t$, where
    $$p_i(a) = (a + e_i) \bmod n$$
    defining a permutation on an alphabet of size $n$

  - Example: English ($n = 26$), with $k = 3, 7, 10$
    $$m = \texttt{THI SCI PHE RIS CER TAI NLY NOT SEC URE}$$
    then
    $$E_e(m) = \texttt{WOS VJS SOO UPC FLB WHS QSI QVD VLM XYO}$$

# One-time Pads (Vernam Cipher)

- A *one-time pad* is a cipher defined over $\{0,1\}$
- A message $m_1 \cdots m_n$ is encrypted by a binary key string $k_1 \cdots k_n$

$$E_{k1\cdots kn}(m_1 \cdots m_n) = (m_1 \oplus k_1) \cdots (m_n \oplus k_n)$$
$$D_{k1\cdots kn}(c_1 \cdots c_n) = (c_1 \oplus k_1) \cdots (c_n \oplus k_n)$$

- Example:

$$c = m \oplus k = (010111)_2 \oplus (110010)_2 = (100101)_2$$

- Since every key sequence is equally likely, so is every plaintext!
- Unconditional (information theoretic) security, if key is not reused!
- Moscow–Washington communication previously secured this way

# One-time Pads (Vernam Cipher)

- Ciphertext is impossible to break if four conditions are met:
  - Key must be true random (independent of the plaintext)
  - Key must be at least as long as the plaintext
  - Key must never be reused in whole or in part
  - Key must be kept completely secret by the communicating parties

# One-time Pads (Vernam Cipher)

- Ciphertext is impossible to break if four conditions are met:
  - Key must be true random (independent of the plaintext)
  - Key must be at least as long as the plaintext
  - Key must never be reused in whole or in part
  - Key must be kept completely secret by the communicating parties
- **Problem?**

# One-time Pads (Vernam Cipher)

- Ciphertext is impossible to break if four conditions are met:
  - Key must be true random (independent of the plaintext)
  - Key must be at least as long as the plaintext
  - Key must never be reused in whole or in part
  - Key must be kept completely secret by the communicating parties
- **Problem?**
  - Securely exchanging and synchronizing long keys

# Transposition Cipher

- For block length $t$, let $K$ be the set of permutations on $\{1, \ldots, t\}$ . For each $e \in K$ and $m \in M$

$$E_e(m) = m_{e(1)} m_{e(2)} \cdots m_{e(t)}$$

- The set of all such transformations is called a *transposition cipher*
- To decrypt $c = c_1 c_2 \cdots c_t$ compute

$$D_d(c) = c_{d(1)} c_{d(2)} \cdots c_{d(t)}$$

  where $d$ is inverse permutation
- Letters are unchanged
  - Apply frequency analysis to reveal if ciphertext is a transposition
  - Decrypt by exploiting frequency analysis for dipthongs, tripthongs, words, etc.

# Transposition Cipher

- Example
  - *C* = Aduaenttlydhatoiekounletmtoihahvsekeeeleeyqonouv

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| A | n | d | i | n | t | h | e | e | n |
| d | t | h | e | l | o | v | e | y | o |
| u | t | a | k | e | i | s | e | q | u |
| a | l | t | o | t | h | e | l | o | v |
| e | y | o | u | m | a | k | e | | |

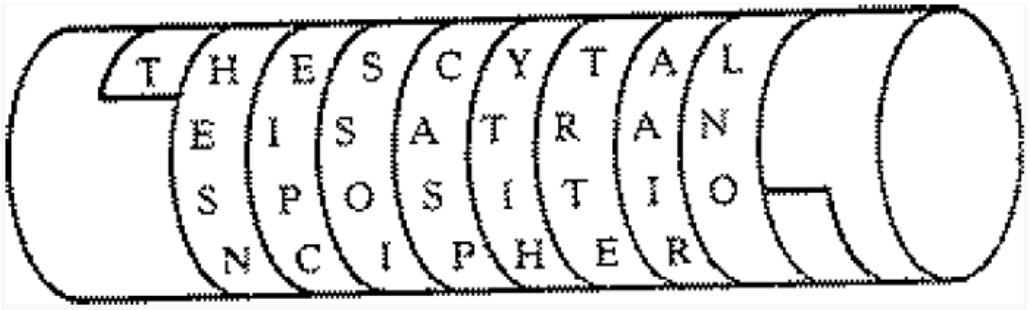
  - Table defines a permutation on 1, ..., 50

# Transposition Cipher

- Example
  - $C$ = Aduaenttlydhatoiekounletmtoihahvsekeeeleeyqonouv

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| A | n | d | i | n | t | h | e | e | n |
| d | t | h | e | l | o | v | e | y | o |
| u | t | a | k | e | i | s | e | q | u |
| a | l | t | o | t | h | e | l | o | v |
| e | y | o | u | m | a | k | e | | |

  - Table defines a permutation on 1, ..., 50




  - Idea goes back to Greek Scytale
    - Wrap belt spirally around baton and write plaintext lengthwise on it

# Composite Ciphers

- Ciphers based on either **substitutions** or **transpositions** are **insecure**

# Composite Ciphers

- Ciphers based on either **substitutions** or **transpositions** are **insecure**
- Ciphers can be combined

# Composite Ciphers

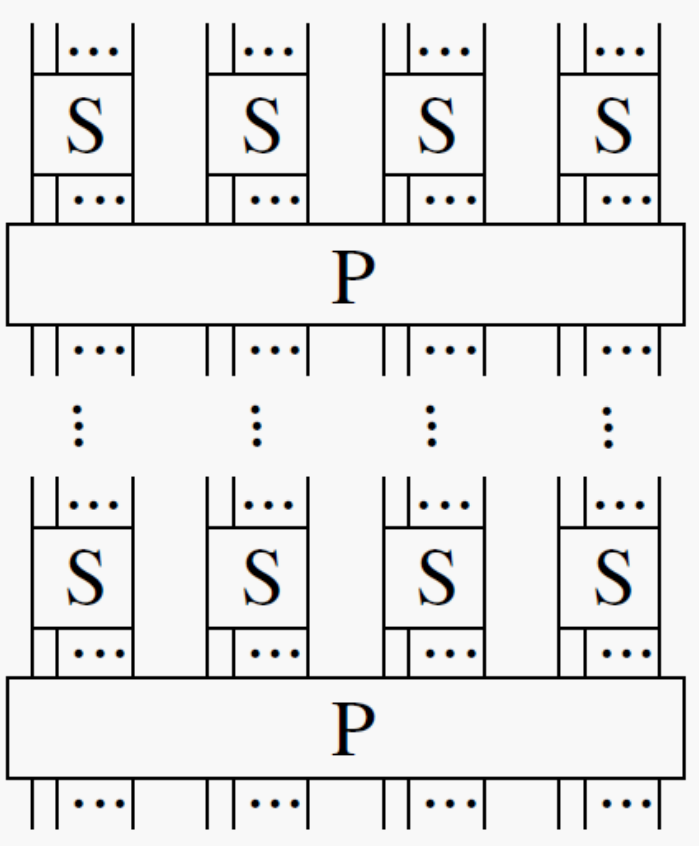
- Ciphers based on either **substitutions** or **transpositions** are **insecure**
- Ciphers can be combined
  - Two substitutions: Only one “more complex” substitution in practice
  - Two transpositions: Only one transposition in practice

# Composite Ciphers

- Ciphers based on either **substitutions** or **transpositions** are **insecure**
- Ciphers can be combined
  - Two substitutions: Only one “more complex” substitution in practice
  - Two transpositions: Only one transposition in practice
  - Substitution followed by a transposition makes a new harder cipher

# Composite Ciphers

- Ciphers based on either **substitutions** or **transpositions** are **insecure**
- Ciphers can be combined
  - Two substitutions: Only one “more complex” substitution in practice
  - Two transpositions: Only one transposition in practice
  - Substitution followed by a transposition makes a new harder cipher
- Product ciphers chain combinations of substitutions and transpositions
  - "S-Boxes" *confuse* input bits
  - "P-Boxes" *diffuse* bits across S-box inputs



# Composite Ciphers

- **Substitution:** Each binary bit of the ciphertext should depend on several parts of the key, obscuring the connections between the two
  - Nonlinear operation
    - Not easily invertible
  - Substitutes message bits according to a lookup table
  - Introduces confusion to the cipher
- **Permutation:** Each plaintext digit (bit) affects many ciphertext digits, or each ciphertext digit is affected by many plaintext digits
  - Linear operation
  - Diffuses substituted bits across S-Box inputs

# Composite Ciphers

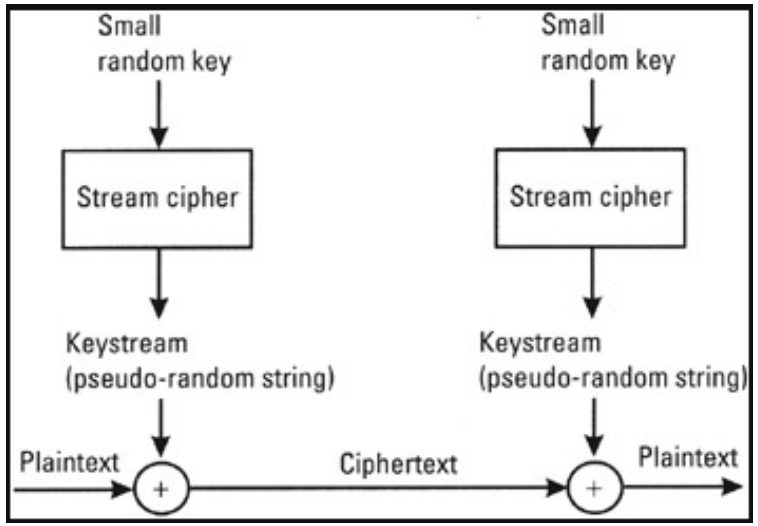
- **Substitution:** Each binary bit of the ciphertext should depend on several parts of the key, obscuring the connections between the two
  - Nonlinear operation
    - Not easily invertible
  - Substitutes message bits according to a lookup table
  - Introduces confusion to the cipher
- **Permutation:** Each plaintext digit (bit) affects many ciphertext digits, or each ciphertext digit is affected by many plaintext digits
  - Linear operation
  - Diffuses substituted bits across S-Box inputs
- Target is to have all bits substituted as soon as possible (in less number of rounds)
  - Otherwise, performance and cost issues in practice
- One bit change in input should have an impact on every output bit
  - So, it should be a completely different ciphertext

# Composite Ciphers

- Symmetric Cryptography
  - Stream Ciphers
  - Block Ciphers

# Stream Ciphers

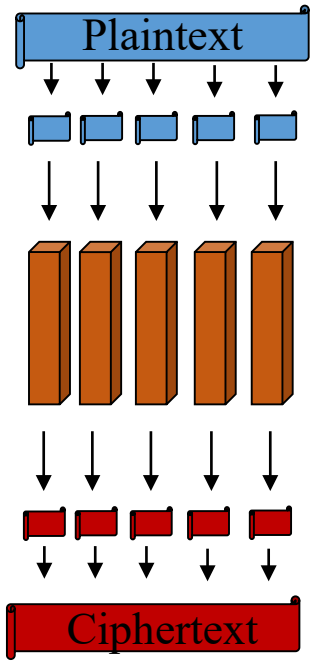
- A stream cipher is one where the block-length is 1
- Plaintext digits are combined with a pseudorandom cipher digit stream (keystream)
- Each plaintext digit is encrypted one at a time with the corresponding digit of the keystream in order to give a digit of the ciphertext stream
- Remember Vigenère cipher (substitution!)



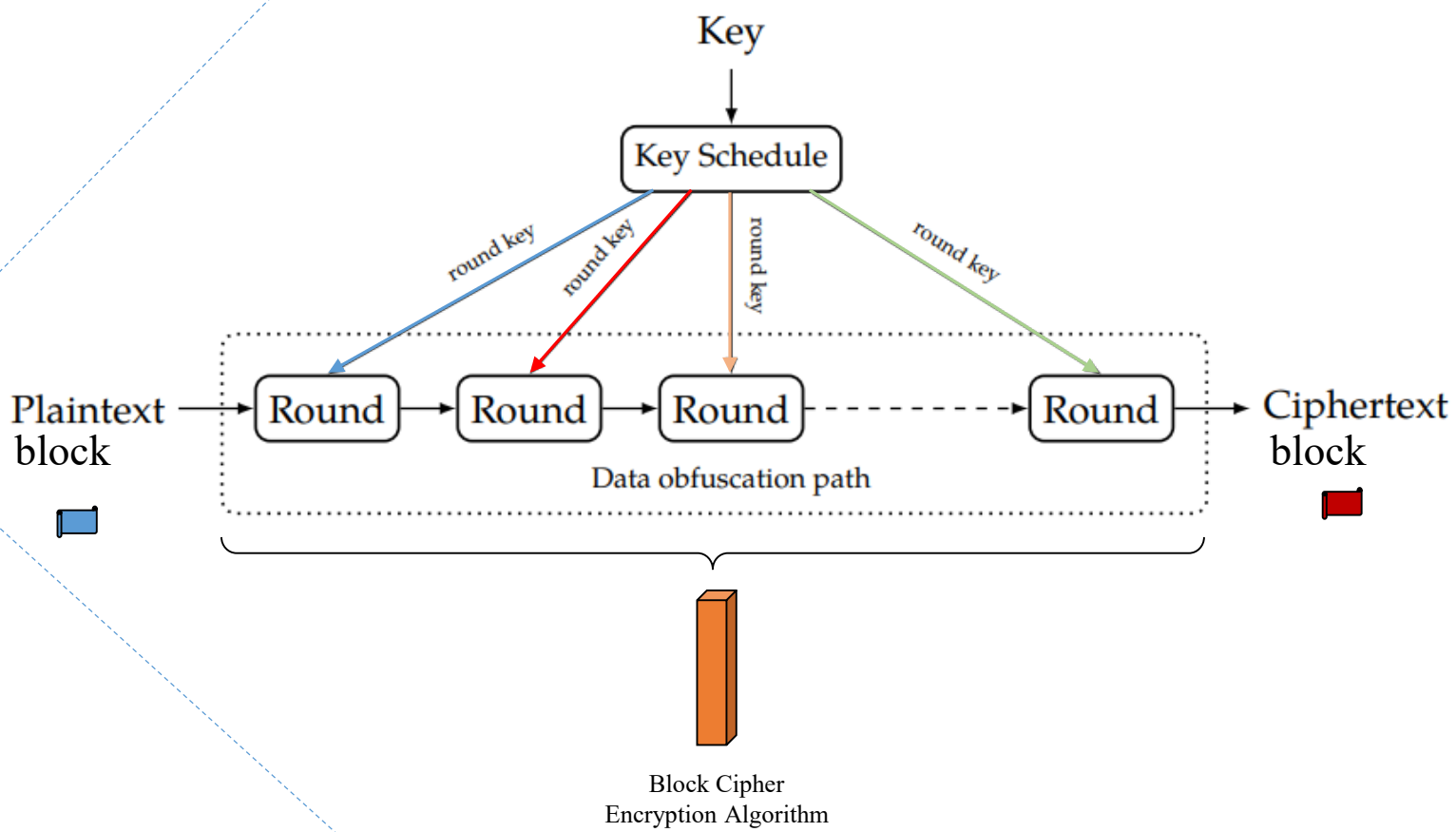
# Block Ciphers

- A block cipher is an encryption scheme that breaks up the plaintext message into strings (blocks) of a fixed length *t* and encrypts one block at a time
  - Take a number of bits and encrypt them as a single unit, padding the plaintext so that it is a multiple of the block size
- 64/128 bits are common block sizes
- Different design strategies (structures) exist

# Block Ciphers




One Block Encryption



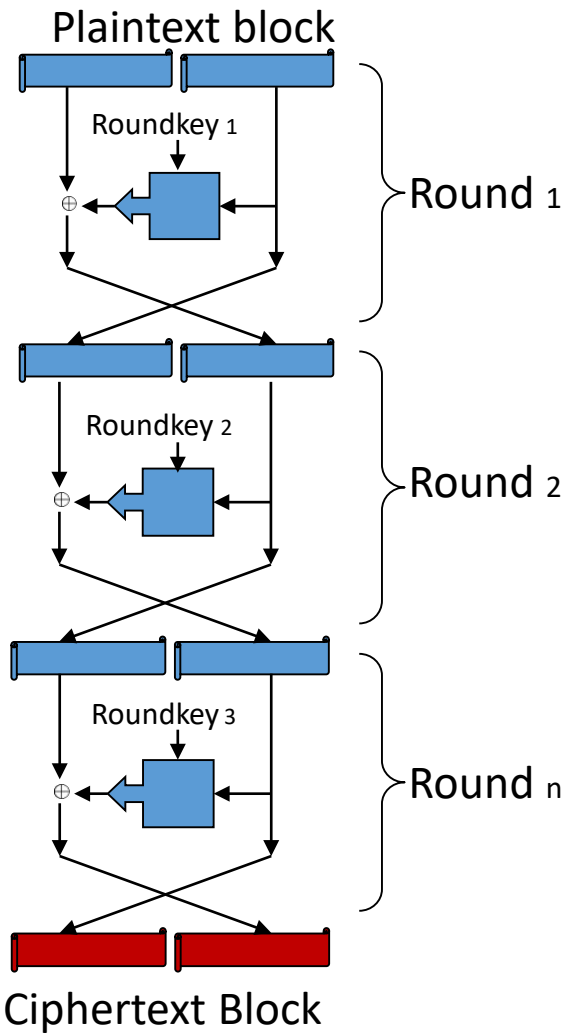
# Block Ciphers

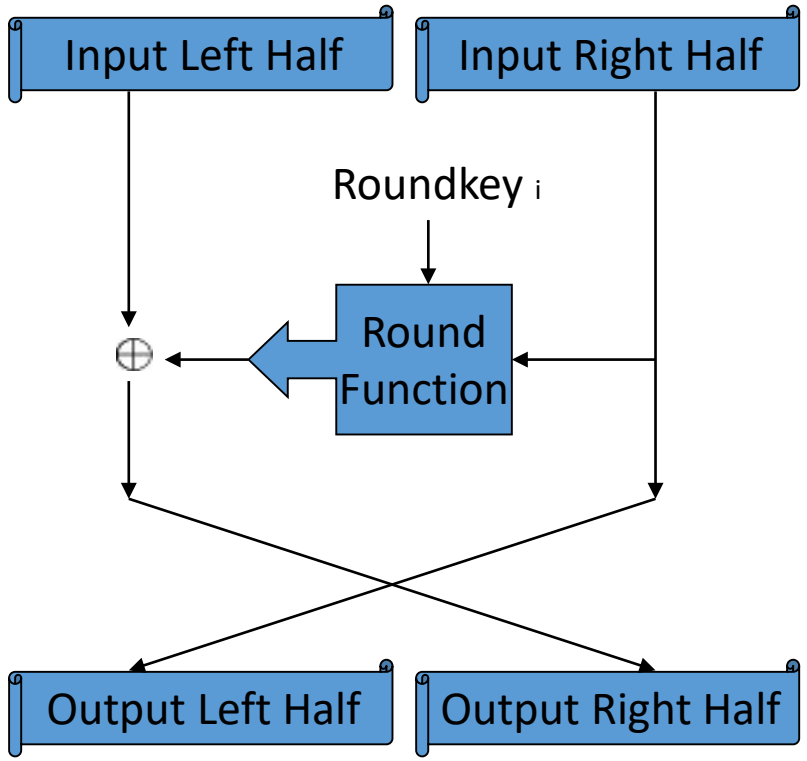
- Design structures for block ciphers
  - Feistel Network
  - Substitution-Permutation Network (SPN)
  - Addition-Rotation-XOR (ARX)
  - Adhoc

Common design strategies

# Block Ciphers

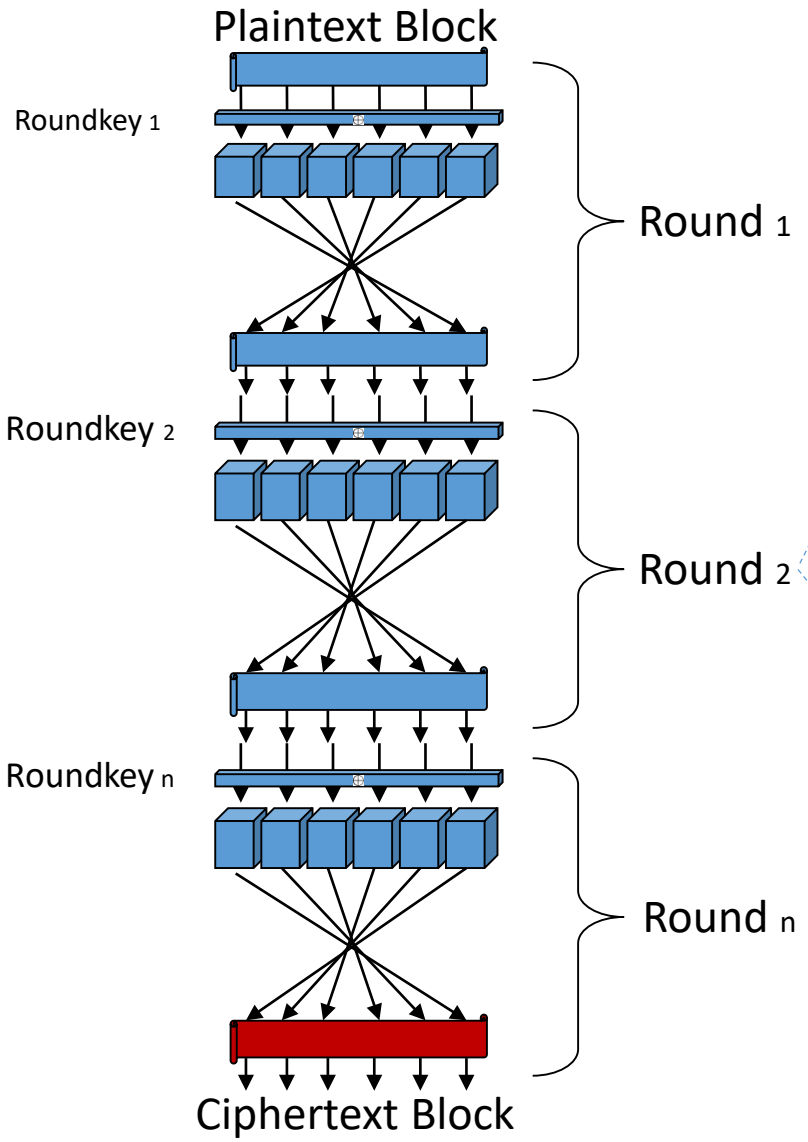
- Feistel Network

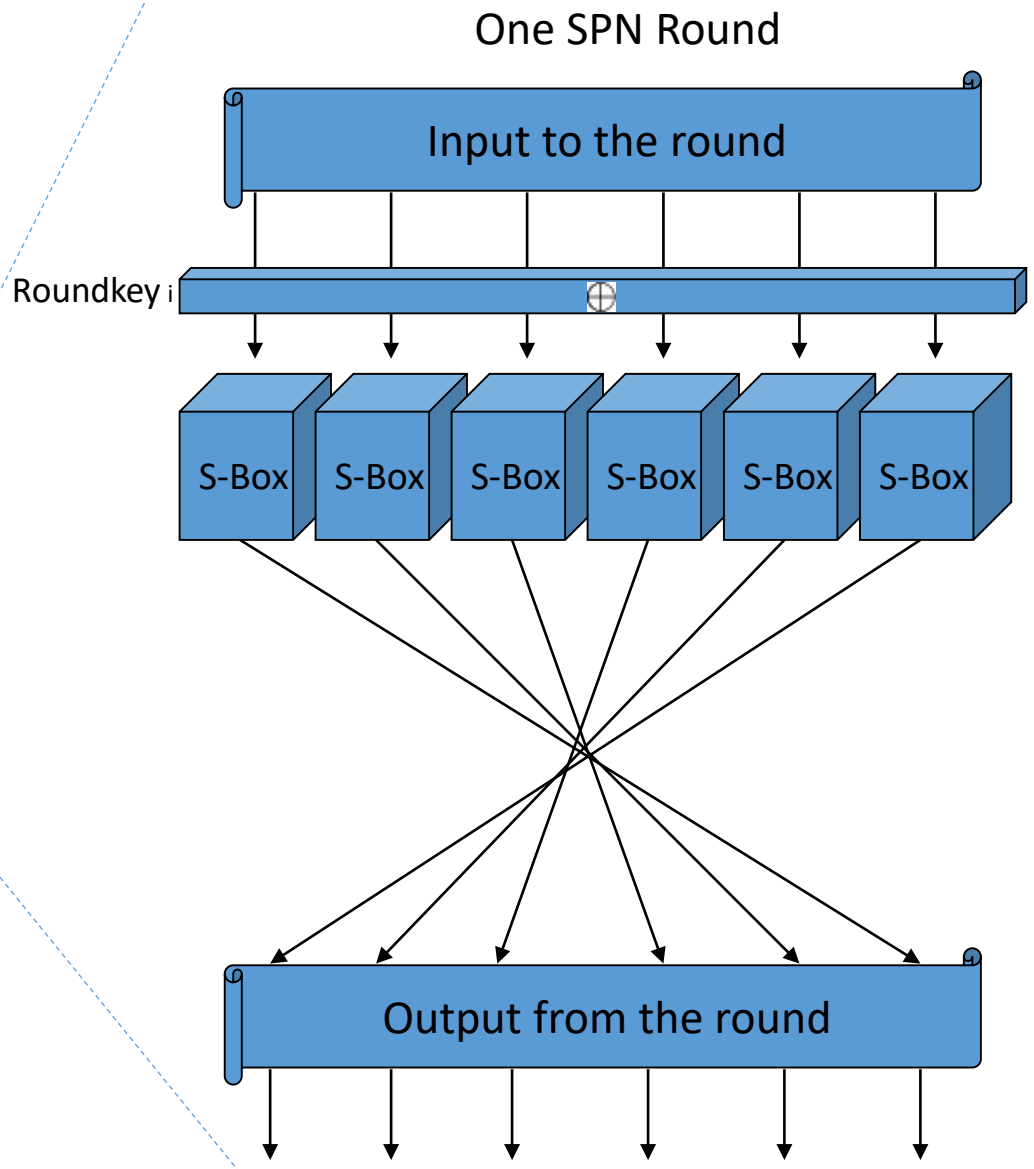


# Block Ciphers

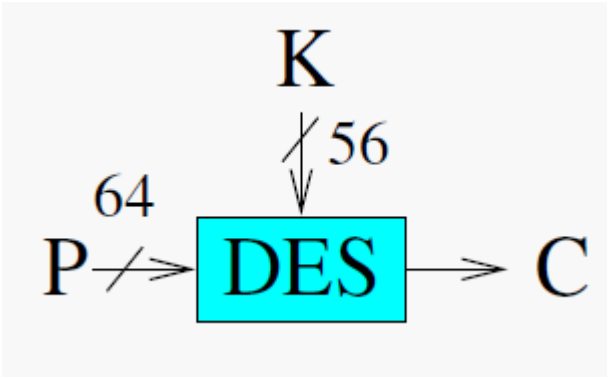
- SPN

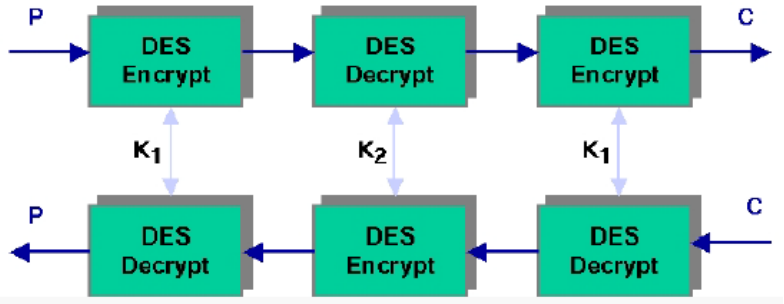


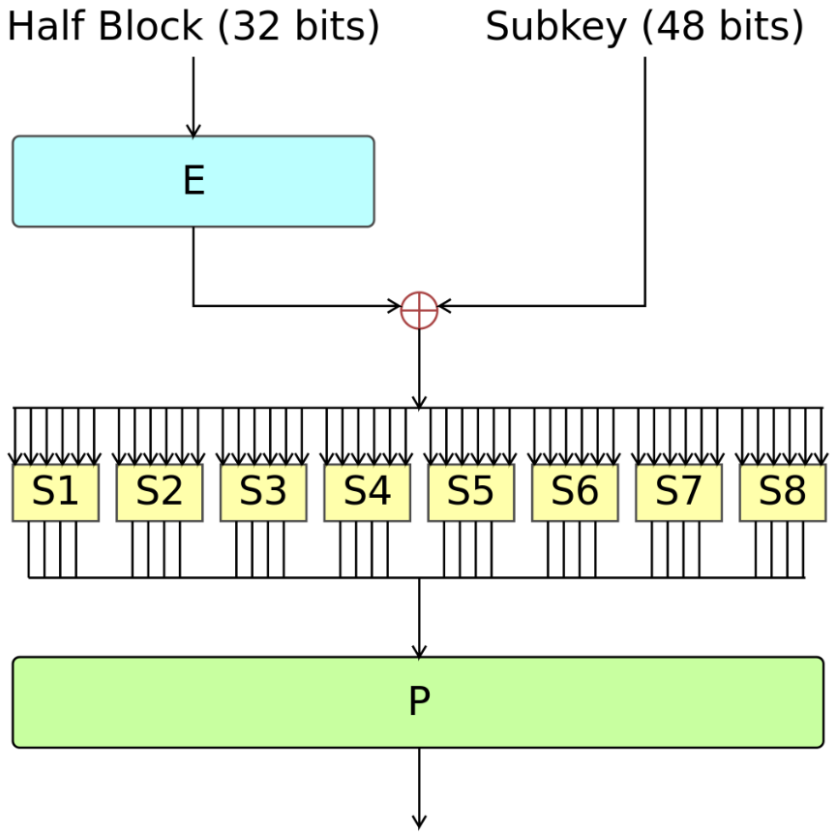
# Data Encryption Standard (DES)

- 1993 NIST Standard
- Feistel network
- Block cipher, encrypting 64-bit blocks, uses 56-bit keys
  - Expressed as 64 bit numbers (8 bits parity checking – key scheduling)
- First cryptographic standard
  - 1977 US federal standard (US Bureau of Standards)
  - 1981 ANSI private sector standard
- Heavily used in banking applications
  - Extensions like Triple-DES (TDES) used to overcome short key-length
  - TDES is the only secure version now



# Data Encryption Standard (DES)



# Security of DES

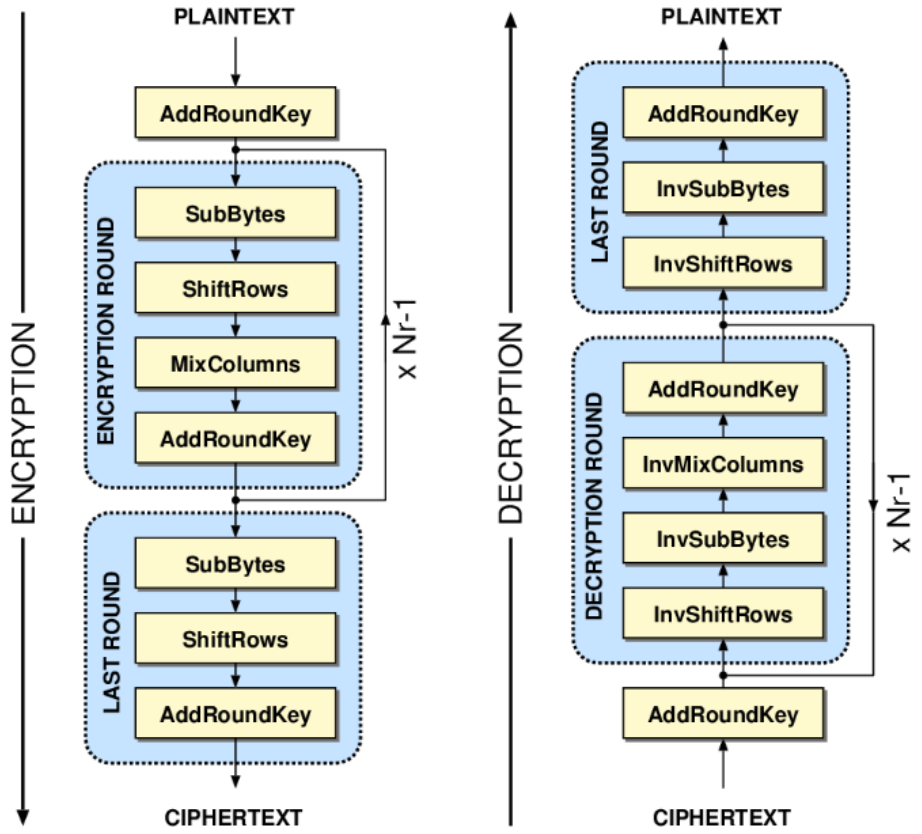
- “People have long questioned the security of DES. There has been much speculation on the key length, number of iterations, and design of the S-boxes. The S-boxes were particularly mysterious - all those constants, without any apparent reason as to why or what they’re for. Although IBM claimed that the inner workings were the result of 17 man-years of intensive cryptanalysis, some people feared that the NSA embedded a trapdoor into the algorithm so they would have an easy means of decrypting messages.” – Bruce Schneier, Applied Cryptography, p278.
- “The National Security Agency also provided technical advice to IBM. And Konheim has been quoted as saying “*We sent the S-boxes off to Washington. They came back and were all different. We ran our tests and they passed.*” People have pointed to this as evidence that the NSA put a trapdoor in DES.” – Bruce Schneier, Applied Cryptography, p279.

# Security of DES

- No security proofs or reductions known
- Main attack: Exhaustive search
  - 7 hours with 1 million dollar computer (in 1993)
  - 7 days with $10,000 FPGA-based machine (in 2006)
- Mathematical attacks
  - Not known yet
  - But it is possible to reduce key space from $2^{56}$ to $2^{43}$ using (linear) cryptanalysis
- Triple-DES: Uses three stages of encryption
  - No known practical attack
  - Brute-force search with $2^{112}$ operations
- DES should not be used for new applications (should at least be TDES)
- “Successor” Advanced Encryption Standard (AES)

# Advanced Encryption Standard (AES)

- SPN
- 128-bit block size, 128/192/256-bit key sizes (key scheduling)
- NIST standard cipher for encryption (2001)
- Widely-used in many applications



# Need for “Tailored” Cryptography

**Lightweight cryptography!**

# Need for “Tailored” Cryptography

**Resource-efficient cryptography!**

# What is lightweight cryptography?

- Reduces computational efforts to provide security
  - Less expensive than traditional crypto
  - Not weak, but “sufficient„ security
    - Reduced level – key size generally below 128 bits

# Lightweight Block Ciphers

- Solutions both from industry and academia

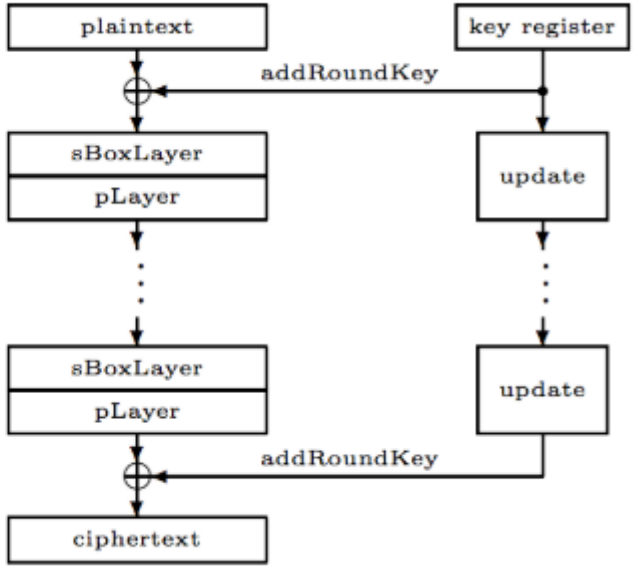
  - Industry
    - In case of propriatery solutions, no public evaluation
    - Generally efficient, but non-standard
    - Lessons learned: MIFARE, Keeloq (stream cipher), etc attacks

  - Academia
    - Good solutions, but sometimes missing industry demands

# Lightweight Cipher Example: PRESENT Cipher

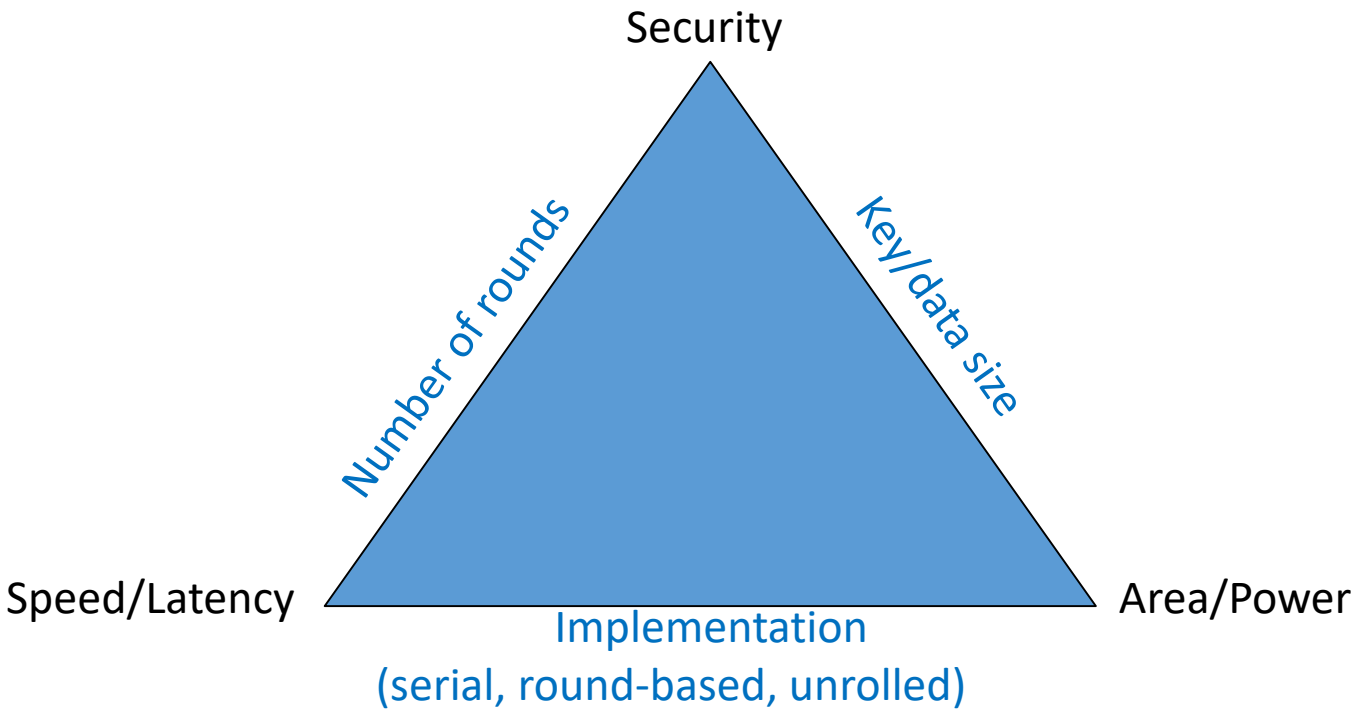
- SPN
- 64-bit block size, 80/128-bit key sizes (key scheduling)
- ISO standard lightweight cipher for encryption

```
generateRoundKeys()
for i = 1 to 31 do
    addRoundKey(STATE, K_i)
    sBoxLayer(STATE)
    pLayer(STATE)
end for
addRoundKey(STATE, K_32)
```



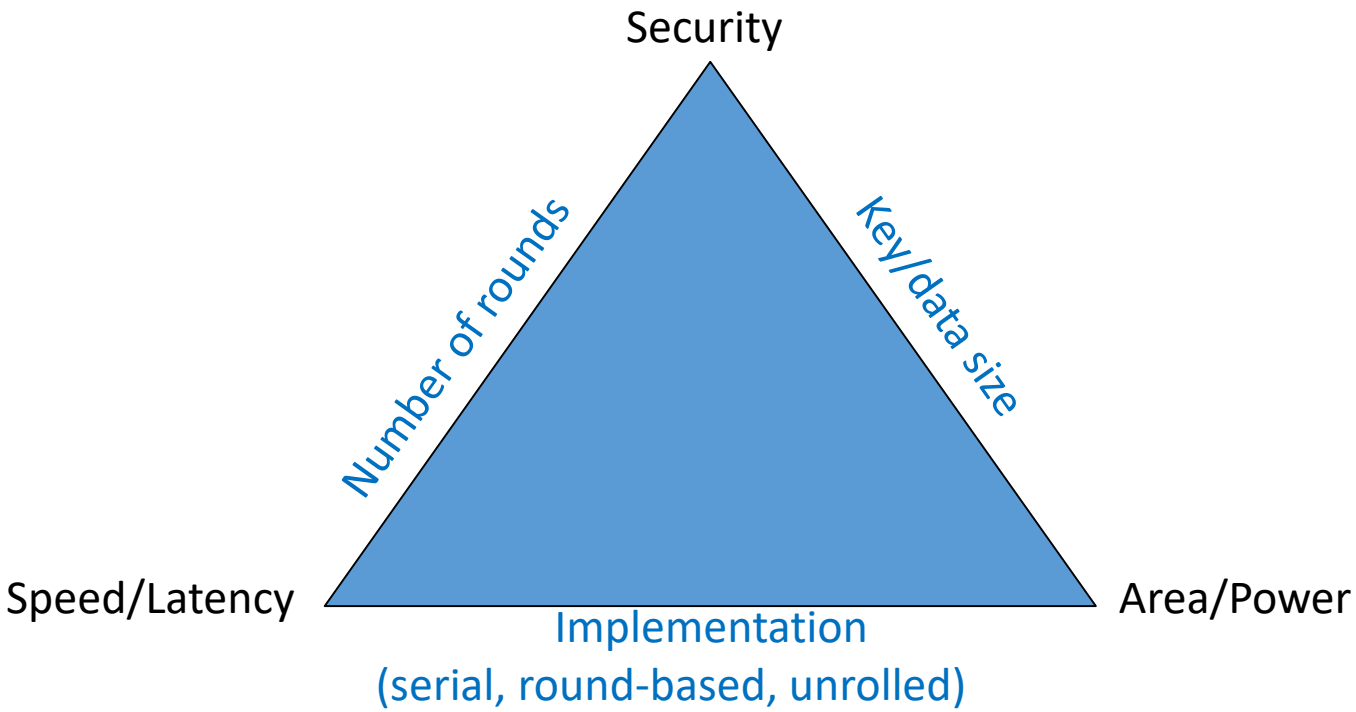
# Proposals vs Metrics

- Initial proposals mostly address area
- There are other important metrics



# Proposals vs Metrics

- Initial proposals mostly address area
- There are other important metrics: Low-latency, less instructions in software, etc.



# Ciphers

- Currently most used
  - Symmetric cryptography
    - Advanced Encryption Standard (AES)
    - Data Encryption Standard (DES) – still, in certain settings…
  - Asymmetric cryptography
    - RSA
    - Digital Signature Algorithm (DSA) – ElGamal
    - Elliptic Curve Digital Signature Algorithm (ECDSA)
    - PQ-Crypto
  - Hash functions
    - SHA2 (SHA256, SHA512)
    - SHA3 (Keccak)
- Standardizations mainly driven by “US National Institute of Standards and Technology (NIST)”

# Key Lengths

- A large enough key space against “brute-force” attacks
  - Symmetric cryptography
    - AES-128: 128 bits
  - Asymmetric cryptography
    - RSA: 2048 bits
    - DSA: 2048 bits
    - ECDSA: 224 bits
  - Hash functions
    - SHA256: 256 bits hash-value
    - SHA3: Different sizes possible

# Bibliography


- Ross J. Anderson. Security Engineering: A Guide to Building Dependable Distributed Systems. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 2001.
  - The complete book is available at: http://www.cl.cam.ac.uk/~rja14/book.html
- Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. Handbook of Applied Cryptography. CRC Press, Inc., Boca Raton, FL, USA, 5th edition, 2001.
  - The complete book is available at: http://cacr.uwaterloo.ca/hac/
- Bruce Schneier. Applied Cryptography. John Wiley & Sons, Inc., 2nd edition, 1996.

# Thanks for your attention!

- Any questions or remarks?