>

DI-FCT-UNL
Segurança de Redes e Sistemas de Computadores
Network and Computer Systems Security

Mestrado Integrado em Engenharia Informática
MSc Course: Informatics Engineering
1st Sem., 2019/20120

Applied Cryptography

Cryptographic Tools,
Methods, Techniques and Algorithms

Outline

- **Applied Cryptography**
  - Typology: objectives and focus of different cryptographic methods and algorithms
  - Cryptographic constructions
  - Secure channels w/ cryptographic protection
  - Tools:
    - Java JCA/JCE for Programming w/ Cryptography
    - Tools in the Java Environment
    - Openssl library and the openssl tool

Cryptography

History / Origins
Ancient Methods, Classical Cryptography
vs. Modern Cryptography
Computational / Applied Cryptography

Cryptography … from classic cryptography …

(from the greek): krypthós (*hidden*) + graph (*"graphein" root… writing*)

Classic Cryptography: secrecy of the algorithm (or the means used to encode/decode functions)

- **Ancient Techniques:**
  - Ex., Bastions of Spartans, Secrets/Codes embedded in scriptures, …
  - Simple text substitution techniques (rotations, additive substitutions)
  - Transpositions (permutations, reordering, geometric, columnar, table-relations, …)
  - Primarily: Monoalphabetic Ciphers,
- **Middle age, 1500s >**
  - Ex., Polyalphabetic substitutions, permutations (ex., 1553, Viginère Cipher) - Algebraic description: $Ci = E_K (P_i) = (P_i + K_i) \bmod 26$
  - Initial Algebraic Descriptions and Methods
- **1920s > …**
  - OTPs w/ Key-Stream Generators and Algebraic Constructions
  - Algebraic Constructions (Polyalphabetic Permutations w/ Matrix-Transf.)
- **1930s-1950 … Rotor Machines**

(Some) Classical Encryption Methods
and related transformations

- **Caeser** Cipher

  (Shift Rotation mod)
- Generalization:

$C=E_3(P) = (P+3) \bmod 26$

$P=D_3(C) = (C-3) \bmod 26$

$C=E_k(P) = (P+k) \bmod 26$

$C=E_k(P) = (P-k) \bmod 26$

Other transformations

- **Monoalphabetic** Ciphers: Permutations and Transpositions
- **Chinese Methods**: Columnar Transformations
- **Viginère** Cipher: Polyalphabetic Ciphers: Polyalphabetic Substitutions 1533
- **Playfair** Cipher: Permutations w/ Multiple Letter Encryptions 1854
- **Vernam** Cipher: bir-XOR w/ Key-Stream Generation, No Statistical Relatioships between Plaintext and Keys 1918
- **Hill** Cipher: Linear Algebra (Matrix-Based Multiplications) 1929
- **OTPs**: Unbreakble One Time Pads 1930s
- **Rotor Machines**: Multiple (chains) Setup-Parameterized Permutations and Transpositions 1950s

Cryptography … Modern Cryptography

Modern Cryptography: algorithm not secret

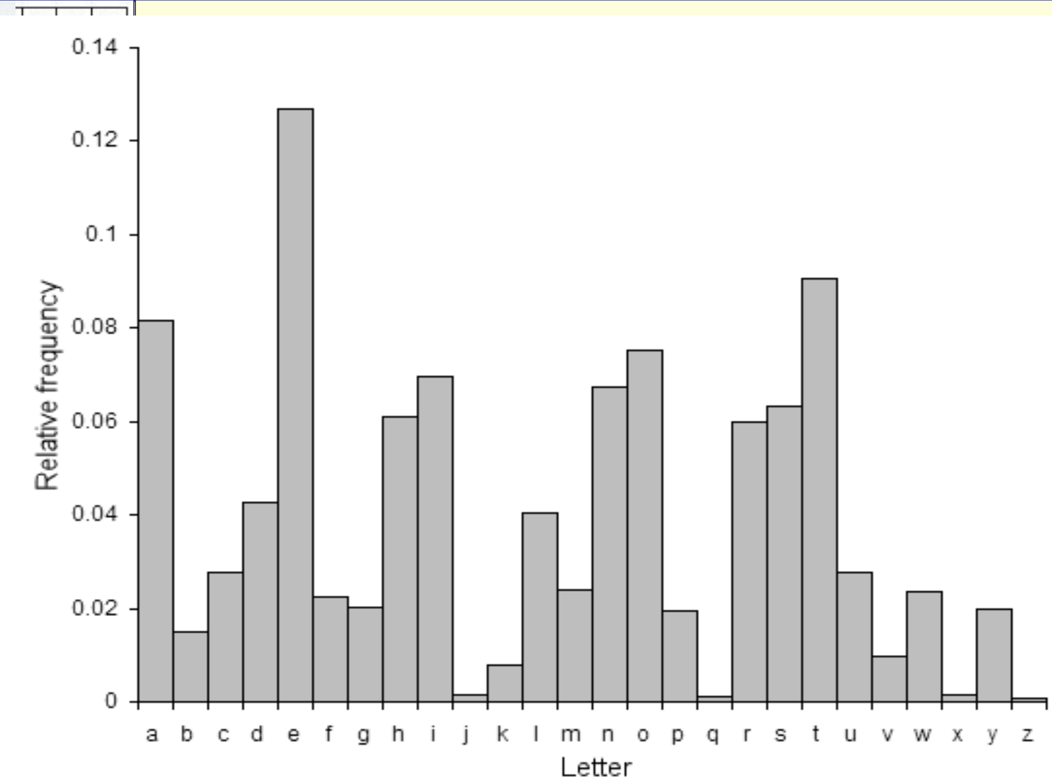
Secrecy is on the algorithm parameters (i.e., Cryptographic Keys)

Research: until the end of 1960s … 1970s … until now

Some examples:

SC	**Lucifer** 1971, Horst Feistel
SC	**Feistel Structure** 1973, Horst Feistel
SC	**DES** 1977, IBM for NBS, later NSA
AC	**Diffie-Hellman** 1976 Whitfield Diffie, Martin Hellman
AC	**RSA** 1977-1978, Ron **R**ivest, Adi **S**hamir and Len **A**dleman
SH	**MD2(**1989, Ron Rivest
AC	**DSA** 1991, NIST
SC	**AES** 2001, NIST, from Rijndael proposal 1998
AC	**ECC** Foundations 1885, N. Kolbitz, Victor Miller
SH	SHA-2 2001- … 2013 … NIST
AC	**ECC Crypto** 2004-… until now, many contributions
SH	**SHA-3** 2006-2015
	from initial Keccak Construction, G. Bertoni, J. Daemen, M. Peeters, G. Assche

Classical Substitutions and Transpositions …
(many examples: Morse, Great, Zodiac, Pipgen, …etc etc)



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

Other classic algorithms: Playfair

Playfair

Plaintext	armuhsea
Ciphertext	RMCMBPIM or RMCMBPJM

Encryption of two letters at a time:

5x5 Matrix Encoding: Key + Alphabet
Ex., Key: MONARCHY

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

ar > RM

mu > CM

hs > BP

ea > IM (or JM)

More sophisticated (Algebraic, Matrix Mult.) Polyalphabetic Substitutions : Hill Cipher

- Ex., Hill Cipher (Sir Lester S. Hill, 1929)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	W	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	28	19	20	21	22	23	24	25

Plaintext:
ACT

$$\begin{pmatrix} 0 \\ 2 \\ 19 \end{pmatrix}$$

Key:
GYBNQKURP

$$\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix}$$

Encryption:

$$\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix} \begin{pmatrix} 0 \\ 2 \\ 19 \end{pmatrix} = \begin{pmatrix} 67 \\ 222 \\ 319 \end{pmatrix} \equiv \begin{pmatrix} 15 \\ 14 \\ 7 \end{pmatrix} \pmod{26}$$

Ciphertext:
POG

Decryption

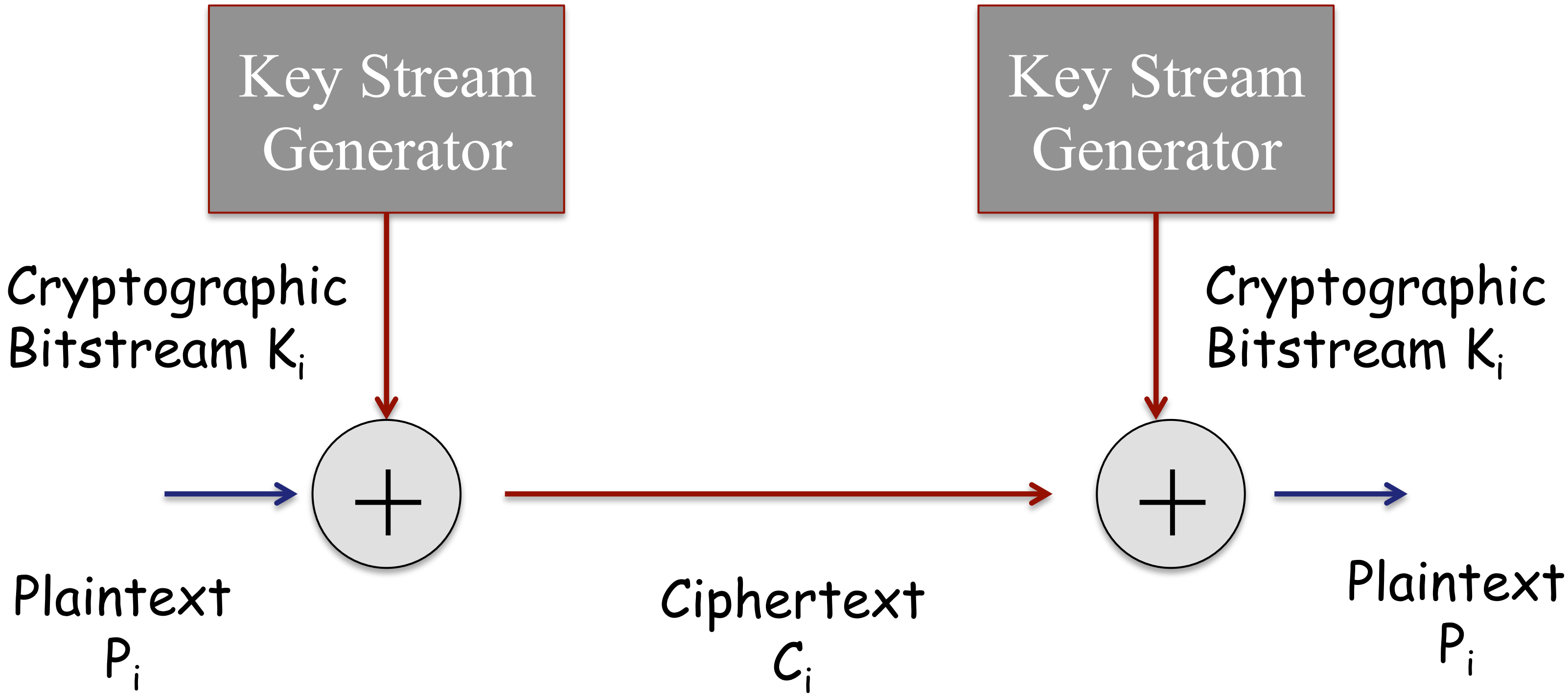
$$\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix}^{-1} \equiv \begin{pmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{pmatrix} \pmod{26}$$

$$\begin{pmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{pmatrix} \begin{pmatrix} 15 \\ 14 \\ 7 \end{pmatrix} \equiv \begin{pmatrix} 260 \\ 574 \\ 539 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 2 \\ 19 \end{pmatrix} \pmod{26}$$

Plaintext:
ACT

Other classic algorithms: Vernam Construction
Idea: choose a keyword as long as the plaintext

J. Vernam, 1918
The very-base idea for a symmetric stream cipher !



The Principle of OTPs (One Time Pads)

J. Mauborgne idea:
Use a random key string, so long as the message size

Random Key Stream … never repeated to encrypt/decrypt Random Key Stream

Supposing we will test a certain number of permutations …:	Time to break (brute force) ?:
10^9 tests/s	$4 \times 10^{26}/10^9 = 4 \times 10^{17}$ s = 6.3×10^9 years
…	…
10^{13} tests/s	6.3×10^6 years

Plaintext P_i CipherText C_i PlainText P_i

Interesting aspects:
Unbreakable
Randomness
Permutations of 26 Chars, (monoalphabeth): $26! = 4 \times 10^{26}$

Practical aspects:
Unbreakable if …
Truly Randomness… vs. Repeatable Keys
Key DistributionEstablishment and Sync.

Rotor Machines
Ex: Enigma Machine

Ex., German Enigma Machine (WW2)

http://enigmaco.de/enigma/enigma.html
https://play.google.com/store/apps/details?id=uk.co.franklinheath.enigmasim&hl=en
https://itunes.apple.com/us/app/mininigma-enigma-simulator/id334855344?mt=8

Rotor Machines
Ex: Enigma Machine

A "Polyalphabetic Substitution Cipher" Machine
Period = 16900 x the more longest encoded message

Summary of the combinatory:

- A Table with permutations of 26 characters = 26! Permutations
  - $26! = 4 \times 10^{26}$
- 3 Rotors: 3 x 26! Permutations (in 15576 possible combinations)
- 4 Rotors: 4 x 26! Permutations (in 456976 possible combinations)
- Plugboard with L leads
  - Combination of letter pairs: $26! / (26-2L)! * L! * 2^L)$
    - Ex., L= 6 => 100,391,791,500 combinations =~ 100×10^{12} => 100 billions
    - Ex., L=10 => 150,738,274,937,250 = 150×10^{15} => 150 trillions

Steganography

Steganography Techniques

- Hidden secret information, encoded in public/available information (concealing the existence of the hided information), Ex:
  - Secret messages (text) in tests
  - Secret messages (text) in images
  - Secret messages (text) in sounds
  - Secret info (text or images) in movies
  - In genera: secret media hidden in public media

  - Examples and demos (will appear in a LAB class)

Example of other interesting approaches (Recent / On-going Research)

- Unobservable Covert Channels
  - Traffic encapsulation
  - Internet Censorship Circumvention
  - Privacy Enhanced Communication (Private Channels / Public Net. Infrastructures)

- Content Withholding Circumvention
- Non-Detectable Network Proxies / Traffic Rerouting combining Data Flows using Steganography Techniques and/or Traffic Obfuscation
- Communication over Randomized Protocols (Circumvention of Protocol Classifiers for Blocking Strategies)
- Protocol Imitation
- Protocol Tunnelling / Staged Protocols and Applications
- Tunnelling w/ Traffic Media Morphing and Multimedia Streaming
  - Ex., Covert Channels for Censorship Circumvention using Staged Conferencing Tools (Skype, Hanghout, etc)

Outline

- **Applied Cryptography**
  - Typology: objectives and focus of different cryptographic methods and algorithms
  - Cryptographic constructions
  - Secure channels w/ cryptographic protection
  - Tools:
    - Java JCA/JCE for Programming w/ Cryptography
    - Tools in the Java Environment
    - Openssl library and the openssl tool

Computational Applied Cryptography

- Conventional Cryptography: Families of Method and Algorithms and related Techniques

Symmetric Crypto	Asymmetric Crypto	Key-Agreement	Secure Hashing

Asymmetric Model

Base Families
(Algorithms)

- Conventional Cryptography: Families of Method and Algorithms and related Techniques

Symmetric Crypto	Asymmetric Crypto	Key-Agreement	Secure Hashing

Symmetric Crypto

Also known as…

- Symmetric Encryption / Shared Key Cryptography

2 Basic Methos/Algs:

- Block Ciphers (or Block-Oriented Crypto Algs.)
- Stream Ciphers (Stream-Oriented Crypto Algs.)

Asymmetric Crypto

Also known as…

- Public-Key Cryptography
- Essentially, block-oriented Crypto Algs.

Key-Agreement

Also known as…

- Key-Exchange Cryptographic Methods and Algs.

Secure Hashing

Also known as…

- Message-Digesting Methods/Algs.

Computational Applied Cryptography

- Conventional Cryptography: Families of Method and Algorithms and related Techniques

Symmetric Crypto

- Primary use for **Confidentiality** (Message-Data Encryption)
- Possible use for Authenticated and Key-Establishment Protocols using KDCs

Asymmetric Crypto

- Primary use for **Authentication**
- Possible use for **Confidentiality**
- Also used for "Key-Distribution and Secure Establishment of Security Associations using PKCs (or CAs)

Key-Agreement

- Primary use for **Key-Exchange**

(or Key-Establishment among two or more principals (multi-party)

Secure Hashing

- Primary use for **Integrity** (Message or Data Integrity Proofs)

Applied Cryptography: Typology

- Conventional Cryptography: Families of Method and Algorithms and related Techniques

Symmetric Crypto

- Two Types of Algorithms

Block Ciphers

Block Encryption Algs.

(Various

Stream Ciphers

Stream Oriented Encryption Algs.

(Various

Asymmetric Crypto

- Two Types of Algorithms

Oriented for Digital Signatures (Authentication Only)

Oriented for Authent. And Encryption (Confidentiality)

Key-Agreement

- Different Methods and Algs for different purposes

Secure Hashing

Different Methods and Algs with a common set of properties

"Each monkey on its correct branch" ☺

Confidentiality (Encryption)	Auth. + Key Exchange	Integrity Proofs	Establihment of Keys and SA Parameters
Symmetric Crypto	**Asymmetric Crypto**	**Secure Hashing**	**Key-Agreement**
Shared Key (sender/receiver) Encryption and Decryption functions use the same key Require secure Key generation and distribution Block encryption: Use of different encryption modes and related parameters Block and key sizes fixed for each algorithm	Use of Keypairs: Private and Public Keys (What you encrypt with one key of the pair you can decrypt with the other key of the pair Main use for Digital Signatures and Secure Envelopes for Sym Key Distribution or protected secrecy parameters	Use for Integrity Proofs	Based on Asymmetric Methods (using private and public Parameters as Key-Values) Used for secure establishment of symmetric keys or security assotiation parameters

(Some) Examples (“*of Kids in Town*”)

- Conventional Cryptography: Families of Method and Algorithms and related Techniques

Symmetric Crypto

Block Ciphers:
- DES
- 3DES
- Camelia
- IDEA
- Blowfish
- ChaCha-Poly
- Skypjack
- AES
- Tofish
- RC6
- …. **etc**

Stream-Ciphers
- RC4
- A5…A8
- VEST
- SEAL
- SNOW
- Truring
- WAKE
- … **etc**

Asymmetric Crypto

Sign. + Encr:
- RSA
- ECC/ECC Variants
- El Gammal
- Paillier
- Knapsack
- … **etc**

Sign. Only
- DSA or DSS
- ECDSA

… **etc**

Key-Agreement

- Diffie-Hellman
- GDH Variants
- … etc

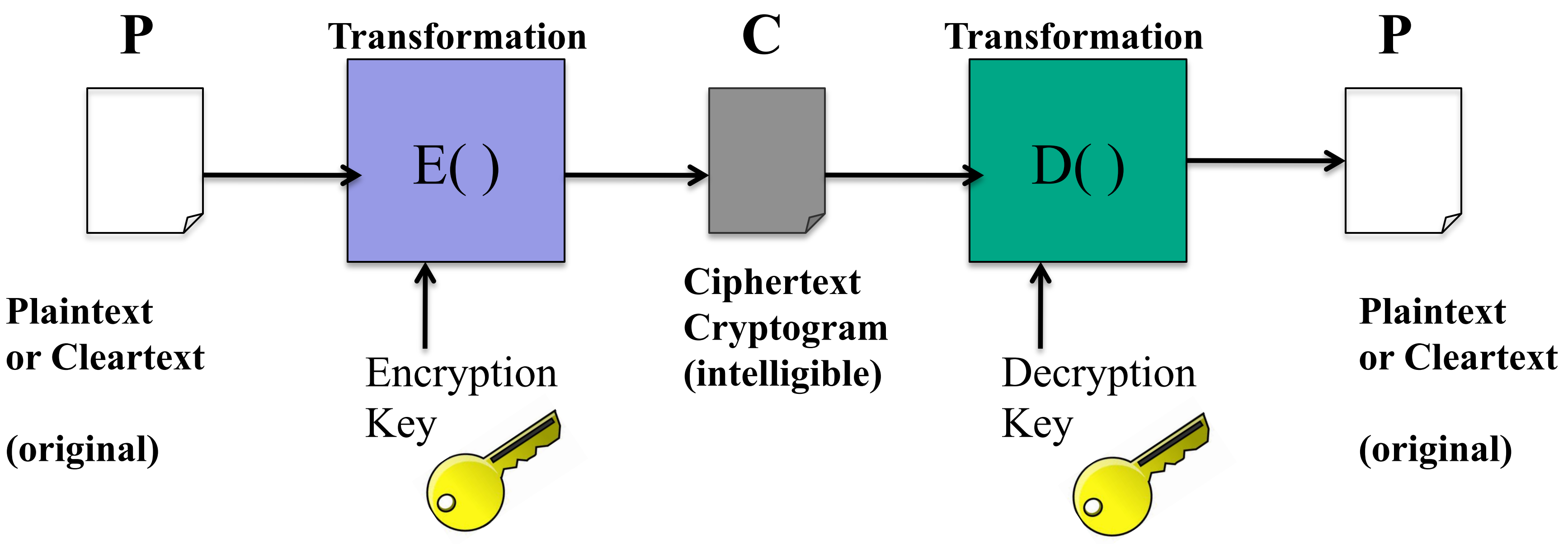
Secure Hashing

- MD5
- SHA-1
- Wirhlpool
- SHA-2
- SHA-256
- SHA-384
- SHA-512
- Blakw 2
- SHA-3
- … **etc**

Symmetric cyphers:
Block Ciphers vs. Stream Ciphers

- **Block-Oriented (or block ciphers)**
  - Used (parameterized) with different **block modes of operation, possible Initialization vectors** and **padding processing (as security association parameters)**
  - Key sizes and block sizes defined (fixed) for each algorithm (you must know it …)
    - … Characteristics for each algorithm
- **Stream-oriented (or stream ciphers)**
  - Byte-stream-oriented or bit-stream-oriented operation
  - Variable key-sizes (algorithm dependent)
  - Fast to operate on stream-oriented inputs, ex., Real-Time Processing (bytes, bits)
    - Ex., real-time bit-streaming, iterative-traffic and/or low-latency communication requirements
  - Security issue: the security and period of the keystream generation

Block Ciphers (Block-Encryption ALGs)

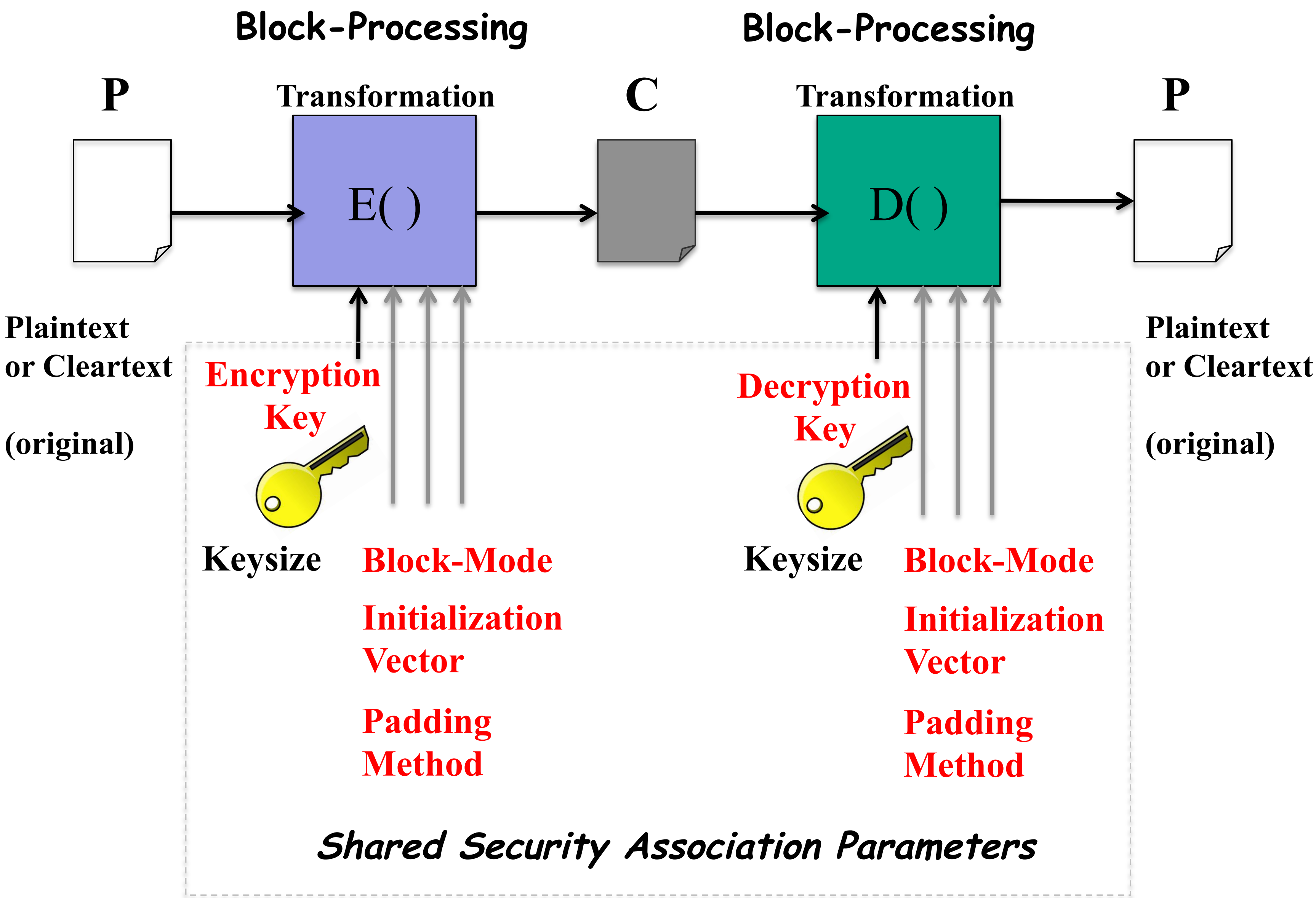


Notation:

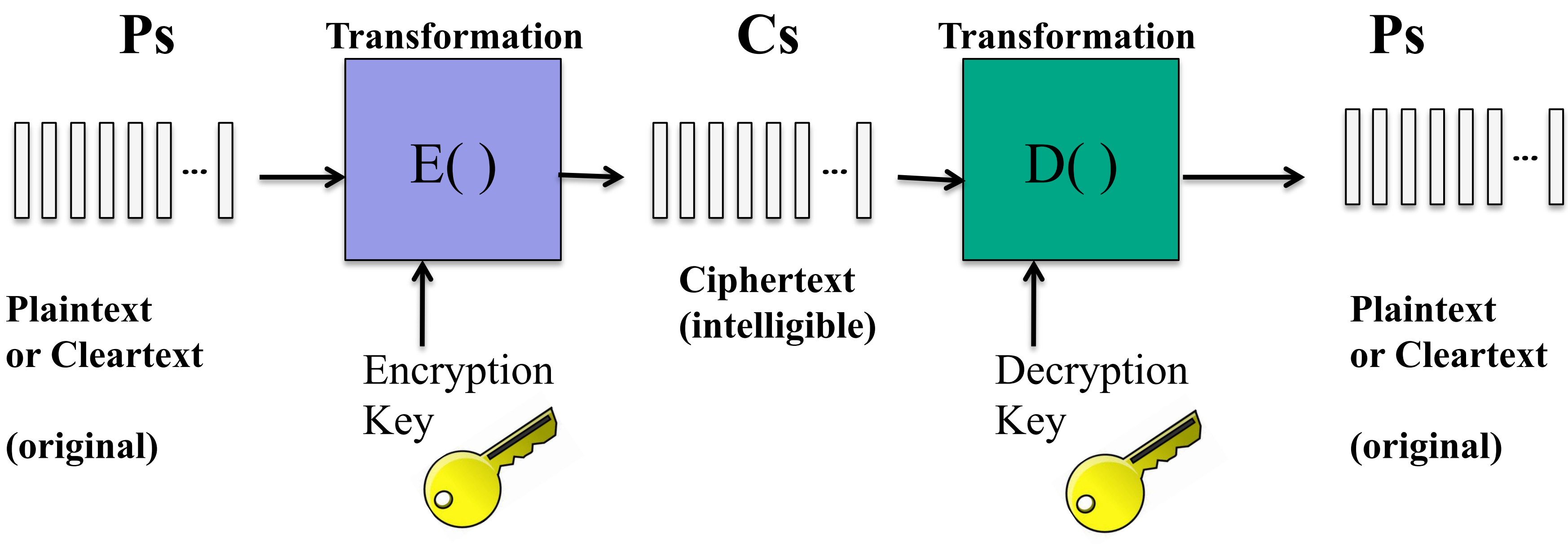
$C = \{P\}_K$; $C = E(P, K)$; $C = E_k(P)$ // M encrypted with key K

$P = \{C\}'_K$; $P = D(C, K)$; $P = D_k(C)$ // C decrypted with key K

Symmetric Block Encryption and related Security Association Parameters

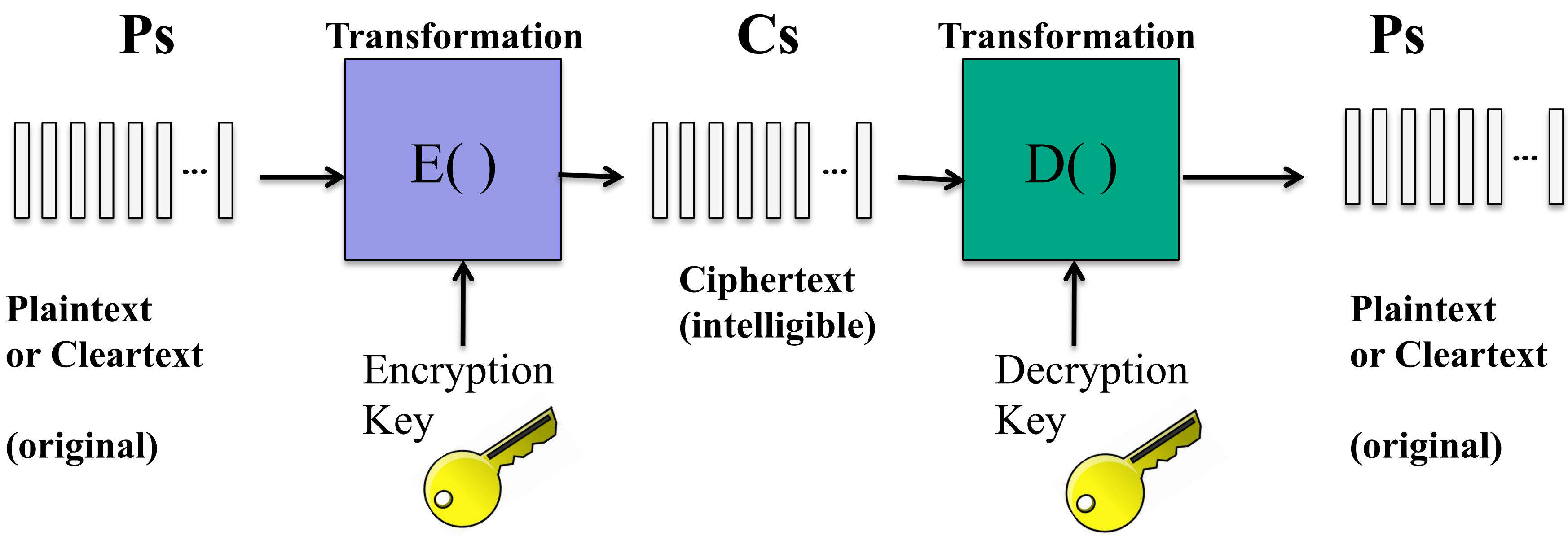


Stream Ciphers



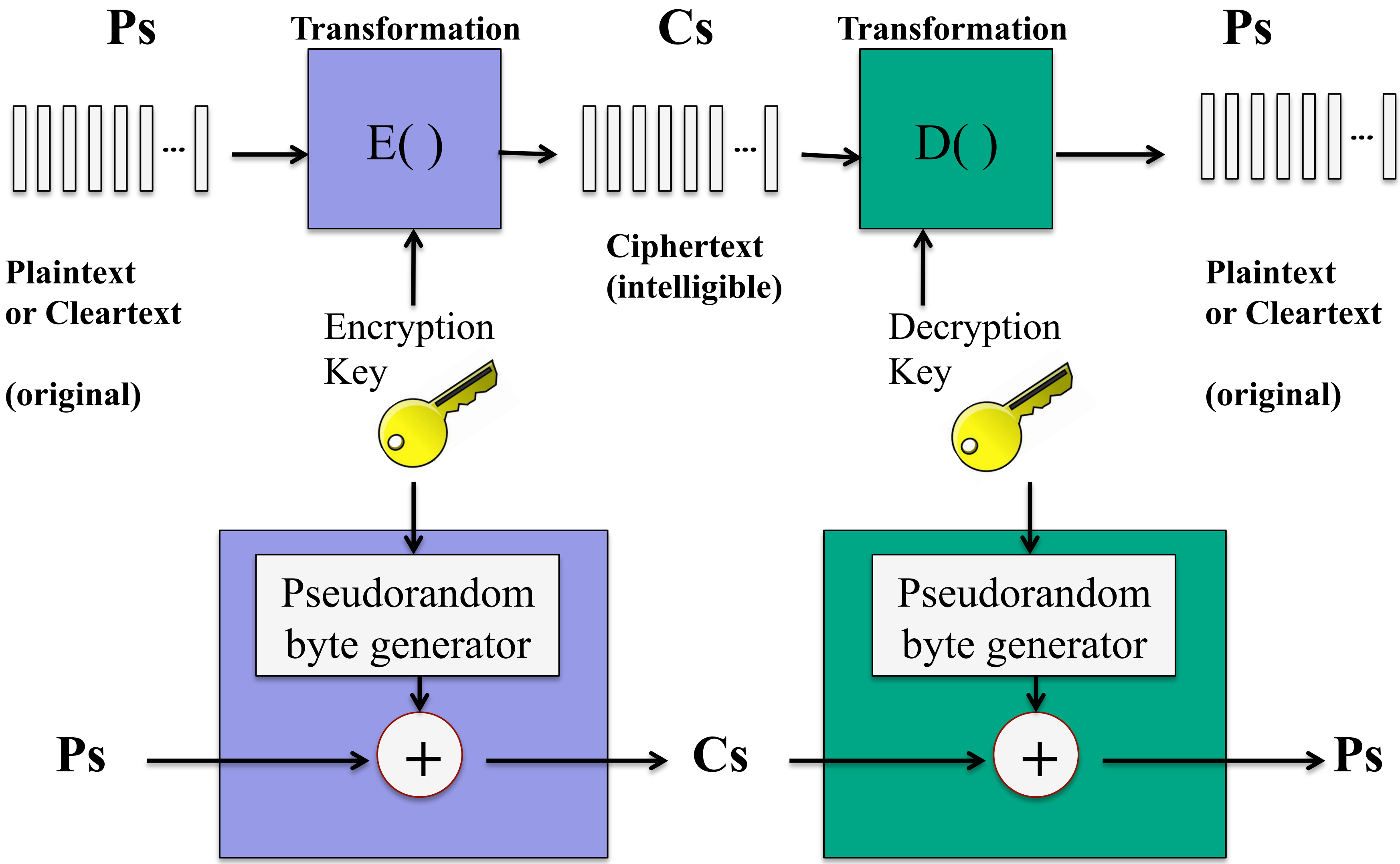
- **Use for stream-encryption, ex., bit-streaming**
- **Interesting: real-time bit streaming** (ex., radio-frequency communication)

Stream Ciphers



- **Use for stream-encryption, ex., bit-streaming**
- **Interesting: real-time bit streaming** (ex., radio-frequency communication)

Stream Ciphers (Typical Structure)



Asymmetric Crypto

Asymmetric Cryptography or Public-Key Cryptography

- **Robustness** (security and correctness of the symmetric encryption algorithms on their trust-execution criteria)
  - Resistance against brute-force attacks and cryptanalysis attacks
- Need **security association parameters** for the intended purpose
- Need of **strong keys**: generated w/ randomness, distributed and maintained with security guarantees
  - TRNGs, PRNGs, PRFs for Key-Generation and other parameters
  - Ex., possible use of HSMs, Smartcards, Crypto-Tokens
  - Avoidance of "possible weak keys" in the key generation process for a specific algorithm
- Need of **secure key-distribution and establishment protocols and services** (shared keys and related security association parameters)
- **Minimization or Avoidance of key-exposure**
- **Fast and secure "rekeying"** services with perfect future and past secrecy, with key-independence
  - Ex., Rekeying for temporary session keys, or keys used with OTP assumptions

Asymmetric Cryptography Model

Need Two related keys (or a **Key-Pair Generation)**

- **a public-key**, known by anybody: can be used to encrypt messages, and verify (or recognize) digital signatures
- **a private-key**, maintained as private: used to decrypt messages, and sign (create) digital signatures

In general*, what we encrypt with one key, we can decrypt with the other key of the pair

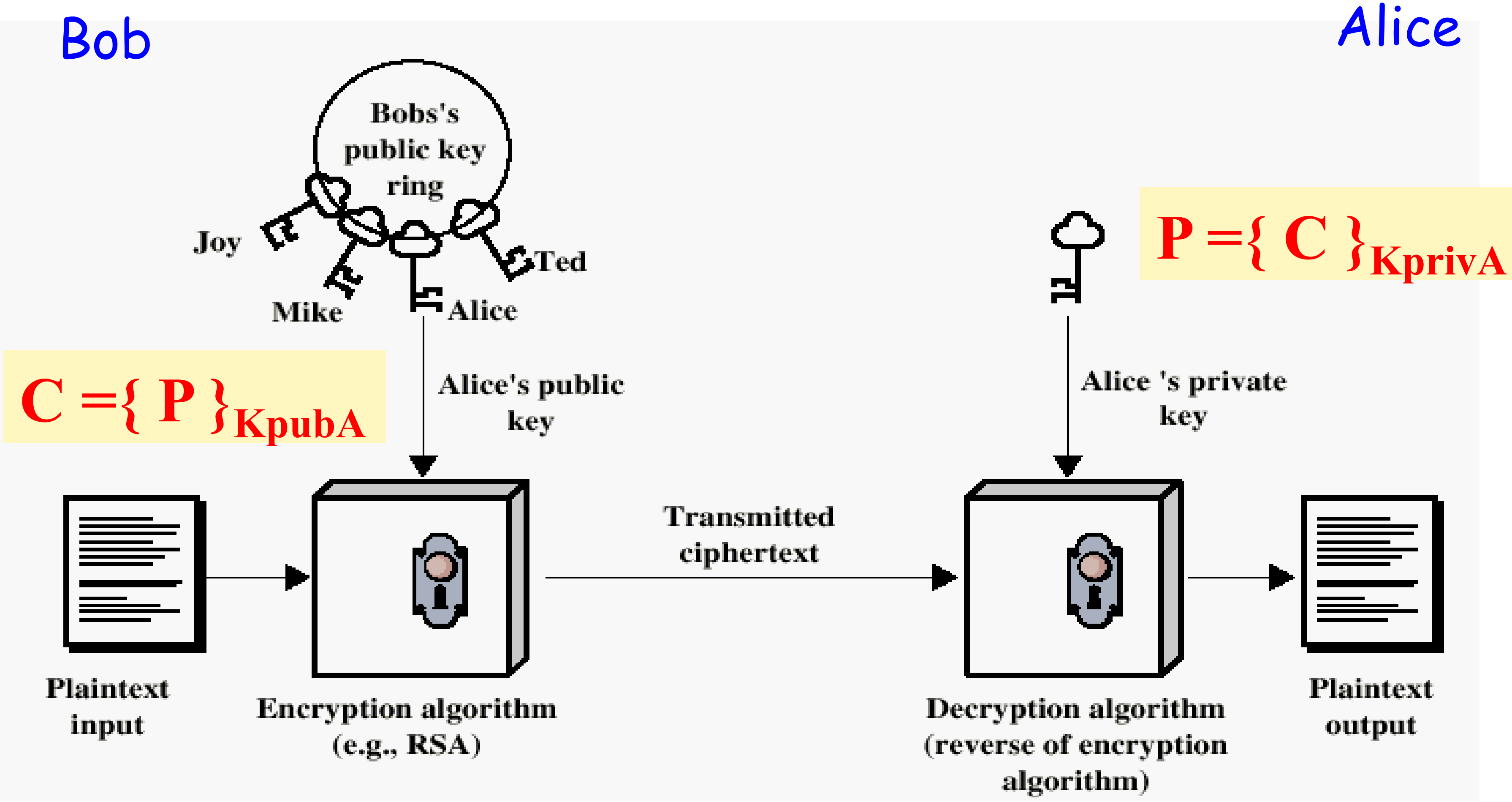
**) Sometimes not supported in specific algorithms and constructions*

Same function (same computation) for encryption and for decryption, different keys

Note: Contrary to Symmetric Encryption: different functions (inverse functions), with the same key

Encryption using a Public-Key Crypto System

For Confidentiality principles:
Encryption with the destination Public key

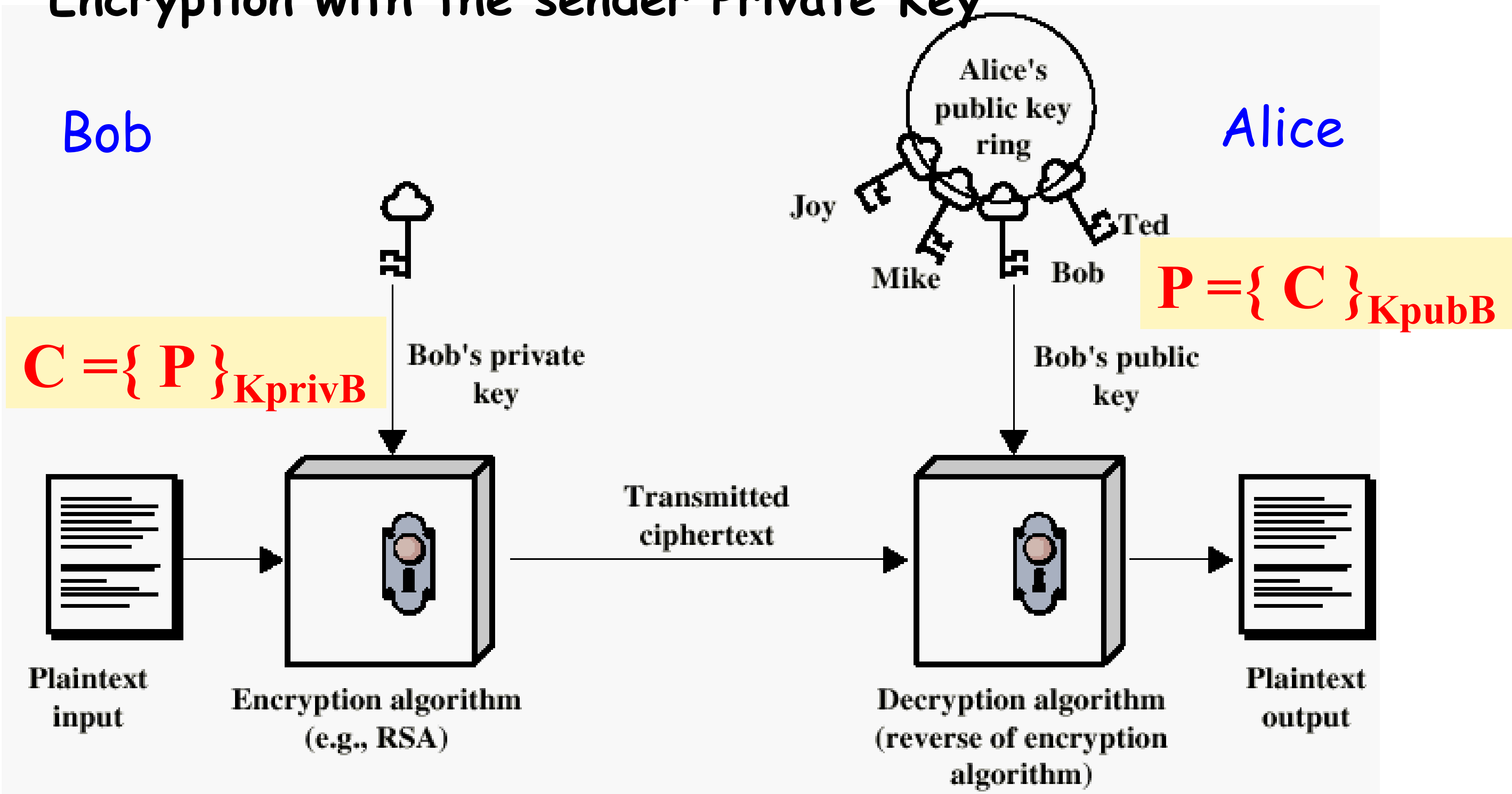


Asymmetric Crypto

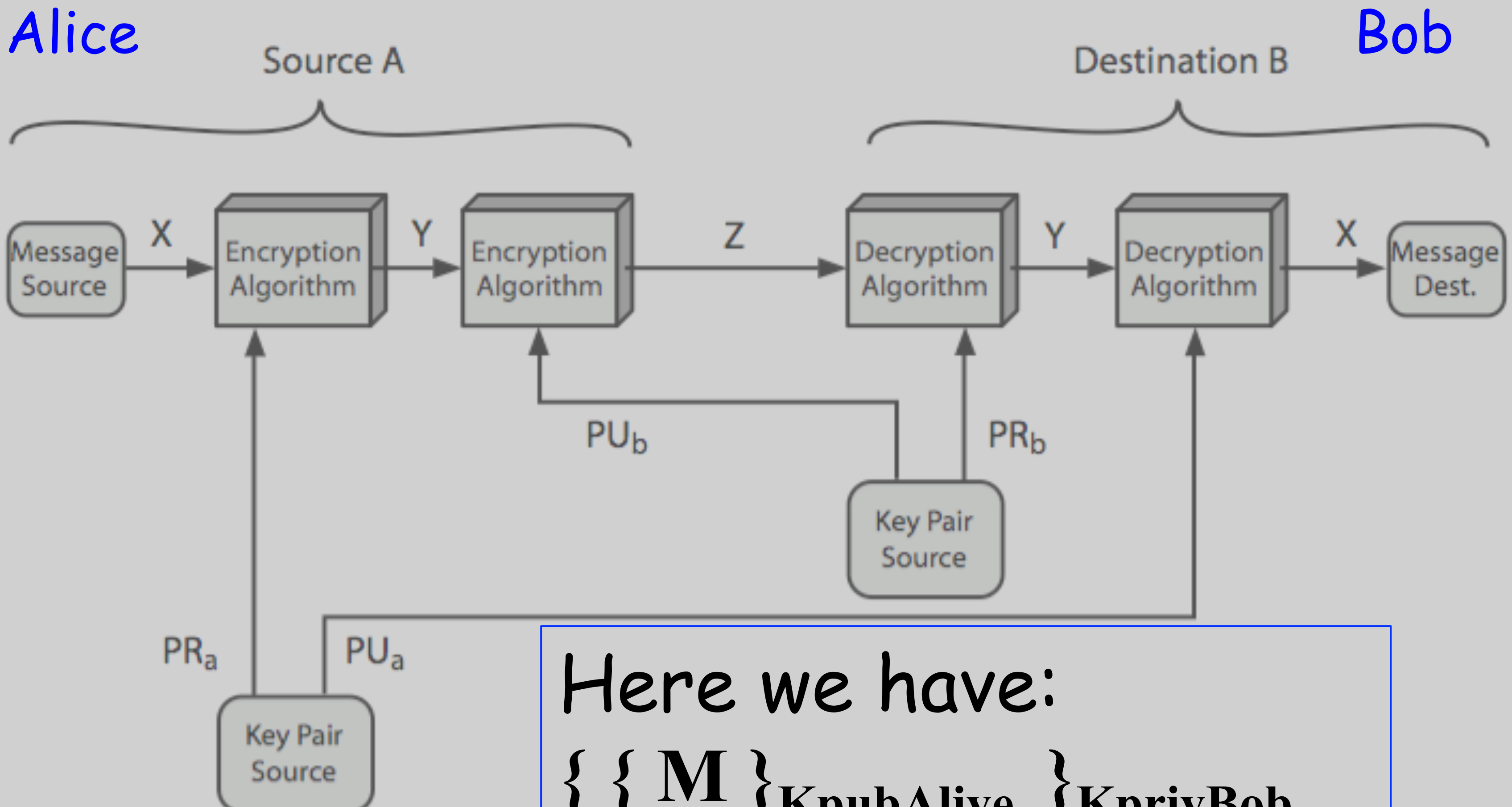
Authentication using Public-Key Crypto System

For **Authentication** principles:

Encryption with the sender Private Key



Confidentiality + Authentication



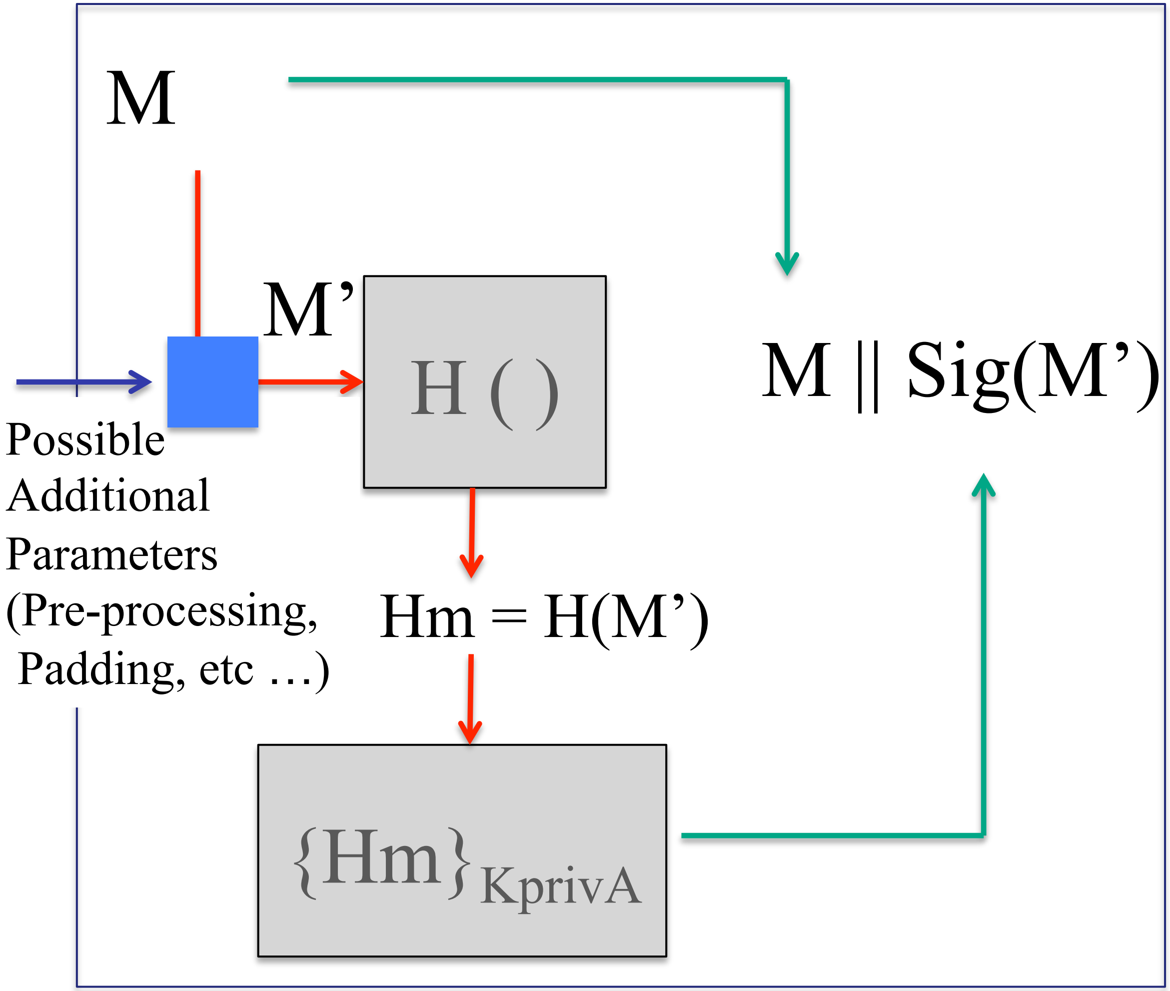
Here we have:

$$\{ \{ \mathbf{M} \}_{\mathbf{KpubAlive}} \}_{\mathbf{KprivBob}}$$

Uhm … Not good (practical) idea !!! Why ?
Can we do better for secure use ? How ?

Basic Scheme for Authentication: Base Digital Signature Construction

- Principle of construction of Digital Signatures Schemes: Sender

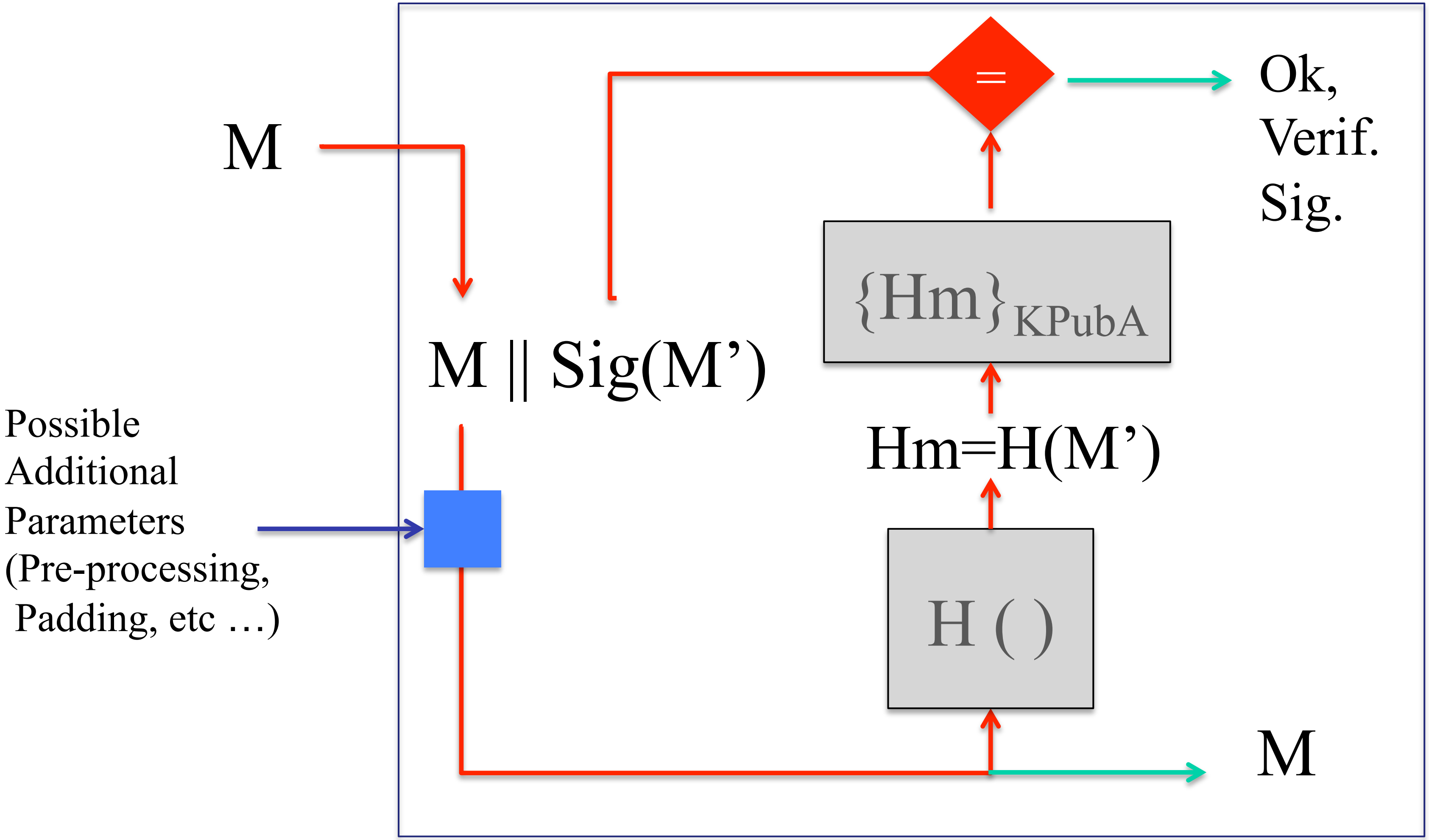


Send Signed (Peer-Authenticated) Message M

Relevance of Standard Constructions or Schemes.
Why ?

Base Scheme for Authentication

- Principle of Verification of Digital Signatures Schemes



Use of Secure Hash Functions

- Use for Integrity
  - Message (or Data) Integrity Proofs and Guarantees
  - Integrity of records in a Database (Data Base Integrity)
  - Integrity of message flows with hash-chains (as aggregated integrity proofs in a chain of ordered messages in the flow

    H(Mi) = H(M || H(Mi-1)) w/ H(Mi-1) = H(Mi-1) || H M(i-2) … and so on

    This can be used as Traffic-Flow Integrity Proofs
  - Other examples:
    - Integrity of Chains of Data Blocks (Integrity and Irreversibility of Blocks in Blockchains, where Blocks are "Hashed-Chained", With Blocks and Hash-Proofs maintained persistently, for example and typically, in a Merckle Tree Structure in a Data-Base (used as a LEDGER), decentralized (replicated) with Certain Consistency and Ordering Guarantees
    - Also used for Proof-Of-Work Verification. How ? Why ?
    - Also usable for possible DoS Avoidance Protection. How ? Why ?

Emergent Cryptography (Beyond the Base-Cryptographic Algorithms and Methods)

- New Arithmetic Constructions in Emergent Cryptography
- (Some) examples (recent research)
  - Lattice-Based Cryptography (Post-Quantum) and important constructs (ex.,):
    - ZKP (Zero Knowledge Proofs), IBE (Identity Based Encryption)
  - Homomorphic Encryption Alg. And Methods
    - FHE (Fully Homomorphic Encryption)
    - PHE (Partial Homomorphic Encryption)
  - Searchable Encryption

    Applications:
    - Privacy-Enhanced Content-Based Searchable Information Retrieval
    - Multimodal Searchable Encryption
    - Privacy-Enhanced Cloud Storage and Computation Services

Outline

- **Applied Cryptography**
  - Typology: objectives and focus of different cryptographic methods and algorithms
  - Cryptographic constructions
  - Secure channels w/ cryptographic protection
  - Tools:
    - Java JCA/JCE for Programming w/ Cryptography
    - Tools in the Java Environment
    - Openssl library and the openssl tool

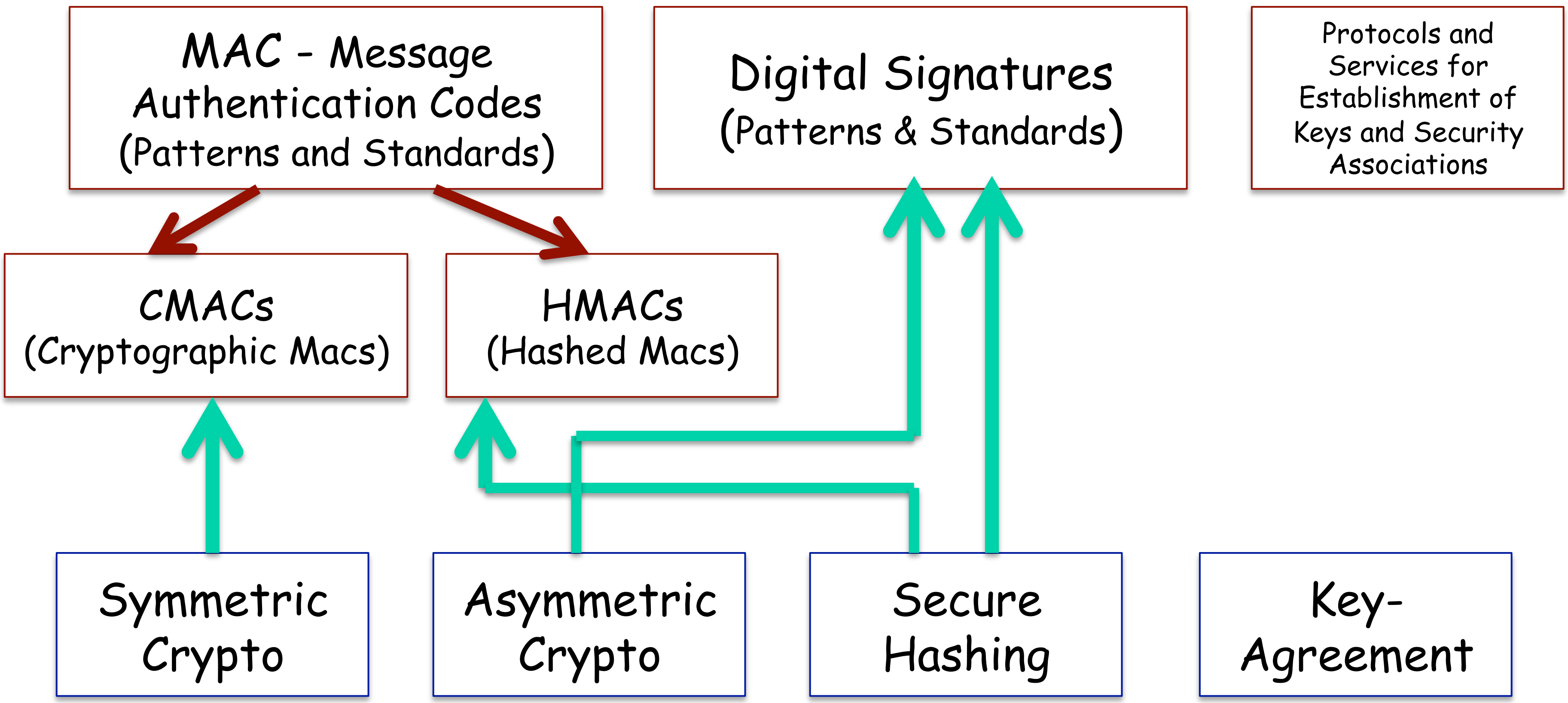
Computational Applied Cryptography Constructions (or Schemes)

- Conventional Base Cryptography:

Symmetric Crypto	Asymmetric Crypto	Secure Hashing	Key-Agreement

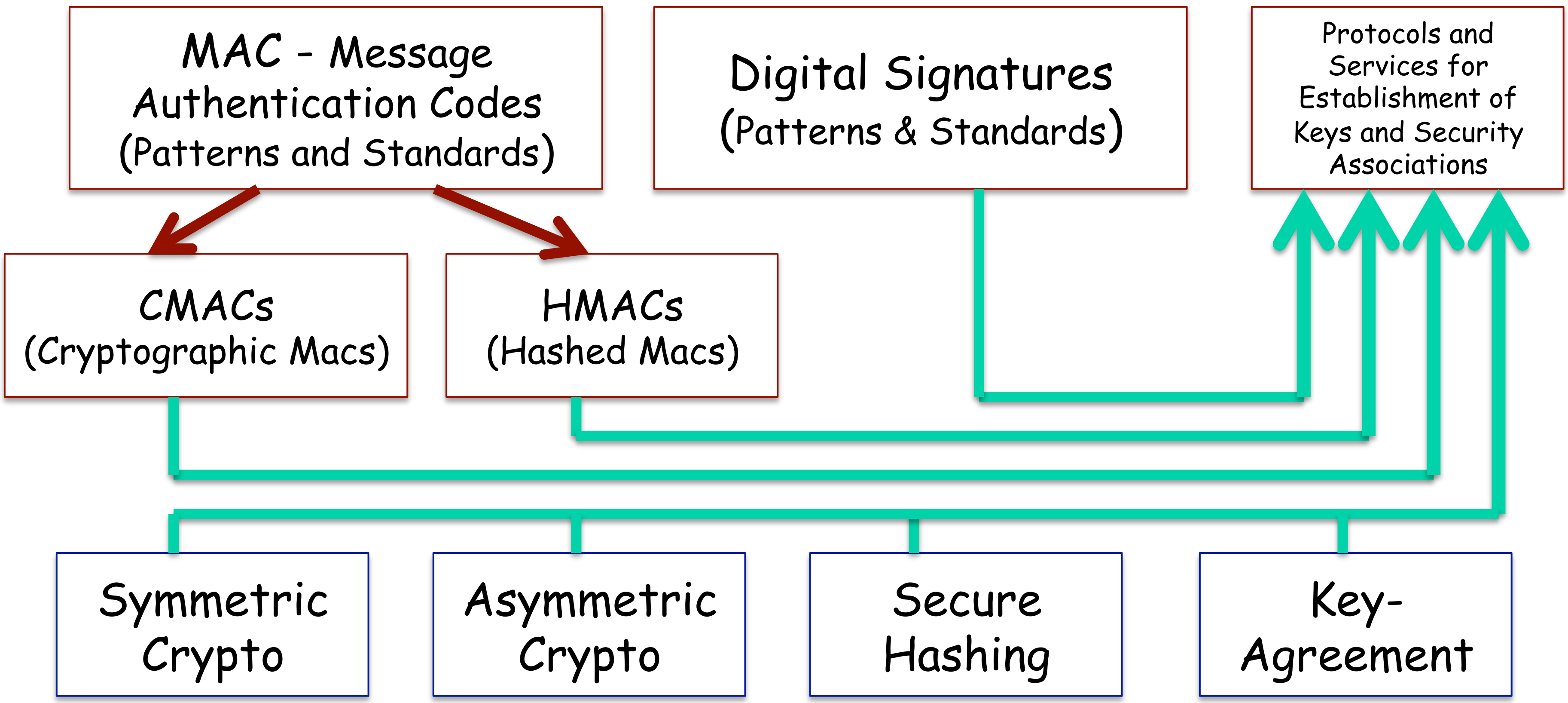
Hybrid Cryptographic Constructions

- Combination of Cryptographic Algorithms



Hybrid Cryptographic Constructions

- Protocols and Services for Secure Key-Distribution and Establishment of Security Association



Outline

- **Applied Cryptography**
  - Typology: objectives and focus of different cryptographic methods and algorithms
  - Cryptographic constructions
  - Secure channels w/ cryptographic protection
  - Tools:
    - Java JCA/JCE for Programming w/ Cryptography
    - Tools in the Java Environment
    - Openssl library and the openssl tool

Cryptographic Message Formats for Secure Channels

- Constructions with the combination of different methods in message multi-parts

Ex: A to B (Discussion)

Cleartext-Metadata || SIG || Secure Envelope || Confidental Data || Mess. Auth and Integrity || Fast Integrity Checks

Example:

Plaintext header w/ metadata	*: Public Metadata*
|| Sig (Info and Msg Data)$_{Sig\text{-}KprivA}$	*: Digital Siganture*
|| {secrecy params}$_{KpubB}$	*: Dist. of Symmetric Session Key*
|| {Msg plaintext data}$_{Ks}$	*: Ciphertext data w/ Symm. Encryption*
// can include Integrity Proof	*// can include Hash (Msg)*
|| MAC$_{Km1}$ (Msg plaintext Data)	*: HMACs or CMAC Cnstruction*
|| MACopt$_{Km2}$ (Ciphertext Data)	*: HMAC or CMAC Construction*

Outline

- **Applied Cryptography**
  - Typology: objectives and focus of different cryptographic methods and algorithms
  - Cryptographic constructions
  - Secure channels w/ cryptographic protection
  - Programming Tools

Tools, Practice, Hands-On (in our LABs)

- Java JCA/JCE for Programming w/ Cryptography
  - JCA / JCE Model and Framework
  - Tools, Algorithms, Prog. Techniques
  - Hands On Practice
- Cryptographic tools and demos

Programming w/ Crypto Algorithms and Methods:

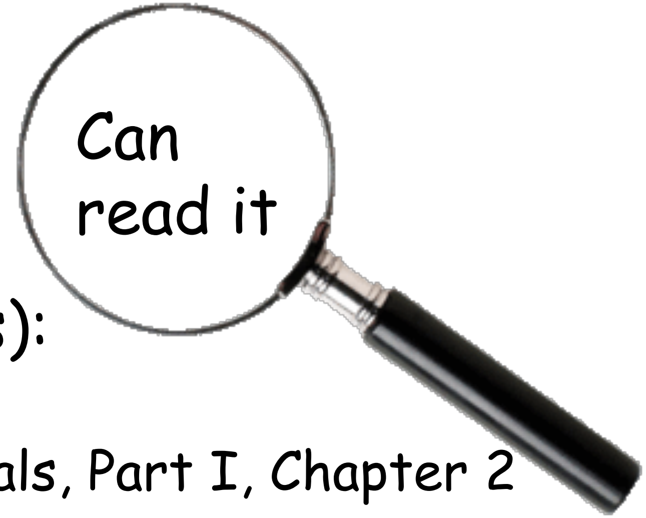
- Lab JCA/JCE, Setups and Prog. Model, Java Platform Policy Enforcements, and Programming w/ Crypto Providers
- Lab: JCA/JCE Symmetric Encryption (Block and Stream Ciphers), Block Modes, SAPs, Key Generation
- Lab: Secure Hash Functions and MACs (HMACs and CMACs)
- Lab: Public Key Crypto and Digital Signatures' Constructions
- Lab: Key-Exchange w/ Asymmetric Methods

Next Lecture Topic …

- Symmetric Encryption … in more detail ☺

For those interested: Optionally Suggested Readings

on Symmetric Encryption



Suggested Reading (study for tests):

- W. Stallings, Network Security Essentials, Part I, Chapter 2

If you want more about Classical Encryption Techniques (including classical methods, rotor machines, steganography):

- W. Stallings, L. Brown, Cryptography and Network Security – Principles and Practices, Part 2 – Symmetric Ciphers, Chap 3 – Classical Encryption Techniques
- Or see the first slides (can also talk with me for practical demos…)