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., 2020/2021

Applied Cryptography

Cryptographic Tools,
Methods, Techniques and Algorithms

Outline

- **Classic Cryptography**
- **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

Outline

- **Classic Cryptography**
- 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

Classic Cryptography

History / Origins
Ancient Methods: Classical Cryptography

Classic 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, and other** 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

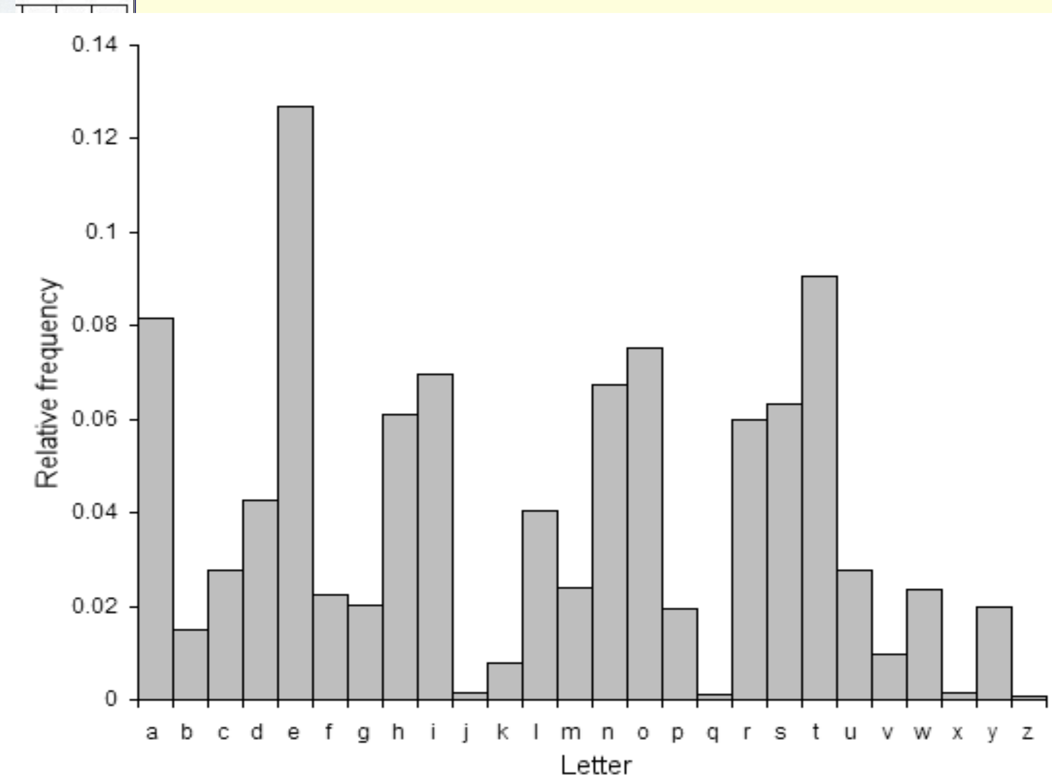
Secrecy is on the algorithm parameters (i.e., Cryptographic Keys)

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 Rivest, Adi Shamir and Len Adleman
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 …
(CAESER Cipher and many other 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

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

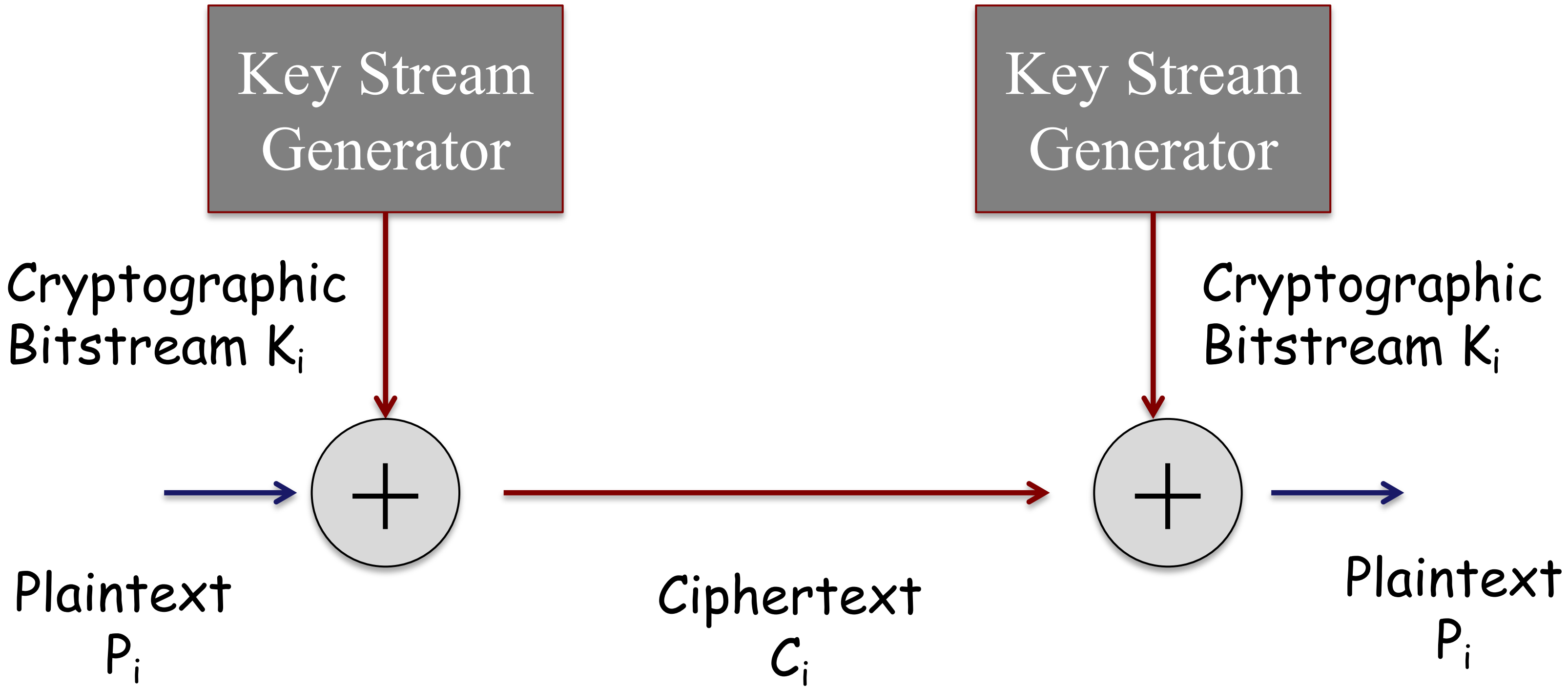
$$\begin{pmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{pmatrix} \begin{pmatrix} 15 \\ 14 \\ 7 \end{pmatrix} \equiv \begin{pmatrix} 260 \\ 574 \\ 539 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 2 \\ 19 \end{pmatrix} \pmod{26}$$

Plaintext:
ACT

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

J. Vernam, 1918

The base idea for symmetric stream ciphers !



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	$4\text{x}10^{26}/10^9 = 4\text{x}\ 10^{17}$ s $= 6.3 \times 10^9$ 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): $26! = 4\text{x}10^{26}$

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

Rotor Machines
Ex: Enigma Machine

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 Techniques

Steganography Techniques

- Hidden secret information, encoded in public/available information (concealing the existence of the hided information), Ex:
  - Secret (hidden) messages (text) in texts
  - Secret (hidden) messages (text) in images
  - Secret (hidden) messages (text) in sounds
  - Secret (hidden) messages (text) in movies

  - In general: secret media (any) hidden in media (any)

Outline

- **Classic Cryptography**
- **Applied Cryptography: Computational-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

X.800 framework – relevant mappings
- attack typology vs. services vs mechanisms
- the role of applied cryptographic tools as
  specific security mechanisms

Mappings in X.800 (remember from last lecture)

	Release of message contents	Traffic analysis	Masquerade	Replay	Modification of messages	Denial of service
Peer entity authentication			Y			
Data origin authentication			Y			
Access control			Y			
Confidentiality	Y					
Traffic flow confidentiality		Y				
Data integrity				Y	Y	
Non-repudiation			Y			
Availability						Y

	Release of message contents	Traffic analysis	Masquerade	Replay	Modification of messages	Denial of service
Encipherment	Y					
Digital signature			Y	Y	Y	
Access control	Y	Y	Y	Y		Y
Data integrity				Y	Y	
Authentication exchange	Y		Y	Y		Y
Traffic padding		Y				
Routing control	Y	Y				Y
Notarization			Y	Y	Y	

	Mechanism							
Service	Encipherment	Digital signature	Access control	Data integrity	Authentication exchange	Traffic padding	Routing control	Notarization
Peer entity authentication	Y	Y			Y			
Data origin authentication	Y	Y						
Access control			Y					
Confidentiality	Y						Y	
Traffic flow confidentiality	Y					Y	Y	
Data integrity	Y	Y		Y				
Non-repudiation		Y		Y				Y
Availability				Y	Y			

Cryptographic tools as base specific mechanisms

Service	Mechanism: Enciph-erment	Digital signature	Access control	Data integrity	Authenti-cation exchange	Traffic padding	Routing control	Notari-zation
Peer entity authentication	Y	Y			Y			
Data origin authentication	Y	Y						
Access control			Y					
Confidentiality	Y						Y	
Traffic flow confidentiality	Y					Y	Y	
Data integrity	Y	Y		Y				
Non-repudiation		Y		Y				Y
Availability				Y	Y			

Symmetric Crypto Methods

Asymmetric Crypto Methods

Secure Hash Functions, HMACs or CMACs

Authentication and Key Distribution Protocols

Cryptographic tools vs. X.800 framework

	Release of message contents	Traffic analysis	Masquerade	Replay	Modification of messages	Denial of service
Peer entity authentication			Y			
Data origin authentication			Y			
Access control			Y			
Confidentiality	Y					
Traffic flow confidentiality		Y				
Data integrity				Y	Y	
Non-repudiation			Y			
Availability						Y

Cryptography methods, Aalgorithms, models, techniques

	Release of message contents	Traffic analysis	Masquerade	Replay	Modification of messages	Denial of service
Encipherment	Y					
Digital signature			Y	Y	Y	
Access control	Y	Y	Y	Y		Y
Data integrity				Y	Y	
Authentication exchange	Y		Y	Y		Y
Traffic padding		Y				
Routing control	Y	Y				Y
Notarization			Y	Y	Y	

Service	Mechanism: Enciph-erment	Digital signature	Access control	Data integrity	Authenti-cation exchange	Traffic padding	Routing control	Notari-zation
Peer entity authentication	Y	Y			Y			
Data origin authentication	Y	Y						
Access control			Y					
Confidentiality	Y						Y	
Traffic flow confidentiality	Y					Y	Y	
Data integrity	Y	Y		Y				
Non-repudiation		Y		Y				Y
Availability				Y	Y			

Role of Cryptographic Tools: Use models, Methods, Techniques and Algorithms

Important:

Cryptography is very important for Computer Systems and Network Security ! ...

... **But it is not a PANACEA... Specific Tools are Specifically Targeted for Specific Properties !**

=> Ex., Must be correctly combined in cryptographic constructions, in designing secure channels

Cryptosystems: Algorithms and Methods

- Encryption: data blocks, messages — **Confidentiality**
  - Symmetric cryptosystems
    - Stream Ciphers vs. Block ciphers
  - Some asymmetric crypto systems (not all)

- Digital signatures: authentication of data blocks, messages — **Peer-Authentication**
  - Some asymmetric cryptosystems

- Message Authentication Codes — **Fast Data/Message Authentication**
  - Sometimes called "Lightweight" Signatures
  - MACs, HMACs or CMACs

- Secure Integrity Checks — **Integrity**
  - Secure Hash Functions

Outline

- **Classic Cryptography**
- **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

- Typology of Applied Cryptographic Methods, Models and Algorithms

Cryptography for Applied Computing

Sometimes mentioned as “modern” or “computational” cryptography

Classic ve. Computational cryptography: … big difference…

- **Classic Cryptography:**
  - Secrecy: the ALGORITHM …. => as secret (not known) processing and in the SETUP, PARAMETERES OR KEYS used by the algorithm !
- **Computational Cryptography (modern era):**
  - Algorithms are known, public, revealed, published, available for study by everybody … and processed by computers
    - the more studied and the more unsuccessful attempts to break the algorithm …. The safer the algorithm is !
  - Secrecy: only depend on KEYS
  - KEYS as used as secret parameters
    - If we use different keys for the same input … the effect of the "same" computation will work as a "random oracle"

Initial "mind setting" ...

- From what it is required, safe cryptographic algorithms are based on the assumption that the related computation problems must computationally hard, i.e, NP-complete.
  - In the unlikely event that someone proves that P=NP, these codes will break !
- For practical use: if we compute with correct keys or related secrecy parameters (as valid parameters), cryptographic operations are executed in polynomial time ... If not, to break the algorithm requires to solve a very hard problem not solvable in polynomial (computational) time
- So the question here is also on practical computing assumptions: computational possibilities and impossibilities
  - What physics do we need ? Is it feasible ?
  - How many time we need to break ? 10^13 years ?

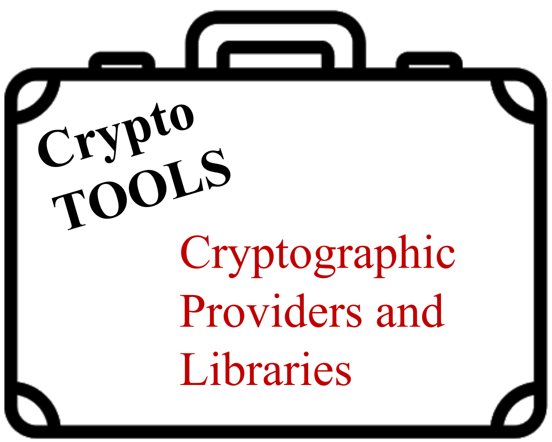
Computational Applied Cryptography: Base Types

- Typology of Applied Cryptography: Families of Methods. Algorithms and Techniques

Symmetric Crypto	Asymmetric Crypto	Key-Agreement	Secure Hashing

Asymmetric Model

All types can be provided in Crypto Providers and Libraries that can be used for programming with different programming languages



Computational Applied Cryptography

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

Symmetric Crypto

- Primary use for **Confidentiality** (Message-Data Encryption)
- But also usable for message/data authentication (**CMACs**) and **Key-Establishment Protocols based on KDCs**

Asymmetric Crypto

- Primary use for **Authentication**
- But also usable (some algorithms, not all)… for **Confidentiality**
- Also used for "Key-Distribution and Secure Establishment of Security Associations based on PKCs (or CAs)

Key-Agreement

- Primary use for **Key-Exchange**

(or establishment (agreement) of Keys and Security Association Parameters (SAs) 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
- Stream Ciphers

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 security properties, possibly different security guarantees

(Some) Examples ("*Some Kids in Town*")

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

Symmetric Crypto

Block Ciphers:
- AES
- Twofish
- RC6
- Serpent
- MARS
- Camelia
- IDEA
- Blowfish
- ChaCha-Poly
- DES
- 3DES
- Skypjack
- .... etc

Stream-Ciphers
- RC4
- A5...A8
- VEST
- SEAL
- SNOW
- Turing
- WAKE
- ... etc

Asymmetric Crypto

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

Sign. Only
- DSA or DSS
- ECDSA
- ... etc

Obs) All Asymmetric Methods process Blocks ...

Key-Agreement

- Diffie-Hellman
- GDH Variants
- ... etc

Obs) Asymmetric Crypto Constructions

Secure Hashing

- MD5
- SHA-1
- SHA-2
  - SHA-256
  - SHA-384
  - SHA-512
- SHA-3
  - ...
- RIPEMD
- WHIRLPOOL
- Etc ...

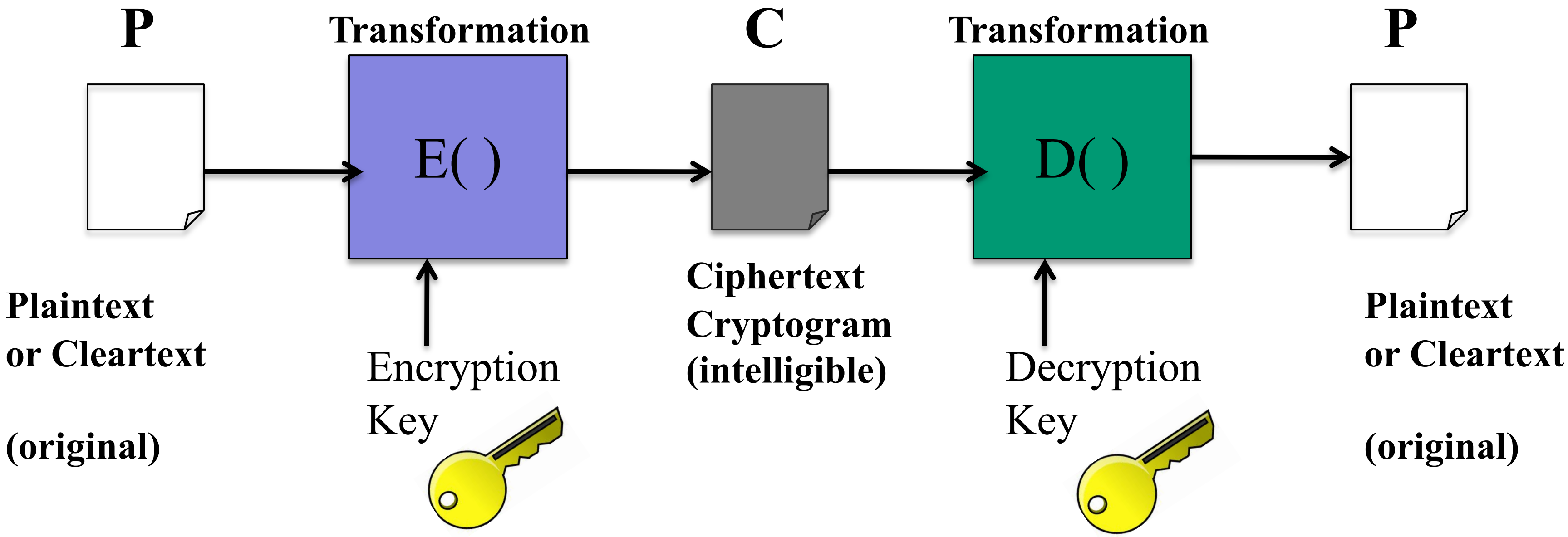
- Symmetric Cryptography

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

Use of Block Ciphers



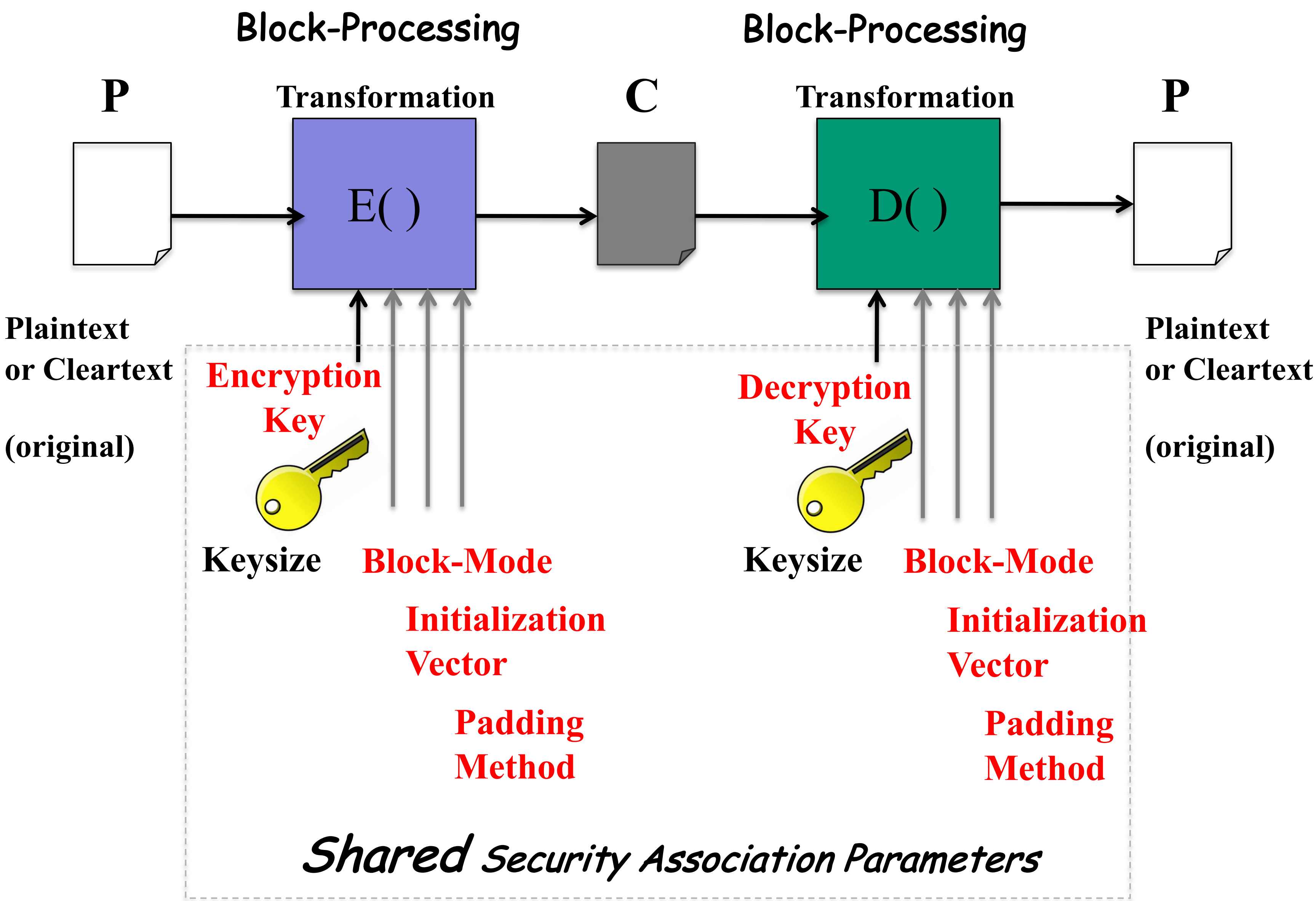
Notation:

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

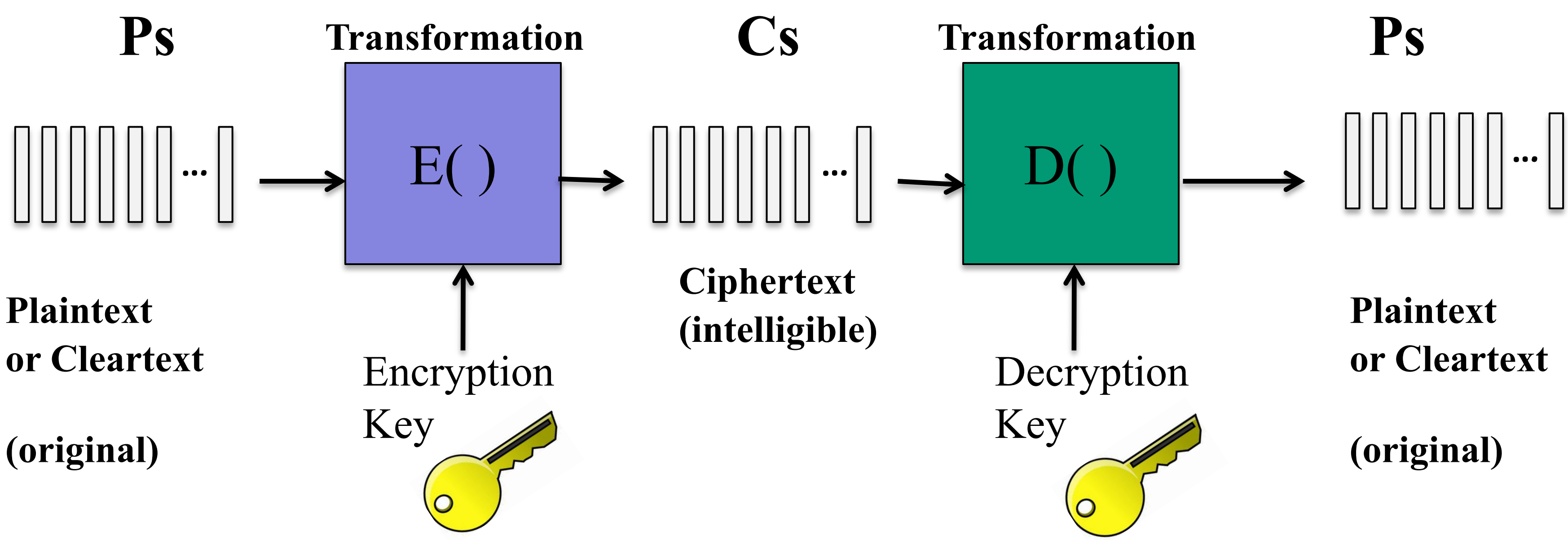
$P = \{C\}'_K$; $P = D(C, K)$; $P = D_K(C)$ // C decrypted with key K

SAME KEY (shared) for Encryption and Decryption
But E( ) and D( ) Functions are Different (inverse)

Symmetric Block Encryption and related Security Association Parameters

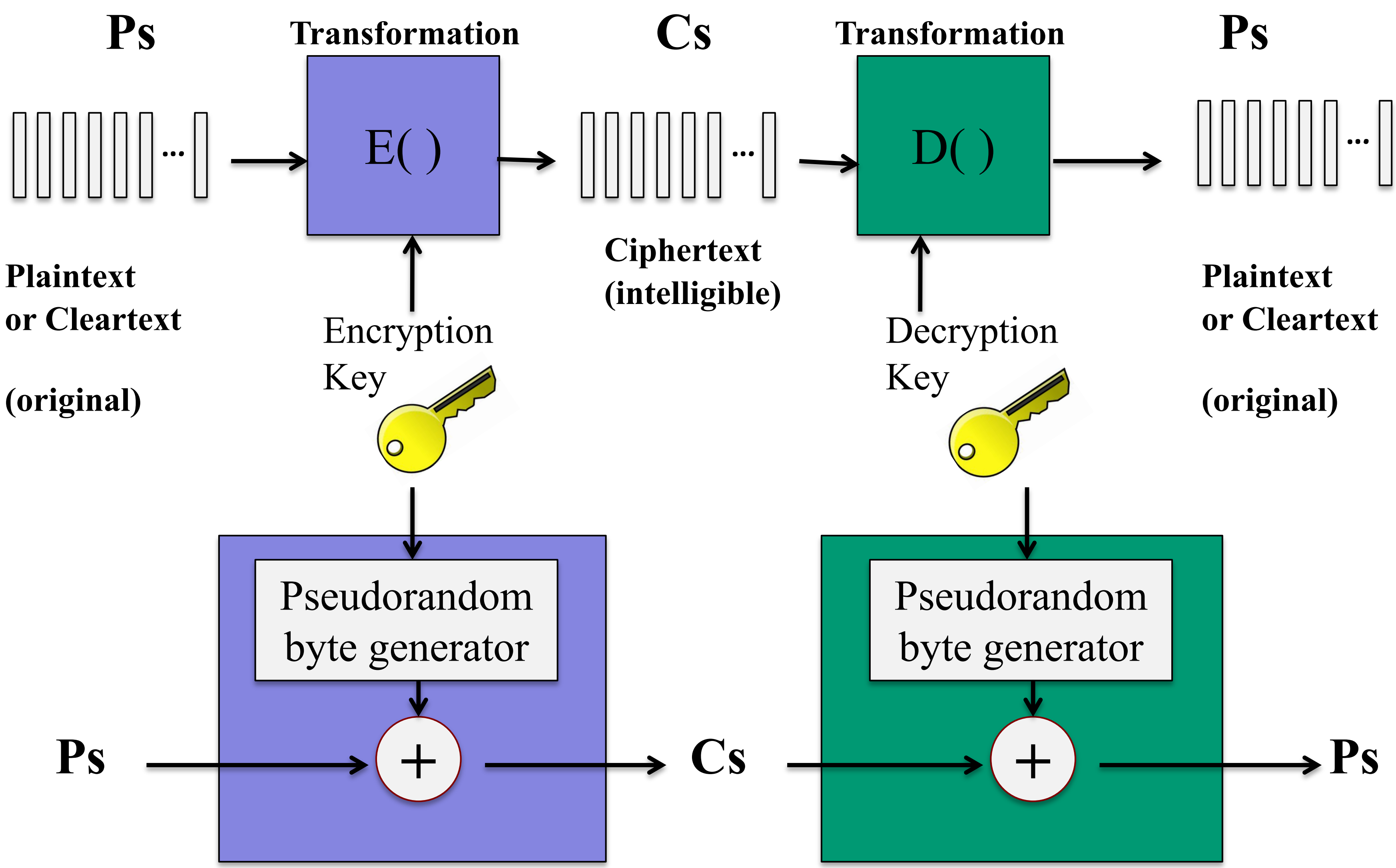


Use of Stream Ciphers



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

Stream Ciphers (Typical Structure)



Use of Symmetric 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

Robustness Criteria

- **Robustness against brute force attacks**
  - Impossibility to conduct brute-force attacks to break algorithms in useful computing time or with required computing physics
    - ⇒For current computers time complexity to break is not possible in Polynomial Time: $O(n^K)$ for some non-negative integer k, as bigger as possible [Ex., 10^{13} anos ….]
- **Robustness against cryptanalysis attacks (or studies)**, under different criteria in trying to break the key, considering the initial knowledge of the attacker (or cryptanalyst)

Cryptanalysis Criteria

Type	Previously Known
Ciphertext Only	Encryption Algorithm and Observed Ciphertext
Known Plaintext	Encryption Algorithm, Observed Cyphertext and one or more Plaintext-Ciphertext Pairs
Chosen Plaintext • Or IND-CPA / Indistinguishability under Chosen Plaintext Attack	Idem but plaintext chosen by the cryptanalyst
Chosen Ciphertext Based Cryptanalysis • Or IND-CCA-1, as well as, IND-CCA-2	Encryption Algorithm, Observed Cyphertext, one or more Purported Chosen Ciphertext, together with correspondent Plaintext
Chosen Text	Combination of everything above

More on symmetric encryption

- Practical use: on LABs (Java JCE Programming and Tools – ex., openssl)
- Also Important practical issues:
  - Block Ciphers:
    - Implications on the proper use of PADDING
    - Implications and relevant issues and tradeoffs in choosing proper MODES for different purposes !
    - Experimental observations in relevant tradeoffs;
      - Sizes of Plaintext vs. Ciphertext
      - Security Concerns
      - Performance Concerns
      - Reliability concerns

- Next Lecture on Symmetric Crytography: details and theoretical issues when using Symmetric Crypto Algorithms for Block Ciphers, Stream Ciphers and PWD-Based Encryption

- Asymmetric Cryptography

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

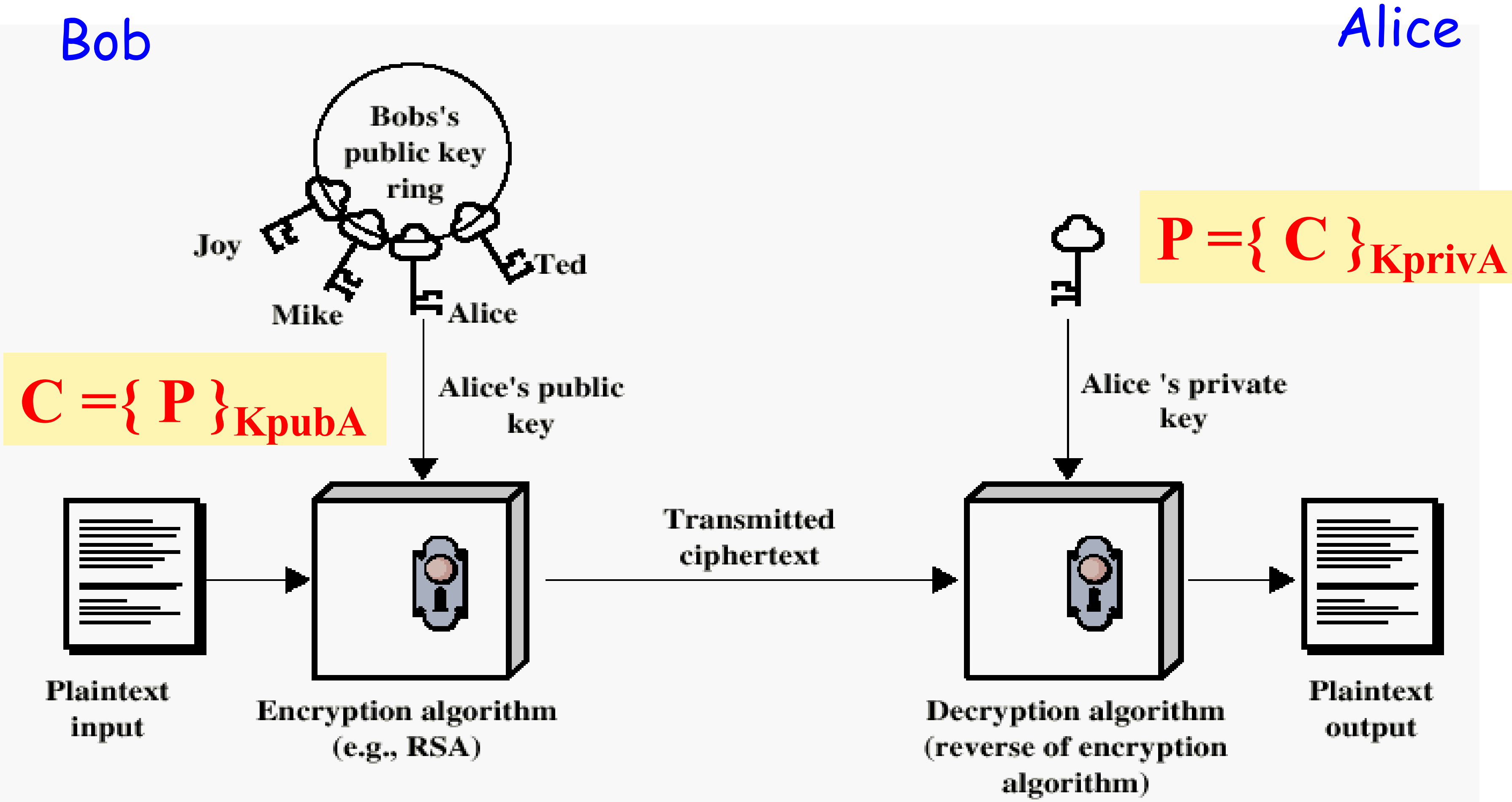
**) Sometimes not supported in specific algorithms and constructions*

DIFFERENT KEYS (Keypair) …. Same E( ), D( ) Functions … or E( ) and D( ) is the SAME COMPUTATION

Encryption using a Public-Key Crypto System

When used for **Confidentiality** principles:

Encryption with the destination Public key

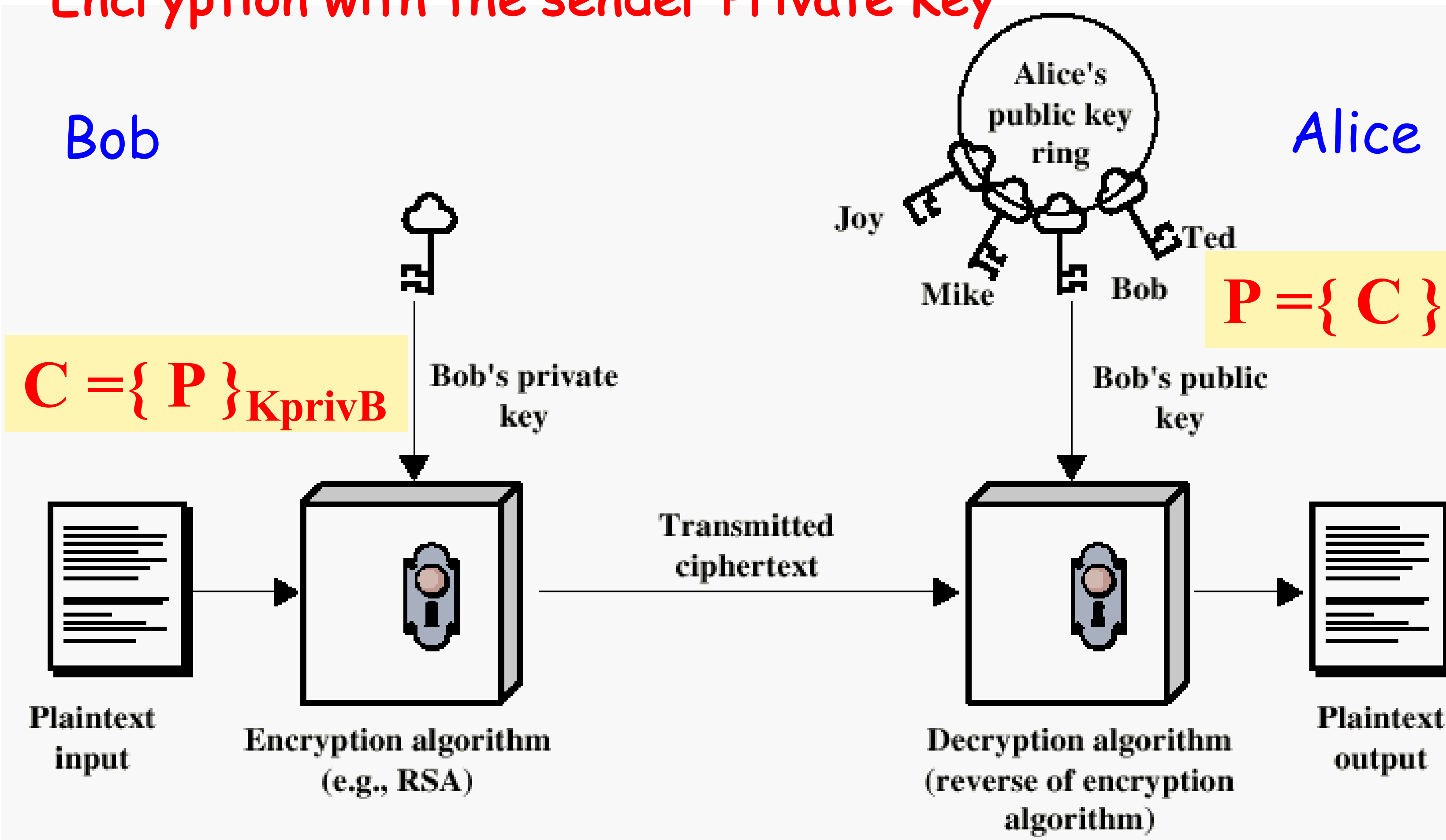


Asymmetric Crypto

Authentication using Public-Key Crypto System

When used for Authentication principles:

Encryption with the sender Private Key



$C = \{ P \}_{KprivB}$

$P = \{ C \}_{KpubB}$

Examples of Asymmetric Cryptography Algorithms

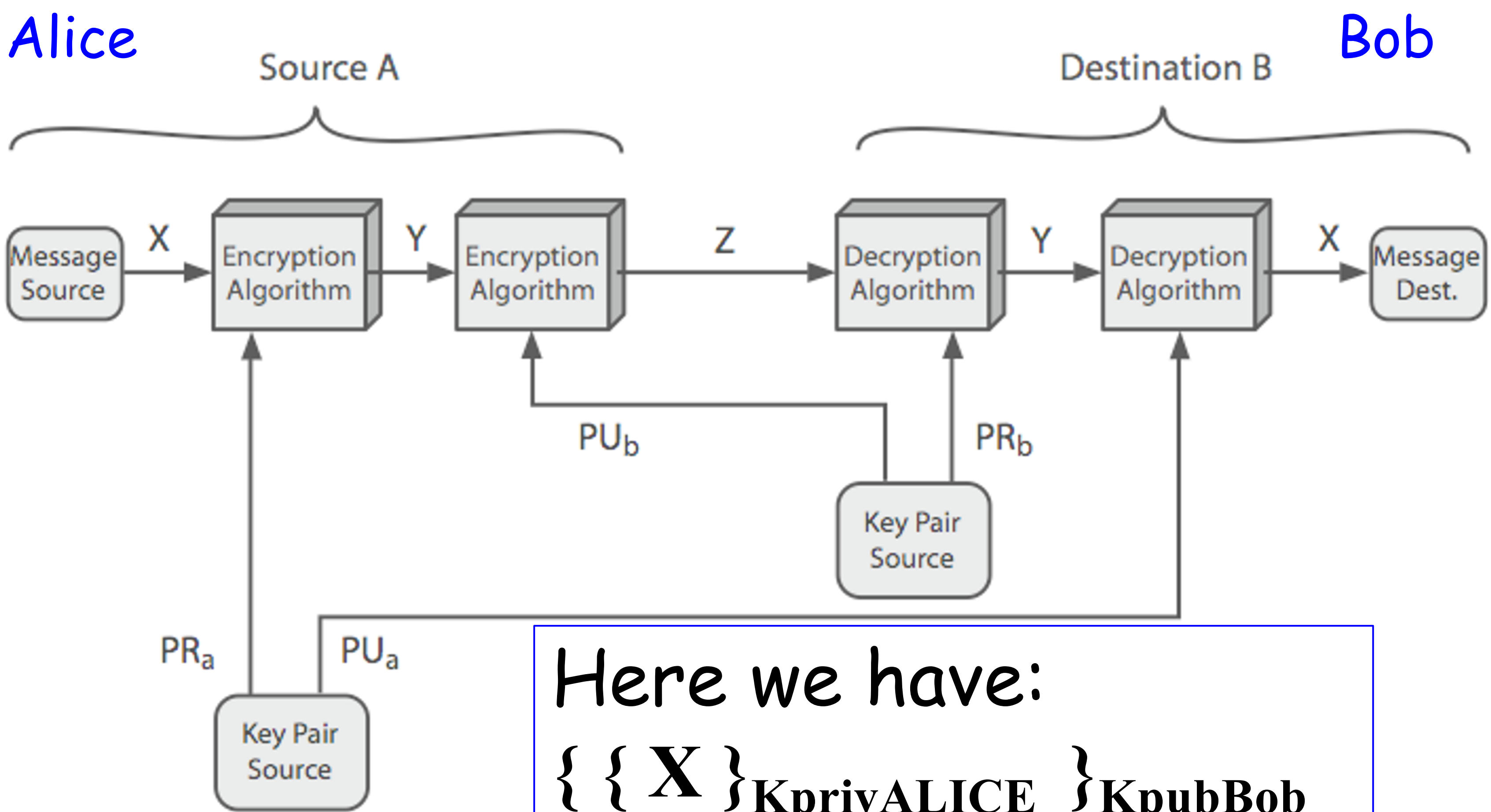
- **Usable for Authentication (Digital Signatures) and Confidentiality:**
  - RSA, EL Gammal, ECC Families or curves)

- **Usable only for Authentication (Digital Signatures' Constructions)**
  - DSA, ECDSA (w/ different ECC families or curves)

- **Usable only for Confidentiality**
  - Crammer-Shoup, Paillier,
  - Paillier, Goldwasser-Mical, Benaloh, ….

OBS: *Diffie Hellman is am Asymmetric-Crypto Algorithm but is not used for Authentciation nor Confidentiality. It is for Key-Exchange/Agreement Purposes*

Combined use for Confidentiality + Authentication

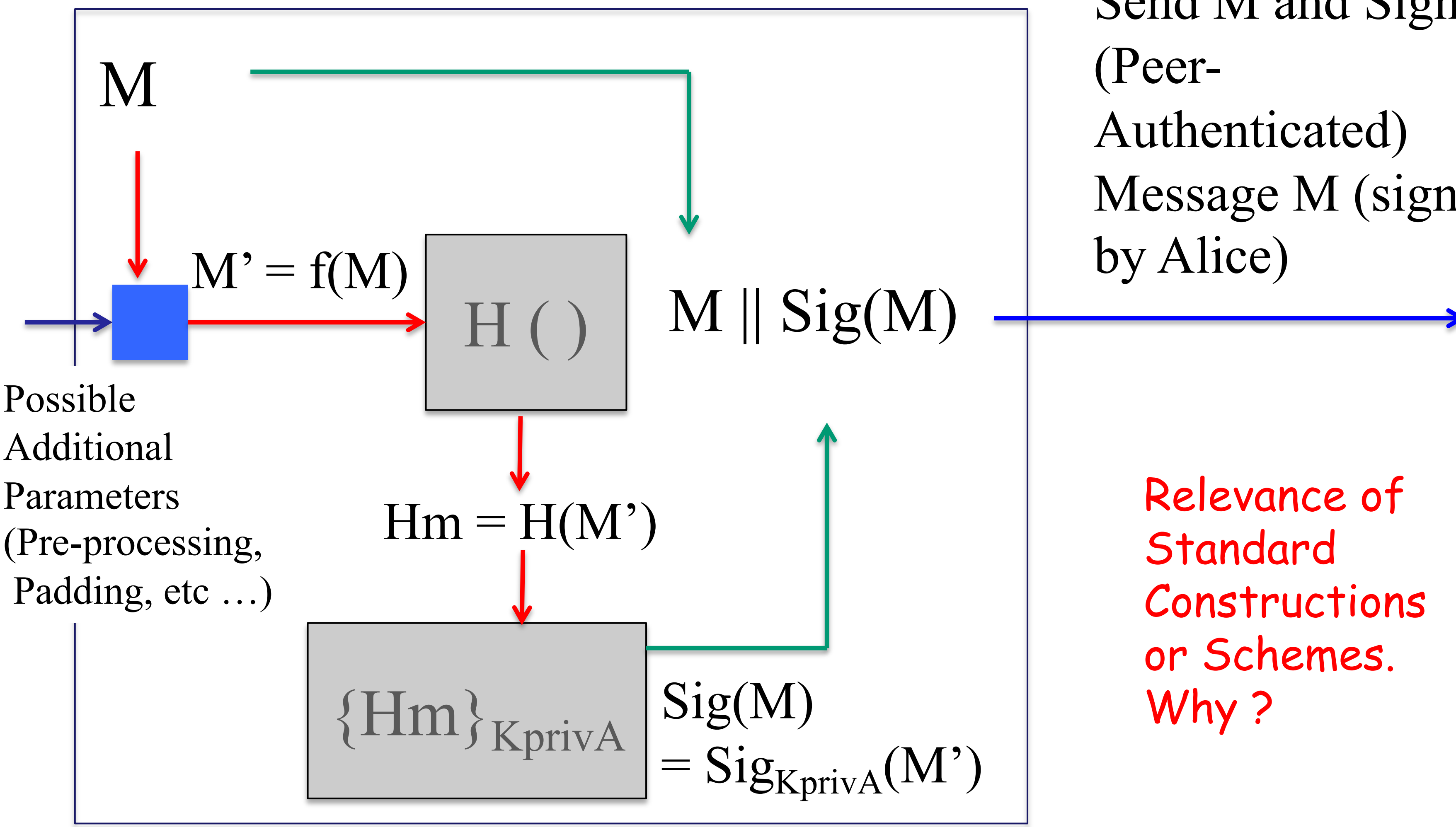


Here we have:

$$\{\ \{\ \mathbf{X}\ \}_{\mathbf{KprivALICE}}\ \}_{\mathbf{KpubBob}}$$

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

- Principle of construction of Digital Signatures Schemes:
  Sender (Alice: A)



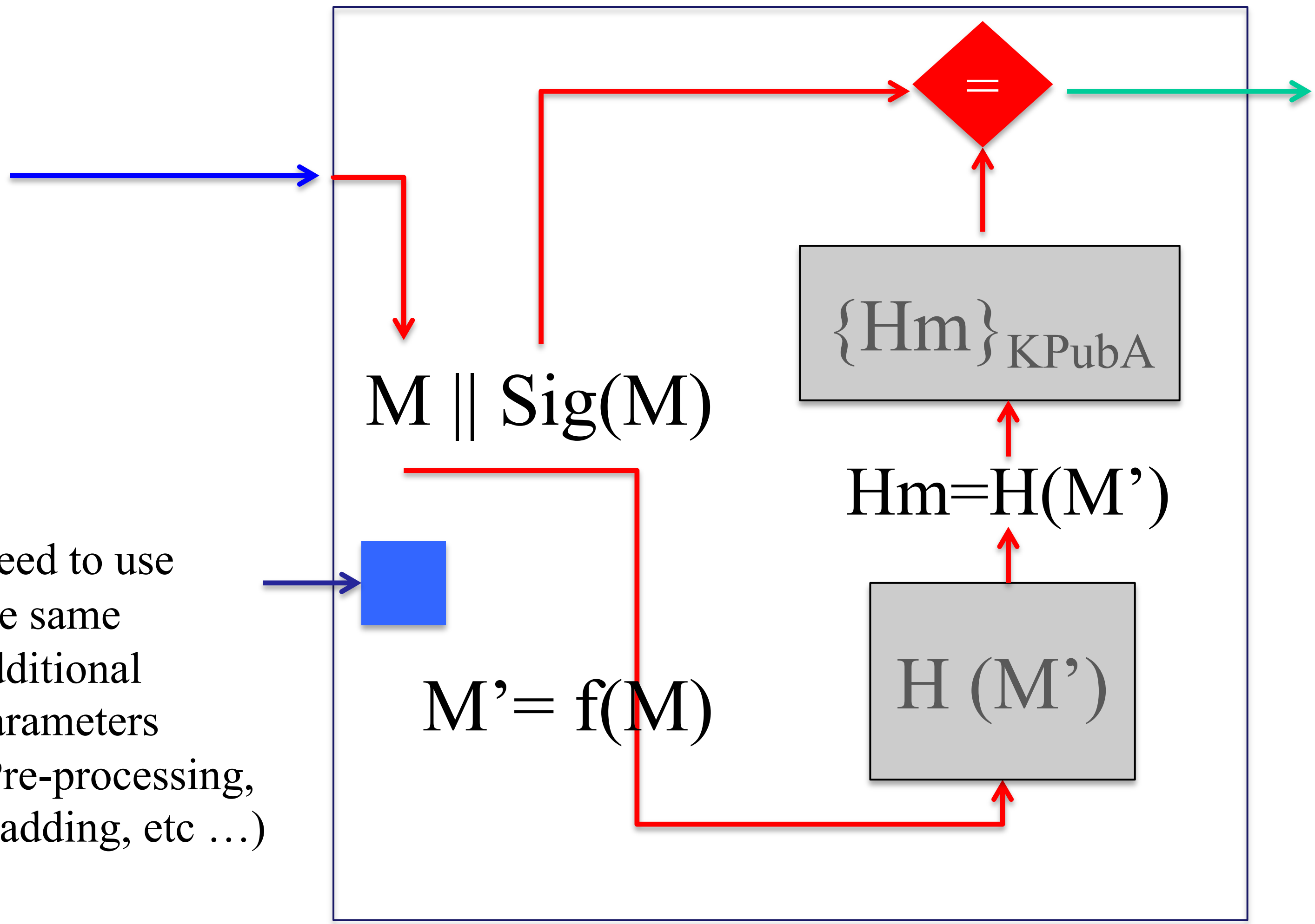
Send M and Signed (Peer-Authenticated) Message M (signed by Alice)

Relevance of Standard Constructions or Schemes. Why ?

Base Scheme for Authentication Proof

- Verification of Digital Signatures

If YES:



OK
Accept M.
Sig.
(verified)

If NO:
Reject M

Need to use the same additional parameters (Pre-processing, Padding, etc …)

Use of Asymmetric Cryptography

- **Robustness** (security and correctness of asymmetric cryptography algorithms on their trust-execution criteria)
  - Resistance against brute-force attacks and cryptanalysis attacks
- Need **security association parameters** for use (ex., pre-processing transformation, padding scheme and the same constructions: hashing, digital signature algorithm)
- Need **strong keys**: keypair generation w/ randomness
- Trust distribution of public keys: association between public keys and the correct principals – Certification of Public Keys
  - Possible use of HSMs, Smartcards, Crypto-Tokens
- Need of **secure public key-distribution and establishment protocols and services under certification guarantees**
- **Need protocols and services for Public-Key Revocation and Status Verification & management**

More on Asymmetric Cryptography

- Practical use: on LABs (Java JCE Programming and Tools – ex., openssl)
- Also Important practical issues:
  - For Encryption and for "Standard" Digital Signatures' Constructions (Signing/Verifying Operations)
    - Implications on the proper use of PADDING in the case of Asymmetric Cryptography – very different purpose when compared with Symmetric Block Encryption
    - Experimental observations in relevant tradeoffs;
      - Sizes of Plaintext vs. Ciphertext
      - Security Concerns
      - Performance Concerns
      - Integrity Concerns
- In a Lecture on Asymmetric Crytography: details and theoretical issues involved in adopting different algorithms: RSA, DSA, ECC-Curves and Key-Exchange (DH and Group-Based DH)

- Secure Hash Functions

Properties of Secure Hash Functions
(Also refereed as Secure One-Way Hash Functions)

$h = H(M)$ // h result is the hash value of input M

- Input (block M) can have any size*

  **) Typically M must have a maximum size for each specific hash function*
- The produced output *h* has always a fixed size
- H(*M*) is relatively easy to compute for any given *M*, with both hardware and software implementations practical.
- Irreversibility (or pre-image resistance):
  - from *h* is computationally infeasible to obtain *M* (One-Way Hash Functions)
- Collision-Resistance (or Collision-Free) :
  - Second-Image Resistance (also referred as weak collision resistance): Given M1, it is impossible to find M2≠ M1 such that h=H(M2)=H(M1)
  - Strong collision resistance: Impossible to find any pair X1, X2, such that H(X1) = H(X2)

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 ?

More on Secure Hash Functions

- Practical use: on LABs (Java JCE Programming and Tools – ex., openssl)
- Also Important practical issues:
  - Use of Secure Hash-Functions
    - Experimental observations in relevant tradeoffs;
      - Sizes of Hash-Values
      - Security Concerns
      - Performance Concerns
      - Integrity-Checking Concerns
- In a Lecture on Secure Hash: details and theoretical issues involved in adopting different algorithms

- Emergent Cryptography

Beyond the current conventional applied cryptography

Is it a Kind of Magic ???

Just Informative …
Details not covered in the CNSS Course

LBE and Functional Cryptography

New Arithmetic Constructions and Emergent and Post-Quantum Cryptography (Some examples from recent research):

- **Lattice-Based Cryptography (Post-Quantum) and important constructs (ex.,):**
  - Applications for ZKP (Zero Knowledge Proofs), IBE (Identity Based Encryption) and similar requirements as the conventional encryption methods but …
    - COMPLEXITY TIMES and SECURITY PROPERTIES for POST-QUANTUM Computing

- **Identity-Based Cryptography**
  - **Identity Elements and Features used as Public Keys in Special Public-Key Cryptograhic Methods**

- **Functional Encryption Algorithms and Methods**

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

- **Homomorphic Encryption Alg. And Methods**
  - FHE (Fully Homomorphic Encryption)
  - PHE (Partial Homomorphic Encryption)

- **Searchable Encryption (can also relate to Practical PHE Methods)**

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

Queryable Encryption for Content-Based Inormation Retrieval

New Arithmetic Constructions and Emergent and Post-Quantum Cryptography (Some examples from recent research):

- Queryable Content-Based Encryption for Privacy-Preserving Information Retrieval (for Encrypted Databases and Encrypted NoSQL Repositories or Key-Value Stores
  - Text-Only, Unstructured Data
    - Ex., Search/Index operations on encrypted unstructured data repositories
  - Text-Only, Structured Data (ex., Graphs, Trees, etc)
    - Ex., Support of Operations directly done on encrypted graphs
    - Ex., Ranking Algorithms directly running on encrypted documents
    - Ex., Encrypted SQL constructions of Encrypted Databases
  - Media-Contents (Privacy-Preserving Multimodal Content Based Information Retrieval or CBIR)
    - Ex., Support of Operations (ex., SEARCHES) directly done on ENCRYPTED IMAGE FORMATs
    - Ex., Given a Repository of Encrypted Images, Search an image that is Similar to

Outline

- **Classic Cryptography**
- **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

- Composition of Cryptographic Constructions

  … In practice we need to combine all the different cryptographic methods. Why ?

What about performance (and computational cost… and energy consumption) of different methods ?

- Easy to "feel" in practice …
- See, ex::

  $openssl speed

What suggests the performance and specialization of different cryptographic Methods ?

- Practical observation:

Hash = H(input)

Symmetric Crypto
Encryption/Decryption
Stream Ciphers

Symmetric Crypto:
Encryption/Decryption
Block Ciphers

x 10^3 to … 10^6 … and more

Asymmetric Crypto

Computational Applied Cryptography Constructions (or Schemes)

- For Confidentiality, Authenticity, Integrity, and Key-Establishment

Symmetric Crypto	Asymmetric Crypto	Secure Hashing	Key-Agreement

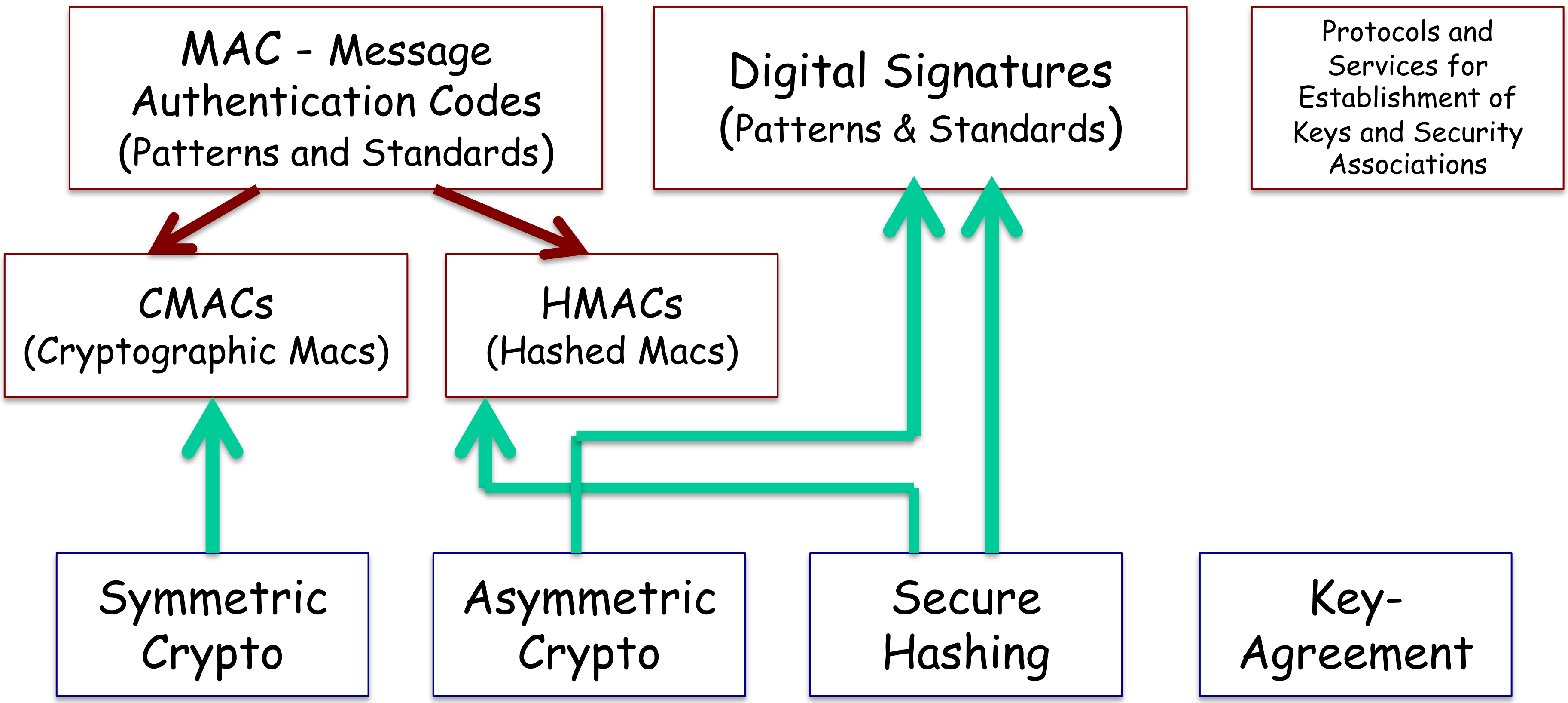
Alice to Bob:

$$\{ Ks \}_{KpubBOB} \mid\mid \{M, H(M)\}_{Ks} \mid\mid Sig_{KprivALICE} (M)$$

$$\underbrace{\{ Ks, Km \}_{KpubBOB} \mid\mid \{M, H(M)\}_{Ks} \mid\mid Sig_{KprivALICE} (M)}_{X} \mid\mid MAC_{Km} (X)$$

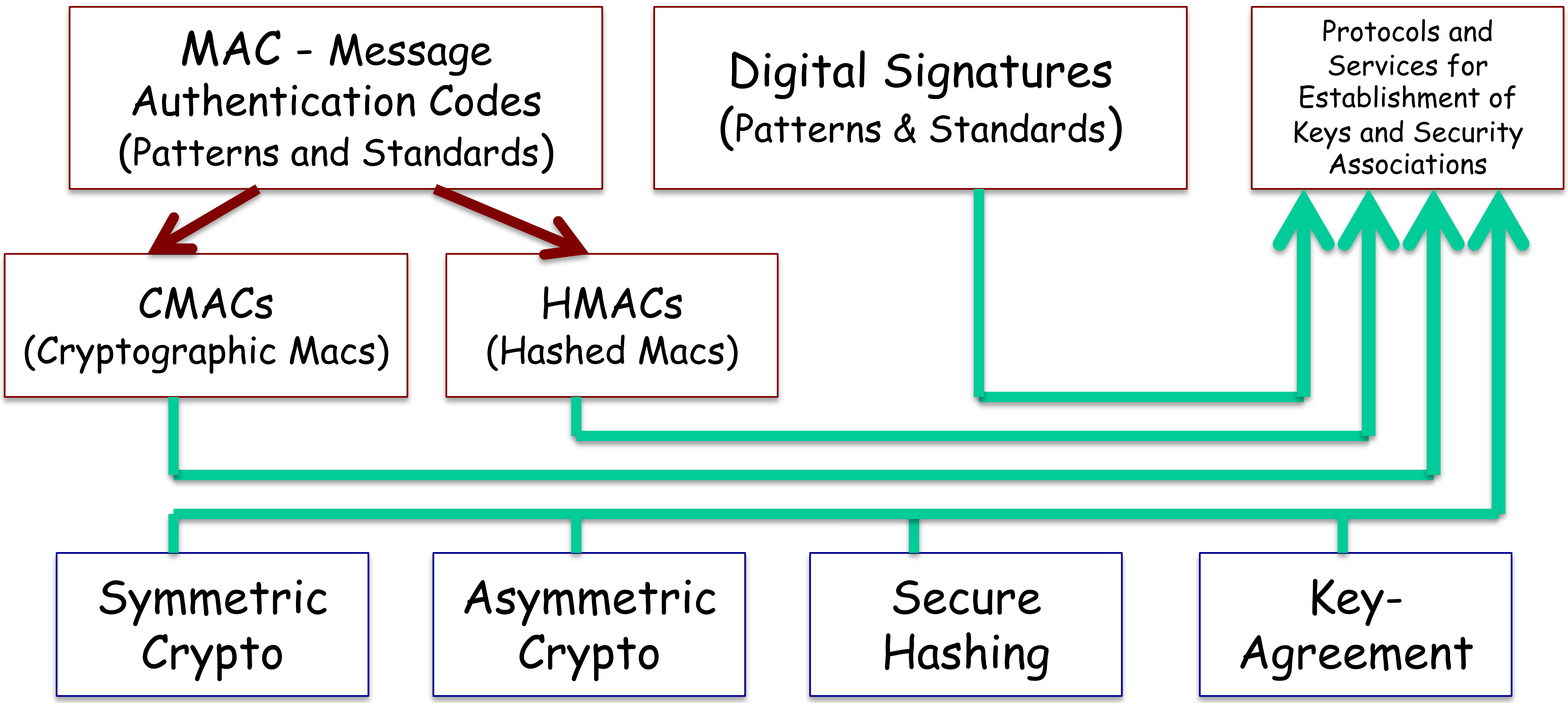
Hybrid Cryptographic Constructions

- Combination of Cryptographic Algorithms



Hybrid Cryptographic Constructions

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



Outline

- **Classic Cryptography**
- **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: metadata, version	*: Public Metadata*
|| Sig (Info, Msg-Data, M)$_{Sig\text{-}KprivA}$	*: Digital Siganture*
|| {Ks, Secrecy Params}$_{KpubB}$	*: Dist. of Symmetric Session Key*
|| {M, H(M)}$_{Ks}$	*: Ciphertext*
|| MAC$_{Km1}$ (Msg plaintext Data)	*: HMACs or CMAC Cnstruction*
… or	*… or*
|| *MACopt*$_{Km2}$ (Ciphertext Data)	*: HMAC or CMAC Construction*

Outline

- **Classic Cryptography**
- **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

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

Practical Considerations … (see in LABs)

Specific symmetric crypto algorithms for block-ciphers

- **Use:**
  - Fixed input Block sizes
  - Valid Key-Sizes
    - Need Secure Key Generators to avoid weak keys
- **Can operate by processing in different modes:**
  - Ex., ECB, CBC, OFB. CFB. CTR, …
- **Can use modes with implicit integrity checks:**
  - Ex., CCM, GCM, …
- Depending on the mode and information to be encrypted/decrypted… we need to use padding schemes:
  - Ex., PKCS#5, PKCS#7, …

How, When, Why and What we must chose
For these different parameterizations ?

Practical Considerations … (see in LABs)

Specific Asymmetric crypto algorithms

- **Uses:**
  - Variable Block sizes
  - Valid Key-Sizes (long, > 1024, 2048 bits)
    - Need Secure Key-Pair Generators to avoid weak keys
- **Not all are used for the same purposes**
  - Digital Signatures Constructions for Privacy or implicit Integrity
  - Encryption constructions for Confidentiality
- Depending on the purpose and information involved, … we need or not to use padding schemes:
  - Ex., PKCS#1, PSS (ex. RSA-Based Signatures)
  - Ex., OAEP (ex., RSA-Based encryption)
  - Ex., NoPadding (ex., DSA, ECDSA Signatures

How, When, Why and What we must chose
For these different parameterizations ?

Practical Considerations … (see in LABs)

Different and specific Hash-Functions, and MAC Constructions

- Operate with:
  - Limited or unlimited input information
  - Have different and specific security guarantees
  - Compute hash-values with different sizes
- Can be used as "Unkeyed" or as Keyed" hash values
  - Unkeyed when used for Message/Data Integrity purposes
  - Keyed when used for Message/Data Integrity and Authenticity purposes
    - This us used as HMAC Constructions (Hash-Based Message Authentication Codes)

How, When, Why and What we must chose
For these different parameterizations ?

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