



Introduction to Cryptography

Gerardo Pelosi

Dipartimento di Elettronica e Informazione
Politecnico di Milano



Outline

- The general goals of cryptographic systems
- History of Cryptography and systematization of cipher design principles
- Categories of cryptographic algorithms
 - Symmetric
 - Asymmetric
 - public key: example of RSA; digital signature, certificates and PKI
 - identity based cryptography
- Hashing functions
- Implementation of cryptographic algorithms
- Categories of attacks



The General Goals of Cryptography

Cryptography is a set of methodologies and techniques for protecting information, aimed at giving a communication channel the following features:

- Confidentiality (or Secrecy)
- Data Integrity
- Authentication (or User Verification)
- Non-Repudiation



Security Services

- **Confidentiality:**
assuring that only authorized parties are able to understand the data.
- **Integrity:**
ensuring that when a message is sent over a network, the message that arrives is the same as the message that was originally sent.
- **Authentication:**
ensuring that whoever supplies or accesses sensitive data is an authorized party.
- **Non repudiation:**
ensuring that the intended recipient actually received the message & ensuring that the sender actually sent the message.



More on Confidentiality

- Confidentiality means that only authorized parties are able to understand the data (authorized from the perspective of the party that encrypted the data).
- It is okay if unauthorized parties know that there is data. It is even okay if they copy the data, as long as they cannot understand it.



More on Authentication

- How can we know that a party that provides us with sensitive data is an **authenticated** party?
- How can we know that the party that is accessing sensitive data is an **authorized** party?
- Possible solutions are:
 - Digital signatures
 - Passwords
 - Symmetric encryption + hashing



More on Integrity

- o This involves ensuring that when a message (i.e. any kind of data) is sent over a network, the data that arrives to the recipient is the same as the data that was originally sent. It is important that the data has not been tampered with.
- o Cryptographic solutions include:
 - Encryption and/or Hashing algorithms



More on *Non Repudiation*

- Ensuring that the intended recipient actually got the message.
- Ensuring that the alleged sender actually sent the message.
- How do we prove that a user's cryptographic credentials have not been compromised?
- Disputes arise when an alleged sender denies that certain actions were taken. For example, one entity may authorize the purchase of property by another entity and later deny such authorization was granted.
- A procedure involving a Trusted Third Party is needed to resolve the dispute.



Basic Terms

- **Encryption:** scrambling a message or data using a specialized cryptographic algorithm.
- **Plaintext:** the message or data before it gets encrypted.
- **Ciphertext:** the encrypted (scrambled) version of the message.
- **Cipher:** the algorithm that does the encryption.
- **Decryption:** the process of converting ciphertext back to the original plaintext.
- **Cryptanalysis:** the science of *breaking* cryptographic algorithms.



History of Cryptography

- Rather non-systematic approach during Ancient Times and Middle Ages:
 - Cesar cipher (it is a substitution cipher):
 - Shift letters of the *plaintext* message 3 alphabetic positions on.
- A systematic approach to cryptography begins during the Renaissance:
 - Discovery of frequency analysis.
 - Disadvantage of Cesar cipher: it does not smooth out frequencies in the *ciphertext*.
 - Introduction of **polyalphabetic ciphers**: use multiple cipher alphabets.
 - First non-trivial permutation algorithms.
 - the positions of the letters of the *plaintext* are scrambled in such a way to alter the ordering and neighbourhood characteristics of the original letters.
 - The shape of the frequency spectrum of the original *plaintext* is not altered.



Example of Substitution Scheme

- Let us try to abstract from unnecessary details, and to understand the underlying mechanisms:
 - Use only two symbols: 0 and 1 (bits)
 - Use only two substitution schemes:
 - $0 \rightarrow 0$ and $1 \rightarrow 1$ (scheme **a**) - IDENTITY
 - $0 \rightarrow 1$ and $1 \rightarrow 0$ (scheme **b**) - NEGATION
(the only possibilities with one bit!)
- Decide that the application order is:
abbabaaaab (just an example).



Example (cont.)

- Encryption example:
 - 0001000100 (*plaintext*: **0**: 80%, **1**: 20%)
 - *abbabaaaab* (application order)
 - 0111100101 (*ciphertext*: **0**: 40%, **1**: 60%)
- The frequency spectrum of the *ciphertext* has been smoothed, with respect to that of the original *plaintext*.
- In principle, the *ciphertext* has been made relatively difficult to relate to the original *plaintext*.
- But it is necessary to remember the application order *abbabaaaab* ... an example of what is meant with a "key".



Example (cont.)

- It is however easy to decrypt, if the application order is known:
 - 0111100101 (*ciphertext*)
 - *abbabaaaab* (application order)
 - 0001000100 (*plaintext*)
- Note that, posing $a = 0$, $b = 1$, encryption and decryption consist of XORing bitwise the bit sequences:

Decryption		Encryption	
● 0111100101	(cipher)	0001000100	(plain)
● <i>0110100001</i>	(applic.)	<i>0110100001</i>	(applic.)
● 0001000100	(plain)	0111100101	(cipher)



Keys

- A **“key”** is a parameter of the cryptographic algorithm, that is necessary to know for reconstructing the original *plaintext*.
- The key avoids the necessity of many different cryptographic algorithms, limiting to just one.
- **The key gives a substantial contribution to making cryptanalysis difficult.**
- However, the key must be kept secret to everybody but to the authorized users.



Example (cont.)

- Encryption example:
 - 0001000100 (plain text)
 - 0101111011 (key – parameter of the algorithm)
 - 0100111111 (cipher text)
- The key parametrizes the algorithm.
- Note how the encryption and decryption algorithms are identical, and share the same key (example of a “symmetric” system).
- This is not a necessity, in general, though many cryptographic algorithms are constructed precisely in this way.



History of Cryptography: Kerckhoffs' Principle

A cryptosystem should be secure even if everything about the system, except the key, is publicly known.

Kerckhoffs' desiderata (1883)

1. The system must be practically, if not mathematically, indecipherable;
2. It must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience;
3. Its key must be communicable and retainable without the help of written notes, and changeable or modifiable at the will of the correspondents;
4. It must be applicable to telegraphic correspondence;
5. It must be portable, and its usage and function must not require the concurrence of several people;
6. The system must be easy to use, requiring neither mental strain nor the knowledge of a long series of rules to observe.



Design Principles of Cryptographic Algorithms

- The security of every cryptographic system must not rely upon to the secrecy of the methodology (i.e., of the cryptographic algorithm) but only in the secrecy of the chosen key.
 - It can be theoretically studied and tested, lowering the risk of containing hidden weaknesses and pitfalls.
 - The encryption/decryption device can be efficiently implemented and fabricated in large volumes.
 - Secrecy is limited only to the small amount of information represented by the key.
- Viability of an encryption scheme:
 - The cost of breaking exceeds the value of the encrypted information
 - The time required to break the cipher exceeds the useful lifetime of the information



Composition of Algorithms

- In principle, one may combine several different cryptosystems, in order to obtain a more resistant cryptosystem.
- However, there are practical limitations of **time**, **space** and **complexity**.
- And it may happen that the composite system is not really more robust (against cryptanalysis) than its components, taken individually.



Levels of Security

Three categories of security are distinguished:

Unconditional security:

- When a cryptosystem cannot be broken (inverted) even when infinite computational resources are available [highly theoretic claim].

E.g.: One-Time-Pad (OTP), see next...

Computational security:

- When for a given cryptosystem the most efficient algorithm for breaking (inverting) it requires N operations (with N large and known in advance). Concepts and tools of the theory of computational complexity classify the systems as $\Omega(N)$ (N is a "lower bound")

Relative (or practical) security:

- see next...



Relative (practical) Security

- A cryptosystem is “relatively secure” (or secure from a practical point of view) if the computational cost of its breaking (inversion) is bound to that of a well-studied, very hard to solve problem.
- This classification criterion is not ultimate, in general.
- A cryptosystem might turn from secure to insecure:
 - Because of new methods to approach the “hard” problems.
 - Because of new ways to bound the same cryptosystem to another problem which results to be computationally simpler.



The Best Cryptosystem: *One-Time-Pad*

- The most resistant (symmetric) cryptosystem (against frequency analysis) consists of:
 - Mapping the *plaintext* onto a string of $m \geq 1$ bits (using some conversion technique -- which is not relevant by itself).
 - XORing the string with a key of $m \geq 1$ randomly chosen bits.
- Property of the Key:
 - its length must be equal to the length of the *plaintext*
 - it must be random
 - it must be used only once



One Time Pad

- OTP is **unconditionally secure** (Shannon 1948). Informally, if a cryptanalyst has a *ciphertext* string $(C_1 C_2 \dots C_t)$ encrypted using a random key string which has been used only once, the cryptanalyst can do no better than guess at the *plaintext* being any binary string of length t . (i.e., t -bit strings are equally likely as plaintext.)
- The severe condition on the property of the keys reduces the practicality of the system in all but a few specialized situations.
 - Reportedly, about 20 years ago communication line between Moscow and Washington was secured by a one-time pad. Transport of the key-pad was done by trusted courier.



Attempts to improve OTP

- To reduce the inefficiency of the One-Time-Pad cryptosystem, while preserving its robustness, one might for instance:
 - use a periodic key, with a sufficiently long period.
 - use some kind of key generator, working in a more or less random way.
 - combine the two above methods.
 - and many others ...
- But in all cases, the robustness of the cryptosystem is affected, since the key is no longer truly random ...
- However, the idea of “recycling” or “expanding” the key recurs in modern (symmetric) cryptosystems, though in a much more sophisticated way ...



Design Principles: Confusion & Diffusion

Confusion is the ability to alter (confuse) the letters of the original *plaintext*, decoupling the frequency spectra of the plain- and cipher- texts and making the relationship between the *key* and *ciphertext* as complex as possible.

Diffusion is the ability to spread (diffuse) the letters of the original *plaintext* over the whole *ciphertext*, so that any redundancy in the *plaintext* is spread out over the *ciphertext*

- Let an arbitrary composition of a substitution (Confusion) and a transposition (Diffusion) be called a round. (the two operations are not always clearly separable)
- Modern symmetric cryptosystems apply a number of rounds in succession to encrypt plaintext.



History of Cryptography

- During World War II mechanical devices called rotors were used to secure radio communications (polyalphabetic ciphers)
- The most famous encryption device was the Enigma Machine used by the German forces
 - substantially a keyboard linked to a scrambling device to permute the pressed keys followed by three rotors.
 - The “cryptographic key” was the initial configuration of rotors, plus the initial permutation.
- A famous English cryptanalyst developed systems and method for breaking the adversary algorithm
 - Alan Turing and the Turing Machine



History of Cryptography

- Most of the modern cryptosystems are based on number theory, abstract algebra (in particular, the algebra of finite fields) and computers.
- Modern cryptography allow to address all the issue needed to implement security services (as authentication and non-repudiation) expecially for commercial applications.
- The use of cryptosystems is typically regulated by protocols.
(i.e. sets of rules for ordered communication.)
- Many modern cryptosystems appear as components of network protocols.
E.g.: IPSec, TLS/SSL, the OSI model, etc.



Cryptographic Paradigms

o Symmetric Cryptosystems

- Encryption and decryption algorithm use the same key.
- Generally are widely used to guarantee privacy, but they can also be used for authentication and data integrity.
- E.g.: AES, 3DES2, TDEA, RC4, IdeaNext, BlowFish.

o Asymmetric (or Public-Key) Cryptosystems

- Every user is in possession of a pair of (related) keys.
- PUBLIC KEY: may be known to everybody;
used with the encryption algorithm.
- PRIVATE KEY: known only to the recipient and rigorously kept secret to everybody else;
used with the decryption algorithm.
- E.g.: RSA, ECC, DH, XTR, DSA, ECDSA.

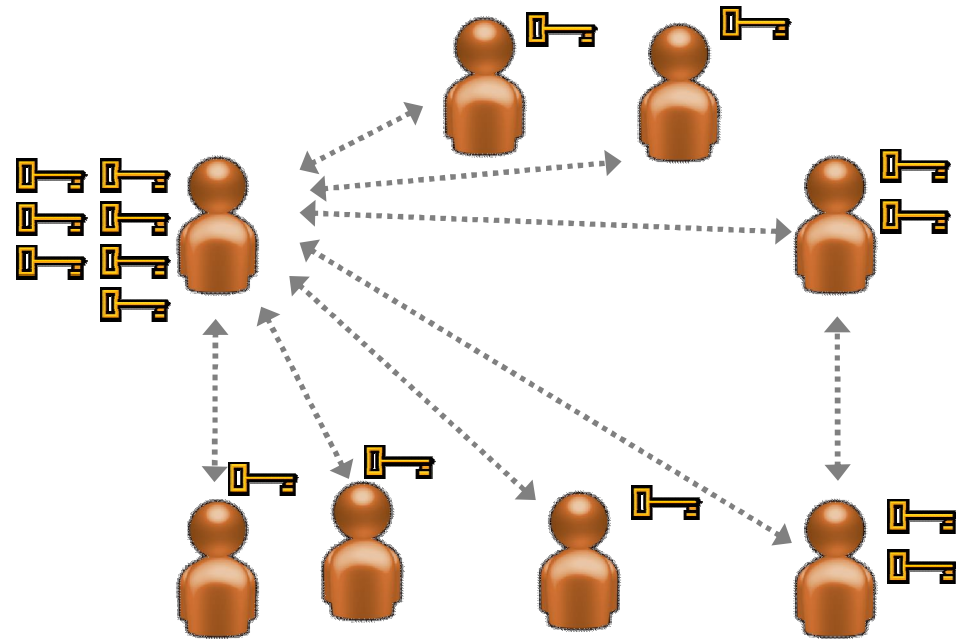
o Identity Based Cryptosystems

- A public-key system where the public key of each user is defined to be its Identity (i.e.: a previously recognized and publicly known piece of information, which is not associated with the cryptosystem parameter generation)

Symmetric cryptosystems



- Confidentiality of transactions
- Fast computation
- "Non repudiation" not guaranteed
- Key Management does not scale well:
 - $\approx n^2$ keys for n parties.
- Expensive Key Storage, Archiving and Backup
- Expensive Disaster Recovery





Public-Key Cryptosystems

- As a physical analogue, consider a metal box with the lid secured by a combination lock.
- The combination (private key) is known only to Bob. If the lock (public key) is left open and made publicly available then anyone can place a message inside and lock the lid.
- Only Bob can retrieve the message. Even the entity which placed the message into the box is unable to retrieve it.



Public-Key Cryptosystems

- A PK-encryption function must be a 'one-way function' with a 'trapdoor'.
 - A 'one-way function' is easy to perform but very difficult to reverse.
 - A 'trapdoor' must be a secret parameter known only to the receiver which makes it easy to reverse the function
- Examples of one-way functions:
 - multiplication of two large primes; the factorization problem is computationally infeasible for sufficiently large numbers.
 - $m \rightarrow m^e \bmod n$; exponentiation modulo n ($n = pq$)
 - $x \rightarrow a^x$ in $GF(2^n)$ or $GF(p)$; *Discrete Log*.
 - $k \rightarrow E_k(m)$ for fixed m where E_k is a symmetric encryption scheme (...secure against known *plaintext* attacks).



Public Key Primitives

- PK Encryption procedure.
A sender computes the *ciphertext* from a *plaintext* message **m** by computing a one-way function parametric in the public key of the recipient

$$\mathbf{c} = \mathbf{Enc}_{K_{\text{pubB}}}(\mathbf{m})$$

- PK Decryption procedure.
To recover the received confidential message the recipient computes a one-way deciphering function (related to the encryption one) parametric in its private key

$$\mathbf{m} = \mathbf{Dec}_{K_{\text{privB}}}(\mathbf{c})$$

- Properties: $\mathbf{Dec}_{K_{\text{privB}}}(\mathbf{Enc}_{K_{\text{pubB}}}(\mathbf{m})) = \mathbf{m}$
 $\mathbf{Enc}_{K_{\text{pubB}}}(\mathbf{Dec}_{K_{\text{privB}}}(\mathbf{m})) = \mathbf{m}$

- Public key cryptographic paradigm enables also the possibilities to realize:
 - Key Agreement over an insecure channel
 - Digital Signatures

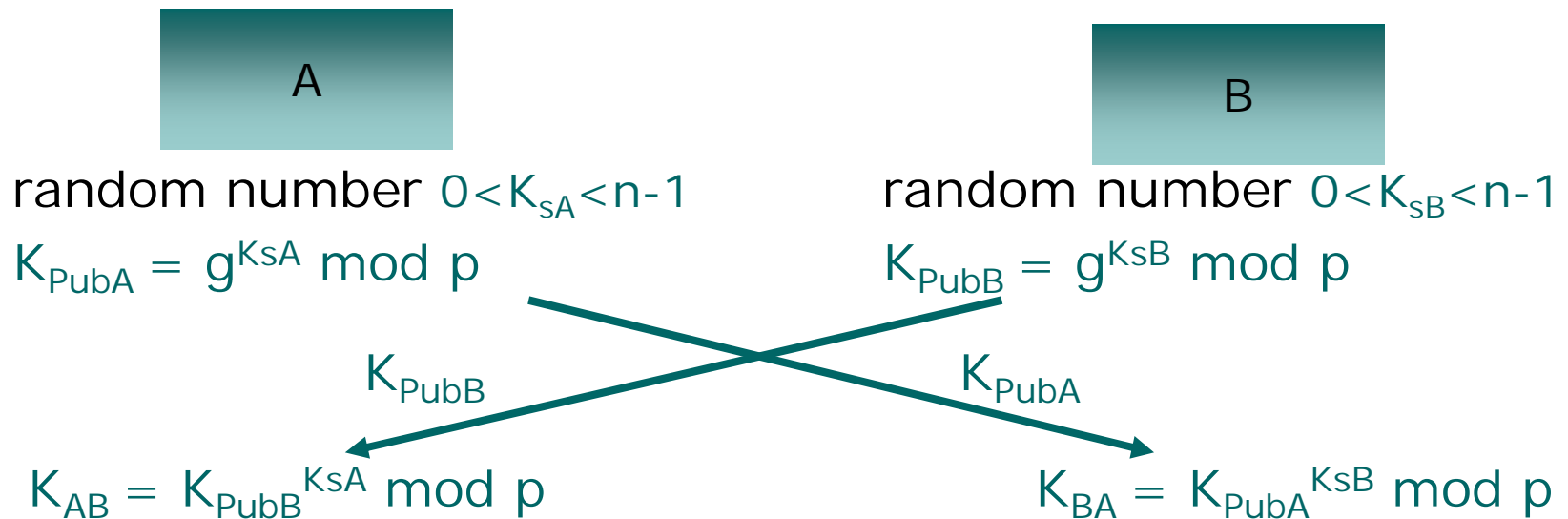


Diffie Hellman Key Exchange

- In 1976 Diffie and Hellman proposed a method for key exchange and basically introduced the concept of public key cryptography
- The method is based on exponentiation in a finite field and on the difficulty of computing the discrete logarithm in such a field

Diffie Hellman Key Exchange

Basic System parameters: a prime number: p , a generator g of a multiplicative subgroup (with prime order: n) of the finite field $(\mathbb{Z}_p; +; \cdot)$



$$\begin{aligned} \mathbf{K}_{AB} &= K_{PubB}^{K_{sA}} \text{ mod } p = (g^{K_{sA}})^{K_{sB}} \text{ mod } p = \\ &= (g^{K_{sB}})^{K_{sA}} \text{ mod } p = \\ &= K_{PubA}^{K_{sB}} \text{ mod } p = \mathbf{K}_{BA} \end{aligned}$$



Security of the DH key exchange

- **A passive attacker** who's monitoring the communication channel to break the cryptosystem can use the knowledge of
 - Z_p finite field properties
 - generator: g
 - public key: g^{K_s}
- Alas the computation of a secret exponent requires the application of a non-polynomial algorithm to solve the discrete logarithm problem (DLP)
- Besides, there is no method to solve the so-called DH-Problem, i.e. given g, g^a, g^b , find g^{ab} .
- Good security settings for current technology require n (the order of the group) be greater than 2^{1024} (traditional cryptosystems) or greater than 2^{160} (elliptic curve cryptosystems)



Man-in-the-middle attack

- The user A has to trust the public key of B
- If **an active attacker** C substitutes B's public key with his own public key, A is not capable to see the difference between two public keys.
- The effectiveness of the attack is due to the fact that the public key is a random element of a finite field which is not easily bounded to a specific user identity
- **Certificates** are tools for the official distribution of public keys, granting the correct and secure association of any *user* to her *public key*.



Public Key Infrastructure

- The use of certificates within a PKI is the most used solution for avoiding the MITM attack.
- The public keys are kept stored in a key server, and a certificate containing the user's key is released to anybody who requests it.
- A "Certification Authority" (CA), itself a trusted organization, manages this service.
- Network accessible directories store the public keys of the users who subscribe to the service.
- But how can certificates be trusted?
- It is necessary to have **digital signatures....**



Digital Signature

- According to ISO, the term Digital Signature *'indicate a particular authentication technique used to establish the origin of a message in order to settle disputes of what message (if any) was sent'*.
- A signature on a message is some data that
 - validates a message and verifies its origin
 - a receiver can keep as evidence
 - a third party can use to resolve disputes.

It has to be available to the sender and depends on

- the message
- a secret parameter

It should be

- easy to compute
- easy to verify
- difficult to forge



Digital Signature

- o Use of a secret parameter
- o Message dependent

Hand-written signatures

- Intrinsic to signer
- Same on all documents
- Physically attached to message

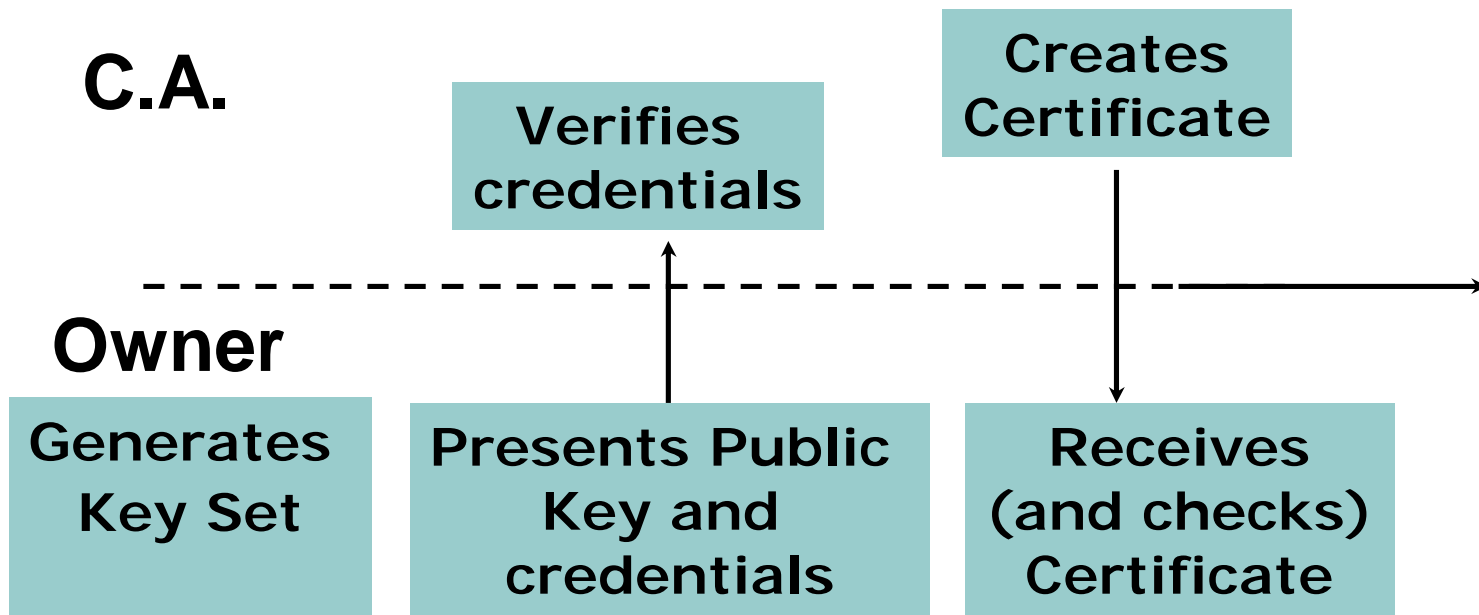
- o Digital Signing procedure:

A (the signer) creates a signature for a message **m** by:

- Compute $s = S_{KsA}(m)$
 - Transmitt the pair $\langle m, s \rangle$; **s** is called the signature for the message **m**.
- o Verification procedure.
To verify that a signature **s** on a message **m** was created by **A**, an entity **B** (the verifier) performs the following steps:
 - Obtain the verification function V_{KPubA} of **A**
 - Compute $u = V_{KPubA}(m, s)$
 - Accept the signature as having been created by **A** if $u = true$, and reject the signature if $u = false$.

Authentication of Public Keys

- A Trusted Third Party a.k.a. the Certification Authority (CA) guarantees the authenticity of a user's public key by signing a certificate, containing both the user's identity and public key, making use of its secret key.
- All users must have an authentic copy of the Certification Authority's public key to verify the signatures of the certificates.





Public Key Infrastructure (PKI)

- The CAs can be organized in trees, the root CA signs the certificates of other CAs, and so on up to the end user.
- Every user is equipped with a set of certificates of the root CAs.
- Once a user receives a certificate, he checks if it is signed by a CA in his list
 - if not it starts to search the certificate of the CA and a hierarchical check of the CA...

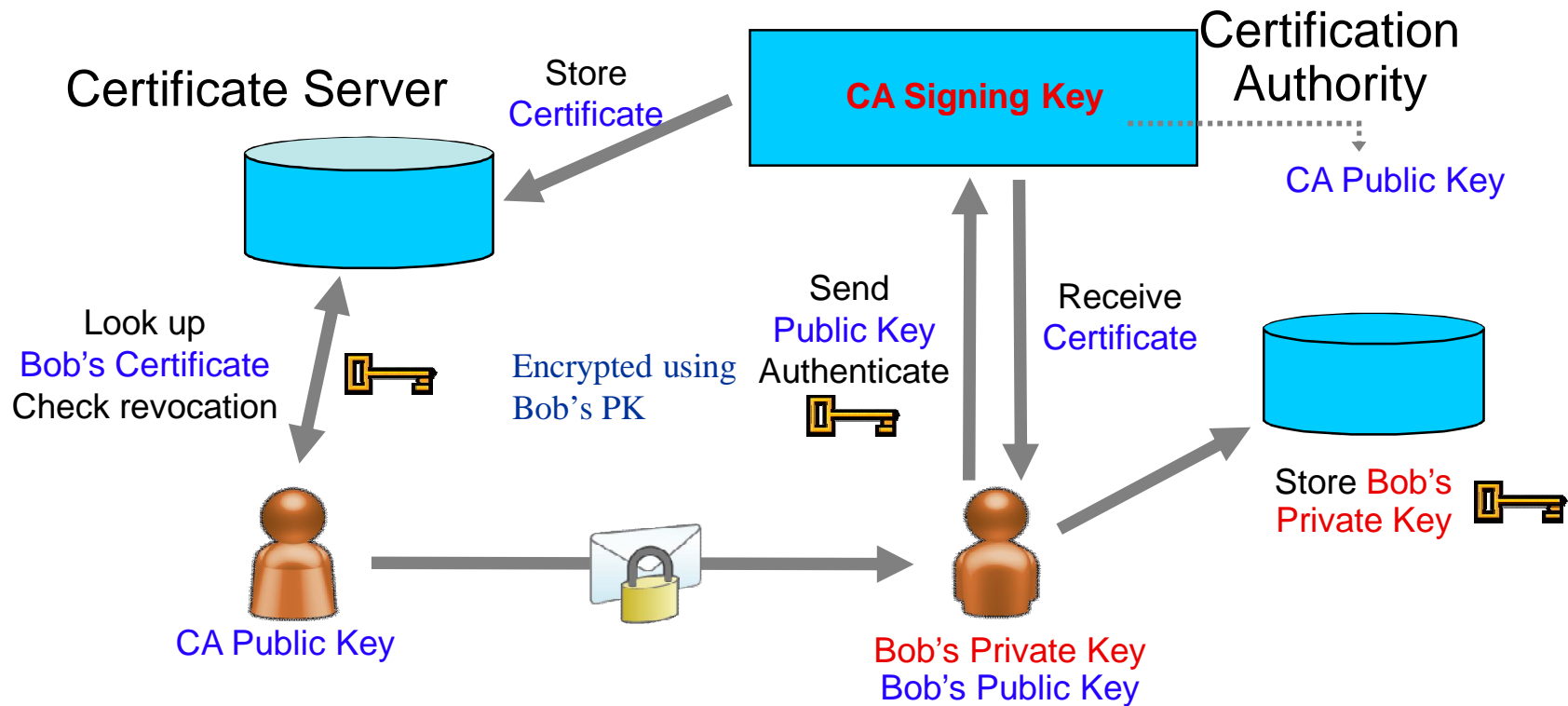


Public Key Infrastructure (PKI)

- Certificates have an expiration date
- Besides, the device where the private key is stored could be compromised (lost or broken).
- It is necessary to have procedures to revoke certificates
- Two standards:
 - Certificates Revocations List (CRL)
 - Online Certificate Status Protocol (OCSP)

Public-Key Cryptosystems & PKI

- o Users have a **Public Key** and a **Private Key**
 - Only need one key per party, total of n keys for n parties





Authenticated DH key exchange

The procedure for the agreement of a common secret session key over an insecure channel, where both parties have to equally contribute to the computation of the secret, follows the following steps:

- exchange of public-key certificates;
- validation of the received certificates;
- application of the DH Protocol where the DH-Public Keys of each user are sent, over the channel, encrypted with the public key of the counter-party.




The RSA Cryptosystem

- It is the most popular asymmetric (public key) cryptosystem.
- It was invented by Rivest, Shamir and Adleman in 1977 at MIT.
- It was under patent until 2000.
- Presently widespread, for instance in the well-known SSH protocol.

The RSA Cryptosystem

Setup of the system. Each user must:

- o select at random two primes p and q 
- o compute $n = p q$ and $\varphi(n) = (p - 1)(q - 1)$
- o select a random integer e such that:

$$0 < e < \varphi(n)$$

$$g.c.d.(\varphi(n), e) = 1$$

Euler's
function

- o use Fermat's th. or Euclid's algorithm to compute d (private key), such that:
 $e d = 1 \pmod{\varphi(n)}$ (d is $e^{-1} \pmod{\varphi(n)}$)
- o keep secret $k_s = (p, q, d, \varphi(n))$ (the private key)
- o make public $k_p = (n, e)$ (the public key)



The RSA Cryptosystem

When Alice need to send a message x (clear t.) to Bob:

- she obtains Bob's public key $k_{p,B} = (n, e)$
- she represents the message x as an integer in the range $[2, n - 2]$ (conversion tech. is irrelevant)
- she computes $y = x^e \bmod n$ and sends y to Bob
 y is the encrypted message (*ciphertext*)

To reobtain the clear text message x , Bob must:

- use his own secret key $k_{s,B} = (p, q, d, \varphi(n))$
and compute $y^d \bmod n = x$.



Example

- TASK: Alice need send to Bob the message $x = 4$.
- Setup of the cryptosystem. Bob must:
 - Choose $p = 3, q = 11$ (two primes)
 - Compute $n = p \times q = 3 \times 11 = 33$
 - Compute the Euler's function:
$$\varphi(33) = (3 - 1) \times (11 - 1) = 2 \times 10 = 20$$
 - Choose $e = 3$, acceptable since $0 < 3 < 20$ and $\text{g.c.d.}(20, 3) = 1$, i.e., the integers 3, 20 are co-prime
 - Compute $d = 3^{-1} \text{ mod } 20 = 7$ (e.g., using Euclid's Ext. Alg.)
- Bob's private key is $k_{s,B} = (3, 11, 7, 20)$.
- Bob's public key is $k_{p,B} = (33, 3)$.
- This completes setup on Bob's side (done only once).



Example

- Setup of the cryptosystem. Alice must:
 - Obtain Bob's public key $k_{p,B} = (33, 3)$.
- This completes setup on Alice's side (done only once).
- Encryption of the message. Alice must:
 - Compute $y = 4^3 \bmod 33 = 64 \bmod 33 = 31$
(e.g., using Square & Multiply Alg.)
 - Send $y = 31$ to Bob
- This completes encryption on Alice's side.
- Decryption of the message.
Bob private key $k_{priv,B} = (7, 11, 3, 20)$
 - Compute $x = 31^7 \bmod 33 = 27.512.614.111 \bmod 33$
 $= 4$ (e.g., using Square & Multiply Alg.)
 - Use x , it is the original *plaintext* Alice need to send to Bob
- This completes decryption on Bob's side.



Security of RSA

- The conjecture is that RSA is as secure as the factorization of n
- The attacker knows e and n .

If the modulus n is factored in p and q , the attacker can compute $\varphi(pq)$ and d , and thus decrypt all messages

- But it has not been proven that other simpler attacks exist



RSA Signature

- If a user decrypts a message with his private key and ships m and $m^d \bmod n$, all the other users can check if $(m^d)^e = m \bmod n$
- If the equality holds, it means that only the user with the belonging of d has computed m^d
- This is a proof of possession of d
- It is a digital signature!



More on Digital Signature

- Usually, instead of computing m^d an unambiguous information associated with M (the *hash* of M) is used.
- It is more advantageous to transmit (or validate) the signature of a short digest, than a signature of a potentially long message. (see next...)

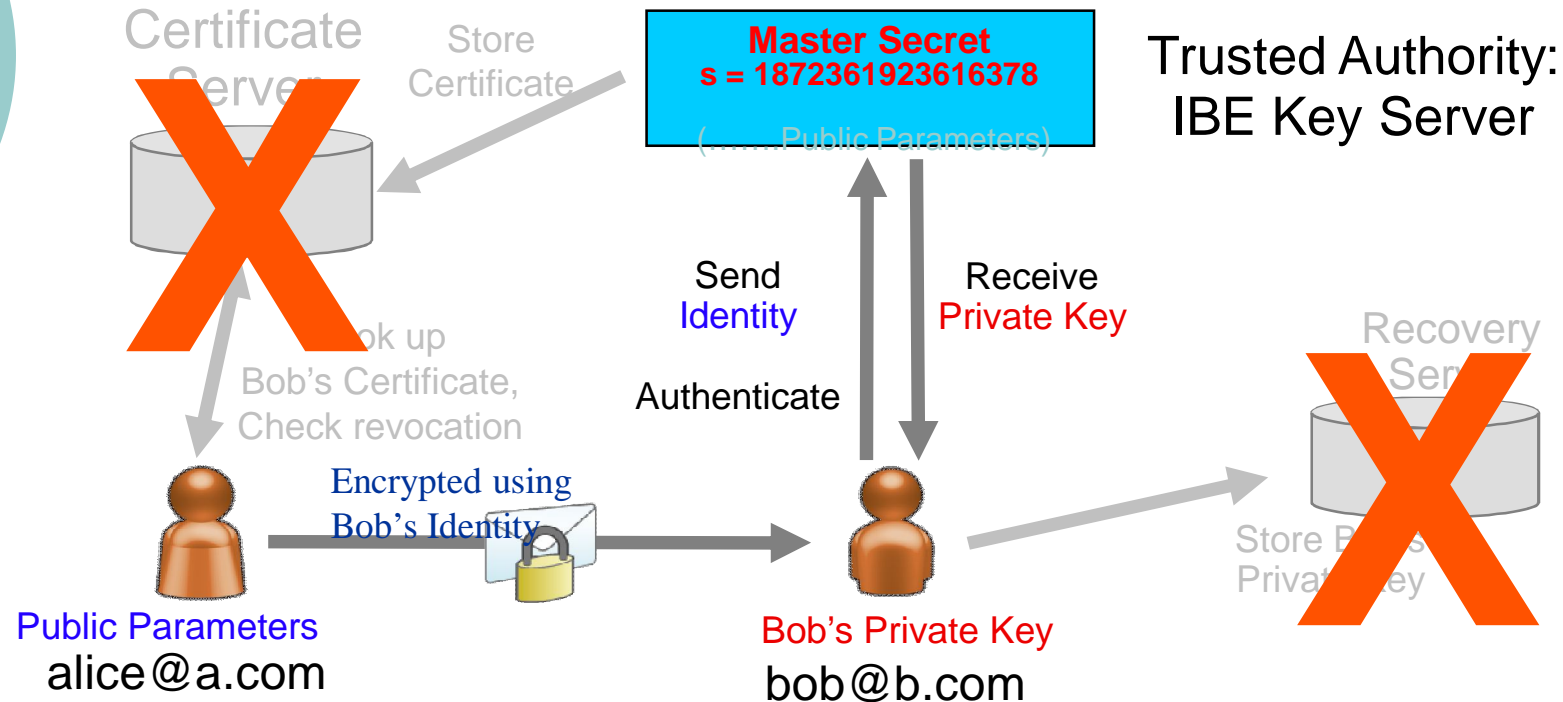


Other Public Key Algorithms

- ElGamal Cryptosystem
 - Based on DLP over finite field (adaption of DH)
 - Variants of the ElGamal Signing Algorithm are adopted as Digital Signature Standard Algorithms: DSA / DSS
- Candidate algorithm for future protocols: ECC - Elliptic Curve Cryptosystem
 - Based on DLP defined over points of elliptic curves:
 - The algorithm is itself not covered by patents, while some implementation is covered
- NTRU: considered very promising but:
 - It is covered by patents
 - Digital signature method has been proven to be insecure 2 time; new method not yet completely analyzed
- XTR: well defined and efficient as ECC
 - It is covered by patents

Identity-Based Cryptography

Main Idea: PK-Encryption where Identities are Public Keys



- o Key Server has a "Master Secret" to generate keys.
- o Private key is generated from Master Secret and Identity.
- o Enabled by mathematical functions defined over points of elliptic curves defined over finite fields (ECDLP)



Hash Functions and Data Integrity

- *Cryptographic Hash functions*, captures the concept of “Integrity checksum”.
- A *hash function* compresses a message of arbitrary length to a digest of fixed length respecting the following fundamental conditions.
 - Looking at the digest, it is not possible to work out the original message.
 - It is computationally infeasible to find a couple of messages that hash to the same element.
 - It is efficiently computable.



Hash Functions and Data Integrity

- When a transmitter sends a message coupled with a hash value, the recipient of the data recomputes the hash from the data that is received and compares that value to the one transmitted by the provider of the data.
 - If the original hash value and the recomputed one do not match, then the data has been changed in some way.
- Hashing functions, per se, provide a kind of digital fingerprint, without authentication.
- A common way to obtain data authentication and source authentication without using a public key scheme is the computation of a digital fingerprint given by the so-called message authentication codes (MACs).



Hash functions and Data Integrity

- The security of the hashing function is related to the size of the resulting digest (in bits).
- SHA-1 is a standard for computing checksum that uses a 160 bit digest.
- SHA-1 does not use secret keys. The checksum is computed with a public hashing function and needs to be stored in a safe way.



Hash Functions and Passwords

- Hash functions are often used to store passwords for users who are logging onto a multi-user system.
- When the user tries to log in with his or her established password, the login program hashes it, and compares the newly hashed password with the stored hash.
- If the two are equal, the system assumes the user typed in the right password.
 - E.g.: Linux login authentication is based on the standard hash function MD5; optionally a MAC with Blowfish cipher (...the key can be a salted version of the login password itself)



Cryptographic Implementation

- When it comes down to the implementation of cryptographic systems both HW and/or SW implementations have to be considered depending on the particular application.
- A good software implementation is highly dependent on optimizations of the algorithm w.r.t. the features of the particular cpu/platform:
 - the specific instruction set
 - memory (cache) system
 - parallelism (ILP, TLP)
- In the case of hardware two types of devices are considered: ASIC or FPGA
- A cryptographic algorithm or protocol could be implemented through a sw/hw partitioning methodology.



Software Implementation

- The most used metric for comparing different implementations is latency, i.e., number of bits encrypted per second
- This could be a secondary metric compared to:
 - Code size
 - Memory requirement
 - Power consumption
 - Security



HW Implementation

- When the CPU is too slow or not available for other reasons a dedicated hardware is used
- There are different metrics for comparing hardware implementations
 - Critical path (or achievable clock frequency) is often used
 - but sometimes the clock frequency is not a metric, it is more a constraint that the crypto block has to satisfy, other blocks of the system (typically CPU or BUSES) impose the clock frequency of the system
 - Silicon area (this is the cost unit of a chip)
 - Power consumption
 - Time to market....



HW implementation

- Silicon compilers are based on heuristics
- The optimal synthesis is depending on the particular constraints imposed:
 - Critical path
 - Area constraints
- Different architectures could turn in non-obvious synthesis results



Attacks against cryptographic systems

- There is no security without an attack.
- If we want to protect data or communications we should know who is the attacker and what are his resources and actual possibilities to success in moving around the implemented security system.
- Not taking into consideration the potentially attack's techniques during the design of a secure system, can make the use of cryptographic tools meaningless



Attacks against cryptographic systems

- Theoretical/Algorithmic Attacks
 - This term generally is referred to the set of attacks that work at mathematical level
 - The aim is to decrypt the message without the key or recover the key from a set of encrypted messages
- Misuse based Attacks
 - Try to find weaknesses in the way the algorithm is used in practice, i.e., storing the key in a external memory, send keys in clear, sign a certificate with the *SnakeOil co.* certification authority private key, etc...
 - [Bruce Schneier] Cryptography plays a role in computer security, but buggy computer systems and vulnerable communications are a reality that cryptography has not solved.



Side Channel Attacks

- Once a crypto algorithm is specified and its security verified with respect to the previously cited threats, it should be implemented in practice...
- This part was thought to be a simple engineering task
- But recently it has been demonstrated that digital devices release important information which enable to conceive powerful Implementation Attacks



Side Channels

- Some “*physical*” effects are involved on the computation of an encryption algorithm
- These are:
 - Power consumption
 - Execution time
 - Electro magnetic radiation
 - Platform Architecture Design can also leak secret information during the execution of a cryptographic algorithm (e.g. Branch Target Buffer Attack)



Side channels

- The encryption process is a function of basically two input data, the *plaintext* and the *secret key*
- Physical phenomenon can be collected, and usually are called traces
- Thus the trace (power, time or electromagnetic) is a function of the key and the *plaintext*
- If the *plaintext* is known, it is possible to relate the trace to the key and thus retrieve information on the secret key

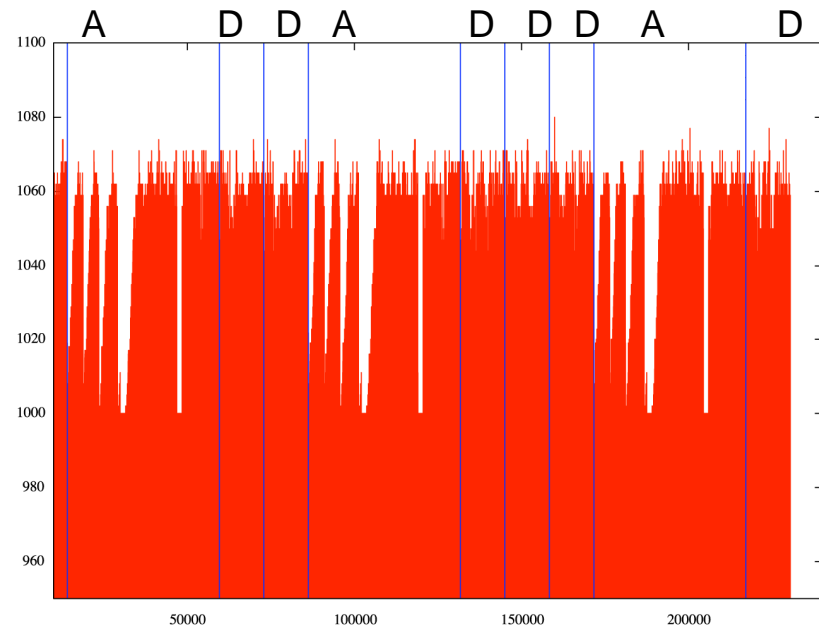


Power Analysis

- One of the most known side channel leak is the power consumption...it allows the power analysis
- Smartcards are particularly subject to this attack since the interface of the card is composed by 5 simple contacts:
 - Vcc and GND
 - Clock and reset
 - Data in/out
- If the current required on the Vcc is measured, then it is possible to obtain the power consumption of the device
- Due to the CMOS logic the power consumption is a function of the switching activity of the device
- Unbalanced execution could be exploited

Simple power Analysis

- In the case of ECC cryptosystem the device computes the analogous of modular multiplication following a sequence of doubling and addition of curve points
- A point addition is executed only when the bit of the key is equal to one
- If the power consumption of the point addition is relevant, it is sufficient to analyse a single power trace to find when a doubling is executed and consequently guess the secret key bits.
- A more powerful attack is called Differential Power Analysis (DPA) and also systems protected against SPA are vulnerable to this technique.





Timing Attack

- This has been the first presented side channel attack
- The SPA countermeasures protect implementation even against timing attack
- Recently a timing attack has been optimized to work even remotely against some SSL server
 - Side channel has been always intended as a threat for physical available devices...



Electromagnetic Analysis

- Like power is consumed, electromagnetic waves are emitted from silicon devices
- But while power need a physical contact for been measured, waves run “free” in the air
- Thus it is theoretical possible to measure it even from long distance
 - IBM has measured the emitted waves of an SSL server from about 20 meters outside the building
- Like Power Analysis it is possible to mount simple electromagnetic analysis and differential electromagnetic analysis
 - Most of the SPA/DPA countermeasures protect implementation against SEMA and DEMA
 - Researches are still open to understand if SPA/DPA countermeasure are good protection again all SEMA/DEMA attacks



Fault injection

- All precedent attacks are called passive attacks, the device computes the encryption algorithm without being disturbed by the attacker.
- The device can be disturbed during the computation, i.e. glitches on the power supply can affect the computation and faults can appear on the nominal values of internal data
- If it is possible to set to 0 all input of the last round of AES then the output is just the last round key
 - This is enough to retrieve the secret key



Devices

- PC
- Smartcard
- Mobile phone
- Set Top Box
- Television
- MP3 player
-



Applications

- Banking
- Digital Right Management
 - Pay TV
 - DVD
 - Music download
- Telecom
 - Mobile phone
 - Wireless LAN
 - Internet
- Car immobilizer
-