Introduction to Cryptography

Gerardo Pelosi

Department of Electronics, Information and Bioengineering (DEIB)
Politecnico di Milano

gerardo.pelosi - at - polimi.it

Cryptography

From craft...

- The word comes from ancient Greek: *kryptos*: hidden, concealed & *graphein*: to write
- Throughout most of its 2500-4000 years long history cryptography meant *secret writing*: Make the text (hopefully) incomprehensible (encrypt) by anyone except the intended receiver (decrypt)

... to science

Nowadays, the study of secret codes (aka cryptology) is the union of two disciplines

- *cryptography*: how to design and implement cryptographic algorithms
- *cryptanalysis*: how to break a cipher (recover key/message), analyzing weak mathematical assumptions or advancements, bad implementations

Crypto History: before seminal Diffie-Hellmann work (1/3)

Typical lifecycle of a cipher (until 1970's):

- New secret code invented (the details could be made public or not)
- Typically claimed *unbreakable* by its inventor
- Used by spies, ambassadors, kings, generals for crucial tasks
- Broken by enemies using *cryptanalysis*

Crypto History: before DH (2/3)

- 1587: Ciphers from Mary of Scots plotting assassination of queen Elizabeth broken; employed as evidence to convict her of treason
- 1860's (USA civil war): Confederacy used a good cipher (Vigenère) in a bad way. Thus, messages among generals were routinely broken by team of young Union cryptanalysts
- 1914: With info from sunken German ships, UK intelligence broke all German codes employed to send telegrams.
  The retrieved information of a German plan to form an alliance with Mexico contributed to push US in joining WWI
- 1939: During WWII the supposedly unbreakable Enigma cipher used by Germans, was broken through exploiting a mix of ingenuity, German negligence and mechanical computation.
  W. Churchill gave the crypto-analysts led by Alan Turing credits with winning the war

Crypto History: before DH (3/3)

Crucial flaws in the design and use of ciphers before the introduction of modern cryptology were (with different levels of importance):

- Secrecy of the encrypting method
- Lack of strong mathematical basis
- Lack of formal definitions of cryptanalytical resistance
- Naivety/negligence of operators

Modern Cryptology (1/4)

- 1976, Approval of DES algorithm as standard cipher for unclassified US documents, acted as catalyst for the academic study of cryptography and cryptanalysis
- 1976, Diffie and Hellman introduce the notion of public key cryptography based on a simple-to-state and hard-to-solve computational problem
- 1977, Rivest, Shamir, Adleman (RSA) propose a full public key cryptoscheme
- 1990+, new cryptoschemes introduced to face the increasing demand of securing computation and storage systems.

Innovative features and applications:
homomorphic encryption, zero-knowledge proofs, electronic cash, electronic voting, auctions, privacy preserving data mining

Modern Cryptology (2/4)

The approach applied in modern cryptology (from 1970s on) completely subverts the previous habits

Kerckhoff's Principle

A cryptosystem should be secure even if everything about the system is publicly known, except a single parameter (a.k.a., the *cryptographic key*)

- The cipher can be studied and tested by anyone thus, lowering the risk of theoretical hidden weaknesses and/or practical design pitfalls

Sound Implementations

The cipher primitives must be efficiently implemented in a wide range of hardware and software systems

- Minimize user induced errors and negligence of the operators

Modern Cryptology (3/4)

The security margin of cryptosystems is provided through analyzing both the mathematical strength and
the information leakage due to implementation details

Security Models

- **Unconditional Security (Perfect Secrecy)**: assumes an adversary with unlimited computing power and proves that she does not have enough information to infer either the cryptographic key or the original message
- **Computational Security**: assumes that any adversary is computationally limited (...as all adversaries are in practice).
  It gives a lower-bound on the computational complexity of the best methods available to break the cipher

Modern Cryptology (4/4)

Security Models

- **Provable Security**: proves that the difficulty of breaking the crypto scheme is "as difficult as" solving a computationally hard problem
- **Applied Security**: defines the resistance against vulnerabilities raising from implementation flaws or weaknesses of the HW/SW platform
  - Quantify the security margin provided by a given cryptographic implementation against active and passive analyses of the hardware platform executing them (Side-Channel Attacks: timing-, power-, EM-, microarchitectural- analyses)

This Course

Roadmap

- Foundations and principles of cryptology
- Review of the Mathematical Background needed to understand the design of most common (Symmetric-Key/Public-Key) cryptoschemes
- Design, implementation and use of ciphers, digital signatures, Message Authentication Codes, pseudo-random generators and secure hash functions
- Mathematical cryptanalysis against secret and public key schemes
- Applied cryptanalysis: passive and active side channel techniques
- Secure (TLS, SSH) and anonymous (onion routing) communication protocols as well as secure data storage schemes

What you will not learn:

- "Hacking"- breaking into systems through finding misuses of primitives and parameters
- Viruses, worms, Windows/Linux bugs, bad programming or social-engineering practices

Administrative infos

- Instructors: Gerardo Pelosi
  gerardo.pelosi -at- polimi.it

  Alessandro Barenghi (teaching assistant)
  alessandro.barenghi -at- polimi.it
- Lectures: Monday (14:30-16:15); classroom: L.26.01 Building 26
  Thursday (14:30-16:15); classroom: L.26.01 Building 26
- Office hours: Wednesday (16:00-19:00) or upon appointment by mail
- Course web page:
  `http://crypto.dei.polimi.it/doku.php?id=courses:csdd`

Courseware

- Slides and lecture notes will be available weekly on the course website
- The reference book for most of the course topics is:
  Nigel P. Smart, *Cryptography, An Introduction*,
  freely available on the course web page or at
  `http://www.cs.bris.ac.uk/~nigel/Crypto_Book/`
- A comprehensive reference for cipher implementations is:
  Alfred J. Menezes et al., *Handbook of Applied Cryptography*
  freely available at `http://cacr.uwaterloo.ca/hac/`

Administrative Infos

- ∼2h written examination with questions and exercises
- A practical project can integrate the exam score yielding at most a +6 increase in the evaluation (groups with at most 2 people).
  - a sufficient score in the written part must be obtained
  - project should be delivered one week before the exam call you want it to be assessed in
  - written part and project can be handed in on different exam calls
  - some topics for the projects will be available on the course web-page starting from **April 2018** (assigned on a first-come-first-served policy)
- Please, contact both the instructor and the TA via e-mail if you are interested in working on a project

Security Services

- Cryptography deals with any protocol or system designed to operate in an environment absent of universal trust
- Fundamental *security properties* (or *services*) ensured by cryptographic primitives are:

1. Confidentiality
2. Authenticity
3. Data integrity
4. Non-repudiation

The first one is the best known and is the basis for obtaining the other ones

Confidentiality

- **Confidentiality** provides encrypted information, thus making it unreadable to anyone except for the intended receivers, who are able to reverse the encryption
- The term *secrecy* is a synonym of *confidentiality*
- The term **privacy** is more generic and usually refers to a different concept, i.e.: the individual right to arrange

  "when", "how", "if", and "to who"

  any information related to her data should be disclosed.
  That is: when/how/where confidentiality is employed or needed !

Authenticity

- **Authentication** is the mechanism that systems use to identify their users
- An Authentication mechanism answers to the questions:
  - Who is the user/counterpart?
  - Is the user/counterpart really who she claims herself to be?
- When a system includes a communication protocol:
  - The parties may need to identify themselves (entity authentication)
  - The parties want to make sure that data is really exchanged between the intended endpoints (i.e., No-one in middle is masquerading as one of them) (data origin authentication)
- **Authorization** indicates the actions permitted to an already authenticated user.

  Discretionary Access Control models the owner of the object specifies which subjects can access the object (e.g., r/w/x permissions)

  Mandatory Access Control models the system – and not the users – specifies the clearance of subjects (Top Secret, Secret,...) to access a specific data object which is also labeled as TS, S, Confidential, Classified, Unclassified

Data Integrity

- **Data integrity** guarantees that data has not been tampered with
- Possible unintended manipulations of the data include:
  - *Insertion*: alien information is inserted in the communication stream or data storage
  - *Replacement*: the original data are corrupted/replaced with unintended contents
  - *Deletion*: some of the data are deleted without substituting them with anything

Data Authentication (data origin authentication + data integrity)

- Data integrity per-se is not really meaningful
  - it is of little help for you to know that the data you have received has not been modified, unless you know who it was sent by (i.e., the sender)
- To prevent *data forgery*, **Data Authentication** is needed
- Data authentication is provided through:
  - data origin authentication (the fact that you know who the sender is)
  - data integrity (the fact that data has not been modified)

Non-Repudiation

- **Non-repudiation** prevents an entity from denying previous commitments or actions later on

  Example:
  one entity, namely Alice, commits to purchase goods sold by another entity (namely, Bob), and later on Alice tries to deny that the action was performed

- a procedure involving a *trusted third party* (acting like a notary) is needed to settle the dispute in front of a judge

Basic Terminology and Concepts

- The need for formal definitions in modern cryptology
  stated some basic concepts employed throughout the whole discipline
- We will now present the basic terminology and concepts used throughout the course

Alphabet

An **alphabet** $\mathcal{A}$ is a finite set of symbols

Example

The binary alphabet, $\mathcal{A}=\{0,1\}$, is the most common choice as any other alphabet can be encoded over it.
E.g., there are 32 different binary strings of length five,
each letter of the English alphabet can be mapped to one of them

Basic Terminology and Concepts

Message Space and Plaintext

message space $\mathcal{M}$: consists of set of strings over an alphabet.

An element of $\mathcal{M}$ is called a **plaintext**

Ciphertext Space and Ciphertext

ciphertext space $\mathcal{C}$: consists of a set of strings over an alphabet (which may differ from the alphabet for $\mathcal{M}$).

An element of $\mathcal{C}$ is called a **ciphertext**

Basic Terminology and Concepts

Keyspace

A **keyspace** $\mathcal{K}$ is a set of elements called **key**s.
The cardinality of the keyspace is one of the figures of merit employed to assess the *security margin* provided by a crypto-system.

Encryption Transformation

Given an element $e \in \mathcal{K}$, the **encryption transformation** $\mathbb{E}_e$: $\mathcal{M} \mapsto \mathcal{C}$, uniquely determines a bijective map from $\mathcal{M}$ to $\mathcal{C}$.

Decryption transformation

Given an element $d \in \mathcal{K}$, the **decryption transformation** $\mathbb{E}_d$: $\mathcal{C} \mapsto \mathcal{M}$, uniquely determines a bijective map from $\mathcal{C}$ to $\mathcal{M}$.

Basic Terminology and Concepts

Cryptoscheme

A cryptoscheme is defined by the following 6-tuple:

$$\langle \mathcal{A}, \mathcal{M}, \mathcal{C}, \mathcal{K}, \{\mathbb{E}_e : e \in \mathcal{K}\}, \{\mathbb{D}_d : d \in \mathcal{K}\} \rangle$$

Fundamental properties:

- Correctness: it is possible to successfully decrypt the plaintext from every ciphertext only employing the correct key(s)

$$\forall\, e \in \mathcal{K} \quad \exists!\, d \in \mathcal{K} \quad \text{s.t.}\ \forall\, m \in \mathcal{M} \quad \mathbb{D}_d\left(\mathbb{E}_e\left(m\right)\right) = m$$

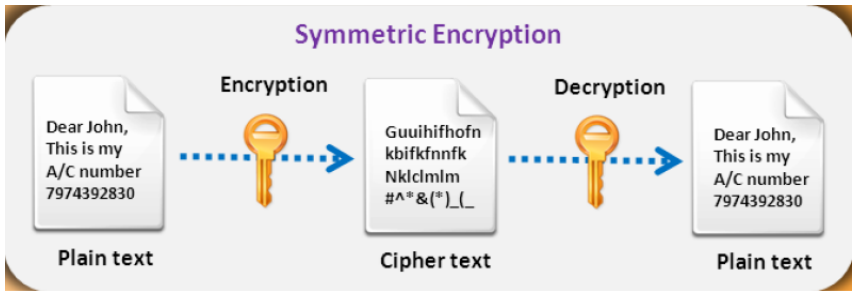
- Efficiency and Strength: Both $\mathbb{E}$ and $\mathbb{D}$ should be *fast* to compute given the correct values of e and d, and *unfeasible* (or impossible) to compute without them.
  (Both $\mathbb{E}$ and $\mathbb{D}$ must be defined as "one-way functions ... with trapdoor")

Cryptographic Paradigms

Symmetric (or Secret-Key) Cryptosystem

Every user is bound to a **single** secret key, with fixed length:
i.e., encryption $\mathbb{E}_e$ and decryption $\mathbb{D}_d$ algorithms use the **same key**: $e = d$

- Provides *Confidentiality* and/or *Data Authentication*
- Cannot provide *Non-Repudiation*



There are two main strategies for defining $\mathbb{E}$ and $\mathbb{D}$, which lead to

- Block ciphers: act on a fixed length plain/ciphertext
  (e.g., AES, 3DES2, CAST5, Camellia, Gost, BlowFish)
- Stream ciphers: act on an arbitrary length plain/ciphertext
  (e.g., RC4, Trivium, A5/1, A5/2, A5/3)

Cryptographic Paradigms

Symmetric (or Secret-Key) Cryptosystem

Pros:

- Structurally, highly computationally efficient

Cons:

- The secret keys need to be exchanged over a separate secure channel among all the parties of the communication
- Key management is quite cumbersome as each distinct pair of communicating parties should share a different key
  - A group of n users, willing to communicate with all the others requires the distribution of $\frac{n(n-1)}{2} \approx n^2$ distinct keys
  - When a user is added/removed to/from the group, he must communicate with potentially n users to send/invalidate his keys

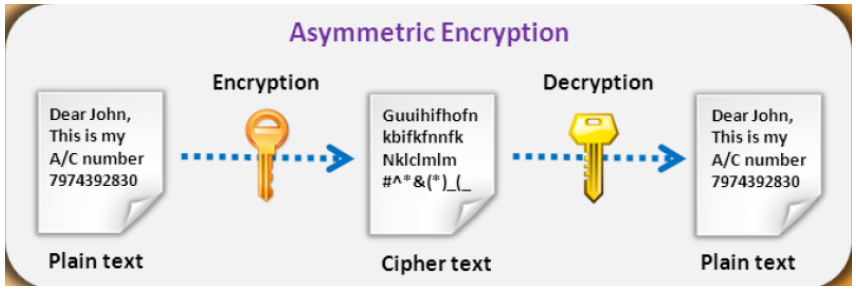
Cryptographic Paradigms

Asymmetric (or Public-Key) Cryptosystem

Every user is bound to a key pair:

- **Public Key** $e \in \mathcal{K}$: is employed to **encrypt** plaintexts ($\mathbb{E}_e(m), m \in \mathcal{M}$). Can be known to everybody
- **Private Key** $d \in \mathcal{K}$: is employed to **decrypt** ciphertexts ($\mathbb{D}_e(c), c \in \mathcal{D}$). Must be known to the recipient only (and rigorously kept secret from everybody else)

PKC provides: *Confidentiality*, *Data Authentication* and *Non-Repudiation*



Public-key cryptosystems exploit well-known mathematical problems

- Factoring based: RSA
- Discrete logarithm based: DH, DSA, XTR, ECDH, ECDSA

Cryptographic Paradigms

Asymmetric (or Public-Key) Cryptosystem

Pros:

- Scalable key management: broadcasting the public key can be managed with *public-key repositories*
- It is possible to provide the *non-repudiation* service

Cons:

- Substantially slower ($\sim 100\times$) than symmetric-key cryptosystems
- Longer key length (from $2\times$ to $10\times$) to achieve the same security level of a symmetric cryptoscheme
- **The public key needs authentication to avoid identity theft. Which guarantee do we have that the public key is really bound to the intended user?**

Cryptographic Paradigms

To provide the *Non-Repudiation* service we need:
Digital Signatures and **Certification Authorities** (CAs)

Digital Signature

A tool to authenticate the public key.
It allows to verify the unambiguous association of a user to her public-key

- A digital signature can be obtained applying the **decryption** transformation to the message m, which should be authenticated: $s = \mathbb{D}_d(m)$
- Everyone can check the validity of the signed message $\langle m, s \rangle$ through checking whether $\mathbb{E}_e(s) = m$ or not, as e is a publicly known value

Cryptographic Paradigms

Certification Authority (CA)

An independent **third party** (that all users trust) certifies the binding of a public-key to the identity of the corresponding user

- The CA digitally signs a document (i.e., a file) containing
  - The identity of the user
  - The public-key of the user
  - Metadata (e.g., expiration date, CA name, etc...)
- The document signed by the CA is called **digital certificate** and is stored in public repositories
- The assumption is that everybody knows the public keys of the CAs, thus every certificate can be checked through verifying a CA signature
- The system can be hierarchically organized with CAs authenticating other CAs: this structure is known as **Public Key Infrastructure**
- we'll see more on PKIs and other models to bind the user's identity with his public-key (e.g., web-of-trust) in other lectures

Cryptographic Paradigms

Identity Based Cryptosystem

A Public-Key system where the User Identity is employed to uniquely derive the public key

- **Identity**: any previously recognized and publicly known piece of information bound to a specified user (e.g., a SSN, an email address, passport No., driving licence No., position in a company)
- **Public Key**: is uniquely derived from the Identity chosen by the user. May be known to everybody
- **Private Key**: is released to each user by a **Trusted Authority** (TA) who combines the user Identity and a *master secret parameter* to compute it
- The private key is known to the user **and** to the TA (*key escrow*)

Cryptographic Paradigms

Identity Based Cryptosystem

Pros:

- No need for digital certificates and their management (i.e., no revocation protocols)
- Multiple private keys can be bound to the same user identity (greater flexibility in yielding deciphering rights)
  - A key can be bound to a specific time lapse (e.g., only in the future)
  - The identity of a CEO can be "split" into multiple keys, one for each of his roles in the company (director of management, director of research...) to partition information access rights for the secretaries (i.e., each of them knows only a specific private key, to access only a subset of the messages addressed to the CEO)

Cons:

- The *key escrow* might not be a desired feature in open networks (the TA can access every message)
- Significantly more complex than both asymmetric and symmetric systems (...built on functions over elliptic curves)

Adversaries and Classes of Attacks

The strength of a cryptoscheme is evaluated against different attack models, where the adversary is classified as:

- **Passive attacker**: she only monitors the communication channel.
  A passive attacker only threatens confidentiality of data
- **Active attacker**: she attempts to delete, add, or alter the messages transmitted over a channel.
  An active attacker threatens data integrity, authentication as well as confidentiality (e.g., MiTM)

Attacks on Cryptoscheme

There are several possible assumptions on information available to a passive attacker

Basic assumptions: Kerckhoff's principle

The alphabet $\mathcal{A}$, the structure of the plaintexts (i.e., the form of $\mathcal{M}$) and the details of the encryption/decryption algorithms are known

Brute force attack (Exhaustive key search)

Given a ciphertext, it checks all possible keys until the correct one is found.

- **The attacker must be able to distinguish the correct plaintext from a valid but incorrect one**
- It is used against any computationally or provably secure scheme
- Perfectly secure schemes are not attackable via brute force... by construction, the adversary is unable to recognize a valid message...

Attacks on cryptoscheme

Ciphertext-only attack (COA)

- The attacker knows the ciphertext of a number of messages encrypted with **same key** (he doesn't known any plaintext)
- He recovers either the plaintext or the key by comparing the statistical distributions of ciphertext and plaintext symbols

Known-plaintext attack (KPA)

- The attacker knowns plain-ciphertext pairs, encrypted with **same key**
- He analyzes the differences among the different ciphertexts/plaintexts and reconstructs the secret key (f.i., permutation ciphers can be easily broken in such a scenario)

Attacks on cryptoscheme

Chosen-plaintext attack (CPA)

- The attacker chooses a number of plaintexts to be encrypted (all with the **same unknown key**) and obtains the corresponding ciphertexts
- The attacker has more control than in known-plaintext attacks. Thus, he may be able to gather more info on the key
  - E.g., Linear cryptanalysis of block ciphers
- If the attacker is allowed to choose the plaintexts adaptively, this is commonly defined as an **adaptive-chosen-plaintext attacker** (CPA2)

Attacks on cryptoscheme

Chosen-Ciphertext attack (CCA)

- Attacker gathers information choosing a ciphertext and obtaining its correct decryption under an unknown key (the same key for all messages)
  - Effective in exploiting vulnerabilities of asymmetric cryptosystems
- An interactive form of CCA where the attacker chooses the ciphertexts to be decrypted knowing the results of previous choices, is called **adaptive-chosen-plaintext attacker** (CCA2)
  - The goal is to gradually reveal information about the plaintext, or about the decryption key itself