Department of Information Engineering and Computer Science

Bachelor's Degree in
Information and Communications Engineering

Introduction To Computer & Network Security

Lecturer

Prof. Silvio Ranise

Students

Mattia Meneghin

Davide Zorrdan

Academic Year 2022/2023

Contents

1 Basic Notions

1.1 Definitions

Information security The practice of preventing unauthorized access, use, disclosure, disruption, modification, inspection, recording or destruction of information. It is a general term that can be used regardless of the form the data may take (e.g., electronic, physical).

Computer security The protection of computer systems and information from harm, theft, and unauthorized use. Computer hardware is typically protected by the same means used to protect other valuable or sensitive equipment, namely, serial numbers, doors and locks, and alarms. The protection of information and system access, on the other hand, is achieved through other tactics, some of them quite complex.

Network security Network Security is the process of taking physical and software preventative measures to protect the underlying networking infrastructure from unauthorized access, misuse, malfunction, modification, destruction, or improper disclosure, thereby creating a secure platform for computers, users and programs to perform their permitted critical functions within a secure environment.

Cyber security The ability to protect or defend the use of cyberspace from cyber attacks.

Cyber space A global domain within the information environment consisting of the interdependent network of information systems infrastructures including the Internet, telecommunications networks, computer systems, and embedded processors and controllers.

Cyber Attacks A cyber attack targeting an organization's use of cyberspace for the purpose of disrupting, disabling, destroying, or maliciously controlling a computing environment/infrastructure; or destroying the integrity of the data or stealing controlled information.

1.2 Remarks on Security

- Security is characterized by protection against an adversary or, possibly, against some other physical or random process
- Security typically focuses on malicious adversaries
- Core to any consideration of security is the modelling of malicious adversaries
  - motivations (in the past... glory, nowadays... money)
  - capabilities (intercept messages, modify messages, read keys pressed, ...)
  - threats (roughly, negative impact on systems, operations, organization, business, ...)
- Arguing that a system is secure without referring to an attacker model does not make much sense
- Indeed, absolute security does not exist even for so-called air-gapped system
- In order to mitigate threats, security proposes controls/mitigation strategies affecting
  - people (e.g., rules for setting passwords)
  - process (e.g., regularly update software)
  - technology (e.g., authentication and access control)
- Controls can be classified in:
  - **preventive**, i.e. before the bad event (e.g., locking out unauthorized intruders)
  - **detective**, i.e. during the bad event (e.g., turning an intruder alert on a dashboard)
  - **corrective**, i.e. after the bad event (e.g., recovering deleted data from a backup)
- Selection of controls is based on:
  - **risk management**: the process of identifying vulnerabilities and threats to the information resources used by an organization and deciding what countermeasures, if any, to take in reducing risk to an acceptable level
  - considering the **human factor**: controls must be easy to use and must neither require stress of mind nor the knowledge of a long series of roules
- The role of trust in software
  - **Dependability** = ability to avoid failures that are more frequent and severe than is acceptable
    - **Failure** = an event that occurs when the delivered service deviates from correct service
  - **Trust** = accepted dependence
    - For a complete discussion on dependability, trust, and security see

- Security controls are human artifacts that
  - can mitigate the impact of an attack but in many cases do not avoid it completely
  - can contain vulnerabilities that can be exploited to mount attacks
- In other words, we are left with a (non-null) residual risk, i.e. the amount of danger associated with an attack after risks have been mitigated by security controls

"Exploiting vulnerabilities allows attackers to violate security"

- At first reading, this may seem obvious but is it really so?
- What does it mean exactly to "violate security"?
- There many possibly answers to the question above including
  - A security violation can be the unauthorized sharing of sensitive information such as personal data of patients in a healthcare system
  - A security violation can be the unauthorized modification of the content of a resource such as modifying the balance of bank account
  - A security violation can be the unauthorized withholding of the content of a resource or service such as the unavailability of network services
- The three answers correspond to three crucial properties characterizing three possible dimensions of security

1.3 The CIA Triad

The CIA triad (**C**onfidentiality, **I**ntegrity and **A**vailability) is to be considered as a high-level characterization of the notion of **Security**, as it provides a general characterization of the notions of secrecy (*Confidentiality*), authenticity and non-repudiation (*Integrity*), and continuity (*Availability*) that need to be refined (and defined) for the use case scenario in which the system/application is going to be deployed.

Importance of the Human Factor

When considering the adoption of **mitigation measures** (also called *security controls*), it is crucial that the **human factor** is adequately evaluated. This means that if user intervention is needed (as it is the case for many security mechanisms/services, such as authentication whereby users are required to prove their presence by providing, e.g., passwords), the user experience should be as frictionless as possible; otherwise, users will either avoid the use of the service or find workarounds that will spoil its effectiveness. Consider for instance passwords; they should be long random strings of symbols but, at the same time, they should be easy to remember.
Indeed, these are conflicting requirements resolved by users with strategies to make passwords easy to remember that can be exploited by attackers to guide brute-force cracking. As a result, nowadays **passwords alone** are considered **inadequate** to protect accounts.

1.3.1 Security Properties

1 Confidentiality

- prevent un-authorised disclosure of information
- permit authorized **sharing** of information

2 Integrity

- prevent un-authorised modification of information
- permit authorized **modification** of information

3 Availability

- prevent un-authorised withholding of information or services
- readily permit authorized **access** to information or services

1.3.2 Confidentiality

- Preserving authorized restrictions on information access and disclosure, including means for **protecting personal privacy and proprietary information**.
- The property that sensitive information is not disclosed to unauthorized individuals, entities, or processes.
- The security goal that generates the requirement for protection from intentional or accidental attempts to perform **unauthorized data reads**. Confidentiality covers data **in storage, during processing, and while in transit**.
- The property that sensitive information is not disclosed to unauthorized entities. In a general **information security context**: preserving authorized restrictions on information access and disclosure, including means for preserving personal privacy and proprietary information.

Confidentiality in Practice

- Unauthorized access to sensitive information could be:
  - **Intentional**: such as an intruder breaking into the network and reading the information
  - **Unintentional**: due to the carelessness/incompetence of individuals handling the data
- How to guarantee confidentiality:
  - **Data encryption** is one way to ensure confidentiality and that unauthorized users cannot retrieve data for which they do not have access
  - **Access control** is an integral part of maintaining confidentiality by managing which users have permissions for accessing data

1.3.3 Integrity

- Guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity.
- The property that sensitive data has not been modified or deleted in an unauthorized and undetected manner.
- The ability to detect even minute changes in the data.
- Ensuring the authenticity of information—that information is not altered, and that the source of the information is genuine.
- The security objective that generates the requirement for protection against either intentional or accidental attempts to violate data integrity (the property that data has not been altered in an unauthorized manner) or system integrity (the quality that a system has when it performs its intended function in an unimpaired manner, free from unauthorized manipulation).

Autenticity : The property of being genuine and being able to be verified and trusted; Confidence in the validity of a transmission, a message, or a message originator.

Non-Repudiation (For auditability) : Protection against an individual falsely denying having performed a particular action. Provides the capability to determine whether a given individual took a particular action such as creating information, sending a message, approving information and receiving a message.

Integrity in Practice

- Data integrity can be compromised both through
  - human errors and
  - attacks like destructive malware and ransomware
- How to guarantee integrity:
  - Implementing version control and audit trails into an IT program will allow an organization to guarantee that its data is accurate and authentic
  - Integrity is an essential component for organizations with compliance requirements. For example, a condition of the compliance requirements for financial services
- Information integrity attack:
  - December 2017: Federal Communication Commission‘s net neutrality comment form witnessed a miracle... the dead returning to life
  - 2 millions identical comments under the name and address of real people... were generated by bots
  - This activity was spotted because users reported that some of the names belonged to their deceased family members and friends!

Violation Of Integrity Impact is on thrustworthines of resources and services available online. Bots can influence social media by massive posting of messages carrying manipulated information with negative impact on social/democratic life.

1.3.4 Availability

- Ensuring timely and reliable access to and use of information.
- Timely, reliable access to data and information services for authorized users.
- The ability for authorized users to access systems as needed.
- A requirement intended to assure that systems work promptly and service is not denied to authorized users.
- The security goal that generates the requirement for protection against intentional or accidental attempts to (1) perform unauthorized deletion of data or (2) otherwise cause a denial of service or data.

Availability in Practice

- Violations of availability include:
  - infrastructure failures like network or hardware issues
  - infrastructure overload
  - power outages
  - attacks such as Distributed Denial of Services (DDoS) or ransomware
- How to guarantee availability:
  - Employing a **backup system** and a **disaster recovery plan** is essential for maintaining data availability should a disaster, cyber-attack, or another threat disrupt operations
  - Utilizing **cloud solutions** for data storage is one way in which an organization can increase the availability of data for its users

Violation of Availability Impact is many fold:

- **Economy**: the longer the downtime, the larger the loss of money for companies or even entire (national) eco-system
- **Funamental Rights**:censorship is obviously bad for normal democratic life

1.3.5 Remark on CIA

- The CIA triad is essential because it allows for achieving security
- Its generality is, at the same time, a positive and a negative characteristic
  - **positive** since it applies to a wide range of situations and use cases

  - **negative** since it must be instantiated to every situation and use case; such instantiations are called security policies that require security mechanisms to be enforced
    * Defining security policies is far from being an obvious task

- Example use case: confidentiality of bank account data
  - Employees of a bank shall access only selected data from each bank account, enough to perform their job
  - A teller of a bank shall access the balance of a bank account to perform withdrawal
  - A manager shall access the history of the transactions of a bank account to decide to grant or deny a loan application

1.4 Security Policies and Mechanisms

Several *security mechanisms* may be needed to guarantee one of the security properties. In contrast, a *security service* fully supports one or more properties in the CIA triad, it's typically much simpler to configure and use than a security mechanism because of the granularity of the latter. Indeed, mechanisms are lower level and present a wide range of parameters that are typically reduced to few with security services.

1 - Security policy The rules and requirements established by an organization that governs the acceptable use of its information and services, and the level and means for protecting the confidentiality, integrity, and availability of its information

2 - Security mechanism A device or function designed to provide one or more security services usually rated in terms of strength of service and assurance of the design.
Implementation of a security policy

3 - Security service A capability that supports one, or more, of the security requirements (CIA). Examples: key management, access control and authentication

1.4.1 Example of security policies

- Purpose
  - $< CompanyX >$ must protect restricted, confidential or sensitive data from loss to avoid reputation damage and to avoid adversely impacting customers. The primary objective is user awareness and to avoid accidental loss scenarios
- Scope
  - Any employee, user or individual with access to $< Companyx >$ systems or data
  - Definition of data to be protected
    - Personal data
    - Financial
    - Intellectual Property
- Policy rules:
  1. Employees need to complete $< CompanyX >$'s security awareness training and agree to uphold the acceptable use policy.
  2. Visitors to <Company X> must be escorted by an authorized employee at all times. If an employee is responsible for escorting visitors, he/she must restrict them to appropriate areas.
  3. Employees must keep a clean desk. To maintain information security, employees need to ensure that all printed in scope data is not left unattended.
  4. Employees need to use a secure password on all $< CompanyX >$ systems as per the password policy. These credentials must be unique and must not be used on other external systems or services.
  5. Terminated employees will be required to return all records, in any format, containing personal information.

1.4.2 Examples of security mechanisms

Authentication Verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in an information system.

Authorization The granting or denying of access rights to a user, program, or process.

Access control The process of granting or denying specific requests:

1. obtain and use information and related information processing services;
2. enter specific physical facilities (e.g., Federal buildings, military establishments).

1.4.3 Example of security services

Security Services are tipically composed of security mechanisms

- Also, security services are easier to use than security mechanisms as the way they can be integrated in available systems is designed to be frictionless
- This means that while security mechanisms may be difficult to configure, security services are easier to set up and deploy
- A prominent example of security services can be found in the cloud
- Authentication and authorization mechanisms are available to be used with services deployed in the cloud in a more straightforward way than when deployed on premises
  - However, notice that the responsibility of configuring the security services is entirely on the shoulder of the user of the cloud platform
  - This is referred to as the shared responsibility model

1.4.4 Trheat Model

While the CIA triad and the associated security policies characterize the security properties, that are crucial for a system/application, the **threat model specifies**, which are the threats that may violate such properties. Threat in details
Such a model is necessary as it describes the capabilities that an attacker will use to exploit vulnerabilities in a system/application and violate one or more of its security properties. Indeed, different capabilities imply varying levels of (in-)security that should be mitigated by adopting different security mechanisms/services that may be technical or non.
In other words, saying that a system/application is secure without precisely specifying with respect to which set of threats is a meaningless statement.
A threat model is based on some simplifying assumptions concerning both the system/service and the capabilities of an attacker. For instance, we typically assume that the code of an application does not contain malware inserted (either maliciously or inadvertently) by the programmer of the application. While reasonable, such assumptions should be carefully evaluated and shall be repeatedly checked over the lifetime of the system/application as the threat landscape is in constant (and fast) evolution. In general, making the "right" **trust assumptions** plays a crucial role in defining accurate and realistic threat models capable of predicting security violations. Threat Model Example

Example 1: TLS Transport Layer Security

The Transport Layer Security (TLS) protocol is used to guarantee confidentiality and integrity of data (in transit) and it is the cornerstone of web security;
Security mechanisms used:
Range of cryptographic primitives to guarantee confidentiality and integrity

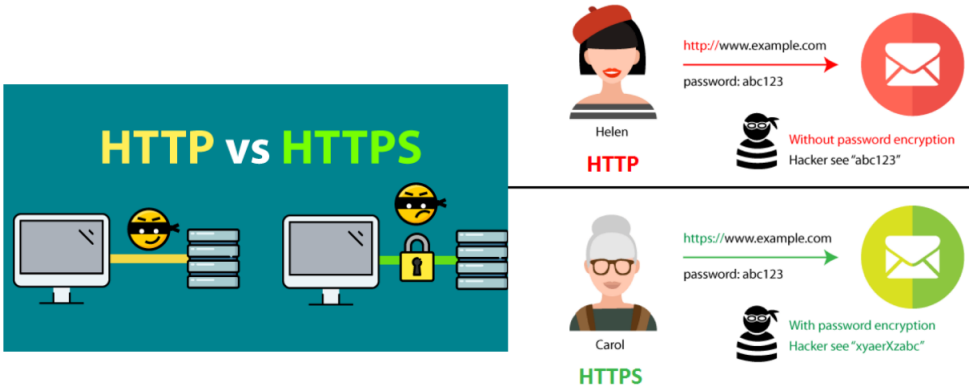


Figure 1.1: Example 1: TLS

Example 2: Access control

Seen as the combination of **authentication** and **authorization**, is typically used to guarantee confidentiality and integrity of data (typically at rest).
Security mechanisms used:
Isolation for integrity and correctness with evaluation of authorization conditions

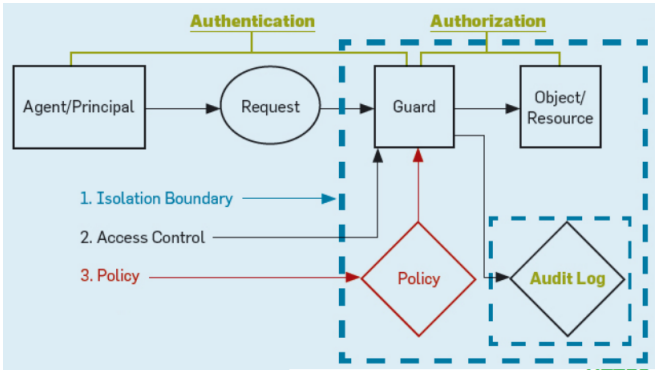


Figure 1.2: Example 2: Access Control

1.5 CIA Security Violations and Migrations

- The CIA triad is not only essential for achieving security but also helps understanding security violations (i.e. what went wrong)
- Example: **ransomware attacks**
  - Ransomware is a type of malware that threatens to publish the victim's personal data or permanently block access to it unless a ransom is paid
  - It encrypts the victim's files, making them inaccessible, and demands a ransom payment to decrypt them
  - Recovering the files without the decryption key is an intractable problem and difficult to trace digital (crypto-)currencies making tracing and prosecuting the perpetrators difficult
- Which security properties of the CIA triad can be violated in a ransomware attack?
  - Availability as access is blocked but also confidentiality if the victim's data is exfiltrated... and even integrity as files are encrypted and can be modified by the attacker
- Mitigation: **Zero Trust**
  - Zero Trust is a security framework requiring all users, whether in or outside the organization's network, to be authenticated, authorized, and continuously validated for security configuration and posture before being granted or keeping access to applications and data
- This is an example of **risk management**
- The CIA triad also helps understanding security violations (i.e. what went wrong) and suggests how to avoid such problems by defining security policies and mechanisms (the latter enforce the former)
- More precisely, the CIA triad is crucial for risk management that involves identifying, assessing, and treating risks to the confidentiality, integrity, and availability of an organization data and systems
  - The end goal of risk management is to treat risks in accordance with an organization's overall risk tolerance
  - Organizations should not expect to eliminate all risks rather to identify and achieve an acceptable risk level
- Risk management main phases
  1. Identification of assets, vulnerabilities, threats and controls (i.e. policies and enforcement mechanisms)
  2. Assessment as likelihood and impact of a threat exploiting a vulnerability
  3. Treatement to reduce risks by selecting appropriated controls

1.6 Risk

1.6.1 Vulnerability

- Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source.
- A weakness in a system, application, or network that is subject to exploitation or misuse.
- A flaw or weakness in a computer system, its security procedures, internal controls, or design and implementation, which could be exploited to violate the system security policy.

1.6.2 Threat

- Any circumstance or event with the potential to adversely impact organizational operations (including mission, functions, image, or reputation), organizational assets, or individuals through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service. Also, the potential for a threat-source to successfully exploit a particular information system vulnerability
- An activity, deliberate or unintentional, with the potential for causing harm to an automated information system or activity
- The potential for a threat-source to exercise (accidentally trigger or intentionally exploit) a specific vulnerability

Example of Threats

1 - Hackers

- Break a password or sniff it off the network
- Use social engineering to get a password
- Taking up resources with irrelevant messages
  - Denial-of-service attacks aim aims to disrupt a service by either exploiting a vulnerability or by sending a lot of bogus messages to a computer offering a service

2 - Viruses and some worms

- A **virus** is a self-replicating program that requires user action to activate such as clicking on Email, downloading an infected file or inserting an infected floppy, CD..
- A **worm** is a self-replicating program that does not require user action to activate. It propagates itself over the network, infects any vulnerable machine it finds and then spreads from it further

1.6.3 Attack

- Any kind of malicious activity that attempts to collect, disrupt, deny, degrade, or destroy information system resources or the information itself
- An attempt to gain unauthorized access to system services, resources, or information, or an attempt to compromise system integrity, availability, or confidentiality
- The realization of some specific threat that impacts the confidentiality, integrity, accountability, or availability of a computational resource.

Further Difficulties

- Growing attack surface
  - How to evaluate the risks from 3rd party sw?
  - How to evaluate the interdependencies from the infrastructure?
  - How to evaluate the dependencies from the physical world?
- Difficulties in getting cyber-intelligence information
  - How to exploit information from other stakeholders to counter threats?
  - How to contribute (in a trusted way) to help other stakeholders?

1.6.4 In a Nutshell

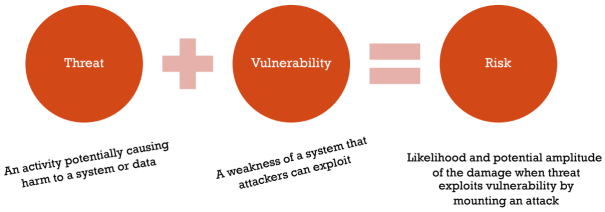


Figure 1.3: In a Nutshell

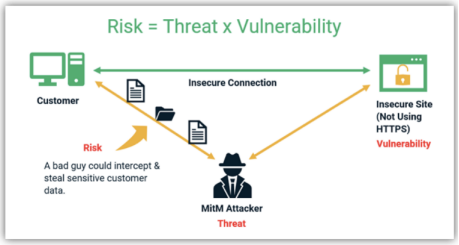


Figure 1.4: Example 1 of risk

1.6.5 In a Nutshell 2

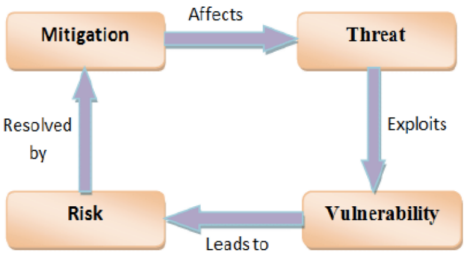


Figure 1.5: In a Nutshell 2

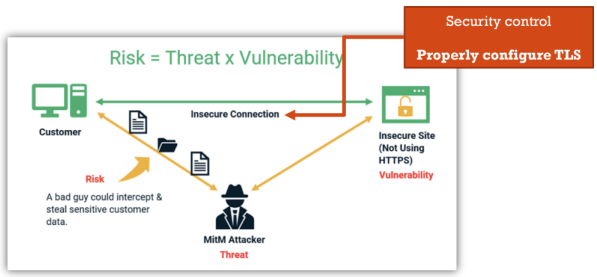


Figure 1.6: Example 2 of risk

1.6.6 Risk

- The probability that a particular security threat will exploit a system vulnerability.
- A measure of the extent to which an entity is threatened by a potential circumstance or event, and typically a function of:
  1. (i) the adverse impacts that would arise if the circumstance or event occurs;
  2. (ii) the likelihood of occurrence.

Note: Information system-related security risks are those risks that arise from the loss of confidentiality, integrity, or availability of information or information systems and reflect the potential adverse impacts to organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, and the Nation. Adverse impacts to the Nation include, for example, compromises to information systems that support critical infrastructure applications or are paramount to government continuity of operations as defined by the Department of Homeland Security.

Remarks on Risk

Risk is the result of threats exploiting vulnerabilities to obtain, damage, or destroy resources together with their impact on the properties in the CIA triad

Threats can be characterized as a combination of:

- **intent** = propensity to attack
- **capability** = ability to successfully attack

Vulnerabilities are characterized by how easy it is to:

- **identify** them
- **exploit** them

Threats and **vulnerabilities** give the likelihood that an adverse event may happen

- Impact should be evaluated with respect to each stakeholder that has an interest in the system under consideration
  - *Example*: unauthorised disclosure of personal information may have catastrophic consequence for the patients involved in the data breach but can be negligible for the organization offering the healthcare service if the number of patients involved is low

1.6.7 Likelihood requires a threat model

- A threat model is a structured representation of all the information that affects the security of an application, i.e. it is a view of the application and its environment through the lens of security
- A threat model typically includes:
  - *What are we working on?*
    - Description of the system to be modelled
  - *What can go wrong under some "reasonable" assumptions?*
    - Assumptions that can be checked / challenged in the future as the threat landscape changes
    - Potential threats to the system
  - *What are we going to do about it?*
    - Controls that can be taken to mitigate each threat
  - *Did we do a good job?*
    - A way of validating the model and threats, and verification of success of controls taken

Threat Model Example

- Description of the system to be modelled
  - Password storage in an application
- Assumptions that can be checked / challenged in the future as the threat landscape changes
  - Offline attacks to passwords are the only security concerns
- Potential threats to the system
  - Decryption of hashed passwords using brute force possible since weak hashing algorithm (MD5) is used
- Controls that can be taken to mitigate each threat
  - Update hashing algorithm to known secure one
- A way of validating model & threats and verification of success of controls taken

Remark on Impact

It is crucial to observe that the impact of an attack cannot be defined in absolute terms but only relative to one of the stakeholders involved in the system/application. This is so because the negative consequences of a given attack are different when considered from different perspectives.

- Nowadays, computers are everywhere and the impact of attacks can be substantial
- Example:
  - Automotive attacks can lead to important consequences both on the vehicles and the passengers
  - For the 2015 attack on a FCA jeep, impact was substantial for all stakeholders
    - Manufacturer obliged to recall 1.4 millions vehicles
    - Drivers and passengers safety put at risk

1.7 Risk Matrix

$Risk = Likelihood \times Impact$

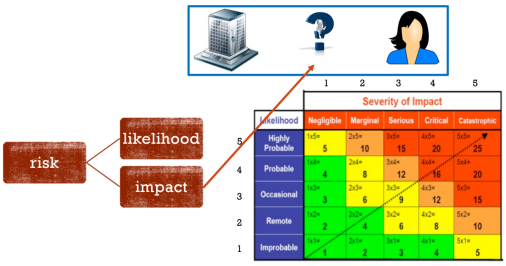


Figure 1.7: Risk Matrix

1.8 Final Remarks on Security

1.8.1 Threat Modeling is Crucial

- Security is characterized by protection against an adversary or, possibly, against some other physical or random process
- Security typically focuses on malicious adversaries
- Core to any consideration of security is the modelling of malicious adversaries
  - **motivations** (in the past... glory, nowadays... money)
  - **capabilities** (intercept messages, modify messages, read keys pressed, ...)
  - **threats** (roughly, negative impact on systems, operations, organization, business, ...)
- Arguing that a system is secure without referring to an attacker model does not make much sense
- Indeed, absolute security does not exist even for so-called air-gapped system

1.8.2 Security Controls

Deploying Security Controls Requires to Consider Several Different Aspects:

- In order to mitigate threats, security proposes controls/mitigation strategies affecting
  - **people** (e.g., rules for setting passwords)
  - **process** (e.g., regularly update software)
  - **technology** (e.g., authentication and access control)
- Controls can be classified in:
  - **preventive**, i.e. before the bad event (e.g., locking out unauthorized intruders)
  - **detective**, i.e. during the bad event (e.g., turning an intruder alert on a dashboard)
  - **corrective**, i.e. after the bad event (e.g., recovering deleted data from a backup)
- Selection of controls is based on:

  - *risk management*: the process of identifying vulnerabilities and threats to the information resources used by an organization and deciding what countermeasures, if any, to take in reducing risk to an acceptable level
  - *considering the human factor*: controls must be easy to use and must neither require stress of mind nor the knowledge of a long series of rules

- Security controls are human artifacts that
  - can mitigate the impact of an attack but in many cases do not avoid it completely
  - can contain vulnerabilities that can be exploited to mount attacks

- In other words, we are left with a (non-null) residual risk, i.e. the amount of danger associated with an attack after risks have been mitigated by security controls
  - Example: automotive seat-belts
    * Use of seat-belts reduces the risk of injury although it does not cancel it as
      · accidents may be quite serious and seat-belts can only mitigate consequences
      · installation of set-belts may be defective and mitigation of risk is reduced

1.8.3 Security and Trust Assumption

Security and Trust Assumption are also crucial

- The role of trust in software
  - **Dependability** = ability to avoid failures that are more frequent and severe than is acceptable
  - **Failure** = an event that occurs when the delivered service deviates from correct service
  - **Trust** = accepted dependence

- Example: SolarWinds attack
  - The attack compromises the infrastructure of SolarWinds, a company that produces a network and applications monitoring platform called Orion, and then uses that access to produce and distribute trojanized updates to the software's users (an instance of a supply chain attack)
  - A trojan is any malware (i.e. a software intentionally designed to cause damage) that misleads users of its true intent

- Impact on 425 of the US Fortune 500, the top ten US telecommunications companies, the top five US accounting firms, all branches of the US Military, the Pentagon, the State Department, and hundreds of universities and colleges worldwide

1.8.4 Security Violation

Security Violation can be traced back to violations of the CIA Triad

What does it mean exactly to "violate security"?

There many possibly answers to the question above including:

- A security violation can be the **unauthorized sharing** of sensitive information such as personal data of patients in a healthcare system
- A security violation can be the **unauthorized modification** of the content of a resource such as modifying the balance of bank account
- A security violation can be the **unauthorized withholding** of the content of a resource or service such as the unavailability of network services

The three answers correspond to three crucial properties characterizing three possible dimensions of security.

2 Authentication

2.1 User Authentication & Digital Identity

Digital Identity An identity whose attributed are stored and trasmitted in digital form

Identity A set of attributes related to an entity

Attribute A characteristic or property of an entity that can be used to describe its state, appearance, or other aspects

Digital identification Can be authenticated unambiguously through a digital channel, unlocking access to banking, government benefits, education, and many other critical services. The **risks** and potential for misuse of digital ID are real and deserve attention.

2.1.1 Digital ID Promises & Properties

- *Verified and authenticated to a high degree of assurance*: High-assurance digital ID meets both government and private-sector institutions' standards for initial registration and subsequent acceptance for a multitude of important civic and economic uses, such as gaining access to education, opening a bank account, and establishing credentials for a job. High-assurance authentication maintains these same standards each time the digital ID is authenticated
- *Unique*: With a unique digital ID, an individual has only one identity within a system, and every system identity corresponds to only one individual
- *Established with individual consent*: Consent means that individuals knowingly register for and use the digital ID with knowledge of what personal data will be captured and how they will be used
- *Protects user privacy and ensures control over personal data*: Built-in safeguards to ensure privacy and security while also giving users access to their personal data, decision rights over who has access to that data, with transparency into who has accessed it.

2.1.2 Risk of digital ID

First, digital ID is inherently exposed to risks already present in other digital technologies with large-scale population-level usage. Indeed, the connectivity and information sharing that create the value of digital ID also contribute to potential dangers.
Technological failure could include problems with the functionality of the hardware or software associated with a digital ID as well as infrastructure problems preventing uninterrupted and effective system use.

Second, some risks associated with conventional ID programs also pertain to digital ID. They include human execution error, unauthorized credential use, and the exclusion of individuals. Digital ID could meaningfully reduce those risks by minimizing opportunity for manual error or breaches of conduct.
High-assurance digital ID programs also reduce the risk of forgery and unauthorized use, which are relatively easier with conventional IDs, like driver's licenses and passports. Furthermore, some risks associated with conventional IDs will manifest in new ways as individuals use digital interfaces.

2.1.3 Individuals use digital ID

Individuals use digital ID in 6 roles to interact with institutions and create share value

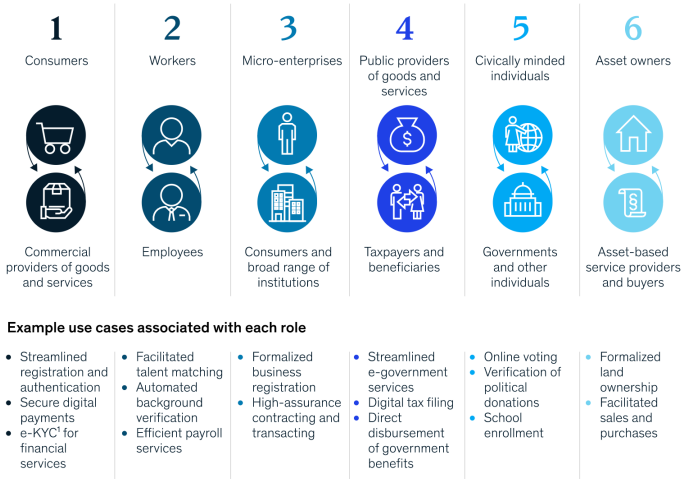


Figure 2.1: Digital ID in 6 roles

2.2 Digital Identity Lifecycle

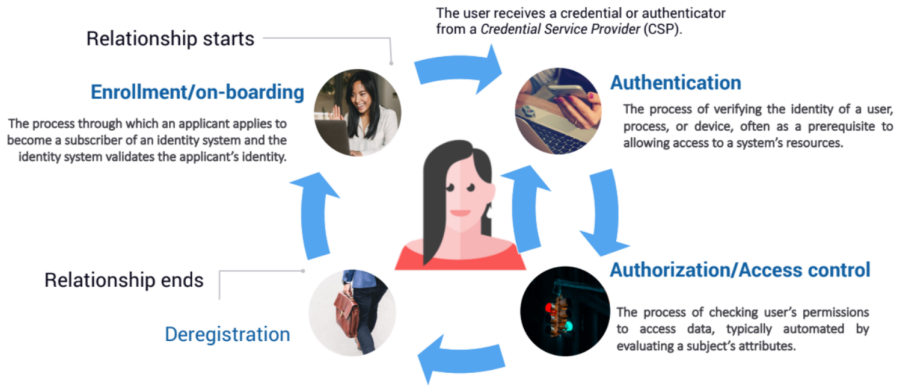


Figure 2.2: Digital Identity Lifecycle

2.2.1 Enrollment / On-Boarding

Resolution

- Collection of Personal Identifiable Information (PII) from the applicant
  - Examples: name, address, date of birth, email, and phone number
- Collection of two forms of identity evidence
  - Example: driver's license and passport by using the camera of a laptop

Validation

- Validation of the information supplied above by checking an authoritative source
- Check of the images of the license and the passport for alterations, coherence of the information on the documents and compliance against standard
- Check with the issuing sources for the license and passport and validates the information matches

Validation

- Ask the applicant for a photo of themselves to match to the license and passport
- Matching the pictures on the license and the passport to the applicant picture
- Sending an enrollment code to the validated phone number of the applicant, the user provides the enrollment code, and check if they match, verifying the user is in possession and control of the validated phone number
- The applicant has been successfully identity proofed

Identity Assurance Level

A category that conveys the degree of confidence that the applicant's claimed identity is their real identity

IAL 1 : Attributes, if any, are self-asserted or should be treated as self-asserted

IAL 2 : Either remote or in-person identity proofing is required

IAL 3 : In-person identity proofing is required. Identifying attributes must be verified by an authorized respresentative through examination of physical documentation

2.2.2 Digital Authetication

Digital Authetication The process of verifying the identity of a user, process or device. The **claimant** must demonstrate to the **verifier** that is indeed the one that claims to be.

Logging Insertion of

- *Username*, to annunce who you are
- *Password*, to prove that you are who you claim to be

This type of 'authentication' is called **user authentication**: the process of verifying a claimed user identity.
Authentication by password is widely accepted and not too difficult to implement (although it may be tricky).

In the context of web authentication No single technology is likely to solve the authentication problem perfectly in all use cases. A synergistic combination of several different techniques is more likely to succeed and cope with the heterogeneous requirements arising in different use cases (such as social networks, financial services, healthcare services, ..).
In practice, several different authentication processes use passwords in combination with other techniques including additional authentication factors (e.g., biometrics or One Time passwords) and Machine Learning for risk based authentication

2.3 Passwords: Attack & Migration

2.3.1 Threat Model for Cracking Passwords

On-line It can be easily mitigated, it is sufficient to set a maximum number of attempts and then to block the authentication procedure for a certain interval of time to make brute-forcing practically impossible.

Off-line

- Attackers is able to steal the file / database where passwords are stored
- The attacker can then be able to carry a brute-force attack
- It becomes crucial to adequately protect the passwords in storage...
- ... unfortunately, still today, there are several cases in which the protection is not implemented (i.e. passwords are stored in clear) or implemented in the wrong way (e.g., the hash functions used are weak or vulnerable to attacks)

Guessing Password

- **Brute force** (exhaustive search): try all possible combinations of valid symbols up to a certain length
- The problem is that nowadays a lot of computational power is available at a reasonable cost
- This is so for several different reasons including
  - Moore's law: the number of transistors that could be housed in a dense integrated circuit doubles about every two years
    - this phenomenon suggests that computational progress will become significantly faster, smaller, and more efficient over time
  - possibility to rent hw from cloud service providers (e.g., Amazon)
- **Being smarter**: search through a restricted name space
  - Ex: passwords associated with a user like name, names of friends and relatives, car brand, car registration number, phone number,..., or try popular passwords
  - Typical example: dictionary attack, i.e. trying all passwords from an on-line dictionary
- You cannot prevent an attacker from accidentally guessing a valid password, but you can try to reduce the probability of a password compromise

2.3.2 Mitigation

Change default passwords!

- Often passwords for system accounts have a default value (e.g., admin)
- Avoid guessable passwords:
  - Prescribe a minimal password length
  - Password format: mix upper and lower case, include numerical and other non-alphabetical symbols
  - Today on-line dictionaries for almost every language exist

2.4 Recap

2.4.1 Hash function

Particular process of hash function that are used to obtain the digest, which is a particular collection of bits that represent in a summarized form the input.
In order to check if 2 very large file are equal, do not compare bit-bit (Large File), just compute the 2 digest (Summarized form 256/512 bits).
This is possible thanks to the HASH function properties (for instance that is very cheap to compute hash function, so you obtain very quickly the output that are robust against collisions). The main property is that it's very computationally heavy to invert.
So if I apply hash function to the passwords, I can forget about them and keep in memory only the digest, although the brute force attack is still possible.
So I need to add something to protect the vulnerability of my password (MFA):

- Knowledgd
- Inherence (fingerprint)
- Possession (Smartphone)

I need to prove to the verifier of the authentication procedure that you control this authenticator, which stores and handles this addictional authentication factor.

OTP is base on challenge-response protocol, so the server that needs to check the authentication procedures issues a challenge: insert something (For istance password, pin), once returns to the verifier, it is able to check that this manipulation is made by the same user that inserted the password.
If both factors are checked and verified and associated with the same user, the authentication procedure is accepted. So in this case, the attackers has 2 factor to compromise. (Stealing OTP is not 0, but very low.)

2.5 FIDO

Namely, **F**ast **ID**entity **O**nline.

Main idea: Avoid password all together replacing password with cryptographics technics, tipically encryption and decription (mainly encryption). It is all stored in a hardware token

- No password needed
- No human interaction (Just insert usb)

"Is it a good idea that an authentication not require any interaction by the user? "

You need to be sure that the user is there, present of the user is a base requirement o the authentication procedure. If you don't have it, you are in trouble: Anyone can take over your NB or Smartphone.
A solution can be biometric authentication over the hardware token.

2.5.1 FIDO - Phising Resistant Authentication

1. The relying party needs to be contacted by the authenticator(that needs to be registered with the relying party) at that pont the relying party checks if the request is admissionable for its policies;
2. The authentichator sends to the relying party a request to register
3. Relying Party does a check that the request is issued a sufficient protecte device ?? 23:03
4. The authenticator generates a couple of key, mathematically related, one private and one public:
   - **Public key** is used to encrypt
   - **Private key** is use to decrypt, it never leave the authenticators
   - Even if you know the public key it is impossible (very computationally difficult) to guess private key
5. Private Key immediatly store in the safest place avaiable in the auht
6. Auth send to the RP a public key (generated in point 4) with its univoque ID
7. RP stores [ID and Private key]
8. Private key never leaves the authenticator

In this way we are replacing the OTP with a challenge protocol that generates a challaìenge contains some kind of data that is unique to the transaction to perform.

2.6 Outsourcing Authentication

Practically you use an **external service** to log in into a specific service. For example to access to the public amministration services you use spid. You have to choose from a list one of the digital ID provider in which you are already enrolled in.

2.6.1 Problems due to design its own authentication procedure

- Eterogeneity
- Quality of the solution
- Secure recepits for citizen (difficult for old people)

In this way:
Public Amministration can focus on their core business, offering their services.
Digital ID Provider can focus secure Digital ID.

The single point of failure is when someone infiltrates in one of Digital ID providers, will be a big problem.
This technology allows us to get access to a large set of services using just one set of credentials, so it's important to create one strong password.

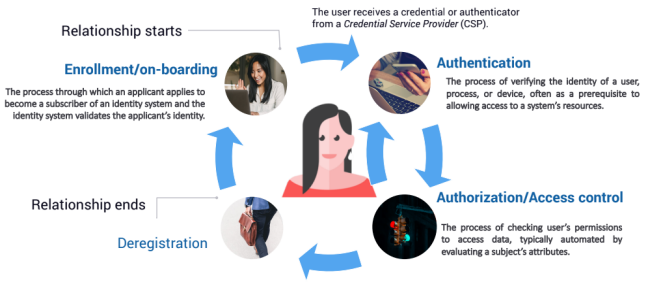


Figure 2.3: Outsourcing Authentication Problems

2.6.2 Outsourcing Authentication

The idea is that the user has to authenticate to the ID Provider, which has to proof that the user has been successfully authenticated.

2.6.3 Solution

Design an authentication protocol, that involve 3 different entities:

- ID Provide
- Serivice that the user is authenticating on
- The Frontend or client

How it works

Using Single Sing On (SSO): Just one set of credentials to access at several different applications or services. All the services trust the same ID provider, which proof the correct authentication.

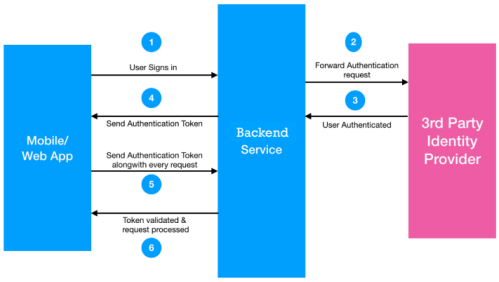


Figure 2.4: Outsourcing Authentication Solution

1. User uses the Frontend to sing in
2. The backend service forwards the authentication request to the third party
3. The third party Checks the user authentication procedure
4. The backend service issues an authentication token(proof)
5. The token is sent back to the client
6. From there on, all the requests will be complemented with this authentication token

This is a distributed solution, not only one single identity (Split Client and ID provider). Nevertheless should be a problem in case an attacker steals the authentication token and uses it(pretends to be another person).
Once you get the authentication, your **session** will expire (for example in 4h), it avoids to insert a lot of time your credentials in a day.

2.6.4 WRAP UP

Authentication is the first security service, it is crucial to guarantee several different properties of CIA Tirad(In particular the correct management of the digital ID.
It can be the front door to access to your system or application.

- LAN base: someone in your network or inside your VPN, can do an offline attack based on brute force
- Phishing: Targeting the user and not the technical part
- Man in the middle: Spoof messages throwing over the network, over the internet if appropriate protocol are not implemented:
  - HTTP protocol provides 0 security, everything is in clear.

3 M-F Authentication Analisys

3.1 Problems of Password - Recap

Guessing:

- Brute force: try all possible combinations of valid symbols up to a certain length
- Dictionary attack: try a large number of commonly used usernames/passwords

Credential stuffing: attackers take billions of email addresses and corresponding cracked passwords from compromised databases and see how many of them work at other online services. (Reuse of the same password for multiple services)

Phishing/Social engineering: a user voluntarily sends the password over a channel, but is misled about the end point of the channel

Resetting technique

Storing (plaintext, without hash/salt, weak hash functions...)

3.2 Multi Factor Authentication

Authentication process based on multi-factor **authenticators** that attest more than a single **authentication factor**.

Authentication Factor: the three types of authentication factors are something you know, something you have, and something you are.

Authenticator: something the claimant possesses and controls that is used to authenticate the claimant's identity. Every authenticator attests one or more authentication factors.

Single Factor

- passwords need to be known, so they attest a knowledge factor
- (enrolled) apps need to be possessed, so they attest an ownership factor

Multi Factor

- advanced smart cards (e.g., eID cards) may require both the possession of the plastic card and the knowledge of the PIN
- USB tokens may require both the possession of the device and the use of a biometric factor (e.g., a fingerprint)

Identity Assurance Level (link): A measure of the strength of an authentication mechanism and, therefore, the confidence in it, as defined in NIST

3.3 MUFASA

- **Mu**lti-**F**actor **A**uthentication **S**pecification and **A**nalysis
- Automatic tool for the security analysis of multi-factor authentication procedures
- Starting from a high-level specification provided through a questionnaire, evaluates the security of the modelled protocol in terms of resistance against a set of potential attackers

3.3.1 MuFASA Phases

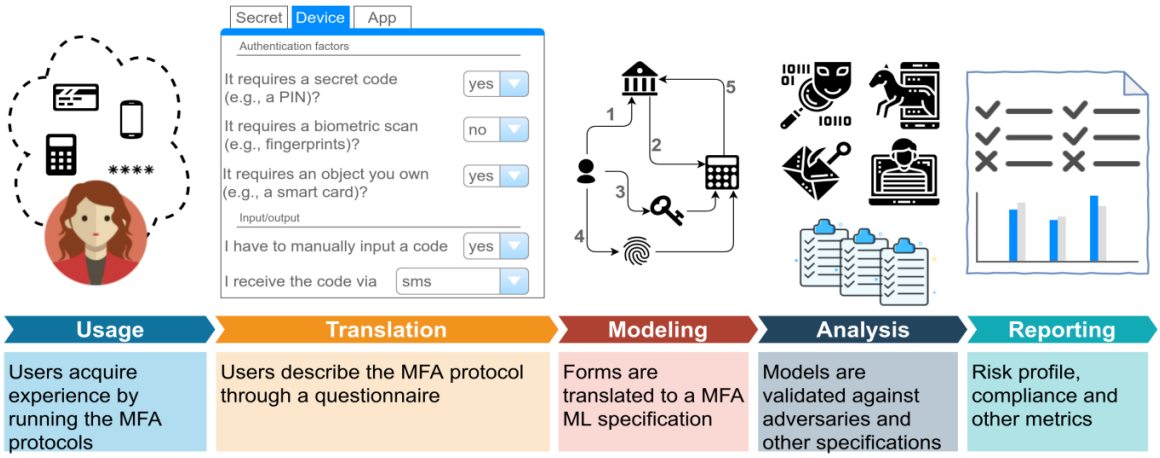


Figure 3.1: MuFASA Phases

MuFASA Phase 1 - Usage

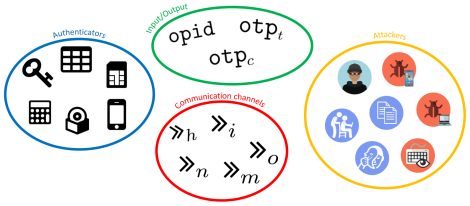
User acquire experience by running the MFA protocols

MuFASA Phase 2 - Translation

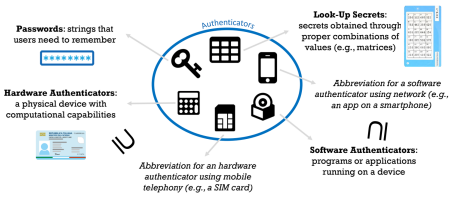
Questions based on the usage of the service to do the analysis, need to select the things to do, similar to a questionnaire

MuFASA Phase 3 - Modelling

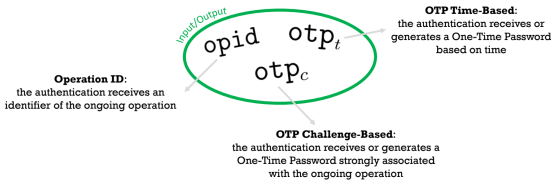
Use of the internal language of MuFASA



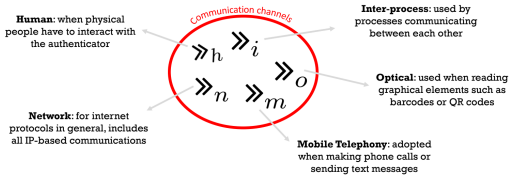
Authenticators



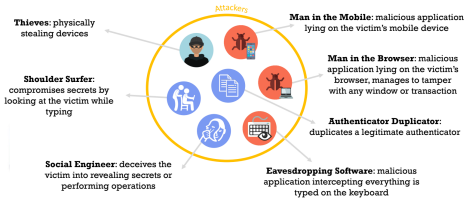
Input/Output



Communication Channel

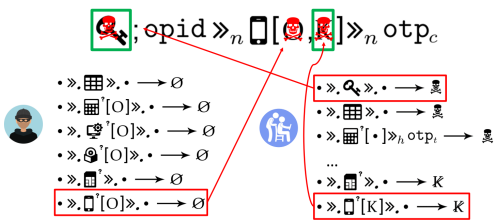


Attackers



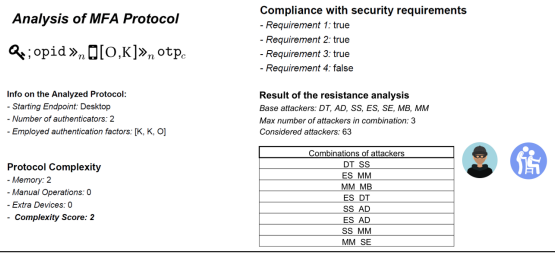
MuFASA Phase 4 - Analysis

Collect and merge info from previos phases. Analize all the combination starting from attackers. **details (link)**
"What can attackers do using info I had collected before?"



MuFASA Phase 5 - Report

Report what MuFASA had found



3.4 MuFASA Analysis Methodology - Phases 4

1. **Security Analysis** To detect the attackers that manage to compromise the protocol (Manage the attackers)

2. **Risk Analysis** To evaluate the risk connected with the successful attackers detected (Priritize attackers, choose the most urgent to fix)

Likelihood: Difficult to access
Impact: Damage possibility

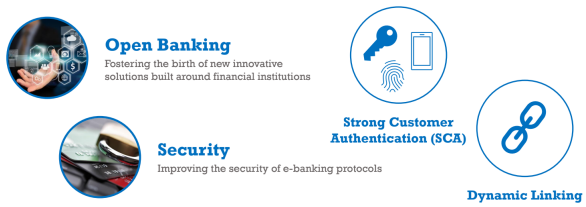
3.5 MuFASA Evaluating the Risks

- **Static Analysis**: the values used to compute the risk are static, thus they do not depend on the considered scenario.
- **Flexible Analysis**: the values used to compute the risk start from static references, but then they are adjusted in order to fit as best as possible the considered scenario. (This can be done, for instance, through a questionnaire)

DA FINIRE SLIDE 35-76

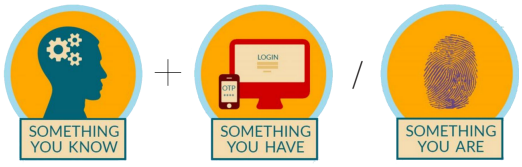
3.6 PSD2 - Payment Services Detective

Directive (EU) 2015/2366 regarding payment services in the internal market.



3.6.1 PSD2 - Strong Customer Authentication

Authentication relying on more than a single authentication factor:



3.6.2 PSD2 - Dynamic Linking

During a transaction, the authentication code must be strongly connected with the ongoing operation. So if the operation info are stealed, the risk is inherent at the ongoing operation, nothing more. Moreover, the user is always displayed the operations' details before the authorization.

3.7 Conclusion

MuFASA: automatic tool for the security analysis of multi-factor authentication procedures

- Security: detect the attackers that manage to compromise the protocol
- Risk: evaluate the risks connected with the successful attackers
- Usability and Compiance checks

Analysis for evaluate **existing solutions** + **what-if analysis** for the design of **new solutions**

4 Cryptography

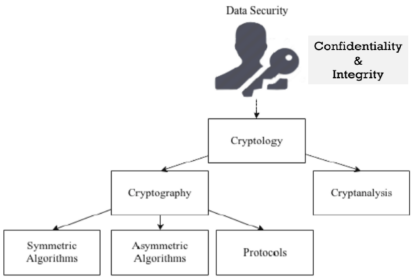
Cryptography is the science and study of secret writing (*Security Mechanism)*, the design of new approaches (create cryptographic primitives and protocols)

Cryptanalysis is the science and study of methods of breaking ciphers, a set of technics that allows to analyze the security of cryptographic primitives.

Cryptology cryptography and cryptanalysis

Today: Cryptography is the study of mathematical techniques related to aspects of information security, the goal is to provide:

- **confidentiality** (i.e. secrecy)
- **data integrity** (i.e. no edits)
- entity authentication
- data origin authentication



Cryptography is a security mechanism that includes a set of techniques for **scrambling or disguising** data. So that it is available only to someone who can restore the data to its original form. In current computer systems, cryptography provides a strong, economical basis for offering data security as a security service, i.e. a service for keeping data secret (**confidential**) and for verifying data **integrity**.

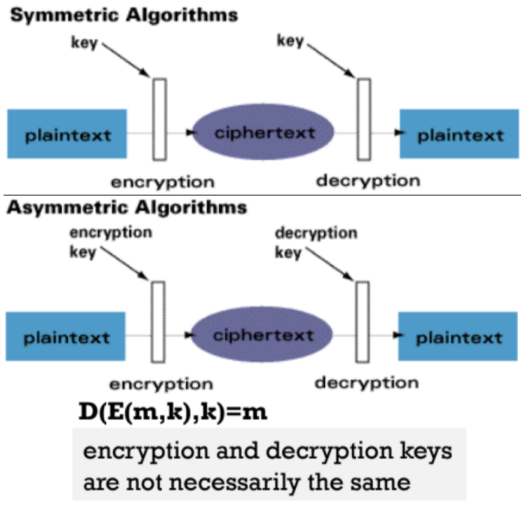
4.1 Cryptosystem

A cryptosystem is a **5-tuple** (E,D,M,K,C):

- **E** is an encryption algorithm
- **D** is a decryption algorithm
- **M** is the set of plaintexts
- **K** is the set of keys
- **C** is the set of ciphertexts

E and D can be characterized as **functions**:

- E: M × K → C
- D: C × K → M



Properties of Function If you apply *decryption* to an encrypted message with the K key, and later you will decrypt with the same k key, you should get back exactly the same plain text(for symmetric cryptology), it is the **Kerckhoffs' principle**:
Do not rely on the secrecy of algorithms, *the key is the only secret that needs protection.*

4.1.1 Application: Communication Security

The scenario is when you are exchanging messages through an internet channel, in the picture [4.1] Channel C is not secure, because there's an attacker ready to spoof. So Alice will send to Bob a ciphertext, once receive he has got the key to decrypt the ciphertext and will obtain the plaintex. Of course, the key must be transmitted in a secure channel. The trick to guarantee confidentiality, is that the receiving part (Bob) knows the secret key, so he will easily invert the chipher text. Here there's a one-way function (*"One way Trap Door Function"*), if you know the secret, in this case the key, it's easy to invert the function.

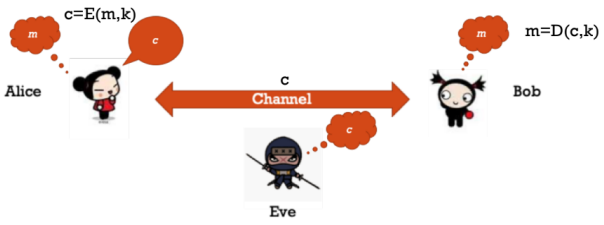


Figure 4.1: Application 1: Communication Security

4.1.2 Application 2: Data in Cloud

Other scenario is **Data at Rest** & **Data in Usage** In case of searchable encryption you have a database in cloud and it's not a good idea to put data in clear, because the cloud service providers can watch inside your environment, they do this because their business is profile user (for example to personalize ads).
You need to compute a complex function on this data set in a distributed way, only the result is disclosed to the right stakeholder of the computation:
SMPC Secure Multi-Path Computation. it is your duty to protect your data, applying a security level between the phyisical cloud infrastructure and your client to access to them.

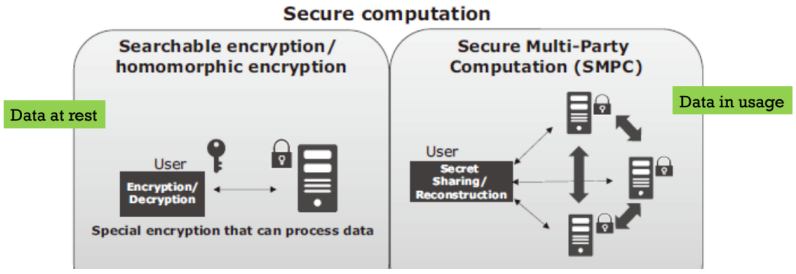


Figure 4.2: Application 2: Data in Cloud

4.1.3 Other Problems to consider

There's another part, which often hasn't been consider by cryptographer(they work on mathematical and theoretical level), they don't pay too much attention: It about the **use** and the **implementation** of this function. For example the implementation can be difficult to handle because you've to manipulate a lot of integers, need to use libraries, etc.. Of course bugs and overhead can be generated very easily. Also the key generation can be a problem in terms of heavy computation. So it's not a good idea to implement own cryptographic primitives and use an available library.

4.2 Key Distribution

- The two parties to an exchange must share the same key, and that key must be protected from access by others
- Frequent key changes are usually desirable to limit the amount of data compromised if an attacker learns the key
- The strength of any cryptographic system rests with the key distribution technique
- Example approaches for entities A and B:
  1. A key could be selected by A and physically delivered to B
  2. A third party could select the key and physically deliver it to A and B
  3. If A and B have previously and recently used a key, one party could transmit the new key to the other, encrypted using the old key
  4. If A and B each have an encrypted

In case of geographical distributed you are in trouble, key distribution can be a problem.

4.3 About "Computationally Secure"

- An encryption scheme is computationally secure if the ciphertext generated by the scheme meets one or both of the following criteria:
  - **Cost** of breaking the cipher exceeds the value of the encrypted information
  - **Time** required to break the cipher exceeds the useful lifetime of the information
- Very difficult to estimate the amount of effort required to cryptanalyze ciphertext successfully
- Assuming there are no inherent mathematical weaknesses in the algorithm, then a brute-force approach is indicated, and so one can make some reasonable estimates about costs and time
- **Brute-force** approach = trying every possible key until an intelligible translation of the ciphertext into plaintext is obtained (On average, half of all possible keys must be tried to achieve success)
- The same cryptographics technics that allow to guarantee integrity, in some years, this technics could not be still secure, because the computational power will increase.

4.4 Cryptography is no Silver Bullet

- Cryptography is not a the solution to your security problem
  - It transforms a problem into another
- Cryptographic keys are sensitive data stored in a computer system
  - Access control mechanisms in the computer system have to protect these keys
- Cryptography (alone) is rarely ever the solution to a security problem
- Cryptography is a translation mechanism
  - usually converting a communication security problem into a key management problem
  - ultimately into a computer security problem or into the security of a cryptographic protocol for key distribution

4.5 Encryption

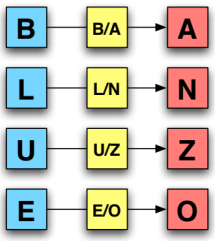
Two types of transformations (mechanisms):

- **Substitution**: each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element
  - In other words, letters are replaced by other letters
- **Transposition**: elements in the plaintext are rearranged
  - In other words, same letters but arranged in a different order

A fundamental requirement: no information shall be lost (i.e., all operations be reversible). Most encryption systems use multiple stages of substitutions and transpositions.

4.5.1 Substitution 1 - substitution cipher

- Substitutes one symbol for another
- The key is the substitution
- It is a method of encrypting by which units of plaintext are **replaced** with ciphertext, according to a **fixed system**.
- The "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above
- The receiver deciphers the text by performing the inverse substitution
- Very few possibility for substitution
- The frequencies analysis is the main threat



4.5.2 Substitution 2 - caesar cipher

Substitution cipher in which every character is replaced with the character, (e.g., shift three slots to the right).
The **key** is the number of characters to **shift** the cipher alphabet (In this case three).
It is also called **ROTk** for *rotate* by k

Figure 4.3: caesar cipher

ROTk Mathematically

1. Translate all of our characters to numbers: 'a' → 0 'b'→1 'c'→2, ... , 'z'→25
2. Represent the ROTk encryption function, $e(x)$, where x is the character we are encrypting, as: $e(x) = (x + k)(mod 26)$, where k is the key (the shift) applied to each letter
3. After applying this function the result is a number which must then be translated back into a letter
4. Decryption function: $d(x) = (x - k)(mod 26)$

The brute force attack requires maximunm 25 attemps

4.5.3 Substitution 2 - vigenere cipher

- It can be seen as a generalization of the Caesar cipher whereby several Caesar ciphers in sequence with different shift values are used
- Assume to associate the letters a, b, c, ..., z with the numbers 0, 1, 2, ..., 25
  1. First, **select a keyword**(LEMON)
  2. Then, **repeat** it up to padding and have the same size of the plain tex 2 time at (here there are 2 entire words and a part of the 3rd)
  3. Finally, **encrypt each plaintext letter using a Caesar Cipher**, whose key is the number associated with the keyword letter written beneath it, for example

A	T	T	A	C	K	A	T	D	A	W	N	plaintext
L	E	M	O	N	L	E	M	O	N	L	E	keyword repeated to match plaintext length
L	X	F	O	P	V	E	F	R	N	H	R	ciphertext

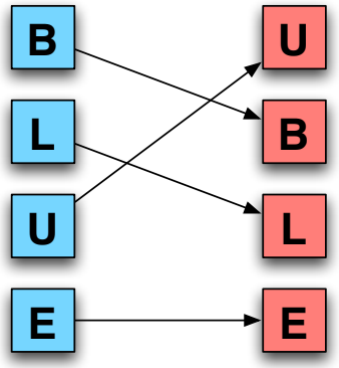
1st column
- A associated to 0
- L associated to 11
- 0+11 mod 26 = 11 associated to L

2nd column
- T associated to 19
- E associated to 4
- 19+4 mod 26 = 23 associated to X

- Every time you consider different letter, you use different shift, induced by the keyword selected at the beginning
- Notice that the same letter A is encrypted with 4 **different** letters, namely L, O, E, and N depending on the position of A in the plaintext
- Notice that letter T occurs 3 times but it is mapped to 2 different letters
- **frequency analysis more difficult** on this cipher than with Caesar cipher
- Since the key is the keyword, its length l determines the size of the key space (l^{26})
- For increasing length l of the keyword, brute forcing becomes increasingly complex until it becomes impossible:
  - when the length of the keyword is the same as the length of the plaintext, a separate Caesar Cipher is used to encrypt each plaintext letter, which makes it impossible to determine the correct plaintext without the key
- In this case, Vigenère cipher can't be broken, if additionally the key is used only once
- it is secure from the point of view of computation attack, but every time you change the message, you have to change the key, otherwise frequent analysis is still a threat

4.5.4 Transposition 1 - transposition ciphers

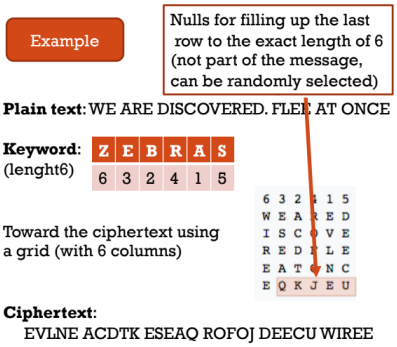
- No more replacing, but scrambling the symbols maintaining the same ones, to produce output
- The key is the permutation of symbols
- In a transposition cipher, the units of the plaintext are rearranged in a different and usually quite complex order, but the units themselves are left unchanged
- Several variants possible



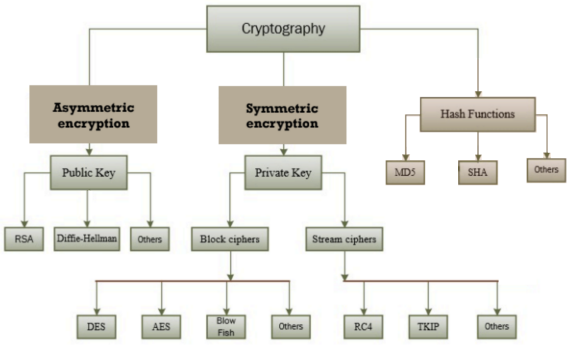
- Comparison with substitution ciphers:
- In a substitution cipher, the units of the plaintext are retained in the same sequence in the ciphertext, but the units themselves are altered

4.5.5 Transposition 2 - columnar ciphers

- In a columnar transposition(permutation), the message is written out in rows of a fixed length, and then read out again column by column, and the columns are chosen in some scrambled order
- Both the width of the rows and theì permutation of the columns are usually defined by a keyword
- For example, the keyword ZEBRAS is of length 6 (so the rows are of length 6), and the permutation is defined by the alphabetical order of the letters in the keyword (In this case "6 3 2 4 1 5")

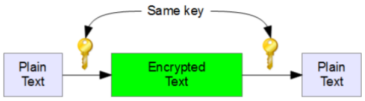


4.6 Overview on Modern Cryptography



4.7 Symmetric keys Cryptography

Symmetric keys, where a single key (k) is used is used for **E** and **D**: $D(k, E(k, p)) = p$
Note: Management of keys determines who has access to encrypted data

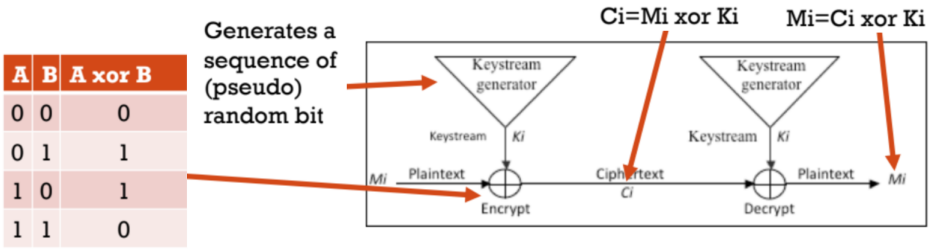


Assuming symmetric encryption uses the same key for (en)decryption, 2 main classes:

- **Straeam ciphers** are an elaboration of the Vigenere cipher's idea, the encryption is very easy. It encrypts sequences of "short" data blocks (Tipically 1 bit/byte) under a changing key stream, the encryption can be quite simple, (e.g., XOR)
- **Block ciphers** is based on combining, permutation and substitution in a very complex way a "long" data blocks (tipically 64bits) without changing the key, that allow one to generate resulting block of the same size, but it's very difficult to invert to get the plain text, without knowing the key

4.7.1 Symmetric Encryption - Stream Cipher

- Operates on **streams** of plaintext and ciphertext **one bit at a time**.
- The same plaintext bit will encrypt to a different bit, every time it is encrypted
- Stream ciphers can be designed to be exceptionally fast, much faster than any block cipher in hardware, and have less complex hardware circuitry
- Encryption is done in a very simple way: just using XOR operation



1. You take **one bit at the time** in the input message (consider a message as a continuous stream of bits)
2. You have the pseudo-random "**keystream generator**"
3. Execute the **XOR** between 1 and 2
4. the results of the XOR operation (3) is the **ciphertext**

So, the nice thing here is that the decryption is very cheap to implement (same encryption operation). Therefore if you are able to generate the same keystream at the receiving side:

- You take the same stream that you have used to generate the first bit of the ciphertext
- You XOR it with the ciphertext
- You will obtain the initial plaintext

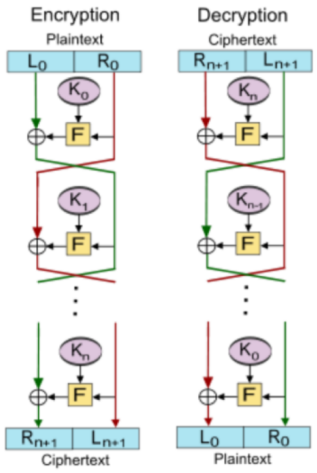
The issue is the generation of the **same** random stream (by keystream generator) at the receiving side, in order to be able to decrypt it:
It is a *pseudo-random* bit generation, hence there is a function that takes a seed (sequence of bit, 8, 16, etc), and if it is applied to the function, it generates one bit as result. This function takes a seed and returns another seed, from that seed peaks one of the bit of the output seed in order to generate the bit stream. Since the function is deterministic, starting from the same seed, you are able to generate exactly the same sequence of bits at the sending side and the receiving side.
Again the stream cipher is a **translation mechanism**, you transform the problem of encryption into the one of generating a pseudo-random stream of bits.(For security reasons are preferred long sequences of bits).
Tipically used in communication and video real-time applications (GSM, BLT, SSL).

4.7.2 Symmetric Encryption - Block Cipher

- A block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. (Typical size of the block between 64 and 256 bits)
- A block cipher breaks(splits) the message (M) into successive blocks M1, M2, M3,..., Mn, and enciphers each Mi with the same key k
- Data Encryption Standard (**DES**)(old and deprecated in 2005) and the Advanced Encryption Standard (**AES**), are well known examples of block cipher systems, it is a gold standard for security (considered the most secure symmetric cipher available nowadays)

DES - Feistel

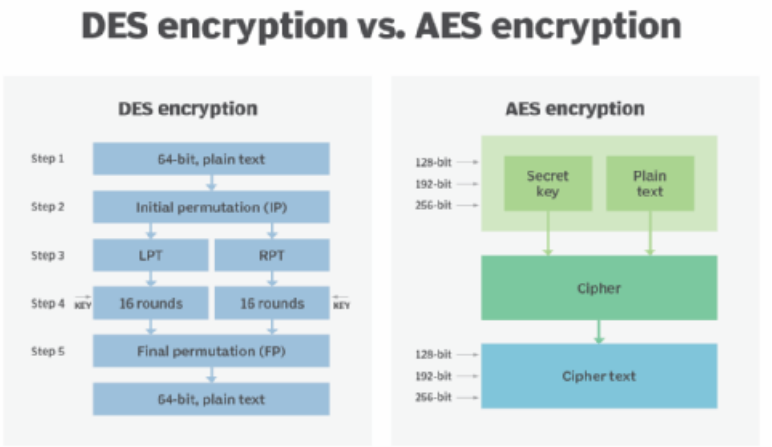
- It is a block cipher, i.e. a deterministic algorithm operating on fixed-length groups of bits, called a block, with an unvarying transformation that is specified by a symmetric key
- How it works
  1. The block of plain text to be encrypted is split into two equal-sized halves (L,R)
  2. The round function F (substitution) is applied to one half, using a subkey K. Then the output is XORed with the other half
  3. The two halves are then swapped
- DES has 16 rounds
- Instead of using the same key in each round, a different subkey is derived from the main key
- When decrypting, subkeys must be used in **reverse order**



1. Split input 64 bits into R and L parts of 32 bits each
2. Compute X=F(R,K)
3. Compute L xor X
4. Let the new 32 bits R be X
5. Let the new 32 bits L be R

AES - After DES

- Advanced Encryption Standard (**AES**) that became the official successor to DES in December 2001
- AES uses a symmetric key crypto scheme called Rijndael, a block cipher
- The algorithm can use a variable block length and key length
- The latest specification allowed any combination of keys lengths of 128, 192, or 256 bits and blocks of length 128, 192, or 256 bits
- The encryption process is based on a series of table lookups and XOR operations which are very fast operations to perform on a computer
- Decryption consists of conducting the encryption process in the **reverse order**
- However, unlike for a Feistel Cipher, the encryption and decryption algorithms do have to be separately implemented, although they are very closely related
- It is more expensive to implement than DEs, because you have to re arrange the rounds and change the rounds in order to perform the suffering, but the idea is similar to DES
- The lesson learnt was the fact that DES had a fixed size key
  - In AES, there are approximately: 3.4 x 1038 possible 128-bit keys; 6.2 x 1057 possible 192- bit keys; and 1.1 x 1077 possible 256-bit keys.
  - In DES, there are approximately 7.2 x 1016 possible DES keys
  - There're on the order of 1021 times more AES 128-bit keys than DES 56-bit keys

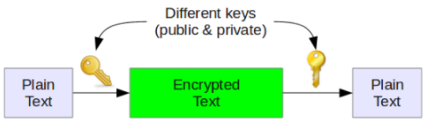


4.8 Asymmetric keys Cryptography

- Also called *"Public key cryptography"*, you don't use anymore the same key for encryption and decryption, the keys are distinct but mathematically related.
- This kind of cryptography is based on hard to solve mathematical problems(they generate a lot of overhead)
- According to the order in which you use the keys, for encryption & decryption, you can guarantee either confidentiality or integrity
- Viceversa, there's no way for the kind of the symmetric chiphers that we have seen to guarantee confidentiality, for this reason they are called **malleable**: even though an attackers cannot decrypt the message, it can tamper the message and the receiving side is not able to get the entire message, hence the integrity is not guarantee.
- In the end, the symmetric cryptography can guarantee only secrecy, instead asymmetric one can guarantee a lot of more. (i.e. confidentiality & integrity)
- It's computationally hard, so for the large messages isn't the best way (hash functions)
- To guarantee the **integrity**, we sign the message to guarantee the authenticity
- **non-repudation**, namely the sender cannot deny having sent the message

Public Key Cryptography - Basic Idea

Use 2 keys: one to encode and the other to decode such that they're mathematically related although knowledge of one key doesn't allow someone to easily determine the other key, even if you know one of the two keys, you can't infer anything about the other one.



The secret key should be protected in the hands of who are in title of use it, instead the public key doesn't need to be protected, but can be widely distributed. It doesn't matter which key is applied first, but that both keys are required for the process to work

Confidentiality Alice wants to send a message to Bob

1. A asks to B for his public key
2. B sends his public key to A
3. A encrypt the message using the public key of B (sign the message)
4. A sends the resulting chiper text to B
5. Only B it will be able to decrypt the message with his secret key (No one else)

In this scenario we are only guaranteeing confidentiality, for Bob there is no way to get another level of assurance about the received message is really coming from Alice.

4.8.1 Asymmetric Encryption - RSA

RSA, namely **R**ivest, **S**hamir, **A**dleman

- Used in hundreds of software products and can be used for key exchange, digital signatures, or encryption of small blocks of data
- Used in hundreds of software products and can be used for key exchange, digital signatures, or encryption of small blocks of data
- Mathematical "trick" of RSA: relatively easy to compute products compared to computing factorizations
- RSA uses a variable size encryption block and a variable size key
- Key-pair is derived from a very large number, n, that is the product of two prime numbers chosen according to special rules
- Primes may be 100 or more digits in length each, yielding an n with roughly twice as many digits as the prime factors
- An attacker cannot determine the prime factors of n (and, therefore, the private key) from this information alone and that is what makes the RSA algorithm so secure
- The ability for computers to factor large numbers, and therefore attack RSA scheme, is rapidly improving and systems today can find the prime factors of numbers with more than 200 digits
- Nevertheless, if a large number is created from two prime factors that are roughly the same size, there is no known factorization algorithm that will solve the problem in a reasonable

RSA - Factorization

RSA is working in this way:

- Take 2 large prime numbers (100 digits) **p**, **q**
- Multiply these 2 numbers, obtaining **N**,the *modulus* $N = p * q$
- Choose a third number **e** relative prime to $(p-1)*(q-1)$
- Find **d**, $d * e = 1 \quad mod(p-1)*(q-1)$, so d is the inverse of $mod\ e$
- Set the **public key** to N,e
- Set the **private key** to d

RSA - A Recent Problem

You should be very careful in picking the 2 primes for the modulus operations that allows you to derive the private and the public key, otherwise you're in trouble. There was an hw device that generates a pair of keys, inserted in a smart card. the keys generation was implemented in a wrong way and the factorization of the primes was possible in some situations

RSA - Integrity

Idea to guarantee integrity

1. You take the message
2. You run the message through the **hash function**
3. You get the digest, a very compact summary of the message
4. The digest is passed to the **signature algorithm** (secret key of the sender to encrypt)
5. You get the resulting chiphertext, called a signature
6. You attach this signature to the original message
7. This pair is the overall message that will be send

At that point, if you want confidentiality you can encrypt the pair (signature and message) with the public key of the receiving entity.

How to check integrity

1. You receive the pair and then you extract the first part of the message
2. You run it through the same hash function that generated the signature and you will get the digest
3. You take the signature and you decrypt with the public key of the sender
4. You get the original digest

Only if the two digest obtained are equal, integrity is guarantee.

Considerations If you use this schema and combine it with the usage of public keys the resulting message, it isn't a good idea, because if the message is very large, you are in trouble (public key chipers are very very slow).

4.8.2 DH Diffie & Hellman

- After RSA algorithm was published, D&F came up with their own algorithm
- DH is used for secret-key exchange only, not for authentication or digital signatures

DH - Idea

You can take one of the 2 primes that generate that particular large prime, as the private key. Even knowing the public key, since is very very difficult factorize large primes, it is also hard to get the private key back.

DH - Scenario

1. Alice fixes a large (random) number **a** (will be the exponent)
   - this is Alice's **private key**
2. Alice computes $A = g^a mod(p)$
   - this is Alice's **public key**
3. Exchanges the public key with Bob
4. Computes $K_A = g^{(a*b)} mod(p)$

1. Bob fixes a large (random) number **b** (will be the exponent)
   - this is Bob's **private key**
2. Bob computes $B = g^a mod(p)$
   - this is Bob's **public key**
3. Exchanges the public key with Alice
4. Computes $K_B = g^{(b*a)} mod(p)$

DH - Overview

- **a** & **b** are kept **secret** (private keys)
- **A** & **B** are **openly shared** (public keys)
- Even if A & B are shared through an insecure channel, it is almost impossible to get a & b, because of the computation effort needed
- After exchanging A & B they are able to compute K_B & K_A, which will be equal

DH - Security Considerations

- There is a possible attack, in particular a "man in the middle" attackers **C**, sitting in the insecure channel between A & B.
- C can pretend with Alice to be Bob and negotiate a key, in the same way C pretend to be Alice with Bob and negotiate another key.
- At that point C can pretend to be Alice with Bob and Bob with Alice.
- The problem is the certainty that the message is really coming from Alice or Bob
- There is no authenticity of the origin of the messages
- We need a way to bind the public key with the identity of sender

5 Attacking TLS

5.1 TLS

It is a suite of a cryptographics protocols (Handshake and Record), it's part of the transport layer

TLS - What is it?

Namely **T**ransport **L**ayer **S**ecurity
It provides

- Authentication
- Confidentiality
- Integrity

Nobody outside client or server can read what the message contains and can't tumper with the message.

TLS - Existing Version

- v1.0 (1999) $\sim$ 35.5%
- v1.1 (2006) $\sim$ 38.7%
- v1.2 (2008) $\sim$ 99.8%
- v1.3 (2018) $\sim$ 56.9%

TLS - Handshake

1. **clien hello**
    -
2. **clien hello**
    -