Introduction to Network Security

COMPUTER NETWORKS A.A. 24/25

Leonardo Maccari, DAIS: Ca' Foscari University of Venice,
leonardo.maccari@unive.it

Venice, fall 2024

Sect. 1 Introduction to Computer Security

Sources

- This and the next two lessons are not based on Bonaventure's book
- You can use the lecture notes I prepared on the course website
- There you can find references to books that you can use if you are interested into going deeper into this subject.
- The whole section 1 (and the material on the lecture notes) is part of the program, the backup slides (the ones without numbers) are there just as an historic reference.

Introduction to Computer Security

↳1.1 The Security Mind-Set

Bruce Schneier on Security

From "Little Brother", by Cory Doctorow.

> *Working in security means knowing a lot about technology. It might mean knowing about computers and networks, or cameras and how they work, or the chemistry of bomb detection. But really, security is a mindset. It's a way of thinking. Marcus is a great example of that way of thinking. He's always looking for ways a security system fails. I'll bet he couldn't walk into a store without figuring out a way to shoplift. Not that he'd do it there's a difference between knowing how to defeat a security system and actually defeating it but he'd know he could. It's how security people think. We're constantly looking at security systems and how to get around them; we can't help it.*

Security Un-definition, a Metaphor

Sachsenklemme (la “stretta dei Sassoni”)

Fortezza

Fortezza: https://it.wikipedia.org/wiki/Fortezza_(Italia)

Fortezza

- 5 years of work (1833-1838)
- 2.6 million florins (around 400M Euro today)
- 900,000 cubic meters of clay
- 20,000,000 bricks
- 6300 people working for 5 years
- dozens of deaths during construction

Arundhaty Roy on Fortezza

Even though it has never been attacked (or so they say) think of how its creators must have lived and re-lived the idea of being attacked. They waited to be attacked. They must have dreamt of being attacked. They must have placed themselves in the minds and hearts of their enemies until they could barely tell themselves apart from those they feared so deeply. Until they no longer knew the difference between terror and desire. And then from that knothole of tormented love they must have imagined attacks from every conceivable direction with such precision and cunning as to render them almost real. How else could they have built a fortification like this? fear must have shaped it, dread must be embedded in its very grain.

Fortezza Was Useless

- . . . it has never been attacked.

Security mind-set

- This is the mind-set of those that design security systems
- . . . regardless their system will or will not ever be attacked.

Introduction to Computer Security

↳1.2 Security Definition

Security: tentative definition

- Security is normally imagined as a process, meaning a dynamic set of actions that make your network secure.
- Security must be guaranteed in all aspects of your system, from physical security to user management.
- From this informal definition there were efforts to try to derive more precise and usable definitions
- X.800 is an ITU standard that defines a systematic approach to the security of an IT system
- X.800 defines three key topics: **security services** and **security mechanisms** with relations to **attacks**

Security Services

6 fundamental themes on which to dedicate resources for a secure system:

- Data availability
- Data authentication
- Data integrity
- Secrecy of data (AKA confidentiality)
- Access control
- Non repudiation

To these we add a seventh: anonymity

Security Mechanisms

- Security mechanisms are functions that can help you get the mentioned security services.
- Most mechanisms are based on cryptographic principles.
- The mechanisms are intended as “building blocks”, or primitive functions that must be composed to obtain a service

Security Mechanisms

Some of the best known mechanisms are:

- Encryption
- Digital signature
- Various access control mechanisms
- Integrity mechanisms
- Authentication Protocols
- ...

These mechanisms can be mapped to cryptographic functions such as hash functions, symmetric/asymmetric key encryption, authentication protocols etc. . .

Attacks: why people study attacks?

- A system that is 100% secure simply does not exist, security is always defined with respect to a certain attacker
- A secure system is a system that resists the attacks of a certain opponent, defined in a certain way
- For this reason it is necessary to study and understand the history of past attack techniques.
- Without this effort one can't design the security mechanisms that are needed to realize the required security services

We can't dig into attacks in this course, I have one whole course on network security at the Master course.

Introduction to Computer Security

↳1.3 Security Services

Security Services: in depth

- Let us now go through security services, with examples of mechanisms to implement them and possible attacks.
- The considerations we do may apply to any system you may want to secure, which deals with moving bytes from one place to another (a web page, an app etc.)

Service Availability

The service must always be reachable.

- Availability is violated if you are a victim of a DoS : Denial of Service attack
- The availability of the service is the most difficult thing to guarantee:
  - There are always physical resource limits: attacker try to saturate them.
- Making a DoS attack must cost as much as possible
- Availability is achieved through accurate network design and doing a proper risk analysis

Note Well: there is always a risk of your service being target of a successful DoS, you should minimize it and make it as hard as possible.

Data Confidentiality/Secrecy

The data exchanged must remain confidential between the parties participating in the exchange

- . . . yet, most of the networks we use were not designed from scratch with this feature in mind.
- For instance, Ethernet networks allow to *sniff* packets from other PCs.
- To achieve Confidentiality, we need need Encryption mechanisms

Data Integrity

Data must reach the destination without being subject to modifications

- Attackers can modify encrypted data without decrypting it (*bit flipping* attacks)
- Integrity is achieved with *Hashing* mechanisms

Authentication

Usually split into two sub-services:

Data origin Authentication:

Who receives information must be sure that the entity of the one that generated the information is actually the one who claims to be (a website, an email address ...)

Peer entity authentication:

Who receives an information must be sure that the source from which the data is received is the expected one (i.e. are you sure data come from your Wireless Router and not from anybody else in the same network?)

- Unfortunately, most of the Internet protocols were not designed with authentication in mind
- Data authentication is achieved with *digital signature* mechanisms

Access Control

Access to the offered service must be restricted to authorized entities only:

- Each entity trying to gain access must first be identified, or authenticated
- Access rights can be tailored to the individual.
- If necessary, use Accounting techniques, or traffic profiling of users' activities (for example to perform billing).

Access Control, unwanted consequences

- Losing access control means to be used as a bridgehead for attacks against third parties (AKA: someone uses your resources to attack a third party)
- The host from which the attack starts is always involved in the legal actions that victims take
- It is up to the manager of a system to prove that the attack did not take place by the will of the manager himself.
- In the meantime, your system may be filtered/banned/shut down
- Therefore, also for systems that do not contain particularly delicate information, access control is essential.

Non repudiation

"Non repudiation prevents either sender or receiver from denying a transmitted message. Thus, when a message is sent, the receiver can prove that the alleged sender in fact sent the message. Similarly, when a message is received, the sender can prove that the alleged receiver in fact received the message."[a]

[a] William Stallings: Cryptography and Network Security

- Important especially at the application level: valid digital signatures today are equivalent to a signature on paper
- Digital signatures, thus, are a mechanism used for this service.

Anonymity

The ability to use a system without the system being able to identify the sender.

- Anonymity is not a canonical security service, but I like to list it because mass surveillance is widespread and anonymous services exist, and people use it:
  - Tor
  - freenet
  - Anonymous remailers
- Why do you want anonymity:
  - Perhaps because you want to commit illegal acts without being traced...
  - ...or perhaps because you are not in the conditions to exercise your civil rights.

Anonymity

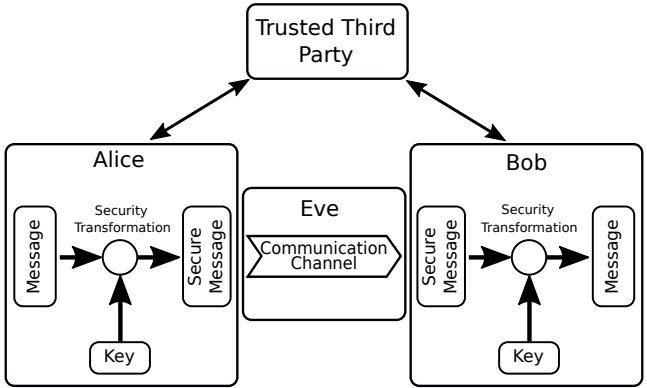
- Anonymity produce both consequences, but we must not consider them with prejudice.
- Anonymity some times is positive for the service provider: the less you know about your users the less you can be asked
- Anonymity contradicts other services (like access control)

Introduction to Computer Security

↳1.4 A System Mode for Security

System Model

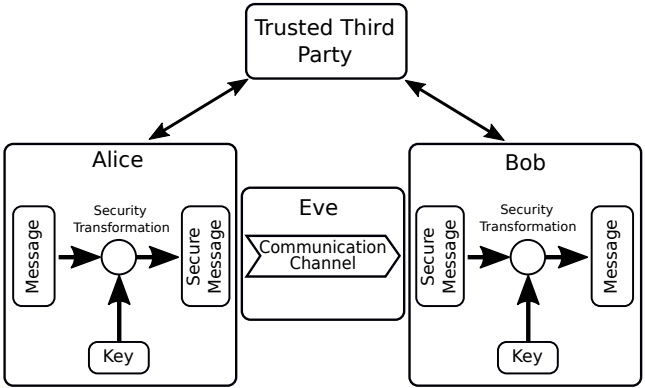
When studying the security features of system, we normally use this abstraction:



System Model Description

- This abstraction can be used for any system one may imagine, being it a website, an app, an email, any situation in which you have:
  - A sender and a receiver of information
  - An adversary that tries to break one of the security services
- This description helps you analyse the security of a system, breaking it down into components.

Model Components



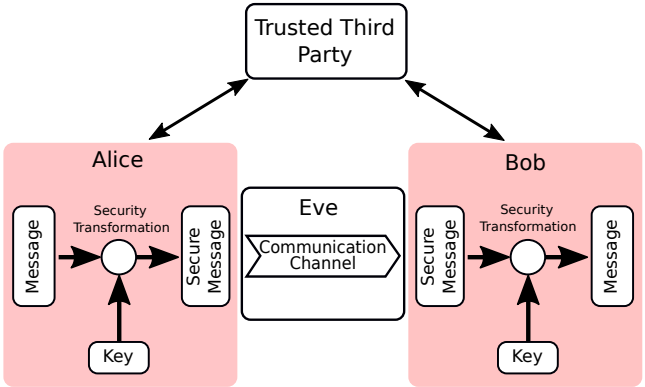
This abstraction can be used for any system in which you have:

- Sender and recipient
- Channel
- Attacker
- Security Transformations
- Trusted Third Party

Interlude: Crypto Folklore

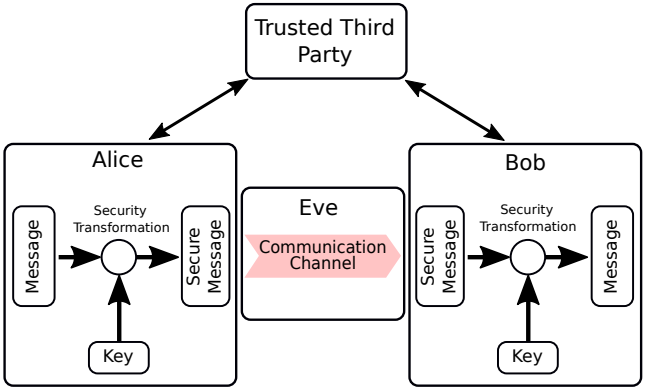
- In the Cryptography community, when you make examples you generally name the characters with standard names, that everybody uses
- Alice: is the sender of a message
- Bob: is the receiver of a message
- Eve: is the Opponent/Attacker
- In case you need more you have Carl, Frank etc...
- These names do not refer to persons, but to any entity involved in the exchange of information.

Sender and Recipient



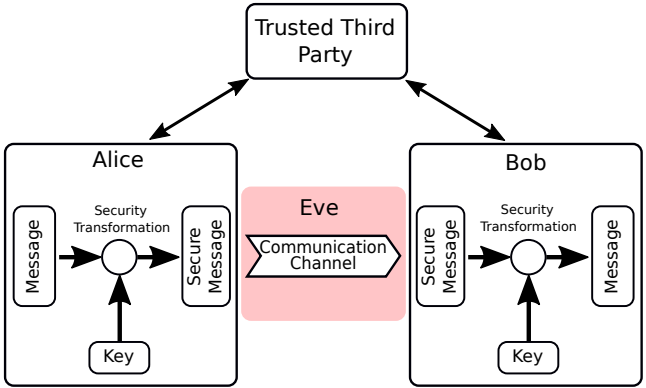
- They have a message to share (whatever kind of data)
- They treat the communication system as a "pipe". They push/pull messages of any kind. They do not control the channel (think of an email).

Channel



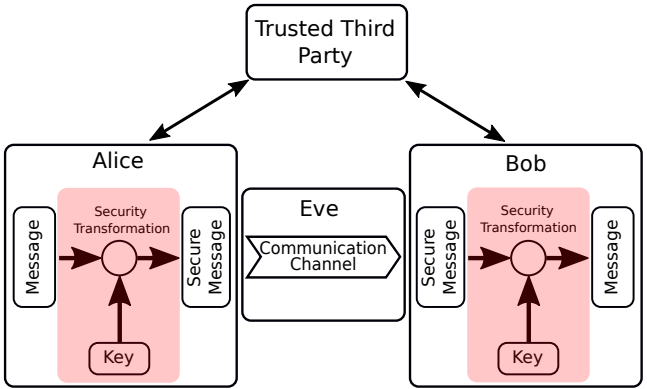
- The channel can be anything: a path through the Internet, a paper letter or a flying pigeon
- Sender and Recipient can only minimally configure/change its behavior.

Opponent



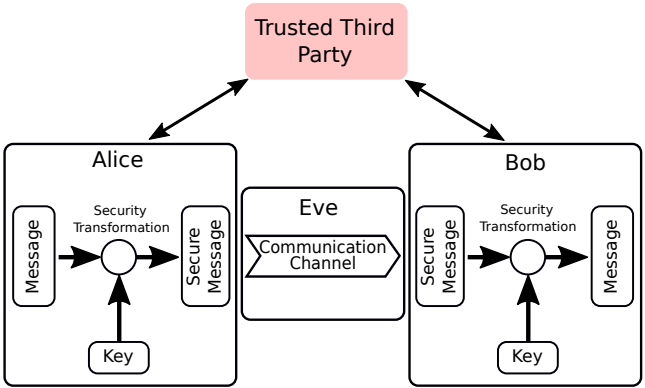
- Instead, the opponent can do anything. Eve owns the channel.
- Eve can intercept information, modify it, prevent it from arriving.
- This is not always the realistic case, but if you design your system to be robust to kind of opponent, it is robust to other ones, closer to real ones.

Transformations



- These are the security mechanisms we mentioned, and that we will study.
- The transformations modify the message from a plain text to some other form.
- They need some input: a secret (password) or any other piece of credentials.
- We assume these mechanisms are robust, the attacker can not break them. Cryptography works if you pick the right functions.

Trusted 3rd Party



- Credentials need to be generated, distributed or certified by a third party
- This is some entity that Sender and Receiver trust, and may be involved (directly or indirectly) in the communication channel.
- Also the opponent can interact with it.

Security Workflow: Risk Analysis

The generic flow of actions needed to achieve a secure system is the following one

- You identify the actors (who is Alice and Bob)
- You identify the assets you want to protect, and their value
- You decide which security services you need to provide
- You Identify a model for the attacker (or more than one) and decide what is the risk you want to run
- You decide on the technical mechanisms that are needed to achieve the security services
- You constantly iterate on this process.

Always a trade-off

- If your house door has one lock, you rely on the security of one producer, and the robustness of that device
- If your house door has two locks, if one lock is insecure (can be opened with a fake key) the other lock still gives you security.
- Yet, you pay more, you need to keep two keys with yourself, and it takes double time to enter your house.

Security is always a trade-off

You need to identify the value of your system, the security services you need, the attackers, the way they can attack you, and make a risk analysis: what security services can you afford? with what mechanisms?

A practical example

- Imagine you want to set-up a app for instant messaging.
- Let's look at what a real app does and map the concepts we described to it.

Telegram-like App

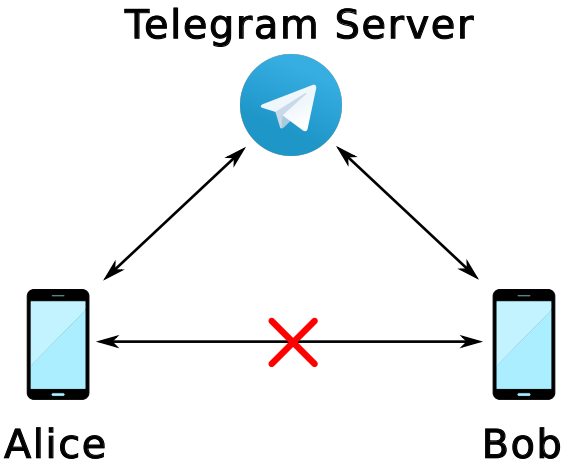
Telegram is a messaging app lik WhatsApp, Slack, ecc. We take its design as an example.

- **Sender and Receivers**: two phone apps
- **Channel**: two internet connections, one server that forwards a message from the Sender App to the Recipient App
- Of course what follows is a simplified description of such a complex system

Assets and Value

- The service is free, so the highest value is the usability of the system, and the reputation of your company that people use because they trust you.
- The security service you decide to provide is secrecy, integrity, authentication and access control. We take only secrecy into account in this example.

Telegram Channel



Attacker 1: Eve

- Eve can interact with Alice's and Bob's **out of Telegram servers**.
- Consider for instance Eve as an opponent that installed a backdoor in Alice's wireless router.
- Eve sees traffic passing back and forth, and she can modify and read it.

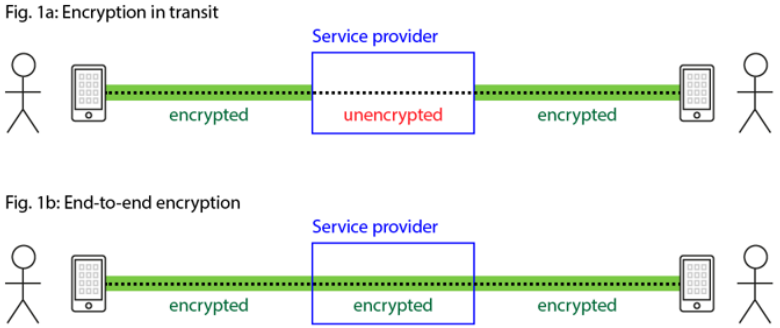
Attacker 2: Mallory

- Mallory instead can control Telegram servers.
- An example is that Mallory is Pavel Durov, the owner of Telegram, or she broke into Telegram servers and can control the traffic passing by, even against Durov will (let's forget for a moment how and if this could be technically possible)

Security goal: secrecy

- We focus on securing Telegram traffic with the secrecy service.
- We imagine two designs that can provide secrecy, and we evaluate them against Eve and Mallory.

Encryption in Transit Vs E2E Encryption[1]



[1]Image from
https://martin.kleppmann.com/2015/11/10/investigatory-powers-bill.html

App-to-Server Encryption

- In the first model (fig. 1a) the message is encrypted from Alice to Telegram server.
- Telegram knows the key to decrypt it, stores it, encrypt it again with another key and send it to Bob
- As long as the encryption/decryption function is robust and the protocol is secure, Eve can not intercept the messages.
- This Telegram mode provides the **data confidentiality** against Eve.
- Mallory instead can read the message in clear, beacuse Mallory controls Telegram servers.

End-to-End Encryption

- In the second model (fig 1b) Alice and Bob are the only ones that know the encryption key.
- Even Mallory will see encrypted traffic passing through, and can not read it.
- The Telegram server acts like a trusted third party that can help Alice and Bob to negotiate a key they only know (there are protocols for this).

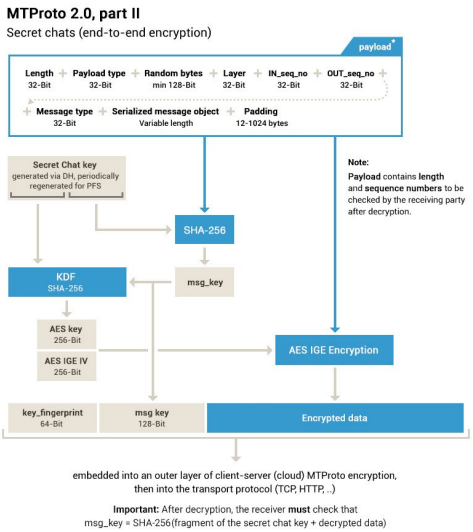
Always a Trade-off

- You may think that End-to-End encryption is just plain better. Under a security point of view, this may be true, but with that mode:
  - You can use only one device (the key is stored in your phone). If your device is lost, your conversation is lost
  - You can not have group conversations
  - The server does not own the cleartext information so it can not provide services like text search.
- that is, security has a price, in this case, in terms of usability of the service.
- A service that is less usable attracts less customers than competitors.

Accepting Some Risk

- so Telegram final choice is: use model 1 as a default and leave the option for model 2 to customers.
- Telegram accepts the risk that the system may be compromised by Mallory for all users that choose the default option (probably the majority).

Telegram App: Security Transformations



- We are not getting into the details, but Telegram uses a complex composition of cryptography functions to transform a plaintext (your message) into a transformed message.
- The keys used for this transformation are generated by Alice and Bob through the third party (the Telegram server)

Sect. 2 Foundations of Cryptography

Encryption

- The term encryption comes from the Greek words kryptós (which means "hidden") and gráphein (which means "write"). It is therefore the science that deals with making information secret.
- Cryptography has a secular history, from the Caesar cipher onwards
- Today cryptography includes studies aimed at providing secrecy, but also all the other security services we've mentioned.
- Among them, the exception is Availability, which has little to do with cryptography. Indeed a widespread use of encryption techniques requires higher computational resources which increases the possibility of being victims of DoS, ad thus, it may hinder Availability.

Cryptography Provides

- Data integrity (through **hash functions**)
- Secrecy (through **encryption**)
- Not repudiation (through **digital signatures**)
- By composing different cryptographic functions you get
  - Authentication of the peer and the origin
  - Access control
  - Anonymity

Components of a Security Service

A Cryptographic Algorithm

- A sequence of mathematical operations that can be applied to a message M

A Cryptographic function

- A block of code that implements the algorithm to provide security mechanisms (for instance, data secrecy through encryption).

Secure Protocols

- We know a protocol is a convention between two entities that communicate with each, that defines how to exchange data and we know that certain protocols guarantee certain services.
- Some protocols combine Cryptographic functions to provide security services.

Example

- The RSA algorithm explains how to perform public key encryption (topic for tomorrow)
- RSA is implemented in the OpenSSL open source library
- The TLS protocol uses RSA to provide security services on top of a TCP connection
- OpenSSL is included in Linux distributions to create secure HTTP (web browsing) services, with the Apache web server.

Example (2)

- We do not expect RSA to be vulnerable, it has been used for decades, and as long as the key is sufficiently long it does not suffer of any known vulnerability
- OpenSSL instead was vulnerable in the past to several problems, which were fixed with time.
- TLS itself has gone through several generations, and each one had vulnerabilities, that were indipendent from RSA/OpenSSL
- Apache has its own record of vulnerability.

Golden Rules

- Never use unknown crypto algorithms
- Never use closed source crypto functions
- Rely on the most used/popular/maintained functions
- Use standard protocols, do not reinvent the wheel
- Restrict the service to users that support the most update version of the protocols (if possible)

This at least minimizes the risk

Today's Crypto

Today we will analyse:

- Hash Functions
- Message Authentication Codes

Foundations of Cryptography

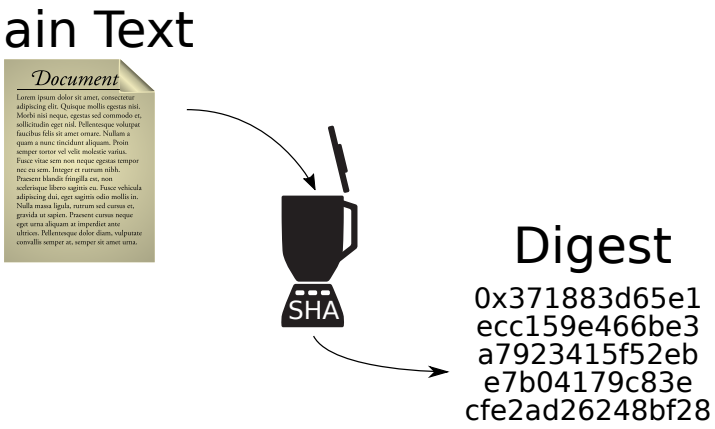
↳2.1 Hash Functions

Hash Functions

Hash functions provide integrity

- a hash function is a unidirectional function H() that once applied to data (any digitized information, an email, a package, a file etc...) generates a **digest**: a fixed size string (a digest can be 128, 160, 256 bits)
- The digest is a function of the data, different input produce different digests

Example: SHA



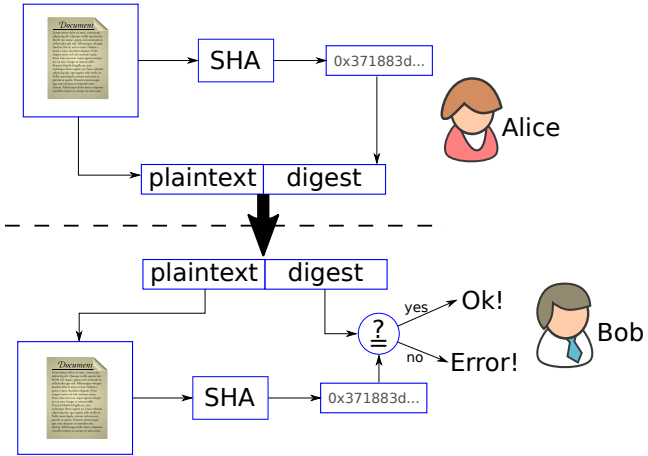
No matter how big is the input, the output size is fixed.

How to Achieve Integrity

Hashes are a mechanism used to obtain the service of data Integrity:

- Alice sends Bob message M, she also calculates $H(M) = D$ and sends also D.
- Bob receives M and D, recalculates $H(M) = D'$.
- If $D = D'$ then the message was not modified along the way.
- The **integrity** service is achieved.

Hash-based Security Scheme



Usage example

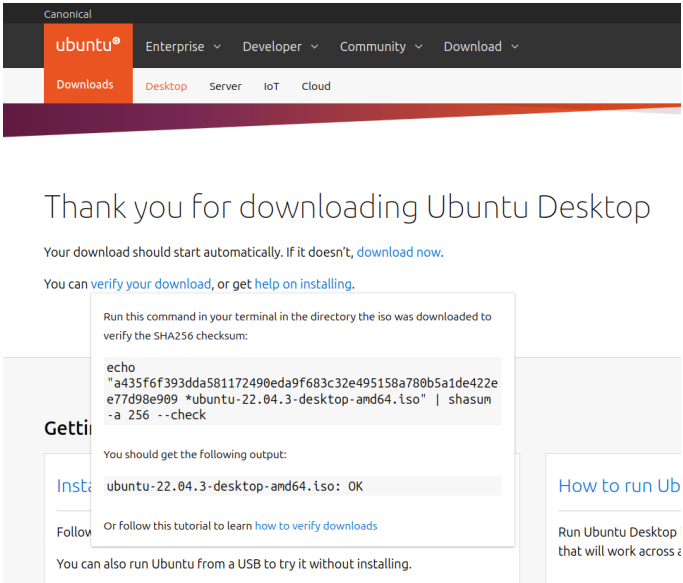
A typical application is the distribution of OS images binary files:

- A new Ubuntu image is published, millions of people will download it at the same time, generating a peak of traffic.
- Ubuntu shares the image with partner sites and puts it on P2P networks
- But when Alice downloads the image from a different place than the Ubuntu website, she can not be sure it was not modified.
- Then Ubuntu publishes the image together with the digest
- Alice can download the image from anywhere, then download the digest from the Ubuntu website (256 bits). Then she computes the hash locally on the image and checks the downloaded digest is the same she computed.

Hashes on GNU/Linux

- Two hashes are often used, `MD5` and `SHA`.
- The math of these hash functions is known (and considered secure for some applications, less secure for others)
- MD5 is generlaly deprecated, while SHA-256 is still OK
- GNU/Linux distributions come with programs that can compute hashes

```
$ echo "test" > test.txt
$ sha256sum test.txt
  f2ca1bb6c7e907d06dafe4687e579fce76b37e4e93b7605022da52e6ccc26fd2  test.txt
$ md5sum test.txt
  d8e8fca2dc0f896fd7cb4cb0031ba249  test.txt
```


Canonical
ubuntu
Enterprise
Developer
Community
Download
Downloads
Desktop
Server
IoT
Cloud
Thank you for downloading Ubuntu Desktop
Your download should start automatically. If it doesn't, download now.
You can verify your download, or get help on installing.
Run this command in your terminal in the directory the iso was downloaded to verify the SHA256 checksum:
echo "a435f6f393dda581172490eda9f683c32e495158a780b5a1de422ee77d98e909 *ubuntu-22.04.3-desktop-amd64.iso" | shasum -a 256 --check
You should get the following output:
ubuntu-22.04.3-desktop-amd64.iso: OK
Or follow this tutorial to learn how to verify downloads
Getti
Insta
Follow
You can also run Ubuntu from a USB to try it without installing.
How to run Ub
Run Ubuntu Desktop
that will work across a

Math → Functions → Protocols

- Math: At the base of SHA there is a mathematical formulation that provides certain properties
- Code: SHA is implemented in a function written in C language, and included in the `shasum` program
- The "*protocol*" adopted is:
  1. Ubuntu releases a new version on their website **and** other places
  2. Alice downloads Ubuntu image from anywhere, but downloads the checksum **only** from the Ubuntu website
  3. Once the download is completed, Alice generates the checksum and checks that it is the same one as the one provided by Ubuntu.

Hash function Properties

Three are intuitive:

1. The input M may be of any size
2. The output must be of fixed-size
3. Given M, calculating $H(M)$ must be computationally very efficient (fast).

Any Size Input

- The domain of the function (the space of the input values) is larger than the dimension of the co-domain (the space of the output function).
- By definition, different input values will be mapped to the same output: there exist two messages M and M' so that $H(M) = H(M') = D$. We say that hash functions admit **collisions**.

Hash function Properties

Then there are three that are less intuitive:

4. One way: given D, find X such that H (X) = D must be computationally impossible
5. Weak collision resistance: given X, find Y such that H (X) = H (Y) must be computationally impossible
6. Strong collision resistance: find two X and Y for which H (X) = H (Y) must be computationally impossible

On "Computationally Impossible"

- Cryptography often depends on something being "impossible".
- Most of the time, this term does not mean that an event is theoretically impossible, but, given the computational power we have today, it is extremely unlikely that the event happens.

Computationally impossible: A problem is said to be computationally impossible if there is no known algorithm of polynomial complexity to solve the problem, i.e. that can run on an input of size n with at most n^c operations, for some value of c.

Prop 4: Not Invertible

- It is intuitive that if $D = H(M)$ has fixed size, a lot of information of M are lost through hashing.
- Moreover, considering that there are collisions, even if Eve finds M' so that $H(M') = D$, she can not know if $M = M'$: **A hash functions is not invertible by design (property 4)**

Brute Force

- However, if Eve knows D and wants to know M, she can try different M' till she finds a collision.
- If the hash size is 256, every attempt has $\frac{1}{2^{256}}$ probability of success, so brute force has exponential complexity on the size of the hash.
- We consider this low enough to be *computationally impossible* to invert a hash by brute force.

Refining a Brute Force

- Let $d(D, D') \rightarrow [0, 1]$ be a distance function between two binary strings, with $d(D, D') = 0 \iff D = D'$
- Let's say that Eve starts from an attempt M', then she loops on minimal modifications on M' until she obtains M'' so that

$$D'' = H(M''); \; d(D, D'') < d(D, D')$$

  and so on till she finds M.
- For this to hold, at each step, it must be true that if $d(D, D'') < d(D, D')$ then $d(M, M'') < d(M, M')$: the distance function is invariant to H().
- This makes the probability of guessing at each step higher than $\frac{1}{2^{256}}$, and the attack could become computationally feasible.
- That's why we have property 5, that explicitly prevents this

Prop 5: Weak Collision Resistance

Given D, it is comp. impossible to find M such that $H(M) = D$

- The digest should not reveal **any** information of the original input file.
- If the input is modified of even one bit only, the difference in the digest should be big
- The output of $H(M)$ must be unpredictable, essentially, $H()$ must be a pseud-random number generator
- Consequence: **If Alice finds a document M' that has a hash similar to D, this does not mean that M' is similar to M. She can not *refine* a brute force attack.**

Prop 6: Strong Collision Resistance

- It you want to break property 7, you need to find any two messages that collide, this, due to the Birthday paradox (a well known statistical paradox) is easier. It takes $\frac{1}{2^{128}}$ attempts in practice.
- So, it is easier, and that's why this is a stronger property
- The bottom line is: Any finite size digest **implies a theoretical non-zero probability of collision**.
- However we consider it *computationally impossible* to find any

Exercise

- Consider the string:
  Dear Student, your final grade is 18, we are going to meet in room A432 at 14:00 to register
- its hash using the Md5 hashing function in hexadecimal is

$$3b3b28a9b694f5eed5f1221575a9cb4e$$

- assume the reader is lazy, he just checks the first and the last 3 chars of it:

$$3b3 \ldots b4e$$

- You want to trick him, modifying the string so that it starts with:
  Dear Student, your final grade is 30...
- how many attempts in average you will have to try to have success (answer this in class)?
- can you implement the code to make this possible (do this if we have spare time at the end)?

Hints

- The string:

  Dear Student, your final grade is 30, we are going to meet in room j356 at 4:27 to register

  has this digest:

  $$3b36ab65a0c6d089628a0628af124b4e$$

- computing hashes in Python is really easy:

```
1  import hashlib
2  digest = hashlib.md5("string to hash".encode('utf-8')).hexdigest()
3
```

- and here you can find the C code (note you need the openssl libries for your system)

Sect. 3 What Can Go Wrong

What can go wrong

- What follows is a list of episodes in which something went wrong
- I will try to divide the cases according to the security services that have been violated
- I will try to refer to old and recent cases, to show the evolution (or missing evolution) of the state of things

Service availability: Mafiaboy against Yahoo

- In February 2000, a 15-year-old boy known as “mafiaboy” started a DDoS attack (a distributed denial of service attack) against some well-known sites (Yahoo, Amazon, Ebay...).
- The attack was carried on from about fifty computers that he previously took control of
- These servers belonged to university networks, public bodies, etc...
- They had a lot of bandwidth available.
- The sites yahoo.com, ebay.com, cnn.com were unreachable for a few hours.
- Mafiaboy got away with 8 months of reform and the ban on activities in forums, lists, groups dedicated to the themes of "cracking"

A few hours?[2]

[2]Image from `https://medium.com/@vikigreen/impact-of-slow-page-load-time-on-website-performance-40d5c9ce568a`

Back to Today: Akamai CDNs

- Akamai is a Content Delivery Network provider (CDN)
- A CDN is a network of servers spread around the world
- If you want to offer someone a service, and you want this service to be fast, you need to be close to the client.
- For this reason, instead of putting your server in one place, you put a replica of it in every big city in the world
- But you don't want to have a data-center in every city
- Akamai does it for you.

CDNs

- The business model of Aakmai (and other companies in the field) is to build and operate large data-centers spread around the world
- You can rent computing power in their centers, and they run your software
- Your clients are always “close” to the service
- They don’t have to go across the world to reach your service, so your service is fast

So you pay Akami for being fast...

Not only. They also give you "Scrubbing Servers"

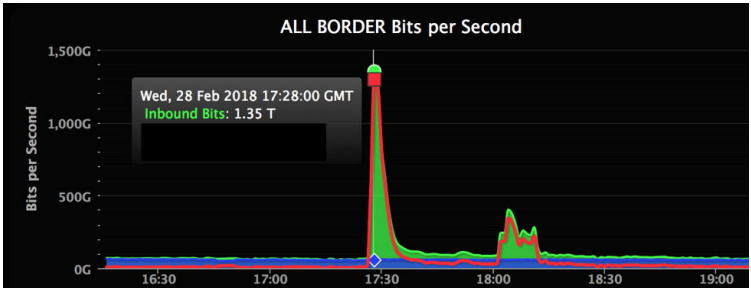
- Scrubbing servers are machines capable of processing Gb/s of data and remove malicious traffic once an attack has been identified.
- When your service is under attack, your traffic is deviated to special servers that "remove" the traffic attack from the traffic your service receives

An Armed Race

- Scrubbing servers have a limited computational capacity
- If the attack is a DDoS, the amount of traffic that the server receives its orders of magnitude higher than the benign traffic
- Cleaning this amount of traffic needs computational power
- Who has more? the attacker or Akami?
- According to Akamai techies: "We modeled our capacity based on fives times the biggest attack that the internet has ever seen"

DDoS against Github

- Despite this, in February 2018 Github was unavailable for around 10 minutes
- The cause was the largest DDoS attack in history, using up to 1.3Tb/s of traffic to Github services



Attack Source

- The attack was generated exploiting a vulnerability of an application that creates web caches
- A web cache is a layer of "memory" that you place in front of your web application
- If two users ask twice for the same content, you serve it from the cache, not from the database (faster)
- Unfortunately, the cache accepted requests from "spoofed" IP addresses, and generated return packets of big dimensions
- In practice, an attacker could send a content request to a cache pretending this request was coming from GitHub.
- The cache was generating a large packet in return, and sending it to Github

Attack Source

- Multiply this for thousands of packets per second, for thousands of caches of different services
- The attacker achieved 1.3Tb/s
- **Relevant Concepts:** the cache service had a **vulnerability**, the attacker exploited it and attacked Github **Availability**

Service Availability: Conclusions

- Service availability is hard to achieve, because it is an armed race between you and the attacker
- Availability is achieved spending a lot of money to make your system part of a large infrastructure, like the one of Akamai. This costs and as we have seen, it is not 100% secure (nothing is...)
- How much does it cost?

Microsoft Azure DDoS Protection Pricing

I could not find Akamai pricing, I found Microsoft prices:

Pricing Details

The DDoS Protection service will have a fixed monthly charge, as well as a charge for data processed. The fixed monthly charge includes protection for 100 resources. Protection for additional resources will be charged on a monthly per-resource basis.

Monthly price for DDoS Protection (includes protection for 100 resources): €2,483/month *

Overage charges (more than 100 resources): €25 per resource per month *, 1

Data Processed:

DATA PROCESSED PER MONTH	PRICE PER GB
0 - 100 TB	€0.043
100 TB - 500 TB	€0.034
500 TB - 1 PB	€0.026
Over 1 PB	Contact Us

Data Authentication

- This examples brings us to another important service, data authentication.
- How could it be that attackers could send requests to cache services pretending to be Github?
- Because the Internet Protocol allows it.

IP Spoofing

- The IP protocol allows you to do *packet spoofing*, AKA you can send packets using the sender IP addresses of a host that is not yours
- Why? In 1981 (when RFC 791 was published, that describes the Internet Protocol) the priority was to make a network work, rather than make it secure.

Packet Spoofing: 16 years after

- Wi-Fi was proposed for the first time in 1997.
- According to the standard, all the hosts in a Wi-Fi network were supposed to share a single encryption key in order to prevent someone to inject packets in a network without being allowed
- Packets were becoming authenticated.
- Unfortunately the first version of Wi-Fi was broken. Up to 2004 (when WPA was published) it was possible to perform many different kinds of attacks to break the authentication of packets.
- *Still today, for some kinds of Wi-Fi packets this is perfectly possible*

Data Confidentiality: the AoL case

- August 2006: the AoL portal, one of the largest Internet Service Providers in the USA, as well as a search engine mistakenly publishes a 440Mbyte file
- The file containing the searches carried out in the last 3 months by 500,000 users.
- For each search an "anonymized" ID of the user is reported, the query done in the search engine, and the link that user clicked
- Some examples:

personal injury, auto accident pictures, divorce law, blackberry, family law, florida divorce law, palm beach county family connection, florida criminal lawyer, [persons name], [persons name HOUSE], [persons name and JUDGE], code of judicial conduct

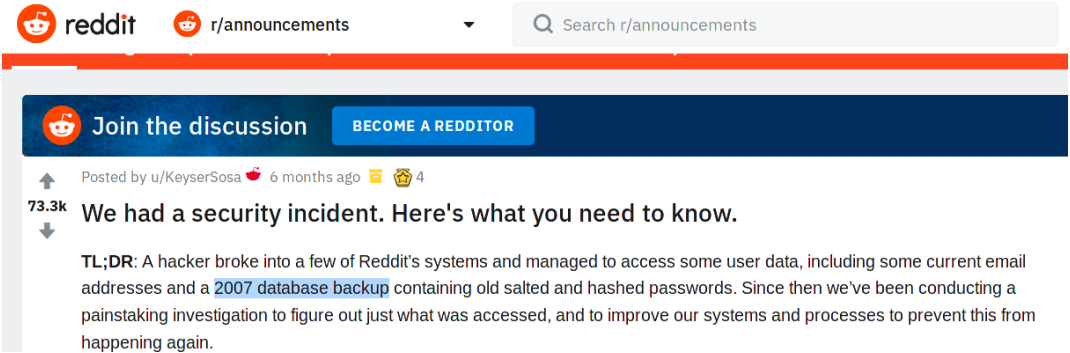
Well, this is probably a lawyer looking for information on a specific case, nothing really interesting. Unless the same person did make other searches:

strippers men, men in briefs, men in speedos, tan speedos, tanning oil, man sexy brief, man swimsuit

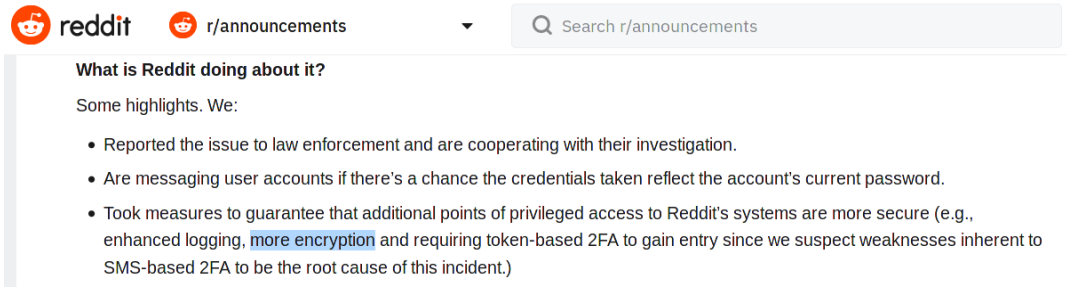
AoL case: take away

- Not all harm comes from malicious attackers
- Some times, mistakes are more harmful than attackers.
- A well engineered organization should make it impossible that the error of one person destroys the reputation of the whole organization
- In practice, these data should have been saved in an **encrypted form**.
- Back-ups are essential in any service, and encrypted back-ups are necessary to preserve **confidentiality** for your customers

12 year later: Reddit leak (2018)



12 year later: Reddit leak (2018)



It is not reasonable that in 2018 Reddit is still maintaining unencrypted copies of databases from 2007. Interestingly enough, the problem seems to be related with 2FA authentication, which brings us to another issue: Access Control.

Access Control: Maryland vs. Diebold

The facts:

- Since 2002 the state of Maryland introduced DRE machines (Direct Recording Electronic voting machines) produced by Diebold.
- The state of Maryland outsourced to independent researchers several security analysis to assess the reliability of such machines.
- The researchers, once given physical access to the machines, were able to:
  - Read the voting data
  - Change the software running in the machines, in order to change their behavior
  - Modify the magnetic cards of the voters to be able to vote several times.
  - Transform a normal magnetic card into an administrator card. Fortunately, the administrator must use a two-factor authentication, he has to enter a numerical PIN
  - In the first generation DRE machines, the hard-coded PIN was 1111

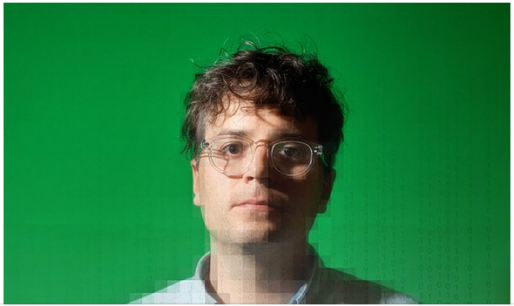
Diebold Case, take-away

- You must properly define your opponent.
- In the case of Diebold, aside some technical failures (1111 PIN...) they did not design machines that were resistant to an attacker that has physical access to DRE
- If you fail to properly design your opponent, your whole access control system may be screwed.

10 years later: Epic Hacking

MAT HONAN GEAR 08.06.12 08:01 PM

HOW APPLE AND AMAZON SECURITY FLAWS LED TO MY EPIC HACKING



- Journalist Matt Honan in 2012 described on the Wired Magazine an attack that led to the cancellation of his entire on-line life.
- All contents of his Gmail, Apple, Twitter, Amazon accounts were erased, with all data.
- The attack took place in the following way, with very little technical component

Attacks Steps: phase 1

- The attacker discovers Honan's home address and some email addresses (including the Apple one). This is not hard with some "google-fu".

Attacks Steps: phase 1

- The attacker discovers Honan's home address and some email addresses (including the Apple one). This is not hard with some "google-fu".
- The attacker calls the Amazon user service, says he is Matt Honan and asks to add a new Credit Card Number.

Attacks Steps: phase 1

- The attacker discovers Honan's home address and some email addresses (including the Apple one). This is not hard with some "google-fu".
- The attacker calls the Amazon user service, says he is Matt Honan and asks to add a new Credit Card Number.
- As this operation is not removing or revealing any sensitive data, the only thing he is asked is to confirm the home address and email.

Attacks Steps: phase 1

- The attacker discovers Honan's home address and some email addresses (including the Apple one). This is not hard with some "google-fu".
- The attacker calls the Amazon user service, says he is Matt Honan and asks to add a new Credit Card Number.
- As this operation is not removing or revealing any sensitive data, the only thing he is asked is to confirm the home address and email.
- Now Matt Honan account on amazon has a new CC number: 

Attacks Steps: phase 2

- The attacker calls Amazon again, he says he is Matt Honan and has lost his password.

Attacks Steps: phase 2

- The attacker calls Amazon again, he says he is Matt Honan and has lost his password.
- This is a critical operation, so Amazon asks to confirm some valid credentials: a credit card number

Attacks Steps: phase 2

- The attacker calls Amazon again, he says he is Matt Honan and has lost his password.
- This is a critical operation, so Amazon asks to confirm some valid credentials: a credit card number
- The attacker provides the number he just added, he resets the password, enters the Amazon account.

Attacks Steps: phase 2

- The attacker calls Amazon again, he says he is Matt Honan and has lost his password.
- This is a critical operation, so Amazon asks to confirm some valid credentials: a credit card number
- The attacker provides the number he just added, he resets the password, enters the Amazon account.
- He is now able to read the last 4 digits of the real credit card that Matt Honan stored on Amazon: 👍

Attack Steps: phase 3

- The attacker calls Apple, and says he has lost his password.

Attack Steps: phase 3

- The attacker calls Apple, and says he has lost his password.
- The operator asks a number of questions to confirm his identity, which he fails to answer, except the last one: tell me the last 4 digits of your credit card

Attack Steps: phase 3

- The attacker calls Apple, and says he has lost his password.
- The operator asks a number of questions to confirm his identity, which he fails to answer, except the last one: tell me the last 4 digits of your credit card
- Now the attacker has access to the Apple account (including reading his personal emails).

Attack Steps: phase 3

- The attacker calls Apple, and says he has lost his password.
- The operator asks a number of questions to confirm his identity, which he fails to answer, except the last one: tell me the last 4 digits of your credit card
- Now the attacker has access to the Apple account (including reading his personal emails).
- He goes on all other accounts, clicks on "I lost my password", these accounts were using the apple email as a back-up email

Attack Steps: phase 3

- The attacker calls Apple, and says he has lost his password.
- The operator asks a number of questions to confirm his identity, which he fails to answer, except the last one: tell me the last 4 digits of your credit card
- Now the attacker has access to the Apple account (including reading his personal emails).
- He goes on all other accounts, clicks on "I lost my password", these accounts were using the apple email as a back-up email
- He resets the password of all accounts and deletes all the contents: 

Epic Hacking: take-away

- Access Control is hard to implement, because it requires users to posses some credentials.
- These credentials can be lost, stolen, invalidated, so procedures are needed to update them, and these procedures can not depend on the credentials themselves.
- Furthermore, users are not tech-savvy and they misuse their credentials (pick dummy passwords, give them away when asked, re-use the same credentials for different services)
- Finally, your system is probably connected and dependent on other systems to work. Security does not only depend on your behavior but also on the behavior of your partner services.