UNITRENTO

Network Security

By

Alessia Pivotto

Handouts of the course 2023/2024

Contents

Chapter 1

Definitions and Properties

Computer/Information security:

The protection *afforded* to an automated information system in order to attain the applicable objectives of *preserving* the Confidentiality, Integrity, and the Availability of information systems resources. (refers to ACI, not to the computer/device/hardware) (preserving not creating)(not possible perfect security)

Cyberspace:

Domain characterized by the use of electronics and the electromagnetic spectrum to store, modify and exchange data via networker systems and associated physical infrastructures.

NB: From IT to cyberspace, same domain and still with electronics but even with physical systems.

Cybersecurity:

Is the art of protecting networks, devices and data from unauthorized (access not authorized to a device) or criminal use and the practice of ensuring integrity, availability and confidentiality. Cybersecurity is the practice of protecting systems, networks, and programs from digital attacks... Prevention of damage to, protection of, and restoration of computer, electronic communications systems, electronic communications services, wire communication, and electronic communication, including information contained therein, to ensure its availability, integrity, authentication, confidentiality and non-repudiation. Cybersecurity is the practice of protecting critical systems and sensitive

information from digital attacks. Also known as information technology (IT) security, its measures are designed to combat threats against networked systems and applications, whether those originate from inside or outside of an organization. Cybersecurity domains are several, education, tech, IoT, matter of governance...

Trust: Is the acceptance of the truth of a statement without evidence or investigation; it is blind faith or wishful thinking if you will.

Trusted:

Able to be depended on → you cannot verify.

Trustworthy:

Worth to be trusted → upon verification.

Safety:

The condition of not being in danger or of not being dangerous: For your safety, keep your seat belt securely fastened.

Security:

Protection of a person, building, organization, or country against threats such as crime or attacks.

Dependability:

The property of a computer system such that reliance can justifiably be placed on the service it delivers. Behaving as expected.

Resilience:

The ability to operate correctly (even with reduced functionality) or restore a safe state while the system is compromised or after an attack.

Usage is important, with Cyber-Physical Systems usage of the system is also a crucial issue besides traditional security properties of its information (Safety issues) and the solution must be a combination of analogue and digital world.

> Es: In a classical IT system, the reaction to an access control violation is to stop the access to the requested resource. In a CPS that could have dangerous collateral effects. Imagine the braking system of a modern car...

PS: The course focuses on digital networks and systems, thus on the traditional CIA and related properties.

We are moving away from cybersecurity to land on cyber resilience.
NB: things are very complicated so it's difficult to protect ⇒ from cybersec to resilience ⇒ the goal it's not anymore to prevent attack (system will be attacked) so you try to regain safe status.
But wait, wasn't computer security about creating security/cracking security/hacking security? No. Rather, it is about preserving security. What does this mean? That you can only preserve something that you already have and that security technologies do not build security properties. They "merely" make sure that the security properties that are already there are maintained. We need to understand two concepts: what is there to be preserved and what is the "built-in" security that needs to be preserved.
NB: What is it that computers do? ⇒ Operate over information ⇒ Information is everything that operates and is operated by a computer (computer program, temporary functions, variables, excel file, picture...) All a computer system is about is information. So preserving what? Properties of information. At the bare minimum, any "piece of information" is only useful if it can be reached and read and is correct. ⇒ These can be seen as "properties" of information.
Computer security is about preserving these properties:

- Confidentiality⇒ Assure that a piece of info can be read only by those who are authorized to read it (only you can read what you need to read);
- Integrity⇒ Prevent unauthorized modification of information (prevent unauthorized writing). While data integrity, integrity synonymous for external consistency.(within the bounding of digital signals(info giuste da me ma devono anche arrivare giuste(external consistency)));
- Availability⇒ Assure that a piece of information can be reached when needed.

NB: This is known as the CIA triad that are the "core" security properties of a piece

of information. On top of this, one can build additional properties like Accountability (The ability to know with certainty who/what operated on a piece of information), Non-repudiation (The entity that acted on the information can not "repudiate" his/her/its action), Authentication (alone has no meaning, it depends of the subject:

- ***User authentication:*** *The ability to prove that a person is who she claims to be;*
- ***Message/Data authentication:*** *assurance that the message has been signed with a particular key;*
- ***Sender authentication:*** *assurance that the message has been sent by a party (IP address, server, computer, process, etc.)) etc...*

But who can act upon information?
Humans are not the only "users" of information, the human user of the system avertedly or un-avertedly modifies information. A system/software/module/thread can act on the whole system (or another system) on behalf of a human user ⇒ "Delegation" is the mechanism by which, for example, software threads are spawned (E.g., with the privileges of the entity that spawned it). While a human user may or may not know what they do a "surrogate" does not. There is no notion of "avertedly" or "un-avertedly". Computer systems do not know what they are doing. They can only execute instructions (information) to operate over some other information. Systems can only be instructed to preserve the security properties of that information by means of some mechanism(es Confidentiality ⇒ crypto, Integrity ⇒ secure hardware, Availability ⇒ redundancy).

Esempietti carini che ha fatto: CBC ⇒ sai che prendi roba e la butti nel blocco, quando esce la unisci con altra e la butti nel prossimo blocco... problema se qualcuno cambia ste cose (change bits) ⇒ attacks because pc doesn't understand that something goes wrong, a human yes but computer no, it's a little bit stupid, so attacker actions produces an invalid value and, because the missing sanity check, there is an attack.

Altro esempietto cutie: Zero trust (è legato alla roba del TCB ⇒ Trusting Computing Base che vedrai dopo, ⇒ Combination of hardware, firmware, and software that can breach your security policy). Practically I need a secure/trusting component to build a system, but what about zero-trust architectures?

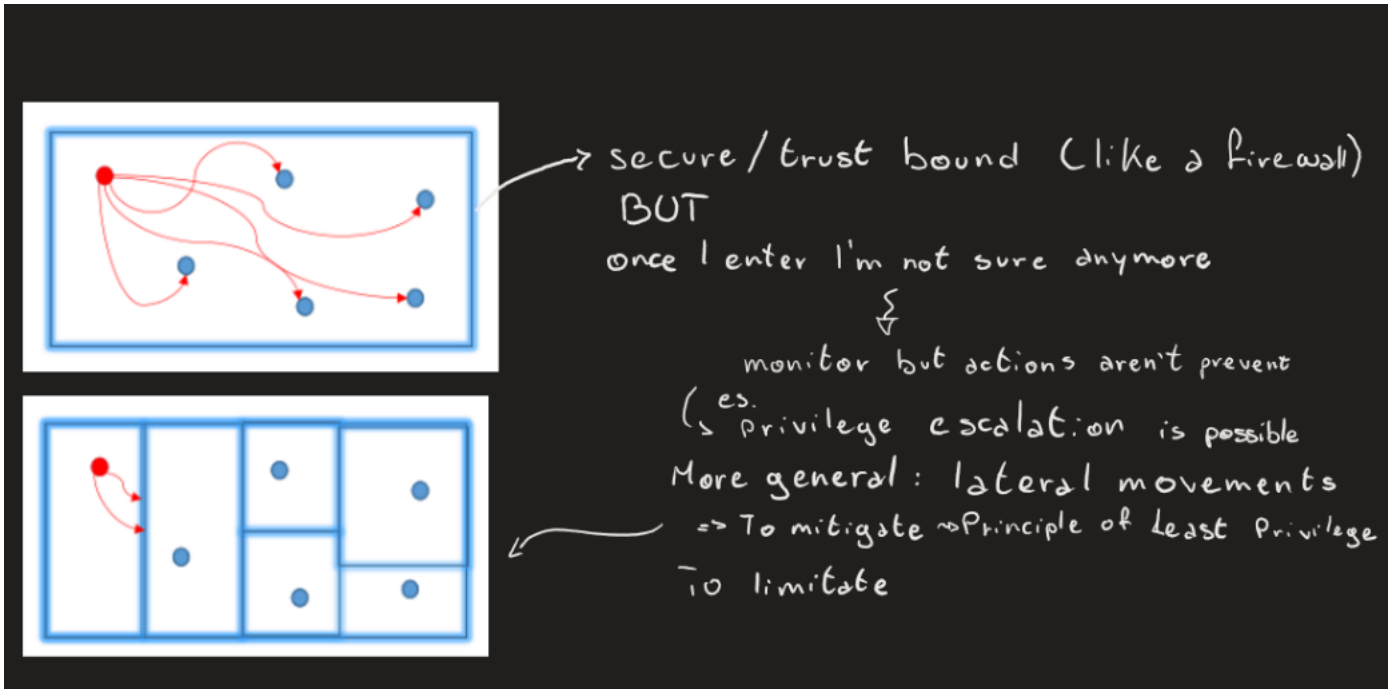


Figure 1.1: Zero Trust

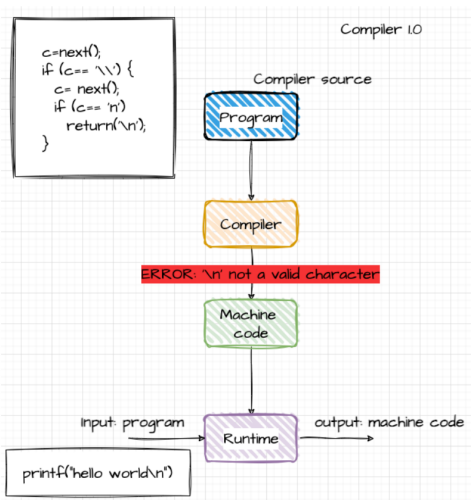
Trust in information security (more formally)

- Ken Thompson, 1983 ACM Turing Award Lecture
- "Reflections on Trusting Trust" is a mandatory reading, http://dl.acm.org/citation.cfm?id=358210
- Suggested reading:
- Bruce Schneier, "Attack trees", Dr. Dobb's Journal, December 1999.

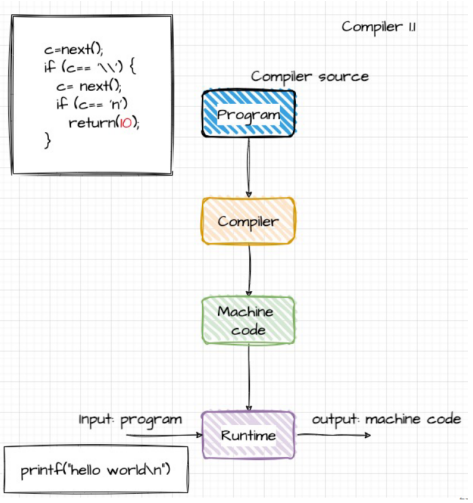
So, what software/system can one trust?
Imagine an authentication mechanism ⇒ User inputs username, User inputs password, User presses enter, User has access to desktop... There are a lot of trust assumptions

here! The user trusts the authentication mechanism. But what really happened? Did the authentication sw do the actual match? Will it only grant access if it matches your credentials? Did it send your credentials to a third party? Did it use your credentials to read and copy your data? How do you increase your level of trust in the software? You can look at the source code, but is this enough?

Example:



(a) Compiler 1.0



(b) Compiler 1.1

What or who did really generate that software? The human being that wrote the sw source code? The compiler that compiled the sw source code? The human being that wrote the compiler that compiles the sw source code? The compiler that compiles the compiler that compiled the sw source code?... So who do you trust? Thompson's view ⇒The compiler can be modified in any way to include code that never appears in the sw source code, and depending on how many generations passed, it won't appear in the previous compiler versions source code either. ⇒ Trusting trust: You can't trust code that you did not totally create yourself. No amount of source-level verification or scrutiny will protect you from using untrusted code."

Computer Security vs Network Security

A computer network is a general architecture that allows computer systems to share information remotely. Network security is based on the same exact idea of preserving the CIA properties of information, it makes for an especially interesting case. Who

can be trusted over the network? Can you trust your own system? Can you trust the communication channel? Can you trust the destination system?

CLIENT-SERVER RECAP FOR DUMMIES, TECHNICAL LOOK, ATTACK POINTS (when 2 component, spinning 2 different processes, that are not suppose to interact, are forced/needed to interact-¿ an attacker can create a gap/bridge-¿ vulnerability!) LEGACY ISSUE

ATTACK A general attacker might infiltrate the communication in between hops, impersonate the client, modify connections/routing, be/infiltrate one of the hops or act "legally" until end of service (after which it may act maliciously)...

Attack trees

Attack models in network setting: outright malicious attacker

Typically, the malicious attacker aims at reading or modifying the messages (in part or fully)-¿That's a confidentiality, integrity, availability problem. In this contest, this attacker is typically called "man in the middle" or "man in the browser". Attackers can intercept and act upon a communication between client and server, such as channel redirection, block communication entirely or spoofing the user's identity. An example could be injection of malicious content, in which there is a manipulation of server response, then client's answer can also be modified by the attacker, so a connection Hijack and finally attacker injects him/herself in the communication and spoofs the victim's identity. NB: Formally known as the Dolev-Yao model, it applies to any entity of the network

Attack models in network setting: Honest-but-curious attacker The goal of this attacker is to use the client's information after correctly handling the service. Typically resides at the service level, e.g. ISP, and implies confidentiality and possibly integrity threats. A possible example could be where DB Server is the attacker, it provides agreed service correctly, like answers queries with correct data. After the query is delivered to the client, the server uses the query's information to perform user profiling.

Chapter 2

Network aspects

Internet communication Internet is made of several logically separated networks -¿ Autonomous Systems (AS) PS: Internet= network of networks Each AS autonomously manages communications within itself through Interior Gateway Protocols (IGP) -¿ route within each AS e.g. two commonly used IGPs are the Routing Information Protocol (RIP) and the Open Shortest Path First (OSPF) protocol. Each AS can communicate to other AS using Exterior Gateways Protocols -¿ route between ASs, such as Border Gateway Protocol.

OSI data link layer Data link layer is the lowest "logical" level, data link interconnects physical interfaces and each physical interface is identified by a MAC address("Ethernet address", 48-bit Network interface identifiers, closest representation of final destination of a frame, HEX notation, used to route packets in local networks).

Mac addresses uniquely identify a network interface and is assigned by the producer according to the standard IEEE 802. eg da fare

OSI NETWORK LAYER The Network Layer provides information on how to reach other systems (addressing functionalities), IP operates at this layer so high-level representation of a host's addresses, conveys information to route the datagram, IPv4 defined in RFC 791 Most IP addresses are dynamically assigned by an authority (e.g. ISP's DHCP server). As opposed to MAC addresses that are fixed by the vendor. "Connectionless" protocol (stateless): no notion of "established connection" at this stage, only

provides the means necessary for a packet to reach its destination.

connection vs connectionless

A communication is made of a number of messages; communications start, develop, and end. Stateful protocols provide means to establish and close a connection(e.g. TCP). Stateless protocols do not have this notion(IP messages are stand-alone packets).

ip vs mac da fare

IP addresses IP provides a structured way to abstract host addresses away from their physical properties There are two versions: IPv4 and IPv6, the first is the most common and is currently used,(32 bits) while the latter is early adoption and will be seen commonly in the future(128 bits). Make it possible to efficiently talk between systems in different AS

routing da fare

ARP protocol ARP = address resolution protocol Allows systems to associate an IP address to a MAC address and allows discovery through broadcast. ARP tables contain information to translate IP addresses into MAC addresses.

arp tables da fare

ARP query

All addresses in an ARP table are added by one of two mechanisms: – ARP request-reply -¿ who is 192.168.0.16 tells 192.168.0.1 -¿ 192.168.0.16 is at 00-10-BC-2c-11-56 – Gratuitous ARP -¿ 192.168.0.16 is at 00-10-BC-2c-11-56

The discovery process happens through queries to neighbor devices – Broadcast message to the desired IP • L2 ethernet address FF-FF-FF-FF-FF-FF

The system with the requested IP replies back with its correct mac address

frame + tables + eg+ broadcast ancora da fare

ARP poisoning

ARP answers or Gratuitous ARP frames do not require an (additional) answer/confirmation, so it's a declarative protocol. This means that nodes are not authenticated, so whomever can say I am x.x1.x2.x3, my mac address is hh.hh1.hh2.hh3.hh4.hh5. • C can tell B "D is at [C mac address]" • C can tell D "B is at [C mac address]" • As a result every

communication between B and D will pass by C

da finire

ARP poisoning - limitations Works only on local networks, where MAC addresses are actually meaningful. NB: when communication is targeted to different networks, IP addresses are used. Routers and DNSs have MAC addresses too.. The poisoning works because systems are not authenticated, some implementations/third party tools can mitigate the problem, such as checking for anomalies.

ip header credo convenga guardare da doriguzzi

Subnets and CIDR + es da fare

Subnets are logical divisions of IP addresses, so possible to split a network in multiple sub-networks IP bits are divided in: – x network bits – y subnet bits – z host bits Subnet mask indicates sections of IP addresses meant for network+subnet – 255.255.255.0 à 24 bits to network+subnet, 8 bits to hosts CIDR à synthetic way to represent subnet masks – Classless Inter-Domain Routing – Indicates number of bits covered by the mask – 192.168.10.1/24 = 192.168.10.1/255.255.255.0

Chapter 3

Reading Materials

In this chapter I'm gonna sum up the extra readings suggested by professor.

3.1 Attack Trees - B. Schneier

Bro tells us that security is all a set-up, not only are systems compromised extra often but moreover you never really know what security means, is the system secure for me? for a few days? etc...

So we need a way to model threats against computer systems to understand the risks and to be able to make decisions about how to mitigate them. Basically, you represent attacks against a system in a tree structure, with the goal as the root node and different ways of achieving that goal as leaf nodes. Each node becomes a subgoal, and children of that node are ways to achieve that subgoal. Note that there can be AND and OR nodes, where AND nodes are subgoals that must be achieved together, and OR nodes are subgoals that can be achieved independently. Once the basic attack tree is completed we have to assign values to the various leaf nodes: I for impossible and P for possible; then calculate the security of the goal. The value of an OR node is possible if any of its children are possible, and impossible if all of its children are impossible. The value of an AND node is possible only if all children are possible, and impossible otherwise.

Assigning "possible" and "impossible" to the nodes is just one way to look at the tree. Any Boolean value can be assigned to the leaf nodes and then propagated up the tree structure in the same manner: easy versus difficult, expensive versus inexpensive, intrusive versus nonintrusive, legal versus illegal, special equipment required versus no special equipment...

Assigning "expensive" and "not expensive" to nodes is useful, but it would be better to show exactly how expensive. It is also possible to assign continuous values to nodes, including probability of success.

NB: Every time you query the attack tree about a certain characteristic of attack, you learn more about the system's security.But to make this work, you must marry attack trees with knowledge about attackers.

TODO: pgp example + images I guess

Our friend then makes an example with PGP to get to say two things: the first is that I can write these attack trees even as if they were an index of a book, and the second is that I don't really need to know how all works, I just need to know the value of root nodes to protect. Which I don't understand too much, but okay, I'm gonna go with trust, he probably know more than I do. Conclusion cute directly from the paper: Attack trees provide a formal methodology for analyzing the security of systems and subsystems. They provide a way to think about security, to capture and reuse expertise about security, and to respond to changes in security. Security is not a product—it's a process. Attack trees form the basis of understanding that process.

3.2 Reflections on Trusting Trust - K. Thompson

Key of the paper: To what extent should one trust a statement that a program is free of Trojan horses. Perhaps it is more important to trust the people who wrote the software. Basically he makes an example to demonstrate that you can't trust the source code of a program, because you can't trust the compiler. He writes a program that, when it compiles the login program, it adds a backdoor to the login program and to the compiler itself. So, even if you have the source code of the compiler, you can't trust it.

Substantially is the same example that professor made during the lecture.
you can't trust anything that is not totally create by yourself. No amount of source-level verification or scrutiny will protect you from using untrusted code. As the level of program gets lower, these bugs will be harder and harder to detect. A well-installed microcode bug will be almost impossible to detect.

3.3 An Introduction to BGP - Geoff Huston

3.4 The Internet:A System of Interconnected Autonomous Systems

3.5 Security Problems in the TCP/IP Protocol Suite - S. Bellovin

3.6 Analyzing Interaction Between Distributed Denial of Service Attacks And Mitigation Technologies - Blackert et al.

3.7 Statistical Analysis of Malformed Packets and Their Origins in the Modern Internet

3.8 Network Support for IP Traceback

3.9 Inferring Internet Denial-of-Service Activity

3.10 A brief introduction to DNSSEC

3.11 The Sad Story of DNSSEC

Bibliography