Advanced Topics in Computer Network and Security Notes

Rovesti Gabriel

1 Summary

Disclaimer

The course is made up by three parts:

- following the speakers (which there are notes here of the ones I was present in, not all of them)
- doing presentations and listening to others' presentations, while making provoking questions
- a third part consisting of projects on papers

Given I was the one who created information files on this, I feel to add no more, but I collected here some notes to give you the idea of such meetings, which from a learning point of view are really good.

Just to give context of a course schedule for this course, just have a look here.

2 First Part of the Course: Guest Presentations

2.1 Course introduction

(Host: Mauro Conti)

The course is structured this way:

1) Advanced Topics

We will cover, through lectures and talks from invited speakers, recent and relevant security issues in traditional and novel technologies, such as:

- IoT and Cyber-Physical System
- (Adversarial) Machine Learning for Security
- Blockchain
- Advanced Cryptography Applications
- User Authentication

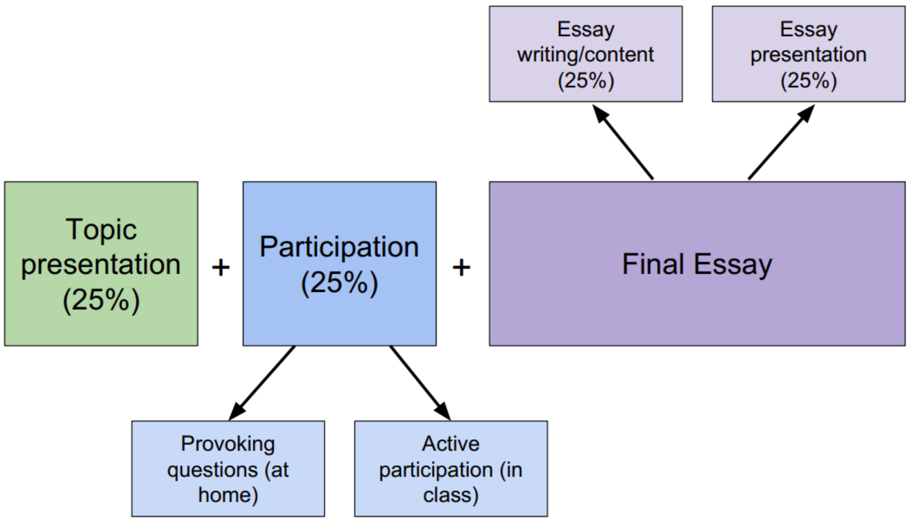
2) Students Presentations

- Students present to the class a given topic
  - Group of about 3 students
  - The topics are assigned (from a list available on course website) through a bidding phase, at the end of the Part I
  - Topics are like the one presented in Part I
- Students are also required to:
  - Send provoking questions regarding the topics presented by other groups
  - Interact with the presenting group during the lecture

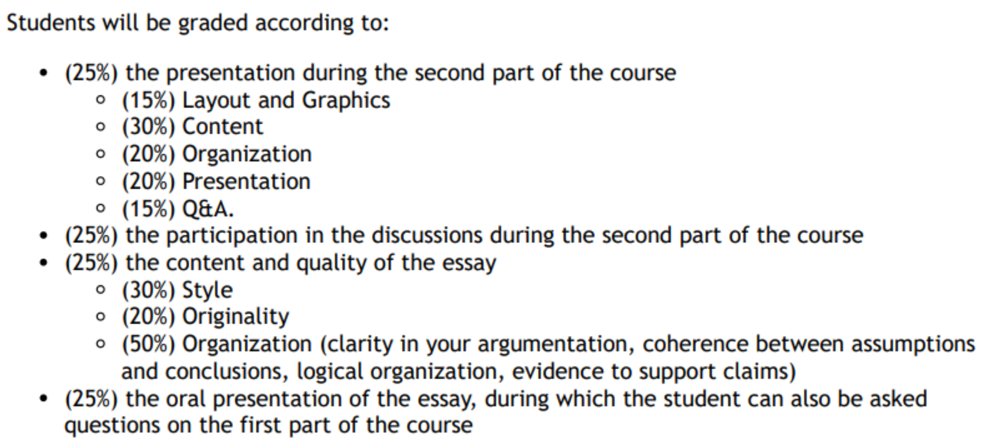
Each group (as identified in Part II) is evaluated through a final project:

- The goal is identify improvement directions of a state-of-the-art problem
- The topic should be "close" to the one presented in Part II (topic shall be identified together with the lecturer)
- The work should be supported by experiments
- Essay (about 10 pages) + presentation of the project

The grading criteria works like this:


Essay writing/content (25%)
Essay presentation (25%)
Topic presentation (25%)
+
Participation (25%)
+
Final Essay
Provoking questions (at home)
Active participation (in class)

Going deeper, this is the evaluation:



Students will be graded according to:

- (25%) the presentation during the second part of the course
  - (15%) Layout and Graphics
  - (30%) Content
  - (20%) Organization
  - (20%) Presentation
  - (15%) Q&A.
- (25%) the participation in the discussions during the second part of the course
- (25%) the content and quality of the essay
  - (30%) Style
  - (20%) Originality
  - (50%) Organization (clarity in your argumentation, coherence between assumptions and conclusions, logical organization, evidence to support claims)
- (25%) the oral presentation of the essay, during which the student can also be asked questions on the first part of the course

Projects can be:

- Proposed by the group and discussed with the teaching team to evaluate the feasibility
- Selected through a list of proposals presented by SPRITZ group members. There will be a project presentation in early December

2.2 How papers are made and carry out research

(Host: Mauro Conti)

There is a methodology in doing research, just like the scientific method (make observations, form hypotheses, experiment, analyze, report and reproduce results). In our case, it's reading papers, attending talks, thinking and discussing (not falling into the *survivorship bias*, so focusing on some parts of the process thinking we're collecting the right data, when actually we're not, to analyze a particular phenomenon). Also, we make patents to create original ideas, and something not seen before (these are considered factual for the most part).

When we're doing research, we're at the edge of the knowledge, when we're consolidating facts and things nobody knows to build new knowledge and facts. We're not alone in this, it's made by the community and other researchers as well, consolidating a clear understanding of themes. Keep in mind we're designing sound experiments, constantly looking at data, criticizing the overall work and questioning constantly.

Papers usually range from 6/10 pages up to 30, made on the outside by:

- a title
- list of authors (affiliations)
- abstract to give the general idea and context
    - give the good idea to the right people, to try to invest further into your text (keep it short)
- introduction on the problem and its history, the motivation behind and the scientific motivations about the work and the contributions
- a section about related work, comparing what's already there with our works, highlighting what our work does more than others

Inside instead, we structure like:

- description of the proposal, giving background knowledge, a formal definition of the problem and its method and the overall components
- experimental evaluation, implanting the experimenting and describing the tools used, presenting results and discussing limitations (supporting claims validly)
- conclusions, summarizing contributions and future research conclusions

The *review process* is made by picking a venue of evaluation to other scientists (journal/conference) keeping an eye to the deadline submission.

The venues can either be:

- scientifical journals, places established for several years, where there is a board of experts responsible of evaluating papers (chief/associate editors/reviewers)
- conferences, mostly one shot in a specifical place (many chairs like conference, submission, publicity chairs, etc./program committee members/reviewers)
    - it's important to understand the quality of the conferences, looking into rankings
    - useful link: https://people.engr.tamu.edu/guofei/sec_conf_stat.htm

The chairs are responsible for the reviews and very few are accepted. We disclose information ethically, presenting good for a career's sake but also having a good paper. To read papers, good places are IEEE Xplore, ACM Digital Library, Google Scholar, dblp, Springer Link, etc.

To assess a paper, it's important to read it, analyzing the person impact, the author reputation, citations (more advanced: assessing researchers, looking for citing, h-index [used to quote the impact of a paper and uses h as the index of number of citations by other authors at least that same number of times. For instance, an h-index of 17 means that the scientist has published at least 17 papers that have each been cited at least 17] and more). ("What is an h-index? How do I find the h-index for a particular author ...")

Also, one can look for the citation graph that describes the citations within a collection of documents, linking all the citation in between and see how problems were linked and solved.

2.3 Containers and Kubernetes Security

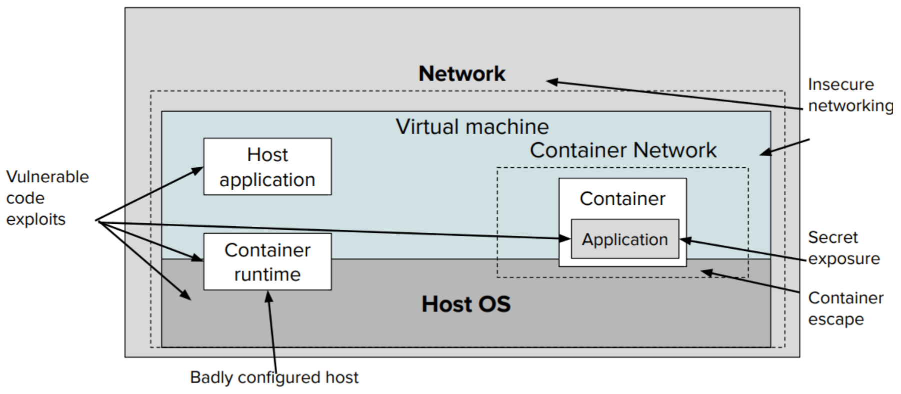
(Host: Alessandro Brighente)

We're talking about security inside the supply chain, given the different connections between people and software. In fact, usually there's the need for software updates, which also regards these kinds of topics, being fast in developing and delivering software (abstract of the context).

In this case, we're talking about containers, which are standards units of software packaging up code and dependencies to run software quicker and bundling a specific configuration of code libraries, configuration files between different environments. They virtualize the OS, so they are lighter and have the OS and its underlying resources belonging to it. This allows containers to be deployed everywhere and allows a usage for velocity and reliability purposes: cloud-native applications, making them ideal for building microservices architecture.

They are different from virtual machines, virtualizing the underlying hardware component so that multiple OS machines can ran and access effectively hardware resources, containing OS image, libraries, apps. The applications need an hypervisor, creating an infrastructure to abstract all the VM images. They are less portable but can be scaled slowly because they are more resource-intensive.

One very well-known platform providing the isolation feature is Docker, allowing multiple side-by-side containers and isolating dependencies from the underlying machine and package versions. There might be threats however; consider the *containers threat model*, because containers can give access to the machine, so giving overprivileged pieces of code can exploit vulnerabilities on the system, given badly configured host (attackers from all sides, external/internal/inadvertent).



A good overview of vulnerabilities:

1. **Insecure Networking:**

Insecure networking in the context of containers refers to the use of inadequate or unprotected network configurations, which can expose containers and their data to various security risks. This includes:

- **No Network Segmentation:** Containers by default can communicate with each other and the outside world. Without proper network segmentation, a compromised container can potentially access or attack other containers on the same host.
- **Unencrypted Traffic:** Communication between containers and external services may occur over unencrypted channels, making it vulnerable to eavesdropping. This could result in sensitive data exposure or unauthorized access.

- **Lack of Network Policies:** Without network policies, containers may have overly permissive network access, allowing unintended inbound or outbound traffic, which can be exploited by attackers.

2. **Secret Exposure:**

Secret exposure involves the inadvertent disclosure of sensitive information, such as API keys, passwords, or encryption keys, within containerized applications. This can happen due to:

- **Improper Storage:** Secrets stored within container images or configuration files can be easily accessed by anyone with access to the container, potentially leading to unauthorized access to services or data.
- **Logging Secrets:** Insecure logging practices may lead to secrets being written to log files, which could be accessible to unauthorized users.
- **Environment Variables:** Secrets passed as environment variables can be visible within the container, especially if other processes or users can inspect the environment variables.

To mitigate this risk, it's essential to use secure secret management tools and practices, such as Kubernetes Secrets or Vault, to store and securely manage sensitive information.

3. **Container Escape:**

Container escape refers to a security breach in which an attacker manages to break out of the container's isolated environment and gain access to the underlying host system. This can be extremely dangerous, as it can potentially compromise the entire host. It may happen due to:

- **Kernel Vulnerabilities:** If a container exploits a kernel vulnerability, it can escape its confined environment and interact with the host system.
- **Misconfigurations:** Inadequate container configurations, especially related to namespaces and cgroups, can create vulnerabilities that attackers may exploit to escape.
- **Legacy Linux Capabilities:** Certain Linux capabilities can be used by containers to gain excessive privileges and break out of the container.

To prevent container escapes, it's crucial to keep host systems and container runtimes up-to-date with security patches, enforce strict access controls, and follow best practices in container configuration and isolation.

4. **Vulnerable Code Exploits:**

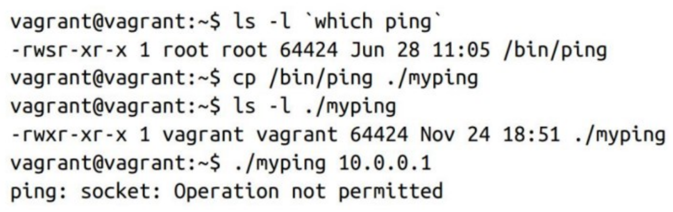
Vulnerable code exploits occur when an application or its dependencies contain known security vulnerabilities that attackers can exploit. In the context of containers, this can happen in several ways:

- **Insecure Images:** Using container images with outdated or unpatched software components can expose your application to known vulnerabilities.
- **Image Scanning:** Failing to regularly scan container images for known vulnerabilities can result in the deployment of images with exploitable weaknesses.
- **Zero-Day Vulnerabilities:** While not known, attackers may discover new vulnerabilities in containerized applications. Regular security updates and monitoring are essential to respond to zero-day threats.

To mitigate the risk of vulnerable code exploits, maintain an up-to-date inventory of container images, apply security patches promptly, and use container image scanning tools to identify and address known vulnerabilities in your images.

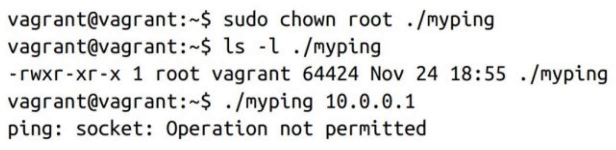
Usually, the applications can run in the *user space*, so if the user wants to access a file, it should ask the kernel to do so; the interface allowing the user space to make there are requests are system calls/syscalls (inside of Linux, there are more than 300 syscalls, both inside and outside the system).

When you execute a file, the process that gets started inherits the User ID.



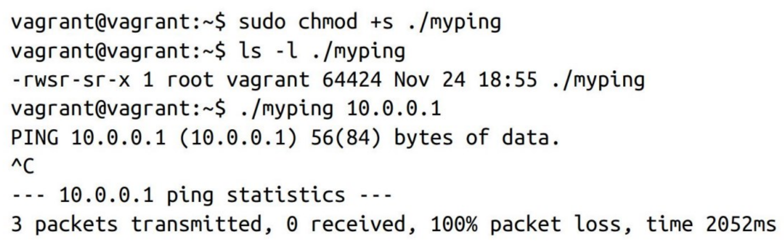
```
vagrant@vagrant:~$ ls -l `which ping`
-rwsr-xr-x 1 root root 64424 Jun 28 11:05 /bin/ping
vagrant@vagrant:~$ cp /bin/ping ./myping
vagrant@vagrant:~$ ls -l ./myping
-rwxr-xr-x 1 vagrant vagrant 64424 Nov 24 18:51 ./myping
vagrant@vagrant:~$ ./myping 10.0.0.1
ping: socket: Operation not permitted
```

We can also change the ownership, but still cannot run it unless root.



```
vagrant@vagrant:~$ sudo chown root ./myping
vagrant@vagrant:~$ ls -l ./myping
-rwxr-xr-x 1 root vagrant 64424 Nov 24 18:55 ./myping
vagrant@vagrant:~$ ./myping 10.0.0.1
ping: socket: Operation not permitted
```

If we set the UID bit (*setuid*, creating executable permissions and *setgid*, to affect both files and directories).



```
vagrant@vagrant:~$ sudo chmod +s ./myping
vagrant@vagrant:~$ ls -l ./myping
-rwsr-sr-x 1 root vagrant 64424 Nov 24 18:55 ./myping
vagrant@vagrant:~$ ./myping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
^C
--- 10.0.0.1 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2052ms
```

To prove more granularity over privileges, there are *capabilities*, which need to be set since version 2.2 of Linux OS and are assigned to assign specific privileges (in case of containers, what they do in a machine). A good overview here: https://man7.org/linux/man-pages/man7/capabilities.7.html

We can set capabilities for command we have *setcap* and *getcap* to know what they are.

We have *control groups (cgroups)*, which are a kernel feature to allow an admin to allocate resources such as CPU, memory, and I/O bandwidth to groups of processes. Managing them involves reading and writing to files/directories within those hierarchies, seeing them as listing content to directories.

Cgroups are like processes in that:

- they are hierarchical, allowing use a set of pseudo-filesystems that involves reading and writing to files and directories within these hierarchies
- child cgroups inherit certain attributes from their parent cgroup.

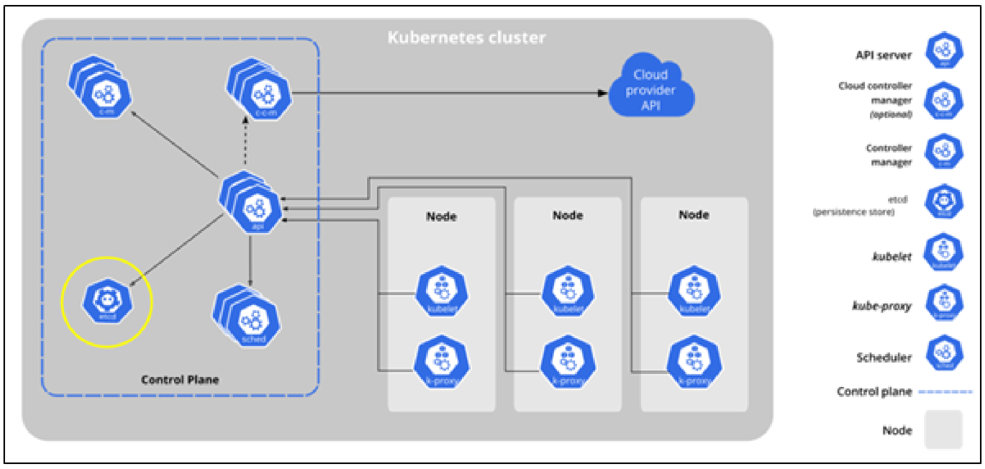
Creating a subdirectory in the memory directory creates a cgroup, where the kernel automatically populates the directory with the various files that represent parameters and statistics about the cgroup. When you start a container, the runtime creates cgroups for it. By default resources (e.g., memory) are not limited, so if a process is allowed to consume unlimited memory, it can starve other processes on the same host. This can open to a resource *exhaustion attack*: use as much memory as possible. They are created automatically within platforms like Docker.

If cgroups control the resources a process can use, namespaces control what it can see, which wrap global system resources in an abstraction that makes it appear to the processes within the namespace that they have their own isolated instance of the global resource. (“difference between cgroups and namespaces - Stack Overflow”) "Changes to the global resource are visible to other processes that are members of the namespace but are invisible to other processes." (“Podman vs. Docker Difference - Which One to Choose? | Codica”) One use of namespaces is to implement containers, this way isolating hostname of a machine from other things.

We can use the *unshare* command, creating a new process from kernel to the program and then some namespace unshared from parent (just like containers in managing resources work, but you need root to do so). In the case of Docker, we can list the whole processes in the host which are independent from the host machine and the *docker build* command to create images, setting instructions in a so-called Dockerfile, carefully managing resources and APIs actions. Remember any user can trigger *docker build* and performing *docker run*.

Anyone who has access to a container image can access any file included in that image. That’s why inside images we have *image layers*, where each one is stored separately, meaning that you must be careful not to store sensitive data, even if a subsequent layer removes it.

We need a framework able to run distributed system resiliently with a lot of containers, so we need an *orchestrator*; for this we have software like *Kubernetes*, designed by Google initially, providing a framework to distribute the system resiliently, operating with *clusters* when deployed.



The smallest units deployable are *Pods*, which are environments where multiple containers run and define a trust boundary. Each one has its own IP access, data linked to the and a trust boundary for its data. What matters is that they have no identity, security context, encryption or something like that (we need to set all these things ourselves). The lifecycle of a pod is controlled by the *kubelet*, the Kubernetes API server’s deputy, deployed on each node in the cluster to manage and run containers (traffic unencrypted here may be easily sniffed by compromised pods/nodes).

By default, every pod can see and talk to every other pod in the cluster and workloads can escalate to host network interface controller, having unrestricted accesses to resources, given the absence of identity, encryption and environmental restrictions in querying data. That’s why they are powerful tools, but they may come with unsecure configurations, poisoned, or reuse pieces of code that are not secure: they may come with unsecure configurations, poisoned, or reuse pieces of code that are not secure. Defending against supply chain attacks is a top-priority for companies.

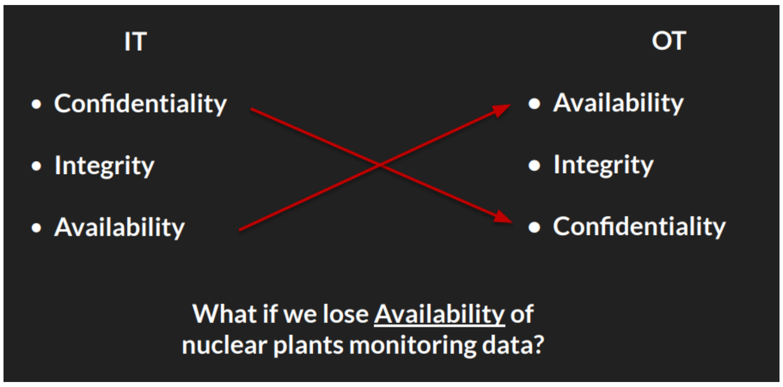
Written by Gabriel R.

2.4 SPRITZMATER – SECURITY PARTNER OF INNOVATION

(Host: 2nd talk – Federico Turrin)

We talk about *cyber-physical systems* as systems that interconnect physical processes (operation technology – OT) and information technology (IT) (some examples might be smart grid, smart cars, industrial control system, e-Health devices). Contrary to traditional IT systems, which only access the internet, these kinds of system connect to the internet, but also physical connections to defined.

We need confidentiality, integrity and availability for those systems (with the last one being very important). Today's vehicle networks are mostly based on those, for example with a technology called the CAN bus (Controller Area Network (CAN) bus is a widely used standard in the automotive industry for communication between various electronic components within a vehicle) to have access to the antennas or vehicle music.


IT
- Confidentiality
- Integrity
- Availability

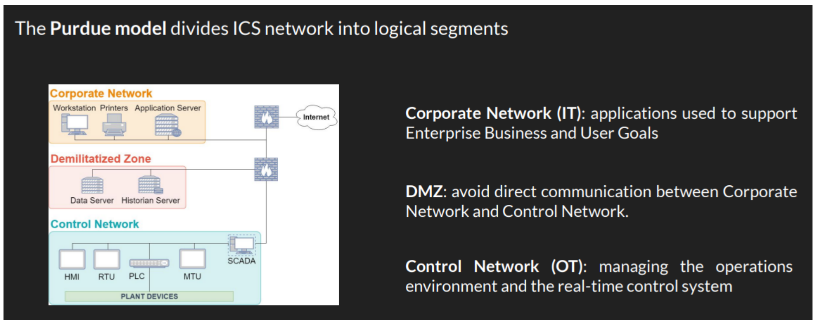
OT
- Availability
- Integrity
- Confidentiality

What if we lose Availability of nuclear plants monitoring data?

Today's vehicle networks are truly distributed electronic systems, via the simple protocol of CAN buses, like drive-by-wire," where electronic sensors and control units manage functions like throttle control, steering, and braking. Also, they provide steering aids, safety features in engine/braking/airbag/navigation control.

A paradigm used to control and monitor vehicles are Vehicle-to-Grid (V2G), with wide spreading electric vehicles, enabling secure communication. Air Ground, instead, we have Automatic Dependent Surveillance (ADS-B), to monitor everything specifically. Here there is *no security at all*, so eavesdropping and injection/modification of messages can happen at any time (e.g. Opensky network).

We also see Industrial Control Systems, categorized also as Critical Infrastructures are nuclear power plants and water treatment systems (damaging them means several damages in business/environment/human lives). In this model, we have a concept of demilitarized zone to prevent threats and firewalls for system, supervising and monitoring system accordingly to how much problem can cause the coming of other entry points (between operation, control and operational devices). An example of its architecture:


The **Purdue model** divides ICS network into logical segments

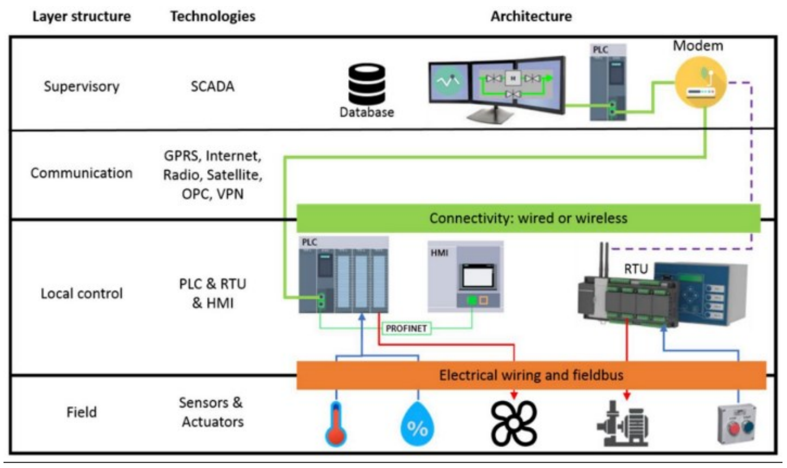
Corporate Network (IT): applications used to support Enterprise Business and User Goals

DMZ: avoid direct communication between Corporate Network and Control Network.

Control Network (OT): managing the operations environment and the real-time control system

Cyber-attacks on this system enabled of many dangerous impacts, even health and lives at times (famous example: Stuxnet). In this case, we lose customer trust or even take fines. Protocols are designed with no auth, no integrity or TCP/IP protection, adapting to communication and connection to legacy devices (e.g. Modbus, which consist in a simple master/slave communication with no security by design).

These systems are called SCADA - Supervisory Control And Data Acquisition, working on such protocols like Modbus, based on ladder logic programming to receive I/O, controlling everything downwards and remotely collecting data, while maintaining control on everything.



Other systems may be:

- sensors, measuring physical and generating continuously domain values
- actuators, which perform actions on the system and generate categorical values
- PLC – Programmable Logic Controllers, which receive I/O and are based on ladder logic programming, structured and sequential
- HMI – Human Machine Interface, made with a GUI to manage system state, alarms, errors and allowing inputs

The SCADA systems are at the higher level of the ICS hierarchy, so they monitor and control acquired from field sites, this way managing the communication between the various devices. They need to be monitored real-time, working with legacy systems most of the time, so they're not well suited for control systems, given their simpler network dynamics.

The communication links lack on security features, also no physical protection and they have a long lifespan. These devices are continuously scanned, looking for leveraging tools to craft attacks. This might not be enough and can lead to outdated results, while exploring vulnerabilities and devices. An interesting piece of software is *Shodan*, an engine which continuously maps internet connected devices and shows ports, modules, names and details.

Some interesting search queries: https://github.com/jakejarvis/awesome-shodan-queries

To protect from these threats we need:

- Security by Design (according to standards, security best practices, challenging the current architectures, etc.)
- Continuous Security Assessment, testing always to many levels the security in networks via for example of VAPT tests, snapshotting the network and then trying to patch it

2.5 Natural Language Processing in Cyber Threat Intelligence

(Host: Puthuvath Vinod)

We analyze an attack as:

- Reconnaissance of an attack, gathering information and vulnerabilities
- Attack to the corporate network
- Expansion, leveraging vulnerabilities to other machines in the network
- Keeping control inside of the network

Inside industries, we try to analyze a lot of data, filtering between a lot of useless data, just to understand what's going on. The main goal is to infiltrate inside of a network with a purpose, given the vulnerability of cyberspace.

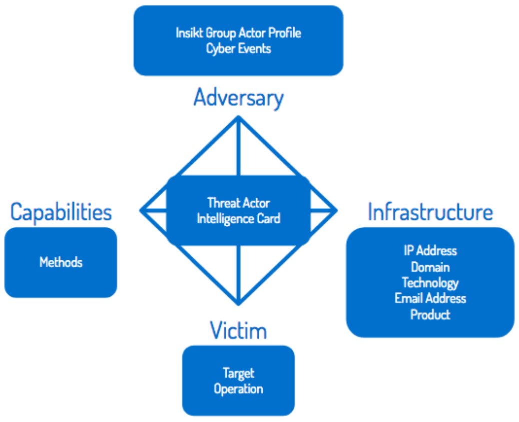
Cyber threat intelligence (CTI) is data that is collected, processed and analyzed to understand a threat actor's motives, targets and attack behaviours. We do preprocessing to extract knowledge from data, understanding the attacker's behaviour, finding out what will happen. This is evidence-based knowledge about existing or emerging menaces/hazards to inform decisions about specific attacks. This way, we will understand tactics, techniques and procedures.

We identify the Indicators of Compromise (IOCs), actionable pieces of CTI to identify and block attacks. They are forensic artifacts signals that a host or a network might have been compromised. IOC can be collected by Structured Threat Feeds or unstructured documents (like blogs, articles, social network, etc.).

The indicators use as parameters:

- Network
- Social
- Blockchain
- Cryptography
- Attack
- Vulnerability
- Location

A famous model available is the *Diamond Model*, which focuses on four core elements: adversary, infrastructure, capability and victim. It helps analysts to understand relationships between these elements and interactions during a cyber-attack.



IOC-centric CTI systems are questionable though:

1) They can't depict the technical details
2) Attackers frequently change infrastructure, so having static indicators may be limited

We have CTI extraction methods:

- Dictionary based
  - Using a list of predefined terms so they cannot capture previously unseen CTI data (they yield high false positive and negative rates) ("Vulcan: Automatic extraction and analysis of cyber ... - ScienceDirect")
- Rule based

    - Here CTI data is represented with letters and numbers, trying to find patterns or rules
- Deep learning based
    - Try to gather a lot of data, giving satisfactory performances when enough will be gathered

The goal is trying to gather as much CTI data as possible, defining *tactics* as high-level strategies used by threat actors (classical example: phishing). We define *techniques* as the specific methods or tools employed with tactics, include social engineering (playing with emotions), forcing victims to click malicious links or techniques. To prevent this, we try to outline the step-by-step process followed by adversaries to execute their tactics and techniques.

The goal of analyzing data is finding links to similar reports of a specific problem and correlating that to specific companies and specific entities. In natural language processing, we want to find processes to extract entities and associate parts of speech and terms to specific things (specifically, it's defined as *named entity recognition*) or grammatically/semantically understand the term (if it's malware, if it's a verb, noun, etc.) via a 1 to 1 mapping.

Each data will be tagged via some terms and via specific libraries (e.g. Python), we can name data and recognize the specific entities (basing itself to English text meanings and not cybersec meanings, e.g. virus which could be a malware and not a disease). Speech tagging, by itself, it's not a big deal: we just give a tag to each word, segmenting things with no problems. Naming entities, also, assigns labels to spans of text, more difficult because of ambiguity in meanings. There will be something called *cross-type confusion*, where same entities can refer to people, locations, organizations, etc.

A model bases itself tries to categorize entity types and boundaries, having huge amounts of data to annotate things and label them according to abbreviations and acronyms without many variations or ambiguity. We then adopt a specific annotation strategy, taking every word in a sentence and tagging labeling sequence for span-recognition problem, where each word has a single label (each word will get categorized inside spans of interest according to the specific context); examples of tagging may be IO/BIOES tagging.

Now we have annotated data and for each sample, we want to label data according to human annotators. Using *Cohen's kappa statistics*, we can calculate measure inter-rater reliability (how independent are the terms) for qualitative (categorical) words (so, how much words are linked together according to labels), putting them in an inference matrix and rate variables and words alike (also *Fleiss Kappa*, analyzing the relationship between multiple terms and correlations).

The goal is creating *dependency parsing*, determining relationships between words via libraries or toolkits. The approach is first crawling data inside the cyberspace, performing breadth-first crawling, starting from homepage and discover each link until no new one is found. This one is based on *blog scrappers*, comparing DOM's pages to template of pages to drop outliers (in this domain, we talk about creating *focused crawlers*). Something specific might be a *topic collector*, where we filter IOC articles (usually longer ones, having lower dictionary) from non-IOC ones, selecting from articles up to *x* words together with their frequencies.

On this, a classification model can be classified over subsets of datasets, label the unknown sets and validating manually selected instances from the classified results. To identify sentences, we look for IOCs using regex and context terms, referring to existing standards. We then check relations between context terms and then parsing sentences into a dependency graph, finding main parts of sentences then parsing all the other parts and finding the shortest paths in between single words (transforming them into vectors and then checking all relationships between every single term).

The relation checking refers to the verb in between two entities in the graph, mapping the verb with action/relation in STIX (Structed Threat Information Expression) and find the relation. The concept of self-attention (which refers to understanding the context of a sentence and what do parts implicitly refer into sentences) is complicated for computers or machines (for example, "the AV did not detect the trojan because it was too complex; a PC does not understand what "it" refers to).

Basically, words are embedded, then transformed according to embeddings and representing values between key/query/value and finding out the attention scores. The *Transformer model* (which labels data weighting words and then stacking them into layers according to the specs) is what's in mainly used in this context; for example, we have the BERT (Bidirectional Encoder Representations from Transformers), learning how contextualized words are bidirectionally linked between semantics and contexts. In this, you mask words and make the model identifying the missing words; also, we can make the model foresee what the next sentence will be.

Federated learning (FL) then aims to collaboratively train a ML model, keeping data decentralized and sharing model between servers, averaging the parameters then sharing all data, repeating the things until the model it's closed. We can also have the P2P FL, selecting the specific server for distributing globally the model, training it with local data, exchanging models and aggregating data as well.

2.6 User profiling in video games

(Host: Pier Paolo Tricomi)

It’s important to gather the right information of users, to construct analytics properly and personal profiles. A lof of scams take over accounts. We try to create a game “fingerprint”, banning harmful players and creating a new “biometric” authentication system. This is done, for example, tracking game habits, for example the way we move in a virtual world on the camera in a game.

This builds a one of a kind “identification framework”, starting from an unknown population of players and then aggregate data, ultimately creating ML models to track habits. This way, given the game features, we can leverage vulnerabilities or typical game characteristics to create a good analysis of habits. In this, we can try to learn from past mistakes and trying to use *tracking websites*, to show you all the player stats and specific things. This data is generally public, to give visibility or to learn tricks from other players and we can exploit it.

For example, we can use ML to infer private data from public data, using a model to track and predict data from the whole data that is being released. The beginning is trying to find the correlation between private and public data from players in-game statistics and real-life data, trying to find matches in players or in a reduced amount of time (also considering player time, say people who play a lot of matches and others who don’t have time and only have a few).

In case of playing games we consider habits even in emotions and age, for example even if players are introverted/extroverted, young (more aggressive), purchasers and workers, etc.

2.7 Breaking AI in practice

(Host: Luca Pajola)

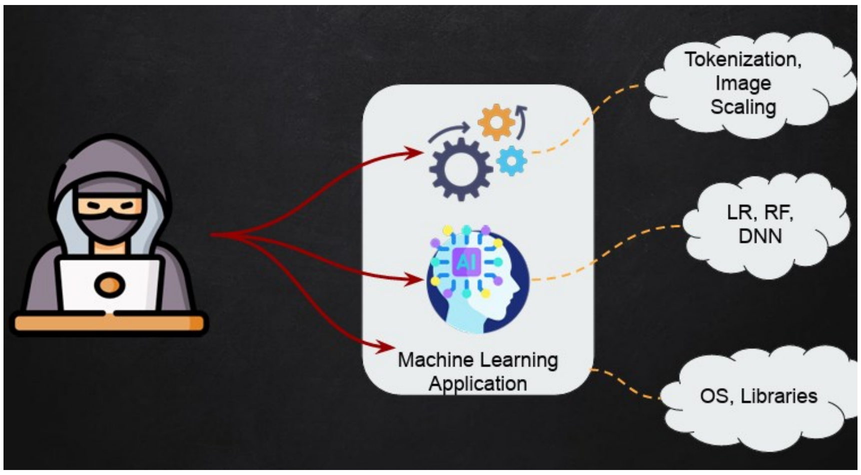
Malicious actors might attack application from all sides, so we should educate people to prevent dangers in this sense; there may be dangerous pieces of software, deploying in many sensible contexts and create a ML model considering security is important.

An interesting example was a chatbot developed by Microsoft called Tay, interacting with it in Twitter, with slogan "the more you chat with Tay, the smarter she gets”. Over time, the model learned to insult people and to be politically incorrect. That was shut it down in a few hours. Let’s then introduce *adversarial machine learning*, where we try to leverage the adversary knowledge for model and data. Then we query info based on input modifications and treat different security levels (*white-boy*, unlikely on a real scenario so very big amount of info and *black-box*, more realistic but less info/capabilities).

We can have MLaaS models (Machine Learning as a Service) models, creating techniques allowing access to a lot of knowledge and platforms to further learn inside the context. We have a behaviour which we train over targeted queries and understand if a sample is used to train a victim gaining knowledge and leaking privacy, while reducing gap between training and validation.

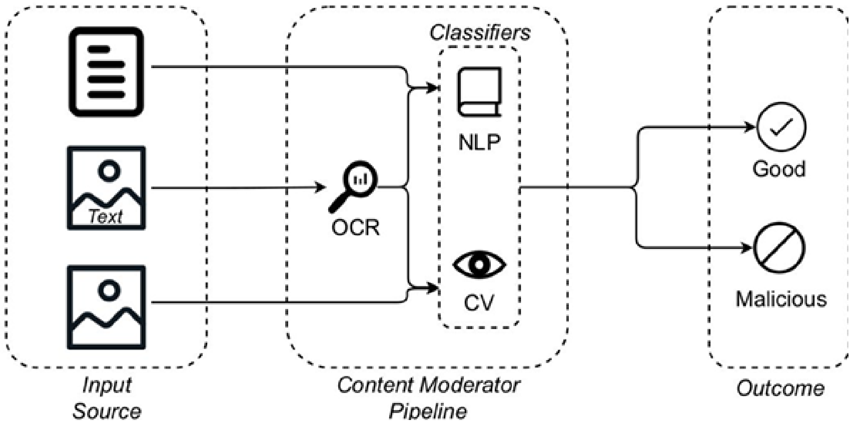
Officially, a possible exploit is trying to evade a model, leading to wrong answer and wrong learning from the model. The defense here is adversarial ML. The threat is model poisoning, reducing model performance, so we try to infer sanitization techniques at training time. Officially, a trigger will be inserted with wrong decisions at test time, then produce a misclassification and noise in classification. A good example might be a speed limit sign, where a STOP sign will be wrongly recognized as a speed limit because of lighting.

In adversarial machine learning, we take an ML application and try to understand the whole context:



For example, a good attack is the *zero-width space*, used in typesetting to indicate word boundaries and leverage the Unicode chars to create commands from containing non-displayed characters, including the zero-width space, and most browsers prohibit their use within domain names because they can be used to create a homograph attack, where a malicious URL is visually indistinguishable from a legitimate one. ("Zero-width space - Wikipedia")

Another one might be a *captcha attack*, so for example on automatic moderators inside of social networks leveraging remotion of content or posts, working specifically this way:



2.8 Deep Learning and Backdoor attacks

(Host: Stjepan Picek)

We start talking about artificial intelligence, gathering big amounts of data, depending on a powerful hardware and novel applications. AI is a big buzzword, so everyone wants a piece of that cake. Machine learning has become mainstream because of usage inside industry, like automotive. Many algorithms are old (over 10 years old; first examples were in 1987 of CNNs – convolutional neural networks).

Of course, there are issues in data confidentiality on how ML models are built and how they are considered trustworthy. We see adversary as a black-box where an adversary can query the model with any arbitrary input and have a result or the white box, where the adversary has information. They may be targeted (chose inputs to have desired outputs) or untargeted (degrading task performance and achieving optimal performance).

Data can be anything, from images, video and sound up to graph and neuromorphic data (used in spiking neural networks, introducing the concept of time in model). In models, there may happen *evasion attacks*, so wrong inputs in inference time (when you train a model, you give data and labels to identify), so you will craft a wrong model as output. For example, in images there may be wrong pixels, and this may potentially lead to wrong analysis.

A completely different class of attacks are the *poisoning attacks*, contaminating the model at training time, bringing misclassifying examples or just altering data. This can happen on data and labels which are misleading created to alter perception. This can happen on the algorithm itself, on the model (manipulating it and touching components).

A category of attacks here are the *poisoning attacks*, which give access to subsets of training data, altering data examples and activating triggers to activate vulnerabilities inside training data. The model this way there would be a *backdoored model*, having activated triggers via random noises for example. This may lead to label which are dirty (poisoned) and clean (not changed), activated by specific inputs and multiple triggers. In doing attacks, we measure the attack success rate, removing samples and inserting triggers to train models accurately considering differences between them.

To activate triggers physical objects can be used (no matter the size), using scaling algorithms to exploit the image rescaling and size. There may also happen invisible backdoors, generating attacks via encoder-decoder networks and training simultaneously string with minimal differences to hide messages then recovered by the decoder (this is like the *steganography* approach (the practice of concealing information within another message or physical object to avoid detection – hidden writing, so no one can read it except the person that has the decrypting key).

There are also backdoors to leverage very small perturbations poisoning the classifiers via small triggers overtime. Some backdoors even try to infer backfunctionality to misclassify the loss function of a ML model to poison the classification, while others may try to infer trojans inside text code.

In text models some backdoor attacks leverage semantic attacks (using different words in a sentence to make it have the same meaning this way misclassifying the model correctly) or in the style of text (writing style to avoid model recognition for patterns in data); this can happen also in the file structure (for example, tabular data is changing the position of values or even the formatting, not difficult to notice but the system may not complain).

It can also happen in frequencies in altitude and their frequent change, superimposing the trigger on those and exploit the linearity of a system, adding noise according to the input type.

2.9 Homomorphic Evaluation of Convolutional Neural Networks

(Host: Huanhuan Chen)

We see machine learning as a service, bringing millions of dollars in revenue given the huge amount of data present (so, it's important to have it privacy-preserving). We analyze Convolutional Neural Networks (CNNs):

- A type of deep learning neural network widely used for analyzing visual imagery.
- CNNs use a variation of multilayer perceptrons (basic unit of an artificial neural network) called convolutional layers, which contains neurons arranged in 3 dimensions (width, height, depth).
- It consists of convolution and pooling layers that help extract features from input data like images which are spatially correlated.

- Commonly used for computer vision tasks like image classification, object detection, etc.

Everything is analyzed via different activation functions because they impact the speed and ability of the neural network to learn patterns from data during training.

- ReLU: Most widely used because it addresses the vanishing gradient problem which slows down training. It allows faster convergence during backpropagation since its gradient is either 0 or 1.
- Sigmoid: Was commonly used historically but suffers from vanishing gradient problem for large negative inputs. Produces outputs between 0-1 like a probability.
- Tanh: Like sigmoid but outputs range from -1 to 1. Also suffers from vanishing gradient issues.
- Softmax: Generalization of sigmoid for multi-class classification problems. Ensures outputs always sum to 1 so they can be interpreted as class probabilities.
- Max: Used in max pooling layers of CNNs to select the most prominent feature and reduce spatial size of representations.
- Sign: Simpler approximation than ReLU but still allows the network to learn the gradient. Provides crude weighting of positive/negative features.

Fully Homomorphic Encryption (FHE) is a form of encryption that allows computations to be carried out on encrypted data without decrypting it first. ("Confidential computing - Wikipedia") Some key points about FHE:

- With FHE, one can perform operations on encrypted data and obtain an encrypted result that decrypts to the same result of the operations performed on the plain text.
- It allows computing directly on encrypted data, preserving the plaintext inside the encryption. This enables secure cloud computing on encrypted data.
- The basic idea is to have an alternative encrypted function F' that behaves similarly to the function F that we want to compute but operates on encryptions instead of plaintexts.
- F' takes encrypted inputs as arguments, performs the computation but keeps the result encrypted, and outputs an encrypted result without ever decrypting the inputs.
- Decrypting the output of F' using the decryption key would give the same result as running F on the decrypted plaintexts.
- This allows third-parties like cloud servers to work with and perform arbitrary computations on outsourced encrypted data without learning anything about the actual plaintext data.
- It provides stronger notions of security compared to other forms of encryption which are limited to only certain operations like search.

So in summary, FHE utilizes an encrypted version of the function F denoted by F' that allows computing on encrypted data and producing encrypted results, thus keeping data confidential even when outsourced for computations.

The activation function has some limitations:

- Prior frameworks only allowed evaluating simple activation functions like the sign function directly on encrypted data.
- More complex functions like ReLU, Sigmoid, Tanh had to be approximated using low-degree polynomials which decreased accuracy.

We also say the same for the message space:

- Earlier FHE schemes could only encrypt a limited message space, often just 4-5 bits of data (e.g. map values from 0-128 to -1, and 128-255 to 1).
- This severely restricted the types of computations that could be done without precision loss.

- The batching Computations could only be done on singly encrypted inputs, not batched inputs. This reduced efficiency.
- Parallel computations were mostly performed in a serial SISD (Single Instruction Single Data) fashion rather than parallel SIMD (Single Instruction Multiple Data). This limited throughput.

So in summary, prior FHE frameworks had limitations in terms of the complexity of activation functions supported, restricted message spaces, lack of batching and limited parallelization - reducing accuracy, efficiency and types of computable functions on encrypted data. Later frameworks aimed to improve on these limitations.

<u>Learning With Errors (LWE)</u> encryption is a computational problem that is widely believed to be hard, even for quantum computers. It forms the foundation for many fully homomorphic encryption schemes.

- Unlike RSA which relies on the hardness of integer factorization, LWE relies on the approximate shortest vector problem aka the learning with errors problem over lattice bases, which is believed to be hard.
- In LWE encryption, the public key is a matrix A sampled from a discrete Gaussian distribution. The secret key is a randomly chosen vector s.
- To encrypt a message m in the message space Zq, a random vector a is sampled from A and the ciphertext is (a·s + e + m) mod q, where e is random noise from a discrete Gaussian.
- To decrypt, the receiver uses the secret key s to compute (c - a·s) mod q which reveals the message m.
- LWE encryption supports plaintext addition directly as ciphertexts can be added together component-wise to encrypt the sum of two plaintexts.
- The hardness of determining s given many public key samples (a, a·s + e) forms the foundation for the security of LWE-based homomorphic schemes like BGV, BFV, etc.

So in summary, LWE introduces learning problems over lattices that modern FHE schemes rely on, along with a symmetric encryption scheme supporting plaintext addition natively.

The noise in data gathering increases significantly with each homomorphic operation performed on encrypted data. This can cause decryption to become unreliable or inaccurate after many operations.

This problem is addressed through a technique called bootstrapping introduced by Gentry in 2009. The key ideas are:

- Noise grows polynomially with the circuit depth of the computation. Without countermeasures, decryption becomes impossible after a certain depth.
- Bootstrapping "refreshes" the ciphertext by reducing the noise level back down, without knowing the secret key.
- It works by fully homomorphically evaluating the decryption circuit itself on the noisy ciphertext. This eliminates the noise while preserving the encrypted value.
- The output is a "fresh" ciphertext with the same encrypted value but very low noise again, allowing more computations to be performed.
- This bootstrapping procedure does incur a high-performance cost, but it allows computing on ciphertexts to an unlimited algebraic depth by periodically refreshing.

So in summary, bootstrapping is a key technique that addresses the noise growth problem and enables arbitrary-depth fully homomorphic computations by periodically reducing noise to usable levels without knowledge of the secret key. It was a breakthrough enabling practical FHE schemes.

2.10 THE GROWING THREAT OF POLARIZATION AROUND ONLINE DEBATES

(Host: Alessandro Galeazzi)

Polarization in Internet data refers to the tendency for people to encounter views mostly like their own, which can occur for these key reasons:

- Homophily - People naturally tend to associate and bond with others who are like them. On social networks, this leads users to selectively follow like-minded people.
- Echo chambers - Surrounded primarily by reinforcing views, users don't encounter opposing ideas that could challenge their own perspectives. Algorithms also may accentuate this by filtering content.
- Selection bias - In picking what content and sources to consume, people exhibit preference for information aligned with their existing leanings. Algorithms study this behavior.
- Feed algorithms - Platforms profile users and curate highly personalized feeds/recommendations based on past engagement. This often maximizes usage but risks insulating users in own echo chambers.

We should care about polarization because some concerning outcomes include:

- Political deadlocks, as polarized camps refuse to compromise on solutions
- Rise of dangerous, extremist subcultures when some individuals only encounter viewpoints that validate extreme beliefs.
- Increased likelihood of collective or individual violent actions taken by those whose polarized views see opposition as enemies.
- A fragmented society where certain portions only interact with others sharing the same worldviews and don't engage with alternative perspectives.

In summary, online personalization and selective exposure risk dividing society through creating homogeneous ideological clusters or echo chambers with less exposure to dissenting views.

The provided URLs are from news outlets, and their content can be related to cybersecurity in the context of polarization and misinformation. Here are some possible ways to compare the 2016 and 2020 elections:

1. Influence of fake news on Twitter: One can analyze the frequency and reach of fake news stories on Twitter during both elections. This can be done by identifying and categorizing tweets containing fake news stories, analyzing the number of retweets, likes, and replies, and comparing the results between 2016 and 2020.

2. Source-based analysis: One can examine the sources of news stories shared on Twitter during both elections. This can be done by categorizing tweets based on the sources they link to, such as news outlets, blogs, or social media platforms. Then, one can compare the popularity of different sources during both elections.

3. Network-based analysis: One can analyze the structure of Twitter interactions during both elections. This can be done by examining the interactions between users, such as retweets, replies, and mentions. Then, one can compare the structure of the interaction networks during both elections to see if there are any changes in how information was being shared and consumed.

4. Content consumption patterns: One can analyze the content consumption patterns of Twitter users during both elections. This can be done by examining the types of content that were most popular, such as news articles, opinion pieces, or videos. Then, one can compare the popularity of different types of content between 2016 and 2020.

Written by Gabriel R.

5. Inferring leanings from contents: One can analyze the content of tweets to infer the political leanings of users. This can be done by analyzing the keywords, hashtags, and URLs shared in tweets. Then, one can compare the political leanings of users during both elections to see if there are any changes in the types of content being shared and consumed.

To perform these analyses, one can use various natural language processing (NLP) techniques, such as sentiment analysis, topic modeling, and named entity recognition. Additionally, network analysis techniques, such as graph theory and social network analysis, can be used to analyze the structure of Twitter interactions.

How the consumption of news evolved:

- Reduction of fake news and extreme bias in tweets.
- Reduction of tweets and users from the political center.
- The policies implemented by Twitter may be effective to combat fake news diffusion.
- Reduction of the fraction of tweets from unofficial clients (likely to be automated accounts).
- Decreased activity of unofficial clients.

The shift of users' leaning in the 2016 and 2020 US Presidential elections:

- Users shifted away from the political center towards left-leaning positions in both 2016 and 2020.
- There was also a reduction in the presence of fake and extremely biased content.

Who leads the debate:

- Retweet networks were constructed for various categories, focusing on retweets of classified links within those categories.
- The Collective Influence algorithm was used to rank nodes based on their influence.
- Top spreaders for each category were extracted.

Top influencers' similarity network:

- This network includes the top 30 influencers, with the top 5 for each category.
- The edges in this network represent the cosine similarity between influencers and their retweeters.
- There's been a reduction in connections between influencers with opposing political beliefs, indicating increased polarization.

Especially influencers tend to polarize people inside certain areas of thought; for example, research was conducted on the climate conferences, analyzing data from sources and topics and identifying the “influencing” people, performing correspondence analysis and estimating opinions and diving into the perspectives of gathering data.

Another interesting point was the political alliances' shift in Pakistan, analyzing similarities between graphs of data via different methods (Pearson coefficient, cosine similarity, etc.). Precisely, we can say:

Certainly, here's a text-based summary of the provided information:

Evolution Of Latent Ideology

- Over time, polarization increased, and shifts in ideology mirrored the evolution of political alliances.
- There was a dramatic increase in users' participation in 2022.

Polarization In Online Debates

- Polarization increased between 2016 and 2020 US elections, between COP21 and COP26, and between 2018-2022 in Pakistani politics.
- In the US, this growth is due to the shift from the center to left/right.
- In COP discussions, it's due to the increased presence of climate skeptics.
- In Pakistan, it's due to the evolution of political alliances.
- In the US debate, right-leaning accounts lead the discussion.
- In COP discussions, both communities debate different topics.
- In Pakistan, there was a dramatic increase in users' participation in 2022.
- In the US debate, the presence of reliable sources and automated accounts decreased.
- In COP discussions, unreliable sources tend to be used primarily by skeptics.
- High polarization may increase the risk of violence and political deadlocks and be exploited for information warfare and disinformation.

So, It Is So Simple?

- Ideology is a powerful method, but understanding why it works and its limitations is essential

Why Ideology Works?

- Ideology works because users tend to be coherent in their behavior and form communities that share similar content consumption. It may be not enough however, when data are not complete or rich enough or users were not analyzed correctly.

2.11 Non-Functional Certification of Modern Distributed Systems

(Host: Claudio Ardagna)

Distributed systems are unpredictable, and this created a lot of safety concerns; billions of devices are expected to exist, reaching a big economic impact. We need evidence of behavior of systems, because they are seen as black-boxes and there is little trust in correctness and reliability of the systems.

Data breaches are everywhere, and regulations are required to protect data. Remote working in enlarging the boundaries of bigger data control, so attackers try effective attacks on those. There is no way departing from distributed systems. The big problem is the non-functional behavior, that brings non-determinism, that brings reliability to a system. When there is enough evidence that a system holds some non-functional properties, there are certifications (starting from the software ones, then service, cloud and modern systems' ones).

Certification schemes detail certification processes, according to the system under observation, collecting models of evidence and prove properties on a system. These are built on the final products of systems, describing completeness (replicas of systems and analyzing of development and evaluation of systems' data), validity, integration and ML.

Non-functional properties are describe as set of dimensions (mathematical functions) which together are evaluated together. Instead, a certification model details the activities to perform evaluation of service and CA, even accredited labs. The target of certification is a set of non-functional mechanisms to logically group dimensions. The execution of a model takes a certification model and processes, executing the model and award certification in case. Each service is filtered according to requirements.

A certificate is valid if there is a trust of chain (binding certificates on each other), using a continuous certification mechanism (making certification be valid according to a system property and across all rcmits changes). In traditional systems, the change in code makes a new check for code every time, here it's consistent with the system state. Using ML models we use dimension data, a dimension process and dimension model, bringing a multi-dimensional scheme.

A system can be poisoned, with attacks carried out at training time, so they need to be sanitized and controlled in models and partitions, splitting data according to strategies and predicting the state via voting.

2.12 Searchable Symmetric Encryption

(Host: Kaitai Liang)

We will see something about Searchable Symmetric Encryption (SSE) and we will describe types of attacks, which will mainly regard cloud-based data (profile info/medical history/genome data) keeping it confidential. To keep control over the data, we need to index it (via sequence numbers/via keys-buckets as hash data).

Here's a more detailed explanation of how SSE works, broken down into its key components:

1. **Data Encryption**: SSE starts by encrypting the data before it's stored. This encryption ensures that the data is protected from unauthorized access. ("How to Encrypt and Decrypt Messages in Laravel - Twilio") However, traditional encryption methods like AES (Advanced Encryption Standard) make it challenging to search within the encrypted data directly.
2. **Index Generation**: To enable efficient searching, SSE generates an index or searchable structure for the encrypted data. This index contains information that allows for fast and accurate searches without revealing the plaintext content. This index generation process often involves using cryptographic hashes.
3. **Querying**: When you want to search for a specific piece of information, you construct a query over the encrypted data. The query is transformed into a form that can be applied to the encrypted data and the index, without needing to decrypt the entire dataset.
4. **Search Process**: The search process operates on both the index and the encrypted data, finding matches according to the query's criteria. Importantly, this process does not reveal the actual content of the data but only the results that match the query.
5. **Relevance and Ranking**: The search results are often ranked by relevance. This ranking is typically done based on the similarity of the encrypted data to the query terms, and it helps in presenting the most relevant results to the user.
6. **Decryption and Retrieval**: Once the search results are obtained, the relevant encrypted data can be retrieved. This data can then be decrypted using the appropriate cryptographic keys to reveal the plaintext content, but only for the specific items that matched the search criteria

In a simple context, we simply index content in a secure and in a $O(1)$ way.

This data must be protected and encrypted in a straightforward way, achieved via using an algorithm like AES and each record will be encrypted with a secure index, which will be used to query data, achieving this way redundancy and safe protection.

Confidentiality here is achieved through hosting encrypted data with secure algorithms, managing securely keys and eavesdroppers try to leverage at best data leakage, targeting data in transfer leveraging vulnerabilities in communication protocols. One to achieve protection tries to make the adversary believe I am who I'm not, faking my identity but creating a convincing proof.

One way to achieve privacy is trying to protect keywords and working on encrypted files, relating to keywords and protecting access to them according to their size.

In the context of databases and data analysis, various types of patterns, including keyword patterns, search patterns, volume patterns, and access patterns, play crucial roles in understanding and managing data. Here's an explanation of each of these patterns:

1. **Keyword Pattern**:
    - **Definition**: A keyword pattern, also known as a keyword search pattern, is a specific set of words, phrases, or terms used to search for information within a dataset or a database. It represents the query terms provided by a user or a search algorithm to retrieve relevant data.
    - **Use Case**: Keyword patterns are commonly used in search engines, content retrieval systems, and information retrieval applications to help users find specific information based on the keywords they enter.
2. **Search Pattern**:
    - **Definition**: A search pattern refers to the structured and repeated way in which searches are conducted within a dataset or database. It encompasses the techniques, strategies, and criteria used to search for information or patterns in data.
    - **Use Case**: Search patterns are prevalent in data analysis and research, where analysts or researchers may follow a systematic approach to querying and filtering data to uncover insights, trends, or anomalies.
3. **Volume Pattern**:
    - **Definition**: A volume pattern relates to the quantity and size of data records within a dataset or database. It helps identify data characteristics such as data growth over time, data distribution, and data size trends.
    - **Use Case**: Volume patterns are often essential in capacity planning, resource allocation, and performance optimization. For example, monitoring the volume pattern of web traffic data can help allocate server resources effectively.
4. **Access Pattern**:
    - **Definition**: An access pattern, also known as data access pattern, characterizes how data is accessed or retrieved from a database or storage system. It includes information about the frequency, order, and types of data access operations.
    - **Use Case**: Access patterns are vital in database optimization and performance tuning. By understanding how data is accessed, administrators can design efficient data structures, employ caching strategies, and optimize queries for better database performance.

We define two types of privacy here:

Forward Privacy:

- **Definition**: Forward privacy, also known as future privacy, refers to the protection of a user's data and privacy for data collected in the future. It ensures that data collected today or in the present remains private and secure as the user continues to interact with a service or application.
- **Use Case**: Forward privacy is essential in scenarios where a user continues to use an online service or application over an extended period. It addresses concerns about how the data collected in the future will be handled, ensuring that it remains private and compliant with evolving privacy regulations.

Backward Privacy:

- **Definition:** Backward privacy, also known as past privacy, focuses on protecting historical data and information collected from users in the past. It ensures that data collected previously remains secure and compliant with privacy regulations.
- **Use Case:** Backward privacy is crucial when users have provided data to a service or application in the past and expect that their historical data will not be mishandled, exposed, or violated in terms of privacy.

Point is, we want to understand how to get context data and how this data can be used to gather new ways of attacking. This way, we're able to deterministically match data based on keyword encryption.

ASE can be vulnerable to various attacks. Here are some common attacks on asymmetric searchable encryption:

1. **Keyword Guessing Attack**:
    - **Description**: In this attack, an adversary attempts to guess the keywords or search terms used in a query. By making educated guesses based on known information or patterns, they can potentially gain access to search results or even the encrypted data.
    - **Solution**: We use multi-server logic to avoid guessing keywords on single devices
2. **Frequency Attack**:
    - **Description**: This attack involves analyzing the frequency of keywords within the encrypted database. By observing the frequency of keywords in the search results, an attacker may deduce information about the data's content, such as popular keywords or data distribution.
3. **Data Leakage through Search Patterns**:
    - **Description**: Adversaries might infer patterns of data access by monitoring search queries over time. By analyzing the frequency and timing of specific queries, they may deduce information about the data or user behavior.

Specifically, we can outlie some key points on those types of attacks:

1. **Leakage Abuse Attacks:**

- These attacks focus on exploiting information leakage, which may occur when using searchable encryption. Information leakage refers to unintentional or unauthorized disclosure of sensitive data or patterns. Attackers look for ways to abuse this leakage to gain insights into encrypted data.

2. **Knowledge of Context (Pre-Knowledge):**
   - Attackers may have prior knowledge or context about the data being encrypted. This pre-knowledge can include information about the data's content, structure, or expected patterns. This knowledge is used as a starting point to launch attacks.
3. **Observing Queries:**
   - Attackers monitor the queries being made to the searchable encryption system. This observation includes tracking the keywords, search patterns, and the timing of queries. By doing so, they may try to deduce information about the data.
4. **Matching Reflection for Data Inside Matrices:**
   - Matching keywords happens insides reflection matrices, which basically try to get frequence of single characters and leverage vulnerabilities on them
5. **Where they can be used**
   - The attacks described aim to use the knowledge gained from observing queries and data patterns to exploit vulnerabilities in the searchable encryption system. This could involve extracting sensitive information, deducing patterns, or revealing more about the encrypted data than intended.

Moving on with other attacks:

4. **Multi-Query Attack**:
   - **Description**: In a multi-query attack, an attacker submits multiple queries and carefully observes the search results. By comparing the results of these queries, they may be able to gain insights into the content or relationships between different pieces of data.
5. **Ciphertext-Only Attack**:
   - **Description**: In a ciphertext-only attack, the attacker has access to the encrypted data and the search index but doesn't know the encryption keys. They aim to exploit any weaknesses in the encryption scheme to derive information about the plaintext data.
6. **Chosen-Keyword Attack**:
   - **Description**: In this attack, an adversary can submit queries for specific keywords they are interested in. By observing the search results for these keywords, they may gather information about the data containing those keywords.

7. **Adaptive Chosen-Keyword Attack**:
    - **Description**: In an adaptive chosen-keyword attack, an attacker can iteratively select keywords for search queries based on previous results. This allows them to refine their search and gather more information with each query.
8. **Intersection Attack**:
    - **Description**: In an intersection attack, the attacker tries to find data records that match multiple keywords simultaneously. By iteratively submitting queries for different keywords and identifying common results, they can potentially deduce information about the data.
9. **Index Attack**:
    - **Description**: This attack focuses on exploiting vulnerabilities in the search index. If the index is not properly protected, attackers may gain insights into the data structure or potentially discover details about the encrypted data.
10. **Time-Based Attack**:
    - **Description**: In a time-based attack, an adversary observes the timing and frequency of queries. By analyzing patterns in when and how frequently queries are made, they may infer information about the data and user behavior.

We can attempt to discover injection vulnerabilities by identifying ordered relationships among elements, arrays, and search keys when interacting with external systems. This approach involves injecting data in a way that exploits patterns in the order of retrieved information.

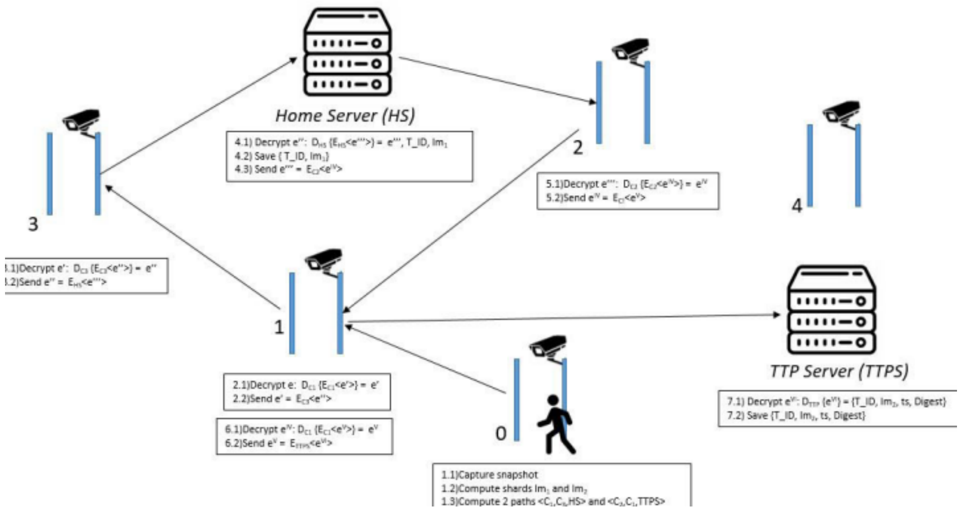
This can also be achieved via padding, sending a lot of dummy/meaningless data to confound other people and data. Data can be easily attacked finding links between samples and copies of that.

2.13 AI-Driven Cybersecurity: Poisoning and Inference Attacks

(Hosts: Antonino Nocera – Marco Arazzi)

We want to create an analysis broadening in many kinds of media, creating structural analysis and graph mining in order to analyze different types of social media. Imagine a network able to inspect content based on engagement over posts and decide/analyze the influences over single posts, then multiplying specific content and understanding the dynamics of share.

The analysis can identify content based also on physical devices connected to the Internet, storing and sharing new data and information:


Home Server (HS)
4.1) Decrypt e'': D_{HS} {E_{HS}<e'''>} = e''', T_ID, Im_1
4.2) Save { T_ID, Im_1}
4.3) Send e'''' = E_{C2}<e^{IV}>
2
5.1)Decrypt e'''': D_{C2} {E_{C2}<e^{IV}>} = e^{IV}
5.2)Send e^{V} = E_{C1}<e^{V}>
3
.1)Decrypt e': D_{C3} {E_{C3}<e''>} = e''
.2)Send e'' = E_{HS}<e'''>
4
1
2.1)Decrypt e: D_{C1} {E_{C1}<e'>} = e'
2.2)Send e' = E_{C3}<e''>
6.1)Decrypt e^{IV}: D_{C1} {E_{C1}<e^{V}>} = e^{V}
6.2)Send e^{V} = E_{TTPS}<e^{VI}>
0
1.1)Capture snapshot
1.2)Compute shards Im_1 and Im_2
1.3)Compute 2 paths <C_1,C_3,HS> and <C_2,C_1,TTPS>
TTP Server (TTPS)
7.1) Decrypt e^{VI}: D_{TTP} {e^{VI}} = {T_ID, Im_2, ts, Digest}
7.2) Save {T_ID, Im_2, ts, Digest}

In AI-Driven Cybersec, there is a huge community, spread across different domains. AI can be used to leverage new design over new menaces, adapting versatile and powerful solutions.

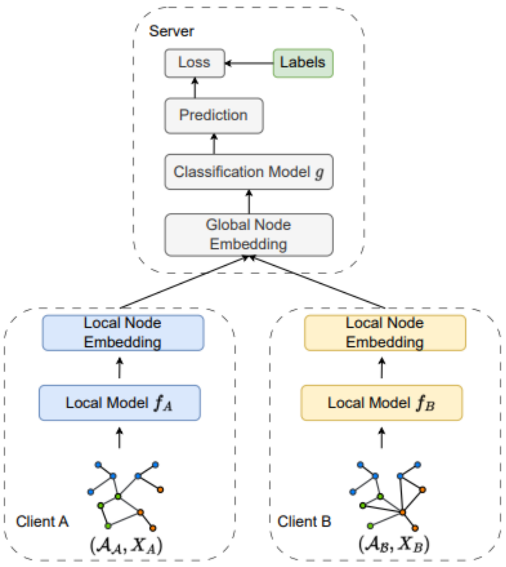
For example, there is behavioral fingerprinting, which works over federated learning, this way communicating with others, with group of objects which try to solve a specific problem, identifying what single nodes are doing and then clustering them via services' calls.

We use *federated machine learning* in order to communication-computation framework train models over workers and aggregators, updating local models with local data and then selecting parties to update global model accordingly. The data partitions over which the ML model work is with decentralized devices which train over data scaled across multiple dimensions. On this multiple attacks can be built, to infer misclassification over single/multiple nodes, according to the node conditions/activity, while also breaking the honesty of particular models.

The *poisoning attacks* class now is targeting models over their availability, breaking their convergence (adversarial attack), or to manipulate the to achieve desired predictions for specific targeted inputs creating a backdoor essentially and breaking integrity of models (backdoor attacks).

A common attack example is label-flipping attacks, modifying labels of training samples maybe with simple classification problems, like AI filters or training models over false data, this way creating images over different people and tricking the prediction to make other things fooling models. The model will predict models over users spreading false data (adversarial), driving decision towards wrong recommendations (backdoor).

This is how it works, assuming to have an idea o classification and generate tests over examples and using inference to get knowledge of context:



3 Second Part of the Course: Students Presentations

This part consists of "student presentations", on which students are called to select topics of interest from a collective list such as this one here.

Before starting this part, people have to form a group of one/two/three people filling a Google form if interested to continue on the course, select a topic and then make a presentation. This is done reading the main paper (or at least a secondary one also, but not mandatory) for the topic selected for the presentation, for which they will be evaluated upon, according to the criteria given in the end of this section.

In this phase, there are also the so-called "provoking questions", which we are asked to give at least two questions per lesson (not per topic) over others' presentations; this requires to read the papers first and create some questions, which can be even not so hard, but they should be interesting. This one is another part of the evaluation.

In reality, on this one, the assistants only evaluate the questions present on Moodle; in class, for each presentation, there will be a lot of people trying to make a questions. I suggest seating in the first rows and fight a bit to ask it. First, say the surname (or name and surname) clearly enough to be heard by the assistants; they will note your name and you should be good to go. The questions done in class, in reality, are always a plus, they seem to be not that strict on this aspect.

So to summarize:

- select a topic and form a group
- fill the Google Form to select this topic (this is "first-come-first-served", so keep that in mind)
- make the presentation and be prepared to receive a lot of questions
- participate to lectures and do provoking questions too (but the indications I gave above were asked by many people and it works like this)

I will give some advice for the presentation (some obvious and other, for some reason, not so obvious):

- the layout of the presentation is important; come prepared and have at least a coherent layout visually, colorfully and don't go over-the-top
- you will have, if there are four groups for presenting a day, 20/25 minutes otherwise for three I'd say from 25 to 30 minutes (each comprehending questions; I was large enough, but consider always less than this)
- remember you are doing a public presentation, and this means (apart from anxiety we all have), but please:
  - you are not supposed to read the slides
  - you are supposed to have a voice which can be heard from the last rows (speak louder please)
  - you are supposed to look people in their eyes when talking
  - you are not supposed to read a script and call it a day
  - given you are presenting a paper, be technical enough, but please, make people understand. Be not too slow with 40 slides (yes, there were people like this), but also don't run so fast I can't even listen to you (some people were talking in 2x speed)
  - you can use whatever you want, from Google Slides up to Typst/Beamer/PowerPoint

The presentation day, one assistant asks people to give him the presentations to load on to the computer; I suggest preparing a USB stick with the PDF format of slides to give and then present to the audience.

4 Third Part of the Course – SPRITZ Projects Proposal

The slides of this meeting are confidential and will not be shared. In case, to collaborate with the SPRITZ grades, sending CVs and exams grades/expected time commitment on projects, etc.

Just write to both Conti and all Teaching Assistants as always (put Conti in cc and use "Reply all")

Each project will have three icons:

- Suitable for MSc Thesis (first symbol)
- Suitable for BSc Thesis (second symbol)
- Suitable for Course/Extra Projects (third symbol)

There are many chances of collaboration, starting from many people and different themes:

1) Federico Turrin (turrin@math.unipd.it)

- Cyber-Physical System Security
    - Anomaly Detection and Intrusion Detection Systems
    - Industrial Encrypted Traffic Analysis

2) Alessandro Brighente (alessandro.brighente@unipd.it)

- Jamming IOT Devices
    - End-to-end attack to 5G Networks
    - Identification of Polyglot Code

- Creating Rules for Secure Testing
    - Tracking users in Wi-Fi
    - Is Easy-Connect Really Secure?

- Drones and Autonomous Driving
    - Drones Startup Safety Check
    - Driving Behaviour Authentication

- Communication Analysis
    - Detection of Location Spoofing Attacks in Air-Traffic Communication: Leveraging Physical Layer and Space Tessellation

3) Tommaso Bianchi (tommaso.bianchi@phd.unipd.it)

- Cyber-Physical Systems security
    - Find exposed UPS Systems on the wild
- Cryptography
    - Break Quantum KD Signal Algorithm
- Vehicles Security
    - ECU Fingerprinting in a Black Box Reverse Engineer manner
    - Remote Keyless Attack and Defenses

4) Denis Donadel (donadel@math.unipd.it)

- Vehicles Security
    - Hyperloop cybersecurity
    - On the feasibility on DoS on vehicles

- Applied ML
    - PoC of an Intrusion Response System (IRS)
    - An LLM-based assistant for sysadmins

- MISC
    - Improving application fuzzing via QR Codes
    - Can we attack channel hopping mitigation against DoS?

- Arxiv-leaks
    - Retrieve redacted data and insight from Arxiv

5) Gabriele Orazi (gabriele.orazi@phd.unipd.it)

- EPZ (Endpoint Privacy Zones) Privacy
    - StravaGANte
- Acoustic communications to evade network security policies
- Document Un-redaction
    - Context Driven Un-redaction of documents

6) Jiaxin Li (jiaxin.li@studenti.unipd.it)

- Covert Channels
    - Solution for Covert Channel for Federated Learning Systems
- Membership Inference Attack (MIA)
    - Modeling the distribution of distinguishable metric for non-member data

7) Francesco Marchiori (francesco.marchiori@math.unipd.it)

- Automotive Security
    - Quantum-Safe Cryptography in Connected Vehicles
    - Authenticating ECUs through CAN bus physical signals
    - Resilience Testing for Connected Vehicles
    - Cyber Threat Intelligence on Automotive

- Cyber Threat Intelligence and NLP (Natural Language Processing)
    - Advanced Persistent Threat (APT) classification

- Adversarial Attacks
    - Attacks on License Plate Recognition Systems

8) Luca Pajola

- Machine-Learning for Security
    - Design on Novel Adversarial Attacks
    - Web Analyses & Attacks

- Creating profiles from audio clips
- NFT Playground

9) Alessandro Lotto (alessandro.lotto@phd.unipd.it)

- 5G and 6G Networks
    - Location Tracking Defenses
    - Authentication in Roaming
- Authentication
    - Physical Layer Device Authentication
    - Behavioral touch-based authentication

10) Saverio Cavasin (saverio.cavasin@phd.unipd.it)

- Deep Fake
    - 1 Shot attack
    - Camaleonet
    - Generative blackmail attack
- Biometrics
    - IDAP (Identity Dialer Acquisition Protocol)
    - Tattoo Identification

11) Giulio Rigoni

- Drones/UAVs
    - FPV Data persistency/authenticity
    - Multilateration vs GPS spoofing

12) Eleonora Losiouk

- Android Security
    - Solving the Android Semantic gap
    - Measurements of Interaction among Android apps

13) Rahul Saha

- Distributed Ledger Technology
- Security analysis of Signal Protocol
- Blockchain Encrypted Protocol

14) Stefanos Koffas

- Adversarial ML
    - Explore the behavior of ChatGPT like a terminal
    - Implement dynamic backdoor attacks in computer vision

15) Harsha Vasudev

- Vehicular Networks Security
    - Secure Protocol for IOV Communication Components
    - PQC-based Protocols for IOVs/VANETs
- CAN Bus Security
    - PQC and PUF Integrated Methods for CAN

16) Giacomo Quadrio (giacomo.quadrio@unipd.it)

- Anomaly detection on IoT networks
  - Creation of a network traffic dataset based on real IOT hardware
  - Survey on most frequent IoT attacks

17) Alessandro Galeazzi (alessandro.galeazzi@unipd.it)

- Cybersecurity and social networks
  - Opinion inference on social media
  - Coordinated behavior in online debates

18) Shuo Wang (shuo.wang@phd.unipd.it)

- Jamming Tool Development

19) Enrico Bassetti (bassetti@di.uniroma1.it)

- IP Security
  - Security of custom IP stacks
  - Multi-path TCP and MP-QUIC
- More IP an E2EE
  - IP retransmissions and accounting
  - E2EE contact verification ceremonies

20) Vinod Puthuvath

- Malware detection
  - Malware visualization
- Threat Intelligence Sharing
  - Large Language Models (LLMs) to revolutionize the way Cyber Threat Intelligence is gathered, analyzed, shared

If the project is not part of these ideas, it depends on the nature of the idea itself and we con contact other collaborators and stuff (given this is the future of the research, the slides will not be shared).

To create the essay on:

- send an email to Conti and assistants to CC
- tell the idea and the project/topic
- they give us a feedback on this (approving/rejecting)
- create an essay of a few pages

The deadline for the project is at least 24 hours before the exam date. It all depends on us.