POLITECNICO DI TORINO

COMMUNICATIONS AND COMPUTER NETWORK ENGINEERING

Master Degree Thesis

Design and Development of the Back-End Software Architecture for a Hybrid Cyber Range

Author: Hussein Farhat

Supervisor: Paolo Ernesto Prinetto

July, 2021

Abstract

In the last decades, reports and news about cyber attacks have showed up a turnover. Such exploits are making use of latest technology and exploiting new techniques and methodologies. Security specialists and researchers are making unique efforts to develop platforms that are capable of emulating threatening situations and allowing to learn how to counter them properly. The development of *cyber ranges*, simulated environments where vulnerable IT infrastructures are reproduced, mainly allow to improve the skills for people operating in the first line of defense, and of course to educate students in the cybersecurity field.

To date, software virtualization techniques allow you to create simulated environments that are very faithful to the original ones, with deeply customizable network topologies and connected devices of all kinds. Virtualization allows considerable savings on the physical equipment required for the implementation of these environments, as well as on the expertise required, given the growing variety of existing automatic design tools. Despite all these advantages, it should not be underestimated that a large part of devices and infrastructures are still irreproducible in the software domain, and that many companies still include physical routers, switches and firewalls in their IT and OT infrastructures, especially when needed to communicate with industrial control devices. Furthermore, the increasingly widespread IoT networks in cities, homes and even wearable objects are hard to be reproduced in totally virtual environments. In fact, training totally abstracted from physical devices is significantly limiting, because a very large part of threats, such as those related to side-channel vulnerabilities or low-level communication protocols, is ruled out.

This thesis present a contribution on the design of a *hybrid cyber range*, entitled PAIDEUSIS, that aims to combine virtualization techniques with real physical devices. In addition to providing an overview of the general features of cyber ranges and the particularities of PAIDEUSIS, the thesis presents the work carried out about the development of the back-end code infrastructure, including the database design and different kind of security features that are fundamental to protect the data over the network. The work is concluded by stating the upcoming features and capabilities to be considered for the future work of the project.

Contents

Chapter 1

Introduction

Nowadays, cybersecurity is certainly one of the biggest trending topics. Industries across the world are steadily increasing the integration of technology into their daily operation, and the evolution of the technology introduce many features to make the human life much smarter and simpler by making everything connected to the Internet. New digital technologies imply a bigger number of possible vulnerabilities, and a wider cyber-attack surface, in a way which was not possible to expect some years ago. The most common cybersecurity threats, related to mainframe devices as personal computers and company servers, are gradually giving way to a wide range of attacks targeting devices such as infrastructure equipment and smart objects, mostly exploiting domains such as low-level software and hardware, which have notoriously been less addressed in literature and companies. New evolved technologies can range among a large set of domains and fields, and the following of this document is intended to present some of them for showing how they are vulnerable to cyber threats to understand the importance of having protection.

Today, the Internet of Things (IoT) is one of the most trending technologies worldwide. This technology is now developed more than before, thanks to several kind of domain that are evolving and making progress such as wireless sensor network, broadband network, radio-frequency, with other communication protocols make a large contribution in evolving IoT [1]. Therefore, these networks of devices become more vulnerable and introduce a lot of challenges for making it secure. An overview for the topology in Figure 1.1 illustrates how IoT devices could be connected to the network through several network layers, pointing out the wide potential attack surface for this kind of devices [2, 3, 4, 5]. The most famous vulnerabilities that could be exploited and used by attackers are categorized in Figure 1.2, showing the importance of several security aspects to be taken into account in order to provide the desired protection [6]. Attackers can exploit vulnerabilities from different access technologies that are used by the IoT systems. For example, it is possible to exploit a vulnerability on the network system, to perform malicious activities such as traffic analysis by intercepting packets on the network [7], or similarly, obtaining informations by intercepting the communication between two nodes. This technique is known as *Man in the Middle* [8]. Other kinds of attacks could be also performed by targeting software vulnerabilities to access the system and inject some kind of malware and to maliciously control or damage the system.

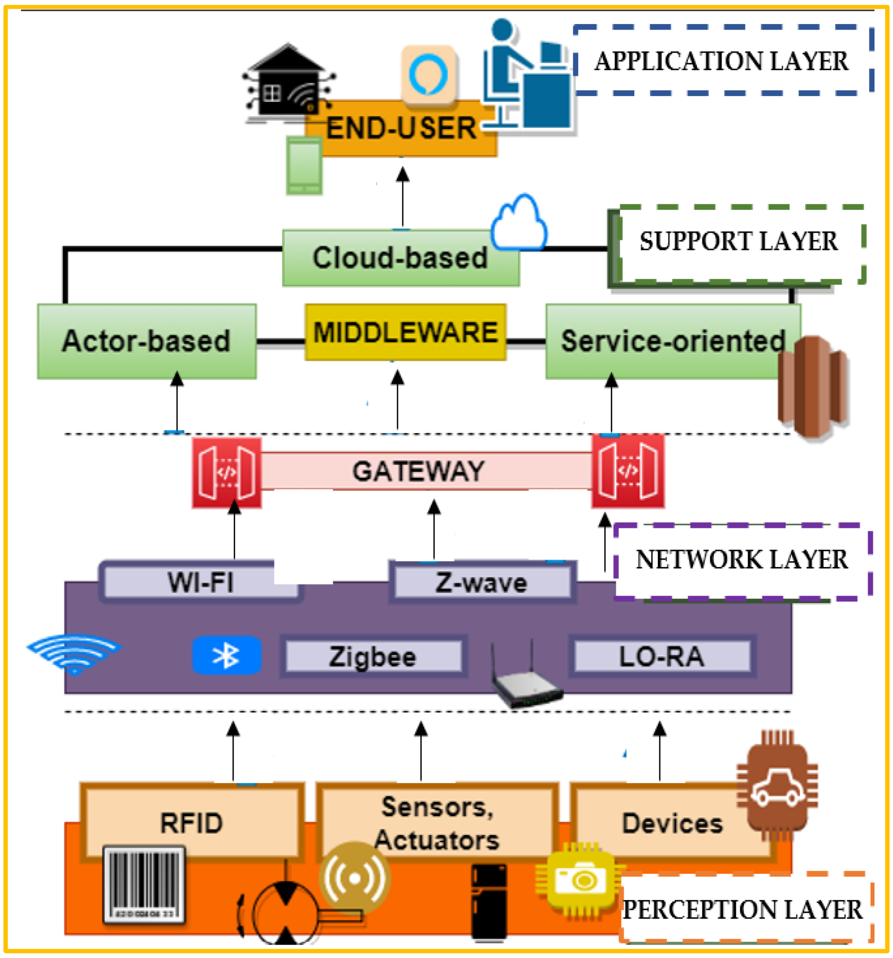


Figure 1.1: IoT topology overview [1].

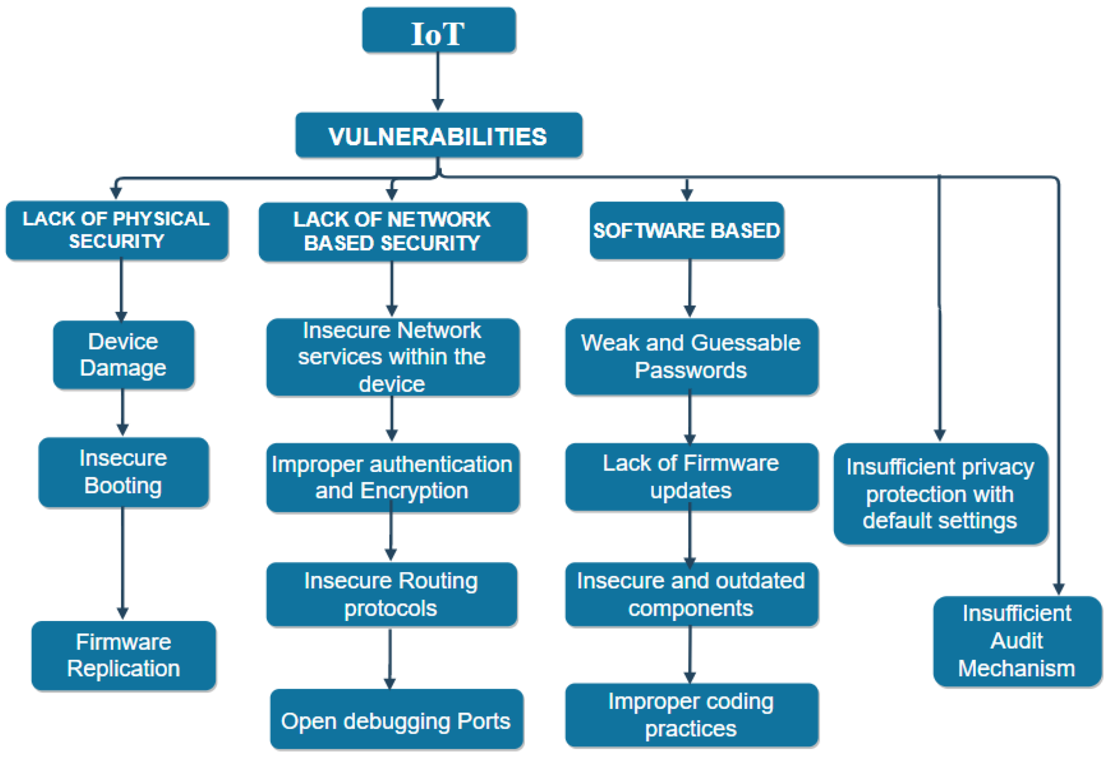

IoT
VULNERABILITIES
LACK OF PHYSICAL SECURITY
LACK OF NETWORK BASED SECURITY
SOFTWARE BASED
Device Damage
Insecure Booting
Firmware Replication
Insecure Network services within the device
Improper authentication and Encryption
Insecure Routing protocols
Open debugging Ports
Weak and Guessable Passwords
Lack of Firmware updates
Insecure and outdated components
Improper coding practices
Insufficient privacy protection with default settings
Insufficient Audit Mechanism

Figure 1.2: Most common IoT vulnerabilities [6].

Another field of technology that is widely used and creates a lot of challenges is the one related to *online banking services*. It is one of the most important domains, that needs an ultimate online protection with a lot of attention, as the data to be transferred are the users' funds. The most majority of people are using online banking services, such as paying bills online, monitoring transactions, transferring money worldwide, managing accounts, and more [9]. This is why the most of fraudulent attacks nowadays revolve around those services, with the purpose of compromising the user's bank accounts and own any available fund [10]. For example, Figure 1.3 shows statistics study for online banking losses in United Kingdom (UK) over several years ranging from 2010 to 2019, where the capital London is considered the fifth largest national economy worldwide with some of largest banks in the world like HSBC and Barclays [11]. Despite the difficulities to estimate the global number of cyber-attacks on the financial sector, the ThreatMetrix 2016 Report [12] state that the financial sector is the highest number of organized cyber-attacks. *McAfee*, an anti-malware company, reports the global estimated cost of cybercrime arround 445 billion USD [13].

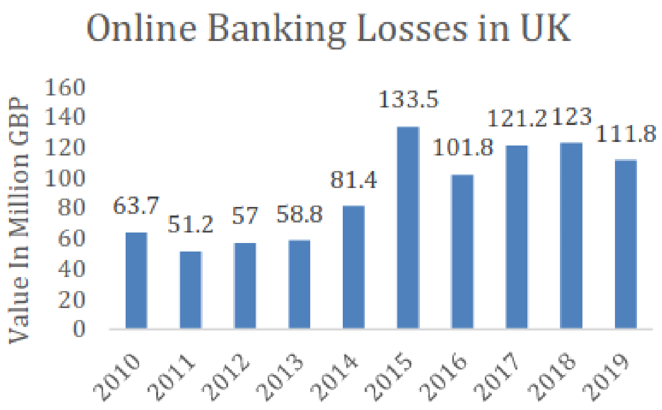


Figure 1.3: Losses due to banking frauds in UK [11].

Additionally, cyber attacks are currently targeting a new sector which is rising on the horizon to become in the near future the dominated technology in the transport sector: we are talking about Connected and Autonomous Vehicles (CAVs). Not only for public requirement, but also for several civilian and military applications across different environments: in air, ground, and sea, by implementing and integrating many new technologies such as Machine Learning, IoT, Mobile Broadband 5G and more [14]. An important category of vulnerabilities could be exploited in order to take down the system that control those vehicles, as illustrated in Figure 1.4. CAVs are connected to the Internet using different wireless access technologies, which make them vulnerable for cyber threats [14, 15].

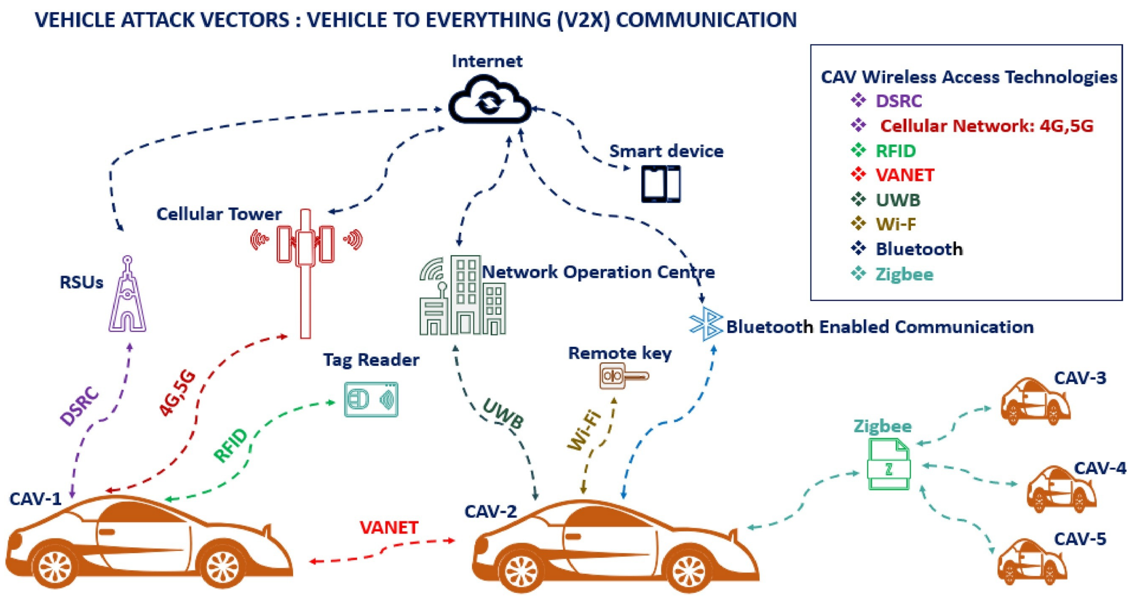


Figure 1.4: Smart vehicle attack vectors [15].

In such an environment, it is possible to recognize different access technology used, like the Ultrawideband (UWB) signal, which is another form of wireless networking that allow the transfer of large amounts of data [16], and Radio Frequency Identification (RFID) to identify objects through radio signals [17, 18, 19, 20, 16]. While networking protocols can simplify the way of using these access technologies, protocols such as DSRS/VANET (Dedicate Short-Range Communication) that guarantee to have a low latency with a secure high-speed networking [21, 22]. The Zigbee protocol is used for low-speed communication within a short-range networking distance [16]. These access technology could be exploited using fake signals: a common attack is known as GPS Spoofing, where attacker send fake signal to the GPS system in order to implement a malware and gain access to the whole system [23].

All these technologies cannot operate properly without using some kind of *cloud infrastructure* that provides a variety of beneficial services. But unfortunately, there exist many vulnerabilities that could be exploited by malicious actors, since the data is transported on the network [24, 25]. Accordingly, it can be possible to state some of the most important vulnerabilities among many of them, and under different categories. For example, some of them could be categorized within *security policies* that define the standards rules that should be taking into account in order to prevent attacks, while other risks could be assigned to the providers of the cloud services since the provider can control all the tasks that operate behind each service such as storing, identification, authentication, authorization, and other issues related to access management [26, 27]. In fact, providers should guarantee the *availability* and *accessibility* of the data at any time, as well as the *integrity* [28]. One of the most vulnerable areas is definitely the software application, since software could be handle on different platform, using different programming languages, each holding a different level of security, architectures, and of course many frameworks and packages as well [29]. Figure 1.5 shows the categorized vulnerabilities for a cloud services.

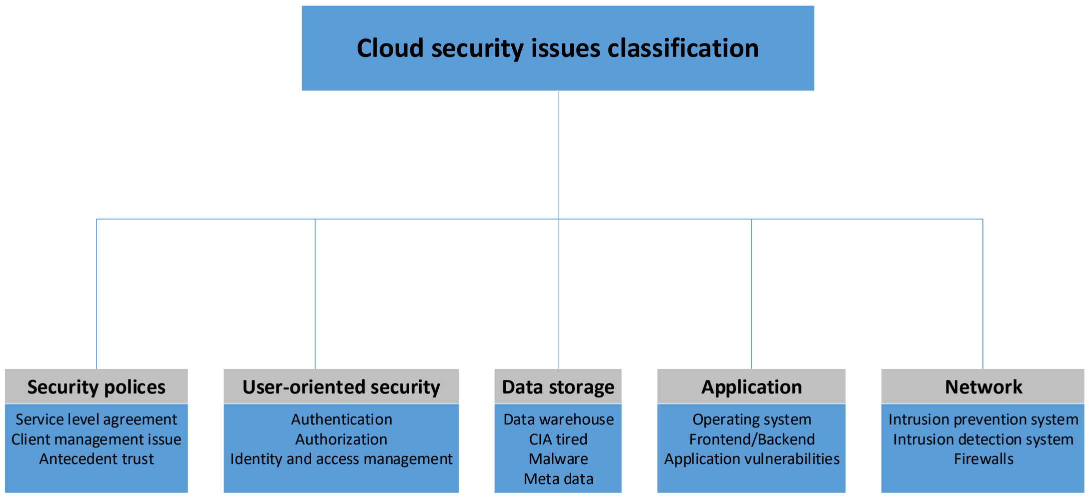


Figure 1.5: Cloud security issues.

Training is the headmost course of action to be considered in setting up a distributed resilience against cyber threats, not only for cybersecurity specialists in companies, but also for students and researchers that belong to this academic field. In this case, a serious challenge is to be faced about the methodology used for training and develop skills in realistic scenarios. Using real infrastructures is impractical, not only for the inevitable interruption of service, but also for flexibility and scalability reasons.

Cyber ranges try to fill this gap. As concerns related to cyber security grow rampantly, more and more cyber ranges are being developed by governments, commercial organizations, and academic research. By definition, *Cyber ranges* are interactive and simulated representations pertaining to an organization's local network, system tools, and applications [30]. *Cyber ranges* provide a safe and legal environment for users to learn, gain experience and develop cyber skills. They usually rely on hardware and software tools, or a combination of both, along with network services. They can also be thought as a virtual environment that deliver skills in cyber security and ensures that professionals work together as a team across several security domain, in order to improve their skills to defend against several attacks. *Cyber range* are not only restricted to an organization's local network, but also range from single standalone ranges in an organization to public ranges which could be accessed around the world such that they can be used by private as well as public organizations along with students, trainers and others.

Cyber ranges could be architected using real hardware devices, software simulation tools or a combination of both. However, the evolution of the technologies in several fields has helped a lot in developing a cyber range. Since the beginning, expert people in this domain start to develop *Cyber ranges* and test beds within secret programs and projects supported by government[1]. Later, when ranges started to come out to the public, professionals and researchers tried to put a lot of effort to integrate new technology and resources, and still doing that whenever a kind of technology evolve, in order to achieve the top realistic experience. But unfortunately, their effort wasn't enough and somehow tend to be limited, especially when they tried to implement some virtualization technologies. Later, the new virtualization technique appeared worldwide as *Software-Defined Network* (SDN). SDN allow to create virtualization of the hardware resources, that tend to be unlimited. Nevertheless, some component play a critical job in term of security or networking, then it will be highly recommended to not virtualize that kind of components such as hardware firewalls and main routers in the infrastructure in addition to IoT devices that make our entire life connected, can be hardly virtualized, and training on such platforms should be based on real hardware.

Based on that, making a combination of real hardware devices and software virtualization tools would be the best option for developers and engineers to build cyber ranges that will maintain such a degree of realism. This is what is called *Hybrid Cyber Ranges.*

[1]`https://www.govtech.com/blogs/lohrmann-on-cybersecurity/cyber-range-who-what-when-where-how-and-why.html`

Chapter 2

Background

2.1 Cyber Range Taxonomy

The word *range* can refer to an environment built for offensive target practice, e.g., like a shooting range for soldiers [31]. Therefore, the environment simulates the real scenario that happens in real life during an attack, such as attacking IoT system, GPS of smart vehicles (CAV), cloud servers, and many other.

A full cyber range taxonomy [32] has been illustrated in a schematic view and divided in two part to better visualize the complete taxonomy (Figures 2.1 and 2.2).

Normally, not all cyber ranges hold the same technology and capabilities, but there are some components that could be categorized as essential in any cyber range. Based on that, they share the same essential categories. Some of them are the following:

- **Scenario:** a scenario is the operating environment created by assigning a limited amount of resources during a practice session, and makes possible of one or more user to perform their activities during this session. They can be characterized within several aspect like the purpose of the scenario, environment (e.g., hardware, software, etc.), domain (IoT, network, etc.) and tools used for development (e.g., software, devices, servers, etc.).

- **Teaming:** a cyber range incorporates certain teams, each depicting their own influence. As many as 5 teams have been mainly discussed so far [32]. They are:
  - A **Red Team** plays the role of attacker, it is the team that perform malicious activities on the target(s).
  - A **Blue Team** plays the role of defender, by performing activities that prevent attackers to attain their goals.
  - A **Green Team** manages the cyber range infrastructure during and before its use.
  - A **White Team** is responsible for setting up scenarios.
  - A **Purple Team** incorporate both **Red** and **Blue** team within particular scenarios.

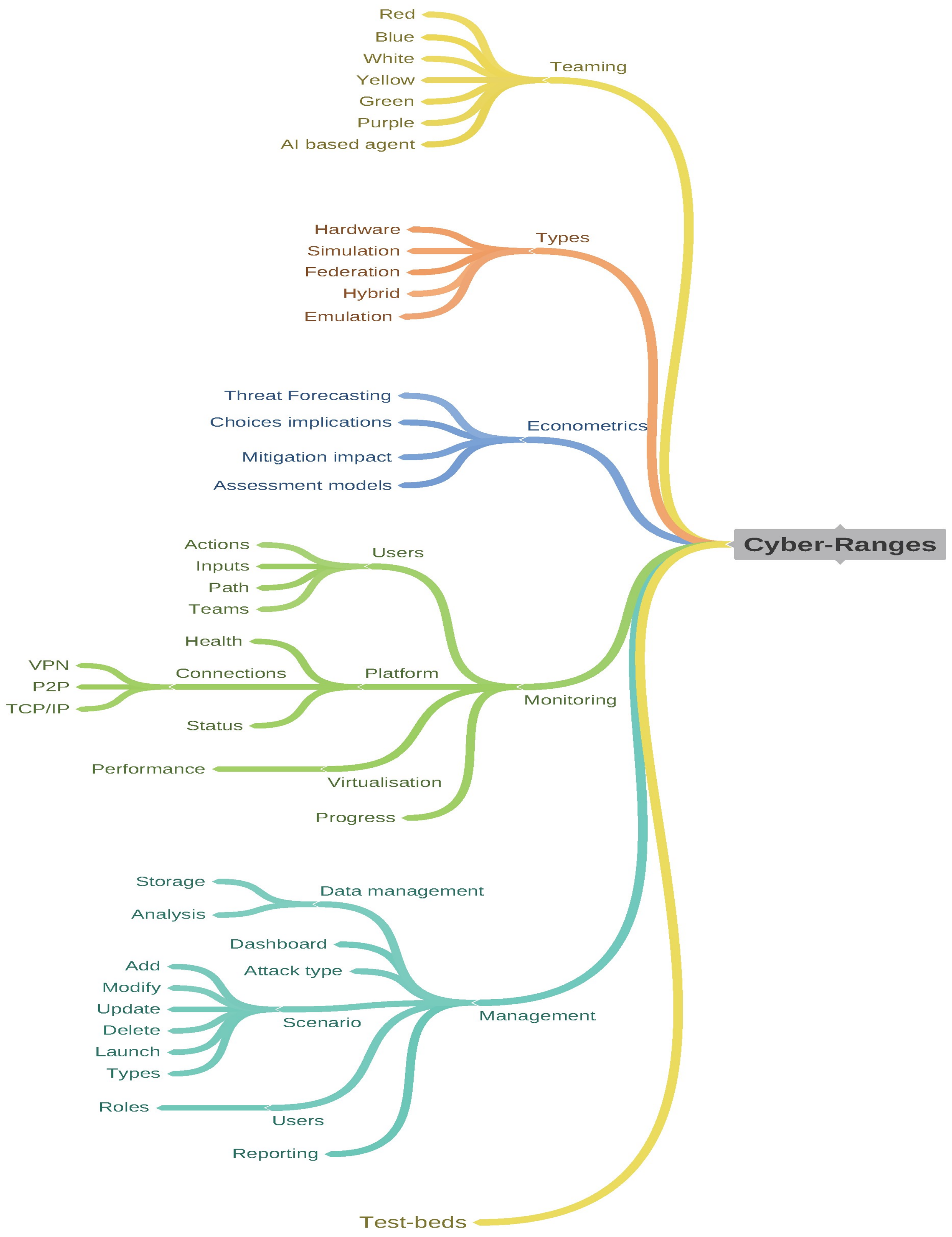


Figure 2.1: Cyber range taxonomy (a) [32].

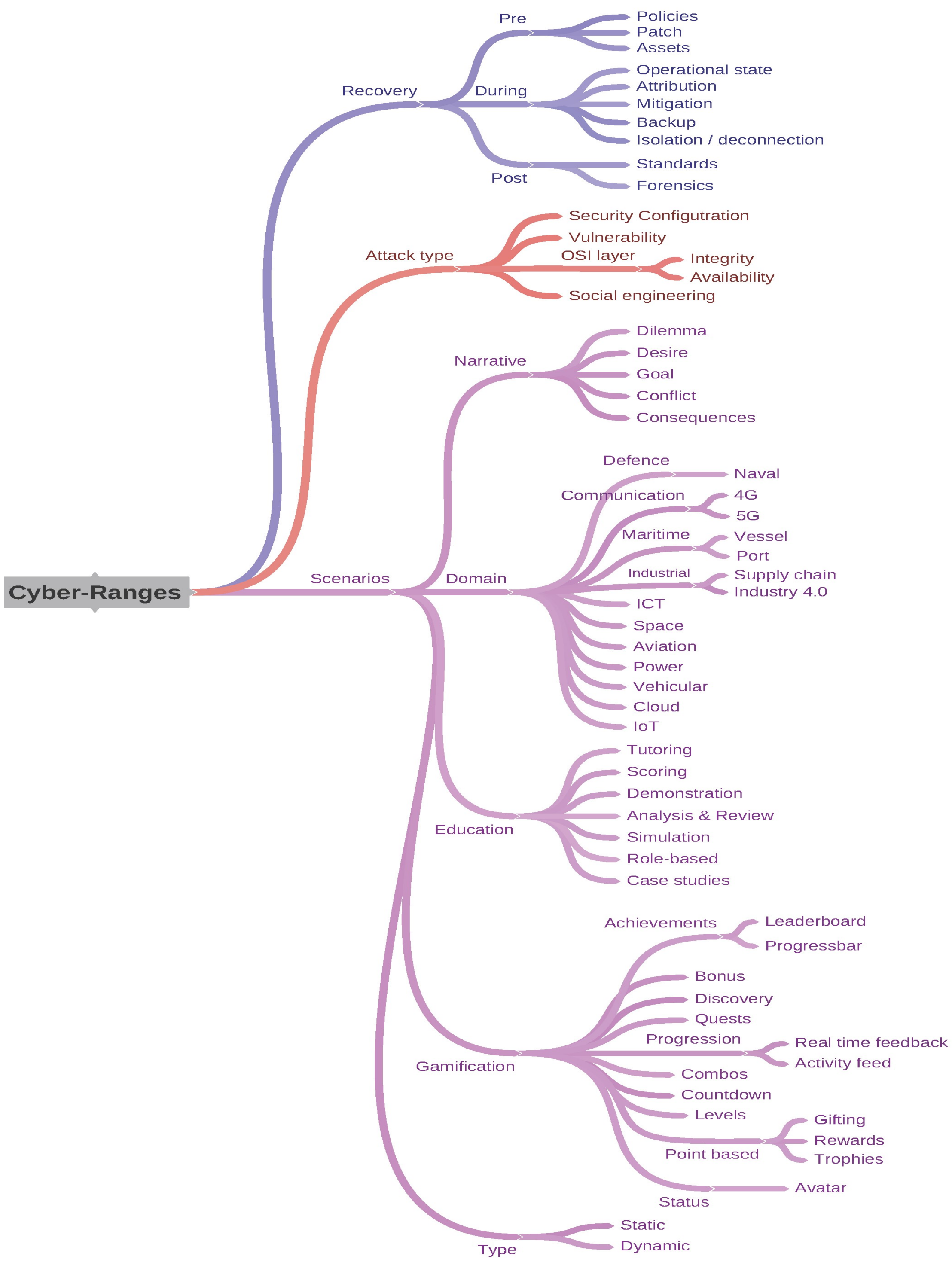


Figure 2.2: Cyber range taxonomy (b) [32].

- **Monitoring:** Monitor and trace users that are connected to the cyber range like actions, behaviors and teams performance. Also, the performance of the cyber range include several element to be monitored, such as services, network layers and storages. It is characterized by the implemented software and tools within the back-end infrastructure at which the data is collected and stored.

- **Econometrics:** Estimate the economic influence of actions taken during a scenario, to bring the possibility for users to develop an idea about the cost of losses when facing the same scenario in future.

- **Management:** the general interface to manage everything related to users like signing, register, logout, setup scenario, and storage management. It is called also *cyber range back-end.*

2.2 Architectures

2.2.1 Types

A cyber range could be developed adopting certain kinds of architectures, or using combination of several types in order to meet a specific requirement. Each type holds its own features and capabilities. The following are considered the main types of architectures until present [33]:

Simulation Ranges : These are ranges implemented in a totally virtual environment and do not need to have a strict mapping with real/physical hardware. The drawback of using this type is obtaining unrealistic network behavior in addition to a unpredictable performance in general;

Overlay/Emulation Ranges : Also known as **Physical Ranges**. They operate on the top of real hardware and provide environment identical to the real one by replicating the desired infrastructure. While the drawback of using this ranges is presented by having a high cost to setup alongside a lot of difficulties to scale, therefore the flexibility tend to be limited.

Hybrid Ranges : A combination based on virtual and physical ranges to provide the best properties of both, using the real hardware devices to provide the realistic experience, and select a set of virtualization technique to upgrade the level of flexibility and scalability.

2.2.2 Design

The design of the architecture will differ depending on several factor, such as the type of the desired cyber range and which cybersecurity sector the range is targeting, in addition to the technology used. But in general, based on the available tools and technology, the architecture of a cyber range should have two fundamental parts (Figure 2.3).

1. *Infrastructure/Back-end Technology*: This domain contains an essential part that manages and controls scenarios and resources. It is also called *orchestrator*. It can be powered through different kind of **Core Technology**.
2. *Front-end Technology*: This area is responsible on delivering the method in which the user will be able to interact with the cyber range;

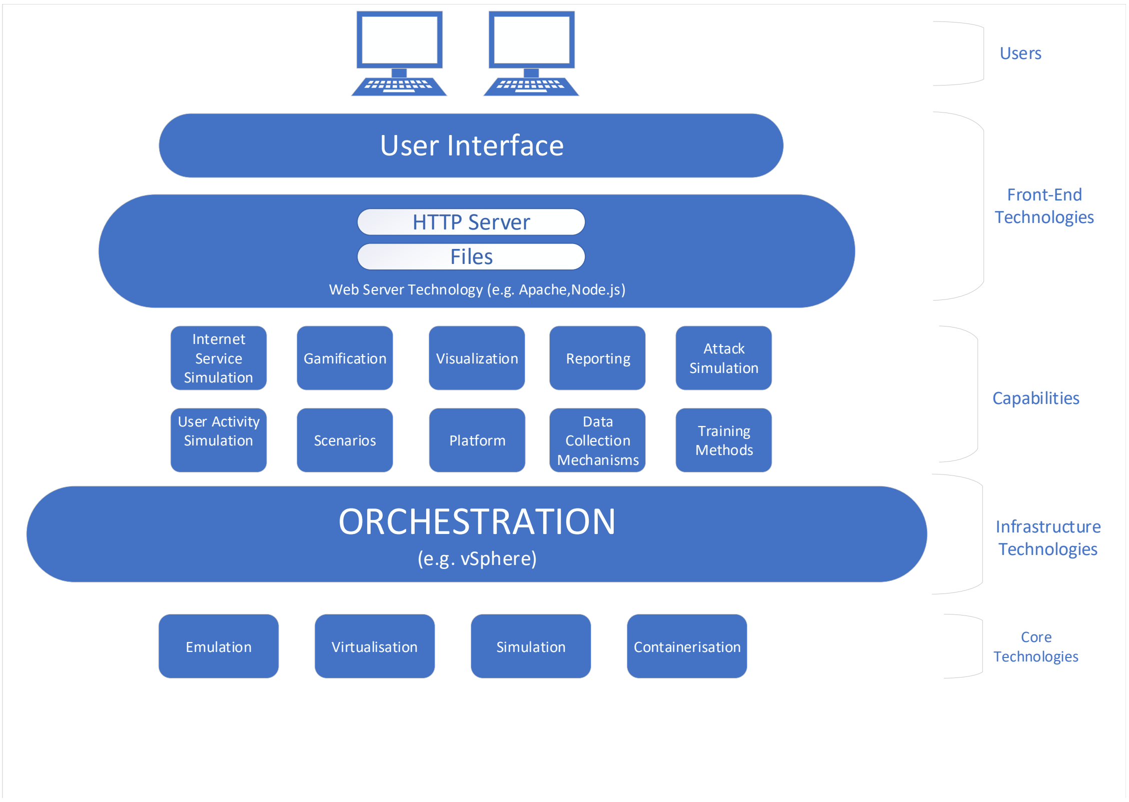


Figure 2.3: Architecture of a cyber range.

Infrastructure Core Technology

The core technology is the backbone of the infrastructure, since it define the environment used to develop and implement the infrastructure. The core technology of a *cyber range* could be one of the following: *Emulation*, *Virtualization*, *Simulation*, and *Containerization*. In particular,

- *Emulation* can be done by replicating exactly the operations of the target infrastructure;
- Instead, the *Simulation* is modelling the behavior of the target system;
- *Containerization* is a technology that takes the application with the related dependencies to build a form of container that follow a specific structure and make sure that application works in every environment [34]. It can be stored/containerized, transported and then executed (cfr. Docker[1]).
- In the other hand, the *Virtualization* beholds a VM (virtual machine) that takes place between the hardware and the host OS (operating system), managed by a hypervisor. In terms of isolation and obtained feature, difference between virtualization and containerization is tenuous (Figure 2.4).

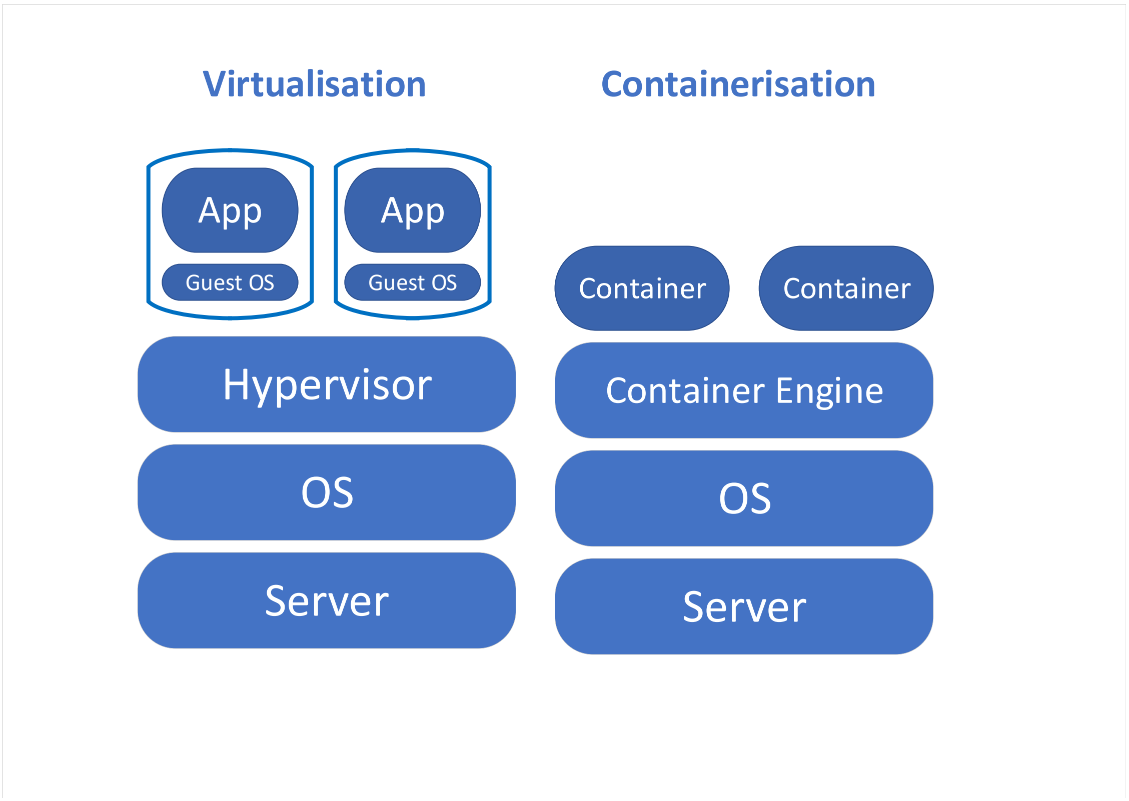


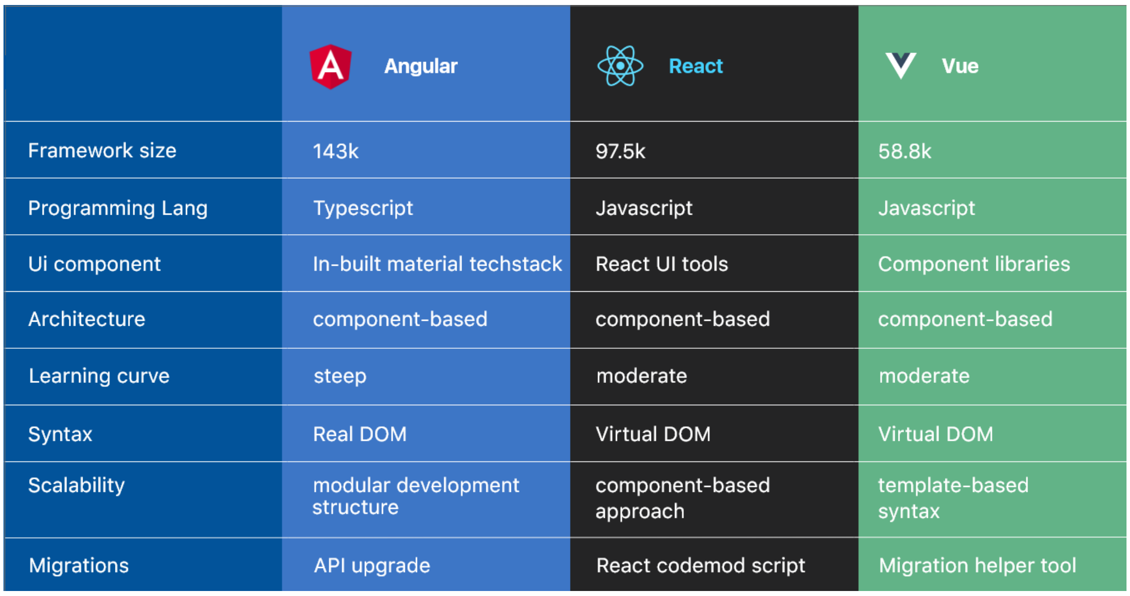
Figure 2.4: Comparison between VM and container technologies.

Front-end

This part of the architecture determines the technology that allow users to access the cyber range. This technology is chosen depending on the purpose and type of the range.

[1]https://www.docker.com

For present, a huge number of front-end technologies are provided as frameworks as well as libraries and packages, and available to use while most of them are open-source [35]. Front-end interfaces are basically architected using common development tools for the Web, i.e., HTML for the webpage structure, CSS for the webpage appearance, and JavaScript for the webpage behavior.



	Angular	React	Vue
Framework size	143k	97.5k	58.8k
Programming Lang	Typescript	Javascript	Javascript
Ui component	In-built material techstack	React UI tools	Component libraries
Architecture	component-based	component-based	component-based
Learning curve	steep	moderate	moderate
Syntax	Real DOM	Virtual DOM	Virtual DOM
Scalability	modular development structure	component-based approach	template-based syntax
Migrations	API upgrade	React codemod script	Migration helper tool

Figure 2.5: Top front-end frameworks comparison[2].

2.3 Scenario

In a cyber range, a *scenario* defines an environment created by assigning a limited number of resources to provide exercises and competitions under specific polices and rules. The range manages to setup a scenario by assigning the required resources, in addition to other process that take place in the back-end part. Moreover, a scenario follows a process (Figure 2.6) that controls its *cycle*. Of course, the process will be different between scenarios, but all of them will use the same concept. Therefore, we can recognize the most common stage of a scenario, starting from initial requirements and ending at execution. The process is composed from 3 fundamental phase: *Design*, *Testing* and *Exercise* [36].

- **Design**: The design of a scenario depend on certain number of pre-requisites like users and teams, capabilities, available resources, etc. [37]. Furthermore, to design a successful and valuable operative scenario, it is necessary to establish a comprehensive study to understand the methods and techniques that could be used by malicious

[2]https://www.angularminds.com/site_data/static/images/angular-react-vue/comparison-angular-react-vue.png

actors followed by an analytical assessment of vulnerabilites to bring out a set of categorized attack for the purpose of realizing the ultimate countermeasures against the attack. [38, 39].

- **Testing**: Testing and validation a scenario is carried out using a framework entitled **TOSCA** (*Topology and Orchestration specification for Cloud Application*) through a *Scenario Definition Language* (SDL) [40, 41]. The framework operate by automating the testing phase of the designed scenario by targeting the errors related to hardware and software within the design. Then if the test succeeds, the designed scenario will be automatically deployed, if not, a design modification should take place.

- **Exercise**: Another framework is reported for testing scenarios as well as deployment, entitled **ADLES** (*Automated Deployment of Laboratory Environments Systems*) [42]. The framework is able to design and deploy scenarios through a set of tools including an instructor guidance. ADLES follow a specific work-flow starting by identifying the primary components and elements needed and converting them into templates, then use these templates to create the desired network for the scenario along with services and other elements. The Figure 2.7 illustarte the ADLES work-flow.

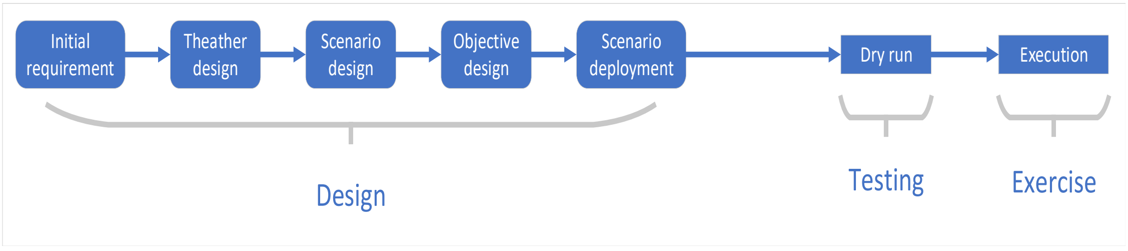


Figure 2.6: Scenario implementation process.

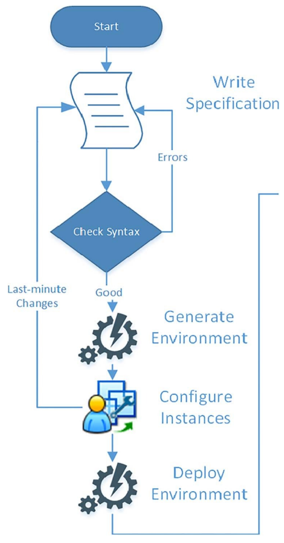


Figure 2.7: ADLES work-flow [42]

2.3.1 Classified Scenarios

It is difficult to categorize all scenarios since a large number of them can belong to different category in the same time. However, the most common scenarios could be classified based on their concept such as **Gamification** that aim to provide training session able to motivate trainees by increasing the level of challenges to extract their abilities, skills and knowledge during the session [43] , **Role Based** scenario assign to users a complete set of tasks related to each role, such as: offensive, defensive, incidence response, etc. [44]. And **Competition** exercise [45] for example, Capture-The-Flag (*CTF*) that aim to provide a hacking competition by involving several teams.

2.4 State of the Art

Cyber ranges could be categorized according to different criteria such infrastructure type, purpose, capabilities, and more. We would like to analyze and review some cyber ranges under 3 main categories: **Military and Government**, **Academic**, and **Commercial**. Since it is not possible to acquire much information about the Military category, we are going to present significative examples on Academic and Commercial ones.

2.4.1 Academic

SECUSIM

A software simulation developed at the Department of Computer Engineering at Hangkong University in Korea in 2001 called **SECUSIM** and aimed to analyse, verify and evaluate some specific attacks [46]. This software simulation is developed using C++ programing language and provides a graphical interface that allow the user to create the desired virtual network.

This simulation provides several mode to operate: **Basic**, **Intermediate**, **Advanced**, **Professional** and **Application**. Each mode holds a different level:

- **Basic**: allows the simulation of basic attack mechanisms;
- **Intermediate**: allows the simulation of attack in addition to setup the attack scenario;
- **Advance**: allows the simulation of attack using direct command level;
- **Professional**: allow sthe simulation of attack in addition of providing advanced analysis for vulnerability;
- **Application**: provides graphic editing that allow users to create, configure, and simulate their own modified network;

This simulator is important because of its simplicity and its graphical user interface (GUI) that provides a high level of customization. The software is developed based on a research published on 1999, that describe initially the constructive simulation of cyber-attacks using a single computer [46].

RINSE

The Real-time Immersive Network Simulation Environment for network security exercises (**RINSE**), developed at University of Illinois within Urbana-Champaign Coordinated Science Laboratory in 2005, has the purpose of managing large-scale real-time human/machine-in-the-loop network simulations, with a focus on security for exercises and training [47].

The RINSE is created on 5 components as shown in Figure 2.8 [47]:

- **iSSFNet**: network simulator developed using C++ programing language based on Scalable Simulation Framework (SSF), a set of APIs used for parallel simulation of large-scale networks;
- **Simulator Database Manager**: connects the **iSSFNet** to the database and delivers data from Simulator to Database and vice versa;
- **Data Server**: allows the user to monitor and control the simulation network using an interface application such as **Network Viewer** (developed using Java programing language);
- **Network Viewer Client**: provides the users with a local view of the network and offer a simple command prompt that allow users to execute commands that belong to several categories (Attack, Defense, Diagnostic Networking Tools, Device Control, Simulator Data).

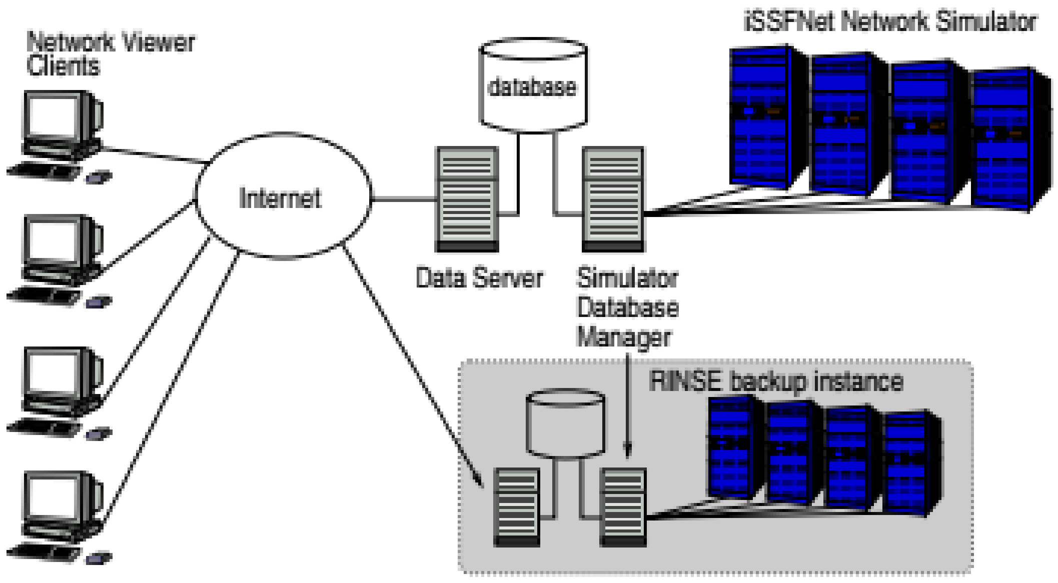


Figure 2.8: RINSE Architecture [47].

This simulator can support hundreds of players and a modeled network composed of hundreds of LANs that form a wide-area network (WAN) [46]. The attacks is performed versus WAN. Users have to defend and counter the attack by execute commands inside the terminal command in the Network Viewer Clients to keep their LAN's network services running.

A useful function allows to save points in a specific time of the simulation and change its velocity. Moreover, it allows the manager to regulate the simulation's resource such as CPU/RAM, that plays an important part when modeling attacks such as DDoS.

ARENA

ARENA is a constructive simulation developed at Rochester Institute of Technology (RIT) and used to simulate cyber-attacks against networks from an external source such as the Internet [48]. The attacks can be modeled automatically using a tool or alternatively pre-stated in XML file that can be loaded by simulation tool [46]. ARENA supports several type of attacks, such as DDoS and Penetration attacks to compromise the target computers.

Each attack is given characteristics of the attacker. Those characteristics are defined through 3 categories: *Efficiency*, *Stealth* and *Skill*. The Efficiency defines the quickness and the lightness of the attacker in moving from one level to another in a multi-tiered networks. Stealthness refers to the capability of the attacker to eschew unneeded steps which may alert network defenders, such as firewall. Last, the attacker's Skill measures the success of the actions taken during the attack to proceed in next step.

ARENA simulator provides several functions, such as determining the IP address inside the network and choosing the operating system as well as the type of Intrusion Detection System (IDS). Actually, ARENA is most importantly used for analyzing the IDS alerts, both positive and negative. Then, after launching the attacks, the statistics can be round up by applying the attack details plus the attacker characteristic against the target network architecture.

LARIAT

The Lincoln Adaptable Real-time Information Assurance Testbed (LARIAT), is an extension of the testbed created for DARPA 1998 and 1999 intrusion detection system (IDS) evaluations [47]. LARIAT provide real-time automated analysis and evaluations of IDS as well as Information Assurance (IA) technologies.

There are two target were proposed for designing LARIAT. The first one supports real-time evaluation, and the second one creates an easily-usable and configurable testbed. It provides a graphical user interface (GUI) to launch and manage attacks against several type of network, including hosts running different operation systems.

LARIAT is a mixture of simulation and real hardware. It is considered a good simulator for security research, as it is fit to develop new technologies, and measure the IDS weaknesses and hardness [49].

2.4.2 Commercial

IBM

IBM's X-Force command centers are the first physical cyber rangeS for commercial sector, with the aim to help clients respond to cybersecurity incidents [50]. IBM X-Force cyber ranges help companies to deal with vast data attacks and how to react, and also how to block future attacks.

IBM X-Force can support more than 100 operators in parallel, and provides the capability to use live malware, ransomware and several hacking tools taken from the dark web. The clients can test their environments by running several attack scenarios and react against scenarios, plus identify weakness of their systems. Figure 2.9 shows the IBM X-Force cyber range.

Figure 2.9: IBM cyber range[3].

EXata

EXata Cyber Range is a simulation software used for planning, testing, and especially for training in a scalable network simulation system [31]. This cyber range is considered a virtual cyber range, and it focus on how the wireless communication will conduct during a cyber-attack such as radio jamming or DDoS. However, even if EXata provide an excellent performance in wireless communication, it still has problem in the process of *ad-hoc* network simulation. In particular, it cannot realize the visualization and the deduction of ad-hoc network simulation [48].

[3]`https://techcrunch.com/2016/11/16/ibm-opens-new-cambridge-ma-security-headquarters-with-massive-cyber-range/?guccounter=1`

Chapter 3

PAIDEUSIS: Hybrid Cyber Range

3.1 Introduction

PAIDEUSIS is the name of a cyber training camp implemented in Turin by the research group of Prof. Paolo Prinetto. Its name is taken from the Ancient Greece and indicates the education of young people to life and war, and by extension also the place where this education was provided. PAIDEUSIS is a *hybrid* cyber range, combining virtualized component with physical devices, and a network infrastructure as well. Through this combination, PAIDESUSIS provide several scenarios and testbeds that offer a real training sessions in the cyber-security domain, such as Vulnerability Assessment, Penetration Testing and cyber defense, by implement scenarios based on software and hardware.

The network infrastructure allows users to access remotely and the ability to reconfigure resources for the scenarios through a flexible mechanism.

3.2 Architecture

3.2.1 CRACK: A Flexible Orchestrator

During the operation process of a scenario, it takes a lot of time to assign resources in order to give the life for the desired scenario. More, performing the execution also takes a considerable amount of time. This issue is considered fundamental, since performance and flexibility are highly recommended properties nowadays. Then, it is nearly impossible to make the usage of an old scenario by making a reconfiguration, since scenarios are complex, especially when dealing with hybrid ranges. Thus, it is necessary to use an *orchestration* solution for this purpose. One of the top orchestrators that aim to give the desired flexibility is called **CRACK** [51]. This project is developed to make the implementation of scenarios more flexible, while this is done by using a specific language called **CRACK SDL**(*Scenario Definition Language*) [51], which basically relies on **TOSCA** (*Topology and Orchestration*

Specification for Cloud Applications) a framework developed by **OASIS** [1] that offers an infrastructure development language.

CRACK SDL are able to represent any element of the scenario as well as other items of the Cyber Range using an essential element called *node*. For example, it can represent hardware and network devices, software, security elements, and more. Using this methodology, engineers are able to design scenarios by making connection between essential elements predefined by CRACK SDL and by consequence, this provides the full flexibility required to implement or reconfigure a scenario, since it depends on elements that could be connected/disconnected when needed.

3.2.2 Range Topology

PAIDEUSIS internal components, that are the basic resources to be assigned in order to setup a scenario, are organized in a systematic view that allow easy modification and reconfiguration for future purposes. They could be presented as an enclosure of several layers (Figure 3.1). The lowest layer of the architecture is called **component**, the most basic part that contribute in creating a scenario (e.g., an embedded device). Then, the second layer is called **subnet**, which is basically defined as a combination of several components connected together.

[1] `https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca`

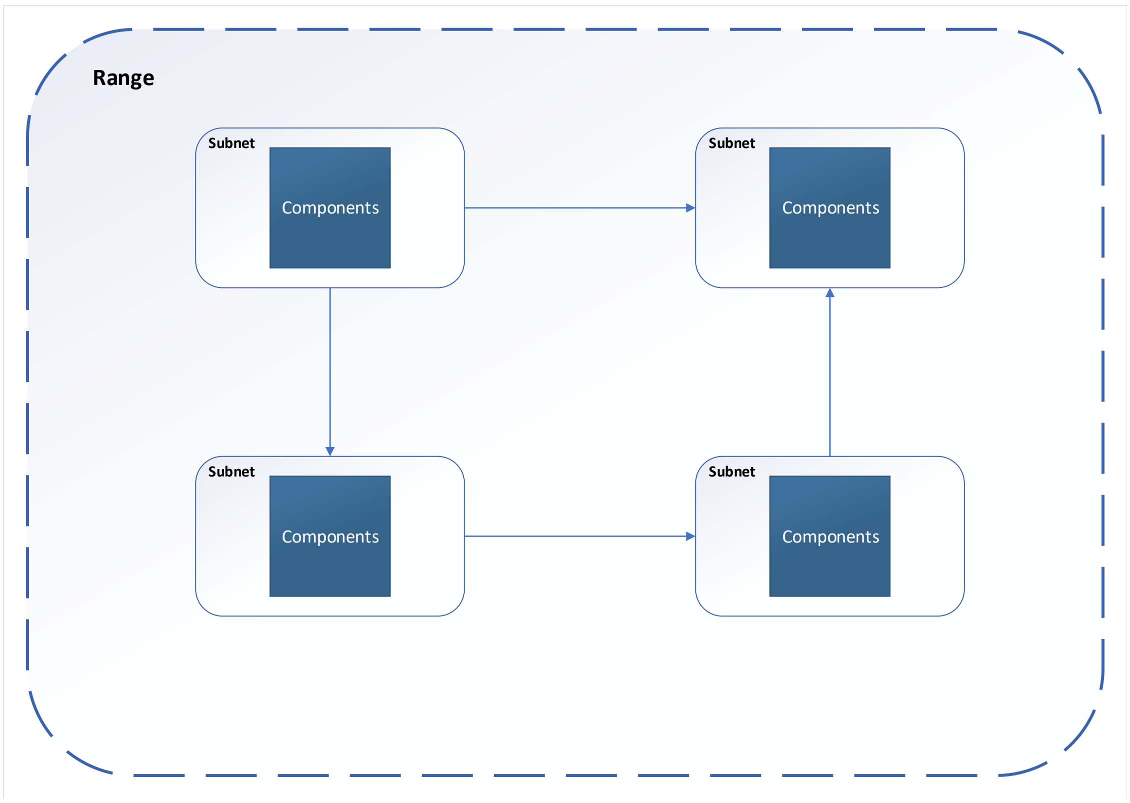


Figure 3.1: PAIDEUSIS internal organization.

We have the possibility to use any communication technology to interconnect components depending on the purpose of every scenario (e.g., wireless, Bluetooth). Finally, the **range** is the third layer that actually holds several subnets to be integrated in order to create a scenario.

Figure 3.2 illustrates the overall topology of PAIDEUISIS starting from the user and ending to the infrastructure. The user can access two virtual servers: the first one is for user teams, and the second one is for maintainer teams. These two virtual servers are connected to the PAIDEUSIS infrastructure that grants connection to the target physical devices, such as routers, switches, IoT sensors, FPGA and more. In the case of scenarios that depend on virtualization process, the servers are responsible of making the virtualization process happen with the devices or directly using the resources for scenarios that depend on the physical hardware. However, the Green Team is responsible of managing the infrastructure including the setting up of new ranges, using tools like **PROXMOX VE**[2], a graphical interface that allow the creation of virtual machines within the infrastructure. **WireGuard**[3] is instead the tool chosen for the project to setup and manage the virtual networks, allowing users to access the cyber range through this virtual network, as well

[2] https://www.proxmox.com

[3] https://www.wireguard.com

as isolating the scenario itself with the assigned components of this scenario. Besides, the White Team is responsible of setting up the scenario process using TOSCA framework and CRACK SDL, that provide all the flexibility and performance for this kind of purposes.

Moreover, the monitoring system alongside the scoring method are important to the cyber range, especially when dealing with a **CTF** (*Capture-the-Flag*) [52] exercise, which is the most famous type of exercise for cyber security expert worldwide. For this reason, the White Team takes care the setup of such systems.

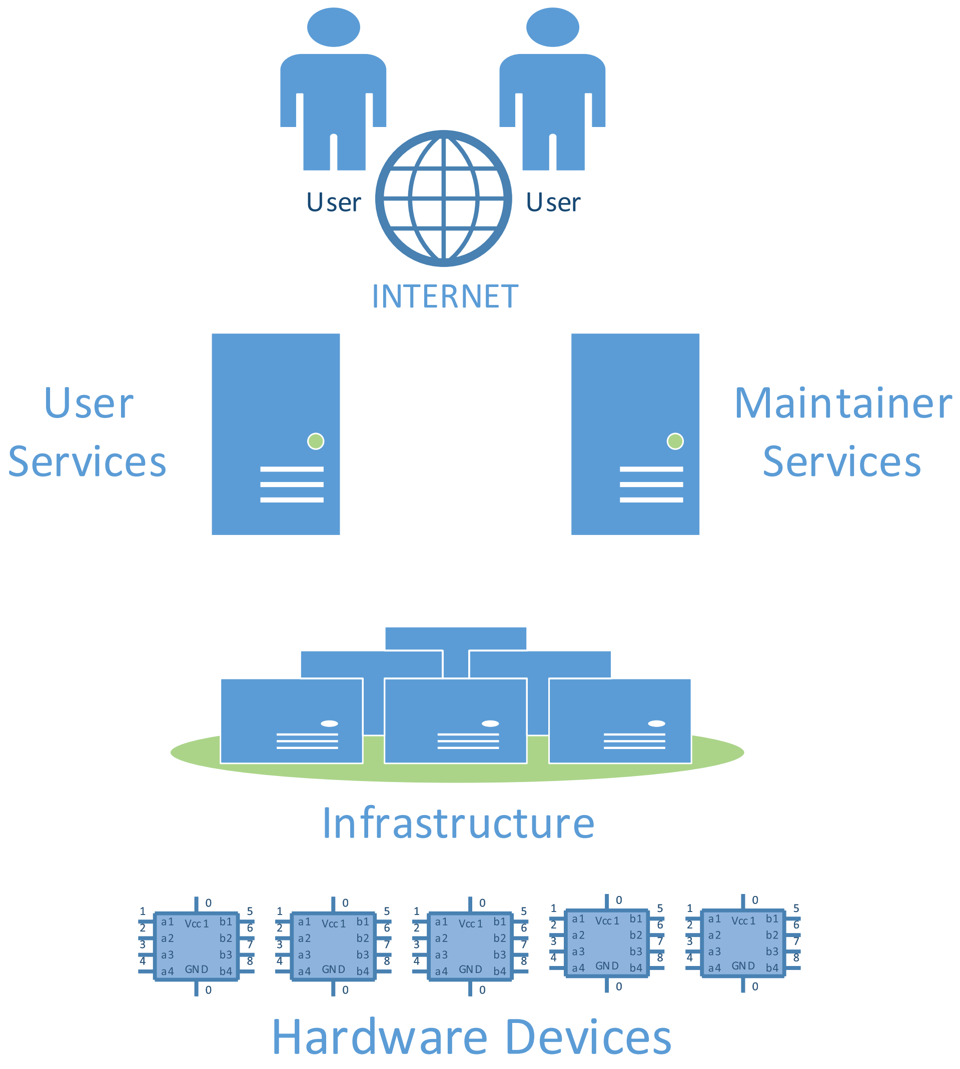


Figure 3.2: Topology overview.

3.3 PAIDEUSIS Ranges and Capabilities

The hosted ranges cover several aspects in cybersecurity to provide the desired scenarios. They are recognized in various categories: **Hardware**, **Network** and **Industrial**.

- **SE*cube*™ Range**: the user is allowed to access training scenarios over the SE*cube*™ physical platform[4], created by Blu5 Group[5] and hosting a three-part chip inside (microcontroller, FPGA, Smart Card). More, it can also host CTFs based on exploiting vulnerabilities in the microcontroller firmware or the hardware FPGA configuration. The access medium to such kind of scenarios is provided by a virtual machine on top of the hardware, equipped with all the tools needed to program the device. Currently, there are 40 virtual machine online, with a Linux operating system based on a Debian distribution entitled **Xubuntu**[6]. Figure 3.3 present the overview of this range with additional technical details.

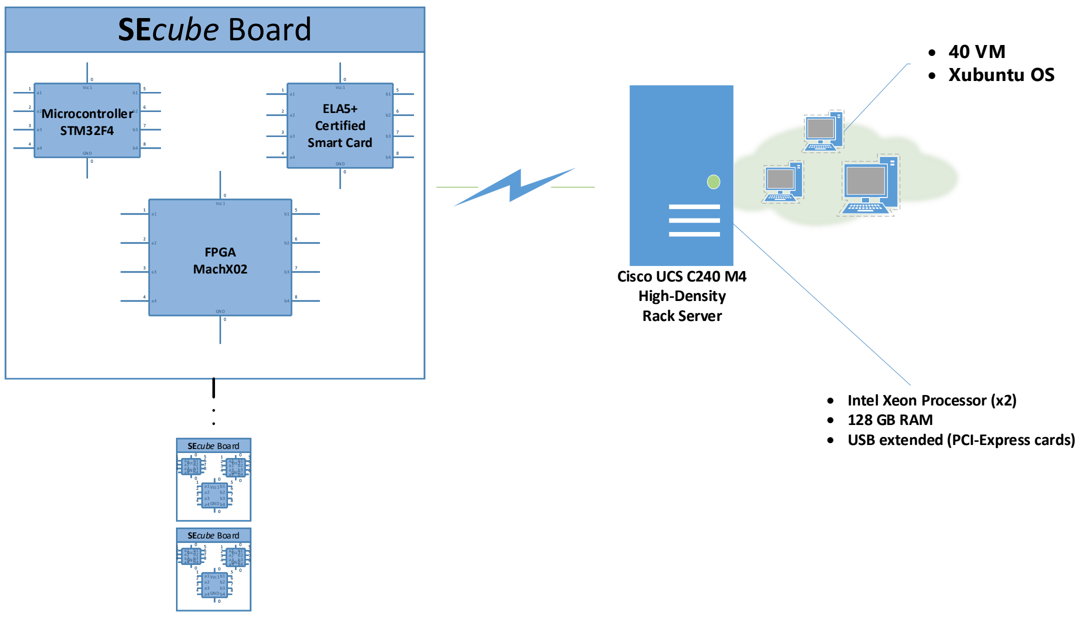

SEcube Board
Microcontroller STM32F4
ELA5+ Certified Smart Card
FPGA MachX02
Cisco UCS C240 M4 High-Density Rack Server
• 40 VM
• Xubuntu OS
• Intel Xeon Processor (x2)
• 128 GB RAM
• USB extended (PCI-Express cards)
SEcube Board
SEcube Board

Figure 3.3: SE*cube*™ Range topology overview.

- **ChipWhisperer Range**: this range provides the necessary resources to deploy scenarios that are based on physical ChipWhisperer-Nano devices[7]. Several chips are integrated on the board, and user attack a specific chip as a target. Scenarios designed in this range provide a number of exercises that allow users to proceed with

[4]https://www.secube.eu

[5]https://www.blu5group.com

[6]https://www.xubuntu.org

[7]https://www.newae.com/products/NAE-CW1101

their physical offensive skills on the target, such as *fault attacsk* [53]. The medium of access is guaranteed using Python scripts: there is no graphical interface, and the requirement resources to be assigned are less than the assigned resources for the SE*cube*™ Range.

- **Digital Hardware Emulation Range**: this theater offer hardware resources that are emulated using specific software application such as *ModelSim*[8], in which White Team is able to use hardware description languages such as **VHDL** (*VHSIC Hardware Description Language*), for simulating all kind of digital circuits, ranging from the fundamental ones (e.g., logic gates, multiplexers, decoders, flip-flop) to complex circuits (e.g., microcontrollers, microprocessors, switches), and to embed vulnerabilities inside.

- **Network Security Range**: this theater is equipped with a certain number of physical network devices. The purpose is to offer scenarios that challenge users to exploit a number of security misconfiguration on the network. Devices can be switches, routers, firewalls, ethernet cards, and any kind of network access technology. In the past, the range has been used to set up an Attack/Defense CTF challenge between several teams of attackers: each team had first to hack its own router, and penetrate inside the others team's routers through Wi-Fi.

- **Industrial Control System Range**: a number of physical devices coming from the industrial sector are hosted on this range. For example, a real watercourse emulator entitled *EVA* [54] allow users to perform malicious activities or security analysis over a realistic testbed for their desired purpose (e.g. training, research). A SCADA control system is present, and the control mechanism over the watercourse is done by sending control instructions to IoT sensors, by using a network protocol through a WI-FI access point.

[8]https://www.intel.it/content/www/it/it/software/programmable/quartus-prime/model-sim.html

Chapter 4

Back-end Code Infrastructure

As previously described (Figure 2.3), the general architecture of a cyber range can be divided into **front-end** and **back-end** domains. The focus of this work is mainly on the back-end domain. Everything that run behind the scenes within the infrastructure can be considered part of the back-end.

4.1 Technical Overview

The development of the back-end code for any infrastructure that offers some kind of service to the users starts with two fundamental choices:

1. the programming language to write the code;
2. the database to store and manage any user-related data.

It is very important to choose the right programming language alongside the right database, in order to optimize the performance and deliver the stability on the long run.

Several tools has been used during the development phase for coding and testing (Figure 4.1), such as:

- *Visual Studio Code*[1]
- *PostgreSQL Database*[2]
- *Microsoft Visio*[3]
- *Python*[4]
- *Django*[5]

[1]https://code.visualstudio.com

[2]https://www.postgresql.org

[3]https://www.microsoft.com/microsoft/visio

[4]https://www.python.org

[5]https://www.Djangoproject.com

- *Postman*[6]
- *HTTPie*[7]

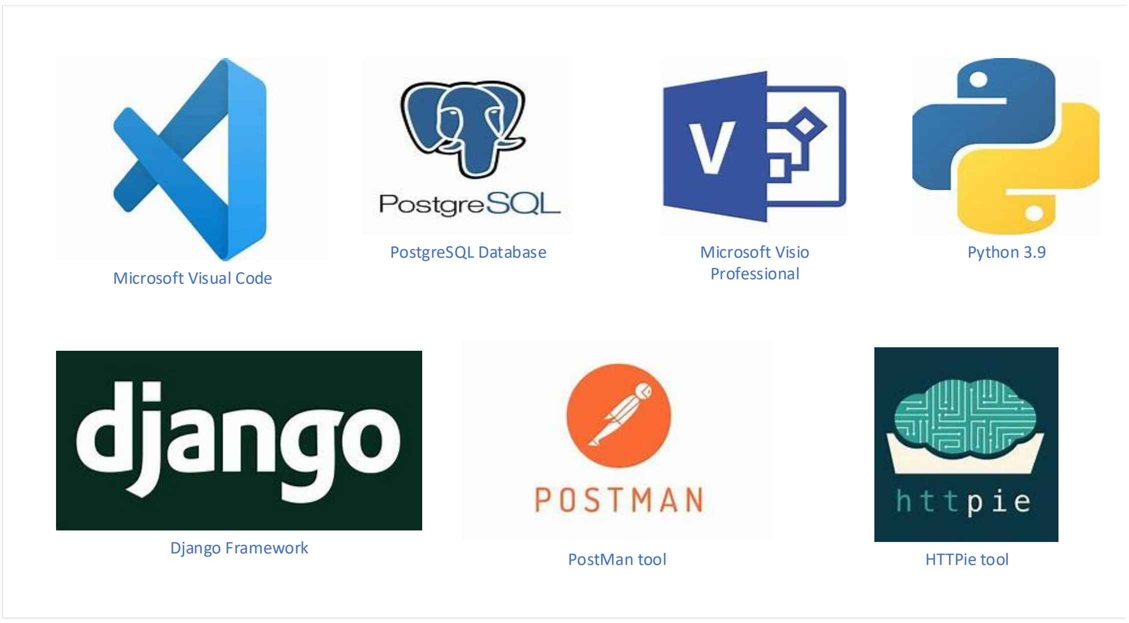


Figure 4.1: Development tools used in the back-end implementation.

Python

Among several programming languages available, the choice was of using **Python**[8]. The reason is actually the high flexibility provided, alongside the enormous community on the web that take in charge to support all kind of libraries and packages (e.g., machine learning, graphs, web application, security, networking, etc.), since the totality of them is provided as open-source code. In addition, a certain number of big companies worldwide use Python in their back-end services, such as Google Search Engine, in addition to companies like Intel, IBM and NASA [55].

PostgreSQL

Besides, the chosen database is **Postgre***SQL*[9], offering a high level of flexibility for designing tables and making relationships. PostgreSQL offers a lot of features that are highly required for PAIDEUSIS such as ranking, monitoring and analyzing. It can perform complex *search lookups* and retrieve results by similarity or by weighting terms: this will be

[6]https://www.postman.com

[7]https://httpie.io

[8]https://www.python.org

[9]https://www.postgresql.org

a useful feature provided for teams and students for several purpose like creating reports, doing research and more.

Django

The back-end is supported by several security features that are needed during any communication process for outgoing/ingoing requests, and provide the protection for the database holding all the data of PAIDEUSIS, including information about registered users. In such kind of project, the database holds a large number of tables interconnected by a complex designed relationship between them. These features are carried out by using the **Django Framework**[10], written in Python language. By using Django, each class is called **Module**, while each module includes a large number of functions and algorithms that make the use of Python's libraries and packages that fit our need. The implementation of the desired features has been done by combining a large number of frameworks, packages, libraries, and extensions that hold all kind of communication process and security behavior over the database and infrastructure.

The flexibility provided by Django is on a superior level when compared to other frameworks. To clarify that, a simple structure of a project implemented using Django could be illustrated such as in Figure 4.2. The project consists of several application where the **Main** application hold the settings of the project such as database type, packages, connection settings, and more. Other applications are simply a part of the project, where each one contains a set of modules that work together to provide a function to the project. Therefore, the flexibility consists in being able to take an application and making it work within another project that require this functionality. It can be done by simply defining the application's name within the main application. In this way, we are telling Django that this new App is available to use.

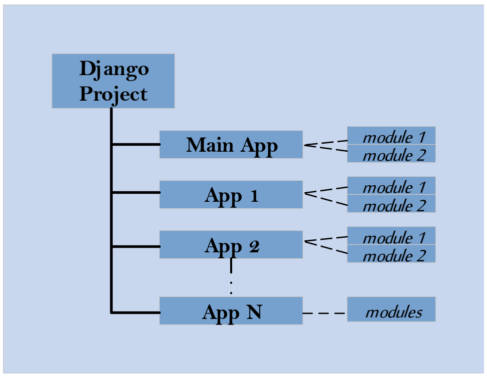


Figure 4.2: Django structure.

Some modules are essential to make things work, while others are optional to be added or modified to provide additional functionalities (e.g., security, admin capabilities, etc.). The following modules are considered fundamental to run a basics Django application. The

[10]https://www.djangoproject.com

workflow describing their behavior when combined together is presented in Figure 4.3. In detail, we have:

- **URLconf**: the navigator of the project that routes and matchs between requests received and project content resource;
- **Views**: presents the database in a specific format that matches a specific set of objects inside a table depending on the received request;
- **Template**: this module can include templates to support the view module for several reason, including security purpose (more details in Section 4.2).
- **Models**: contains the code responsible to create everything inside the database (e.g., tables, queries, keys, etc.).

Whenever a request is receivedm the URLconf module will match the request with the correct view that contain the right methods and functions to allow the manipulation of the database in order to serve the request. Additionally, the module called Template provide a friendly-designer plain text to develop the case view to support a set of views that share in common the same behavior by inherit and extend their views function from templates instead of writing a view several time.

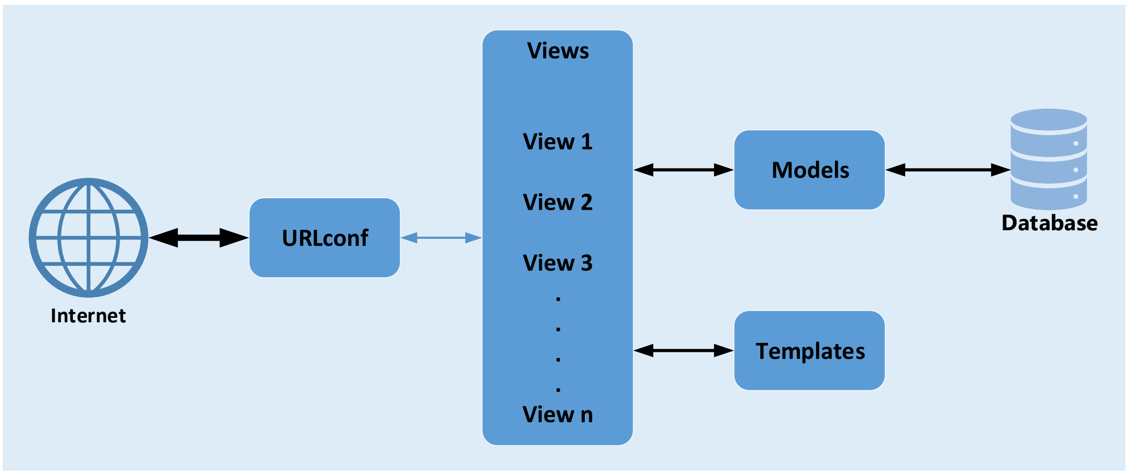


Figure 4.3: Django application workflow.

HTTPie & Postman

For testing purpose during the development phase, it is fundamental to perform some testing activities over the database. **HTTPie** is installed to make it easy to send HTTP requests and monitoring responses over the network[11], by providing a command-line HTTP client written in Python. **Postman** is instead a very popular API testing suite tools,

[11] https://httpie.io/docs

providing a graphical user interface that allow to easily compose and send HTTP requests among other features[12].

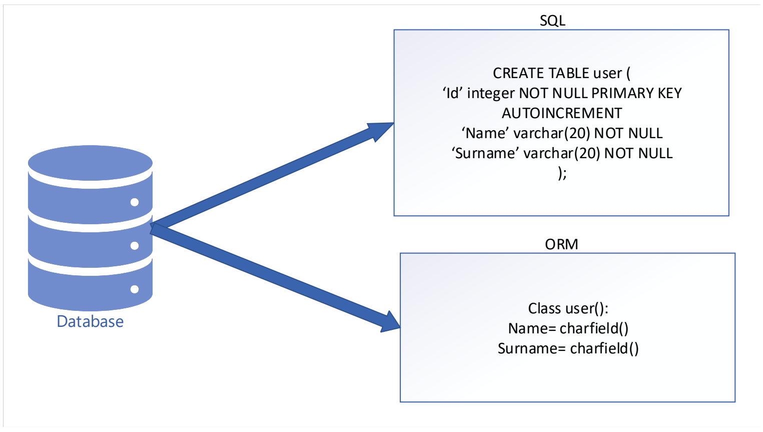
4.2 Technical Implementation

In this Section, we are going to clarify in details how things work and how they can be implemented. Each paragraph will take care of a topic and describe the reason of implementing the desired feature. Not all features and capabilities are mandatory to use, but they are included as options to provide a customizable database for PAIDEUSIS for which the team will be able to use it for now and for any future works.

Create & Configure The Database

Using ORM

Actually, the most common databases allow users to interact with the data by using **SQL** (*Structured Query Language*), where each database provide SQL by its own way. On the other hand, Django models allow developers to use **ORM** (*Object-Relational Mapping*), which is a programming technique that makes easy to interact with a database by providing a mechanism to map between object and database. To clarify that, Figure 4.4 illustrates the difference between SQL and ORM. It is clear that ORM provide simpler syntax to manipulate the database when compared to SQL. Therefore, ORM is used to implement the PostgreSQL database for PAIDEUSIS.



```
CREATE TABLE user (
'Id' integer NOT NULL PRIMARY KEY
AUTOINCREMENT
'Name' varchar(20) NOT NULL
'Surname' varchar(20) NOT NULL
);
```

```
Class user():
Name= charfield()
Surname= charfield()
```

Figure 4.4: ORM vs SQL comparison.

Implement Tables

After chosing to use ORM over SQL, the development process started by creating the tables and their colomns depending on the desired features. To make the database comprehensible and readable as much as possible not only for a presentation, but also for any

[12]https://www.postman.com/postman-features/

future modification to be performed by other engineers, we manage to organize tables into *categories*. This also actuates the *normalization* technique, aiming to eliminate repetitions by keeping related data into separate tables while creating relationship between them. For PAIDEUSIS, the schema presented in Figure 4.5 is optimized and ready to be extended and reconfigured in order to fit any new requirement. It contains **25 tables**, that are organized into 4 categories as follow:

- **User Account**: 9 tables to manage users accounts. An account can be normal user with limited permission and access time related to the scenario chosen (e.g., students, researchers, etc.), or admin user (e.g., green team or white team member) with full access over the database including a full authorized permission to control and customize anything (e.g., create, delete, modify, permission, email, authenticate, etc.). In addition, users can register in groups to create a team with specific permission and session time.

- **Social Account**: 4 tables to provide the capability to perform the registration procedure through social media accounts into the cyber range with full security in mind. The number of supported platforms is very large. The following list contain the most common platforms used worldwide and supported here, they can be disabled or enabled any time alongside others that are not listed here:

  1. Amazon
  2. Apple
  3. Azure
  4. Discord
  5. Dropbox
  6. Facebook
  7. Github
  8. Google
  9. Instagram
  10. Linkedin
  11. Microsoft
  12. Paypal
  13. Twitter
  14. Yahoo
  15. Zoom

- **Ranges Data**: 7 tables currently to manage the available ranges. Each range will have an ID alongside other information like resources available, devices installed and the scenario created and more. In addition, each scenario created within a range will be assigned to the user which is already authenticated and authorized to access this scenario. The Figure 4.6 show the main table that register the list of installed ranges, where a foreign key is poiting to each range table.

- **Log**: 5 tables to record activities and actions in the cyber range for several purpose (e.g., security, archiving, etc.), including the administrator logs and the scripts generated during a migration process (e.g., create, delete, update) with dates and times at which they were applied.

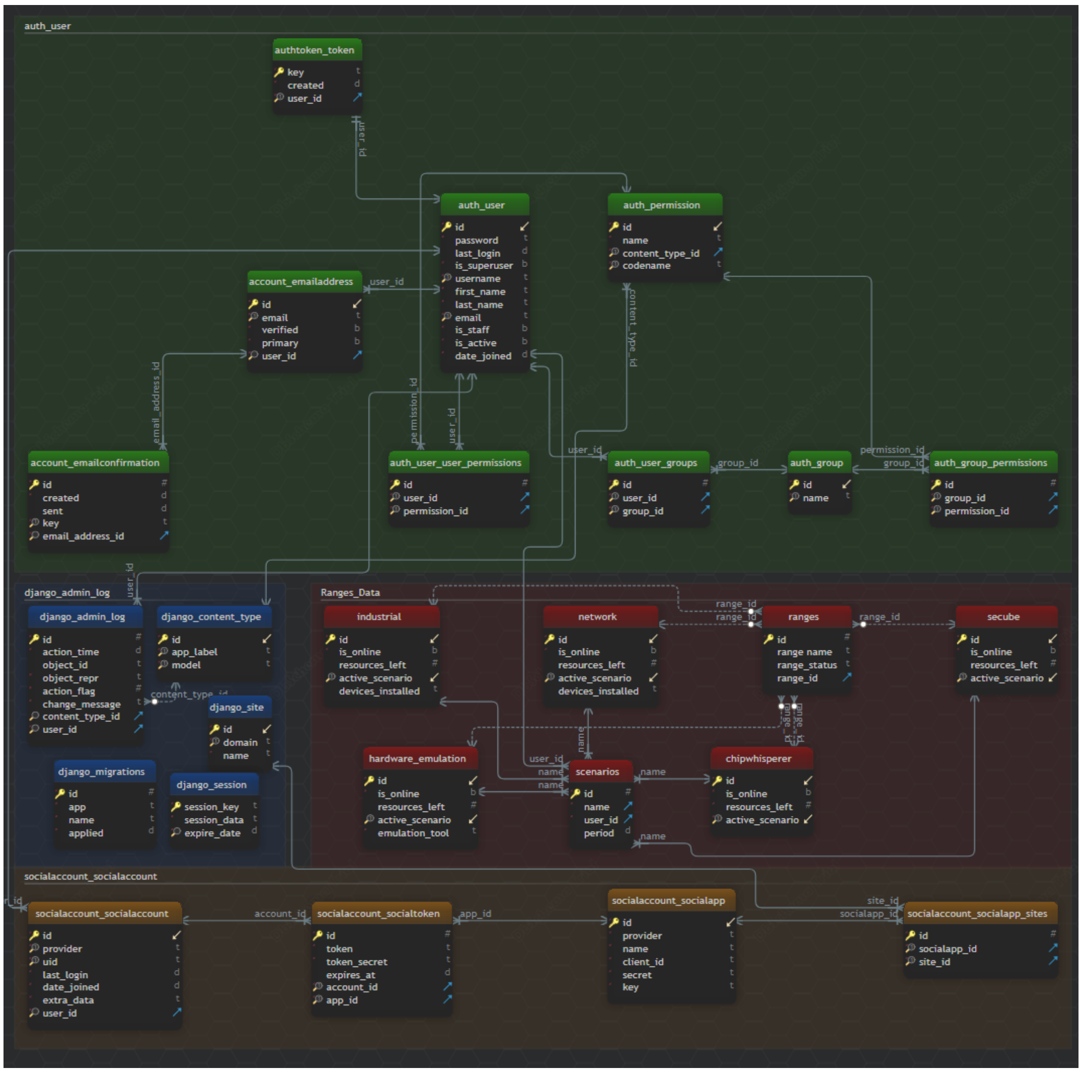


Figure 4.5: Database structure.

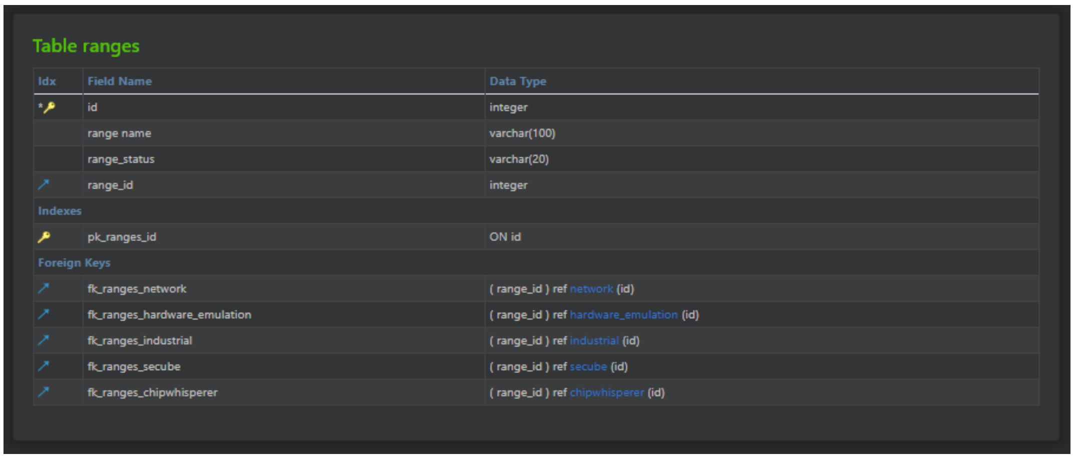


Table ranges

Idx	Field Name	Data Type
*🔑	id	integer
	range name	varchar(100)
	range_status	varchar(20)
↗	range_id	integer
Indexes		
🔑	pk_ranges_id	ON id
Foreign Keys		
↗	fk_ranges_network	(range_id) ref network (id)
↗	fk_ranges_hardware_emulation	(range_id) ref hardware_emulation (id)
↗	fk_ranges_industrial	(range_id) ref industrial (id)
↗	fk_ranges_secube	(range_id) ref secube (id)
↗	fk_ranges_chipwhisperer	(range_id) ref chipwhisperer (id)

Figure 4.6: The Ranges' main table.

Serialization

The *Serialization* process take place to define a specific format of the data to be exchanged over the network, since not all software and network devices are using the same programming language or database. Therefore, this module transforms information to be sent from the database into **JSON** (*Java-Script Object Notation*), a format notation that represent a data into *key-value* pairs (e.g. `name:value`) which is easy to read and recognize within other software tools and languages. Same thing applies when a data is received: data come in JSON format, but our module performs the *Deserialization* process to make the data readable and usable. This process is done by using two classes, *Parser* and *Render*. When a request is received, the incoming data are deserialized by implementing functions from the parser class, while the outgoing data have to be serialized by implementing functions from the render class.

Providing Security

Authentication

The *Authentication* process aims to verify the identity of the user. It is impossible to develop any kind of project without authentication. The community offersa plethora of customized methods, functions, classes, framework, libraries, and other to provide authentication in different level. A certain number of classes are developed using Python's libraries to create a custom authentication process that fit our need. Therefore, several kind of authentication back-end core mechanisms are used, ranging from the basic mechanism that authenticates a user based on username and password, to advanced mechanism that provide authentication based on token, email confirmation and sessions.

Cross-Site Scripting (XSS)

Cross-Site Scripting (**XSS**) is one of the most known security problems for almost any web application running on the Internet today. Attackers are able to run their own malicious script within the web application (usually based **JavaScript**) run by the user. Therefore, the server which is hosting the application (i.e., the back-end code of the application) and serving the request of the user, will also serve the attacker as a trusted client alongside the original user. The malicious script can access the server in different way, mostly by using form fields, fake URLs, and fake Ads. To prevent this kind of attack, it is essential to use template views to extend them within the views module, since template are able to auto-escape the HTML special characters, so that any malicious JavaScript-related code will be encoded to HTML instead of being executed. Then, whenever a user is going to register to the cyber range by using a web application (e.g. browser), he is protected from this attack.

Cross-Site Request Forgery (CSRF)

Cross-Site Request Forgery (**CSRF**) can be a powerful attack when a user is operating on a web application where an attacker is holding a fake element that allow the execution of unwanted actions by using the user's credential that belong to another web application where the user is already authenticated inside and didn't perform the logout yet. The attacker does this without knowing the user's credential in condition. Therefore, when a user is already authenticated in PAIDEUSIS while using another web application in the same time, this could be a vulnerability. The countermeasur taken to protect PAIDEUSIS from a CSRF attack is using a *token* inside each template developed. The token is implemented by embedding a generator function (**csrf_token**) within the template header, where this token will be rendered to a random number recognized by the back-end. The attacker will not be able to make an HTTP POST request, as he cannot generate the same token.

SQL Injection

SQL Injection is considered a very famous web attack beside XSS: the concept is to inject (insert) a malicious SQL code within a query that is going to be executed on the database while sending an HTTP POST request. This attack is very dangerous, as the code could include any kind of SQL instructions (e.g., to create, select from, or drop tables). Fortunately, the database of the cyber range able to execute query using only the ORM structure instead of SQL. This will prevent any injection attack.

Clickjacking

Clickjacking attack is performed by creating an invisible malicious link in a web application by aligning it over a button where user will probably click on. It can be used for several purpose by the attacker (e.g., to locate the target location, activate malware, acquire information). The protection is basically created by fixing the frame of the header to make it usable only by the origin frame provided by the trusted domain. It is possible to prevent such kind of attack by implementing functions within the setting of the project, specifically under the class **Middleware** which is responsible of handling security behavior during a process. This configuration will be recognized by browsers, so it can help to notify user when the domain changed for an HTTP request.

End Points

The back-end software is designed to be flexible for customization and configuration, therefore, many end-point are available to be accessed which mean it's not mandatory to use a web browser or a similar access technology to access the cyber range, other technology can be used. The following are the standard end points of the back-end infrastructure code. They could be changed, re-configured, updated, re-named, extended whenever a new feature is deployed. In addition, since we are dealing with end-points, it is also possible to configure them with a front-end interface whenever a graphical user interface will be developed for PAIDEUSIS in the future.

Basic

1. /app-name/login/ (POST)
    - username
    - email
    - password
2. /app-name/logout/ (POST)
3. /app-name/password/reset/ (POST)
    - email
4. /app-name/password/reset/confirm/ (POST)
    - user id
    - token
    - password1
    - password2
5. /app-name/password/change/ (POST)
    - password1
    - password2
    - password(old)
6. /app-name/user/ (GET, PUT, PATCH)
    - username
    - name
    - last name

Registration

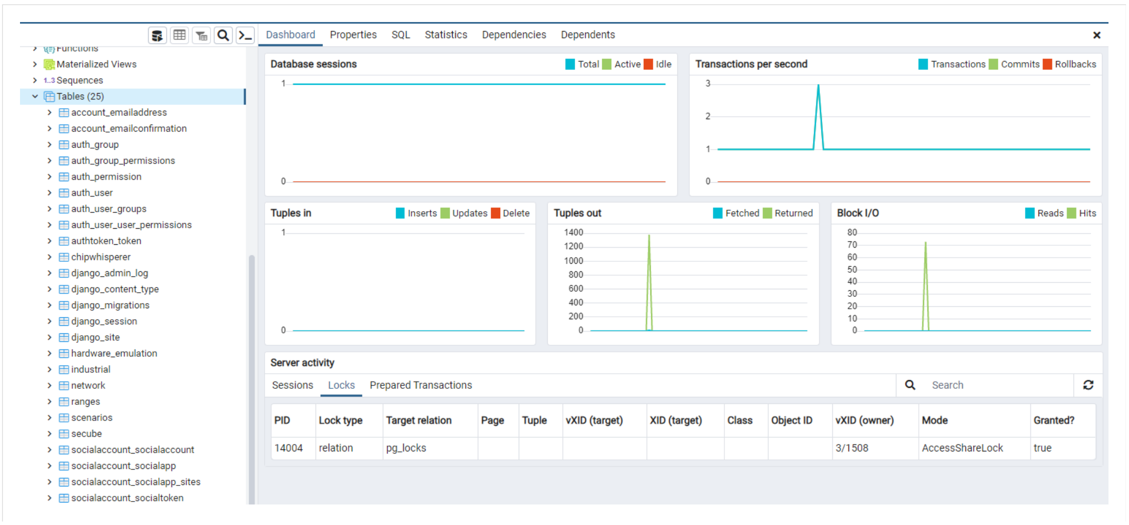
1. /app-name/registration/ (POST)
   - username
   - password1
   - password2
   - email
2. /app-name/registration/verify-email/ (POST)
   - key

Social Media Authentication

1. /app-name/facebook/ (POST)

4.3 Simulation

As a result, a real time simulation is performed in order to visualize how things are working together. Therefore, a virtual environment is created within a localhost server that holds the software including the database and connected to the dashboard provided by PostgreSQL. The simulation proves the functionality of the code including the policies implemented, like managing session of the user, monitoring and tracking. Figure 4.7 shows a snapshot from the dashboard taken during a test simulation.



- Functions
- Materialized Views
- Sequences
- Tables (25)
  - account_emailaddress
  - account_emailconfirmation
  - auth_group
  - auth_group_permissions
  - auth_permission
  - auth_user
  - auth_user_groups
  - auth_user_user_permissions
  - authtoken_token
  - chipwhisperer
  - django_admin_log
  - django_content_type
  - django_migrations
  - django_session
  - django_site
  - hardware_emulation
  - industrial
  - network
  - ranges
  - scenarios
  - secube
  - socialaccount_socialaccount
  - socialaccount_socialapp
  - socialaccount_socialapp_sites
  - socialaccount_socialtoken

PID	Lock type	Target relation	Page	Tuple	vXID (target)	XID (target)	Class	Object ID	vXID (owner)	Mode	Granted?
14004	relation	pg_locks							3/1508	AccessShareLock	true

Figure 4.7: Simulation of the back-end software.

Chapter 5

Future Work & Trends

5.1 Future Work

Throttling

Throttling is the process of making limited amount of request available to manage over an infrastructure, since a user is able to send thousands of requests to be processed without limitation[1]. This will become a kind of denial of services (**DoS**) attack, and the database will be overloaded, especially if resources are limited. When using throttles methods, the rate of request that users are able to send will be under control. For example, it is possible to fix a maximum number of requests per hour to 30 requests. This is can be done by using specific classes provided by Django, such as `AnonRateThrottle` to limit the rate of unauthenticated users by counting the amount of request generated by the same IP addresses. And a class called `ScopeRateThrottle` defines a limited number of requests to be accepted for specific services or features: therefore, it may be useful for several purpose. For example, if in the future the range will be able to simulate scenarios related to banking systems, it is essential to limit the amount of request similar to the real systems, this will introduce a real challenge for users.

Versioning

Versioning is allowing to run several versions of an application in the same time including all features and services[2]. Therefore, it will be possible to run several versions of the cyber range back-end, with the advantage of providing different behaviors between different users. This feature will make possible to provide the same scenario with different level of difficulties, for example by implementing different version of vulnerabilities.

[1]https://www.django-rest-framework.org/api-guide/throttling/

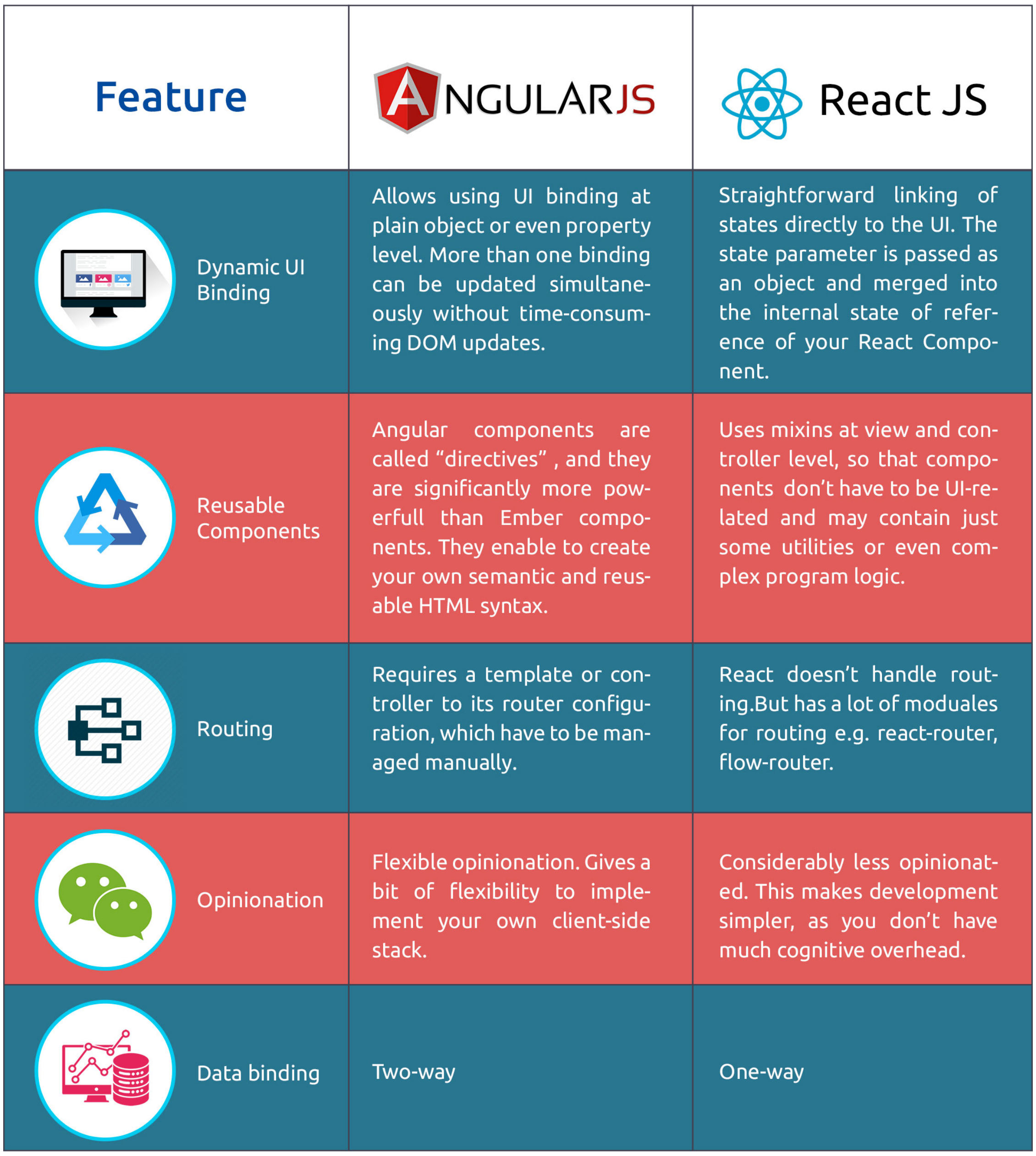
[2]https://www.djangopackages.org/grids/g/versioning/

Front-End

Sooner or later, a front-end interface will be implemented in order to present the cyber range in a stylish graphical interface to administrators and users. This can be done by using Django, but also with other framework that are more powerful to develop a front-end software. However, there exist a large number of frameworks and tools create by companies: for example, **Google** provides *Angular*[3], a framework that uses **TypeScript** and **HTML** to create front-end graphical interfaces. *Facebook* provides instead a framework called *Reactjs*[4], that makes use of **JavaScript** to develop complex user interface. The top advantage of React is the capability of running developed application anywhere, on all platforms: Android, iOS, Windows, and more. Figure 5.1 shows the difference between Angular and React in the general overview.

[3] `https://angular.io/`

[4] `https://reactjs.org`



Feature	ANGULARJS	React JS
Dynamic UI Binding	Allows using UI binding at plain object or even property level. More than one binding can be updated simultaneously without time-consuming DOM updates.	Straightforward linking of states directly to the UI. The state parameter is passed as an object and merged into the internal state of reference of your React Component.
Reusable Components	Angular components are called "directives" , and they are significantly more powerfull than Ember components. They enable to create your own semantic and reusable HTML syntax.	Uses mixins at view and controller level, so that components don't have to be UI-related and may contain just some utilities or even complex program logic.
Routing	Requires a template or controller to its router configuration, which have to be managed manually.	React doesn't handle routing.But has a lot of moduales for routing e.g. react-router, flow-router.
Opinionation	Flexible opinionation. Gives a bit of flexibility to implement your own client-side stack.	Considerably less opinionated. This makes development simpler, as you don't have much cognitive overhead.
Data binding	Two-way	One-way

Figure 5.1: Angular vs React [5]

Docker & Microservices

Docker is the most used containerization technology and it is widely supported among cloud vendors, due to its simplicity to create and manage images, by using a base image (binary

[5]https://krify.co/wp-content/uploads/2018/08angular-vs-react-1.jpg

file) or *Dockerfile* (script). *Microservices* are instead a mean of viewing an application as split into several littler and simpler services for the purpose of containerizing each one alone. This will reduce the usage of resources due to its minimal overhead and application isolation. Therefore, the back-end software of a cyber range can be implemented in several ways through this technology. The simplest one is containerizing similar services together, e.g., in our case a container for PostgreSQL services, then a container for Django services and another one for other services like web server. In addition, this will make possible to run each container in a server with in different locations, and managing such kind of containers is done by using specific technology created for automating deployment and managing containerized applications such as *Kubernetes*[6]. Figure 5.2 shows an example of Kubernetes workflow with containers and microservices.

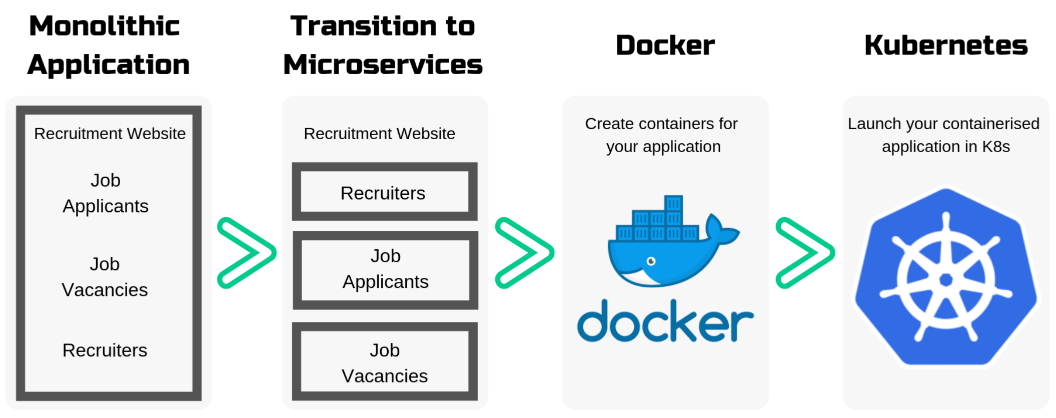


Figure 5.2: Kubernetes workflow example [7]

5.2 Trends

Auto-Configurable Cyber Ranges

An advanced tool for cyber ranges called *Automatic Live Instantiation of a Virtual Environment* (**ALIVE**) [56], developed by the *MIT Linclon Laboratory*, offers the capability of automatically building a virtual network of a cyber range and implementing the ranges and network infrastructure (e.g. hosts, routers, firewall, etc.). Another advanced tool is called *Cyber-Range Instantiation System* (**CyRIS**) [57], which offers the capability to create automatically a cyber range using specifications defined by the manager. The tool has

[6]https://www.kubernetes.io

[7]https://docs.bytemark.co.uk/wp-content/uploads/2019/02/Monolithic-Application.png

the power to create the infrastructure along with its settings.

AR/VR

The **Augmentation Reality** (AR) is a new technology that mainly enhances the real world by adding virtual objects and elements, allowing people to interact with them within the existed real world[8]. The **Virtual Reality** (VR), instead, will simulate a new virtual environment, completely different than the real one. However, they could be used to create an interactive virtual learning platform that provide exercises for this purpose [58]. This technology is very new and needs more time to be available to all people, but soon or later it will be used widely and then new attacking vector will raise against them. Therefore, it will be necessary to develop platform based on AR/VR to educate security people and improve protection.

5G Technology

As we know, the 5th generation of mobile broadband network is almost here. This offers a high connectivity speed with low latency by using new technologies like *Software Defining Network* (**SDN**) and *Network Function Virtualization* (**NFV**) [59]. Though, 5G can be brought to the cyber ranges in order to provide low latency in the network system. In addition, it is important to provide scenario based on 5G network, where attacker will target the 5G infrastructure (e.g., SDN), as this new generation will become more used in the future.

[8]`https://www.immersive.io/blog/what-is-augmented-reality-definition/`

Bibliography

[1] Parushi Malhotra, Yashwant Singh, Pooja Anand, Deep Kumar Bangotra, Pradeep Kumar Singh, and Wei-Chiang Hong. «Internet of Things: Evolution, Concerns and Security Challenges». In: *Sensors* 21.5 (2021), p. 1809.

[2] Hamed Haddad Pajouh, Reza Javidan, Raouf Khayami, Ali Dehghantanha, and Kim-Kwang Raymond Choo. «A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks». In: *IEEE Transactions on Emerging Topics in Computing* 7.2 (2016), pp. 314–323.

[3] Janice Canedo and Anthony Skjellum. «Using machine learning to secure IoT systems». In: *2016 14th annual conference on privacy, security and trust (PST)*. IEEE. 2016, pp. 219–222.

[4] Eirini Anthi, Lowri Williams, and Pete Burnap. «Pulse: an adaptive intrusion detection for the internet of things». In: (2018).

[5] Mahmudul Hasan, Md Milon Islam, Md Ishrak Islam Zarif, and MMA Hashem. «Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches». In: *Internet of Things* 7 (2019), p. 100059.

[6] Nataliia Neshenko, Elias Bou-Harb, Jorge Crichigno, Georges Kaddoum, and Nasir Ghani. «Demystifying IoT security: an exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations». In: *IEEE Communications Surveys & Tutorials* 21.3 (2019), pp. 2702–2733.

[7] SN Uke, AR Mahajan, and RC Thool. «UML modeling of physical and data link layer security attacks in WSN». In: *International Journal of Computer Applications* 70.11 (2013).

[8] Ioannis Andrea, Chrysostomos Chrysostomou, and George Hadjichristofi. «Internet of Things: Security vulnerabilities and challenges». In: *2015 IEEE symposium on computers and communication (ISCC)*. IEEE. 2015, pp. 180–187.

[9] CJ Rawandale, Manish M Deshpande, and Vinayak P Rajadhyaksha. «Banking on online banking». In: *2018 International Conference On Advances in Communication and Computing Technology (ICACCT)*. IEEE. 2018, pp. 240–244.

[10] Joris Claessens, Valentin Dem, Danny De Cock, Bart Preneel, and Joos Vandewalle. «On the security of today's online electronic banking systems». In: *Computers & Security* 21.3 (2002), pp. 253–265.

[11] ADHARSH MANIVANNAN. «CYBER ATTACKS IN THE BANKING INDUSTRY». In: ().

[12] Warwick Ashford. «Financial institutions on high alert for major cyber attack». In: *ComputerWeekly. com, February* 11 (2016), p. 2016.

[13] CSIS McAfee. «Net losses: estimating the global cost of cybercrime». In: *McAfee, Centre for Strategic & International Studies* (2014).

[14] Conor Woodrow and Kevin Curran. «Security Issues in Self-Driving Cars within Smart Cities». In: *Security and Organization within IoT and Smart Cities*. CRC Press, 2020, pp. 243–254.

[15] Shah Khalid Khan, Nirajan Shiwakoti, Peter Stasinopoulos, and Yilun Chen. «Cyber-attacks in the next-generation cars, mitigation techniques, anticipated readiness and future directions». In: *Accident Analysis & Prevention* 148 (2020), p. 105837.

[16] Zeinab El-Rewini, Karthikeyan Sadatsharan, Daisy Flora Selvaraj, Siby Jose Plathottam, and Prakash Ranganathan. «Cybersecurity challenges in vehicular communications». In: *Vehicular Communications* 23 (2020), p. 100214.

[17] Andrew Patrick Hennessy. «Implementation of physical layer security of an ultra-wideband transceiver». PhD thesis. San Diego State University, 2016.

[18] Sourabh Pawade, Shraddha Shah, and Dhanashree Tijare. «Zigbee based intelligent driver assistance system». In: *International Journal of Engineering Research and Applications* 3 (2013), pp. 1463–1468.

[19] VR Vijaykumar and S Elango. «Hardware implementation of tag-reader mutual authentication protocol for RFID systems». In: *Integration* 47.1 (2014), pp. 123–129.

[20] Abdulla O Al Zaabi, Chan Yeob Yeun, and Ernesto Damiani. «Autonomous vehicle security: Conceptual model». In: *2019 IEEE Transportation Electrification Conference and Expo, Asia-Pacific (ITEC Asia-Pacific)*. IEEE. 2019, pp. 1–5.

[21] Fei Hu. *Vehicle-to-vehicle and Vehicle-to-infrastructure Communications: A Technical Approach*. CRC Press, 2018.

[22] Liangkai Liu, Baofu Wu, and Weisong Shi. «A Comparison of Communication Mechanisms in Vehicular Edge Computing». In: *3rd {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 20)*. 2020.

[23] Minh Pham and Kaiqi Xiong. «A Survey on Security Attacks and Defense Techniques for Connected and Autonomous Vehicles». In: *Computers & Security* (2021), p. 102269. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2021.102269. URL: https://www.sciencedirect.com/science/article/pii/S0167404821000936.

[24] Saurabh Singh, Young-Sik Jeong, and Jong Hyuk Park. «A survey on cloud computing security: Issues, threats, and solutions». In: *Journal of Network and Computer Applications* 75 (2016), pp. 200–222.

[25] Issa M Khalil, Abdallah Khreishah, and Muhammad Azeem. «Cloud computing security: A survey». In: *Computers* 3.1 (2014), pp. 1–35.

[26] Ahmed Alzahrani, Nasser Alalwan, and Mohamed Sarrab. «Mobile cloud computing: advantage, disadvantage and open challenge». In: *Proceedings of the 7th Euro American Conference on Telematics and Information Systems*. 2014, pp. 1–4.

[27] Monjur Ahmed and Alan T Litchfield. «Taxonomy for identification of security issues in cloud computing environments». In: *Journal of Computer Information Systems* 58.1 (2018), pp. 79–88.

[28] Prerna Mohit and GP Biswas. «Confidentiality and storage of data in cloud environment». In: *Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications.* Springer. 2017, pp. 289–295.

[29] Victor Prokhorenko, Kim-Kwang Raymond Choo, and Helen Ashman. «Web application protection techniques: A taxonomy». In: *Journal of Network and Computer Applications* 60 (2016), pp. 95–112.

[30] Zhihong Tian, Yu Cui, Lun An, Shen Su, Xiaoxia Yin, Lihua Yin, and Xiang Cui. «A real-time correlation of host-level events in cyber range service for smart campus». In: *IEEE Access* 6 (2018), pp. 35355–35364.

[31] Jon Davis and Shane Magrath. «A survey of cyber ranges and testbeds». In: (2013).

[32] Elochukwu Ukwandu, Mohamed Amine Ben Farah, Hanan Hindy, David Brosset, Dimitris Kavallieros, Robert Atkinson, Christos Tachtatzis, Miroslav Bures, Ivan Andonovic, and Xavier Bellekens. «A review of cyber-ranges and test-beds: current and future trends». In: *Sensors* 20.24 (2020), p. 7148.

[33] Vasileios Linardos. «Development of a cyber range platform». MA thesis. *Πανεπιστήμιο Πειραιώς*, 2021.

[34] Amit M Potdar, Narayan D G, Shivaraj Kengond, and Mohammed Moin Mulla. «Performance Evaluation of Docker Container and Virtual Machine». In: *Procedia Computer Science* 171 (2020). Third International Conference on Computing and Network Communications (CoCoNet'19), pp. 1419–1428. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.04.152. URL: https://www.sciencedirect.com/science/article/pii/S1877050920311315.

[35] Duong Dinh and Zhuanyan Wang. «Modern front-end web development: how libraries and frameworks transform everything». In: (2020).

[36] Enrico Russo, Gabriele Costa, and Alessandro Armando. «Scenario Design and Validation for Next Generation Cyber Ranges». In: *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA).* 2018, pp. 1–4. DOI: 10.1109/NCA.2018.8548324.

[37] Daniele Marrocco. «Design and Deployment of a Virtual Environment to Emulate a Scada Network within Cyber Ranges». PhD thesis. Politecnico di Torino, 2018.

[38] Adam Hahn, Ben Kregel, Manimaran Govindarasu, Justin Fitzpatrick, Rafi Adnan, Siddharth Sridhar, and Michael Higdon. «Development of the PowerCyber SCADA security testbed». In: *Proceedings of the sixth annual workshop on cyber security and information intelligence research.* 2010, pp. 1–4.

[39] Chee-Wooi Ten, Chen-Ching Liu, and Manimaran Govindarasu. «Vulnerability assessment of cybersecurity for SCADA systems using attack trees». In: *2007 IEEE Power Engineering Society General Meeting.* IEEE. 2007, pp. 1–8.

[40] Enrico Russo, Gabriele Costa, and Alessandro Armando. «Scenario design and validation for next generation cyber ranges». In: *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA).* IEEE. 2018, pp. 1–4.

[41] Tobias Binz, Uwe Breitenbücher, Oliver Kopp, and Frank Leymann. «TOSCA: portable automated deployment and management of cloud applications». In: *Advanced Web Services*. Springer, 2014, pp. 527–549.

[42] Daniel Conte de Leon, Christopher E Goes, Michael A Haney, and Axel W Krings. «ADLES: Specifying, deploying, and sharing hands-on cyber-exercises». In: *Computers & Security* 74 (2018), pp. 12–40.

[43] K Boopathi, S Sreejith, and A Bithin. «Learning cyber security through gamification». In: *Indian Journal of Science and Technology* 8.7 (2015), pp. 642–649.

[44] Patricia Toth and Penny Klein. «A role-based model for federal information technology/cyber security training». In: *NIST special publication* 800.16 (2013), pp. 1–152.

[45] Art Conklin. «The use of a collegiate cyber defense competition in information security education». In: *Proceedings of the 2nd annual conference on Information security curriculum development*. 2005, pp. 16–18.

[46] Sylvain P Leblanc, Andrew Partington, Ian M Chapman, and Mélanie Bernier. «An overview of cyber attack and computer network operations simulation.» In: *SpringSim (MMS)*. 2011, pp. 92–100.

[47] Lee M Rossey, Robert K Cunningham, David J Fried, Jesse C Rabek, Richard P Lippmann, Joshua W Haines, and Marc A Zissman. «LARIAT: Lincoln adaptable real-time information assurance testbed». In: *Proceedings, ieee aerospace conference*. Vol. 6. IEEE. 2002, pp. 6–6.

[48] Zhao Niu, Tao Ma, Xiao-feng Zhong, and Lin Jiang. «Research on Topology Inference Method for Ad Hoc Network Simulation on Exata». In: *2018 Eighth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC)*. IEEE. 2018, pp. 174–177.

[49] Michael G Wabiszewski Jr, Todd R Andel, Barry E Mullins, and Ryan W Thomas. «Enhancing realistic hands-on network training in a virtual environment». In: *Proceedings of the 2009 Spring Simulation Multiconference*. 2009, pp. 1–8.

[50] Ishaani Priyadarshini. «Features and architecture of the modern cyber range: a qualitative analysis and survey». PhD thesis. University of Delaware, 2018.

[51] Enrico Russo, Gabriele Costa, and Alessandro Armando. «Building next generation cyber ranges with CRACK». In: *Computers & Security* 95 (2020), p. 101837.

[52] Chris Eagle. «Computer security competitions: Expanding educational outcomes». In: *IEEE Security & Privacy* 11.4 (2013), pp. 69–71.

[53] Duško Karaklajić, Jörn-Marc Schmidt, and Ingrid Verbauwhede. «Hardware Designer's Guide to Fault Attacks». In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 21.12 (2013), pp. 2295–2306. DOI: 10.1109/TVLSI.2012.2231707.

[54] Shabeer Ahmad, Nicolò Maunero, and Paolo Prinetto. «EVA: A Hybrid Cyber Range.» In: *ITASEC*. 2020, pp. 12–23.

[55] KR Srinath. «Python–the fastest growing programming language». In: *International Research Journal of Engineering and Technology (IRJET)* 4.12 (2017), pp. 354–357.

[56] Timothy M Braje. *Advanced tools for cyber ranges.* Tech. rep. MIT Lincoln Laboratory Lexington United States, 2016.

[57] Cuong Duy Pham. «On Automatic Cyber Range Instantiation for Facilitating Security Training». In: (2017).

[58] Kirsi Aaltola. «Empirical Study on Cyber Range Capabilities, Interactions and Learning». In: *Digital Transformation, Cyber Security and Resilience of Modern Societies* 84 (2021), p. 413.

[59] Christos Tranoris, Spyros Denazis, Lucas Guardalben, João Pereira, and Susana Sargento. «Enabling Cyber-Physical Systems for 5G networking: A case study on the Automotive Vertical domain». In: *2018 IEEE/ACM 4th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS)*. IEEE. 2018, pp. 37–40.