

Cyber Range Green Team Training

ACS 2025



CybExer

Agenda

- Introduction
- Cyber Range Components
- Building content manually (vSphere)
- Building content with Ansible
- Building content with vLM
- Visualizing exercise content with ISA
- ACS Gamenet overview
- Workshop of generating new target into ACS

Cyber Range Technologies

Ansible

What is Ansible

- **Ansible® is an open source, command-line IT automation software application written in Python.**
- It can configure systems, deploy software, and orchestrate advanced workflows to support application deployment, system updates, and more. Ansible's main strengths are simplicity and ease of use.
- **Agentless architecture.** Ansible doesn't require any special software to be installed on the nodes it manages.

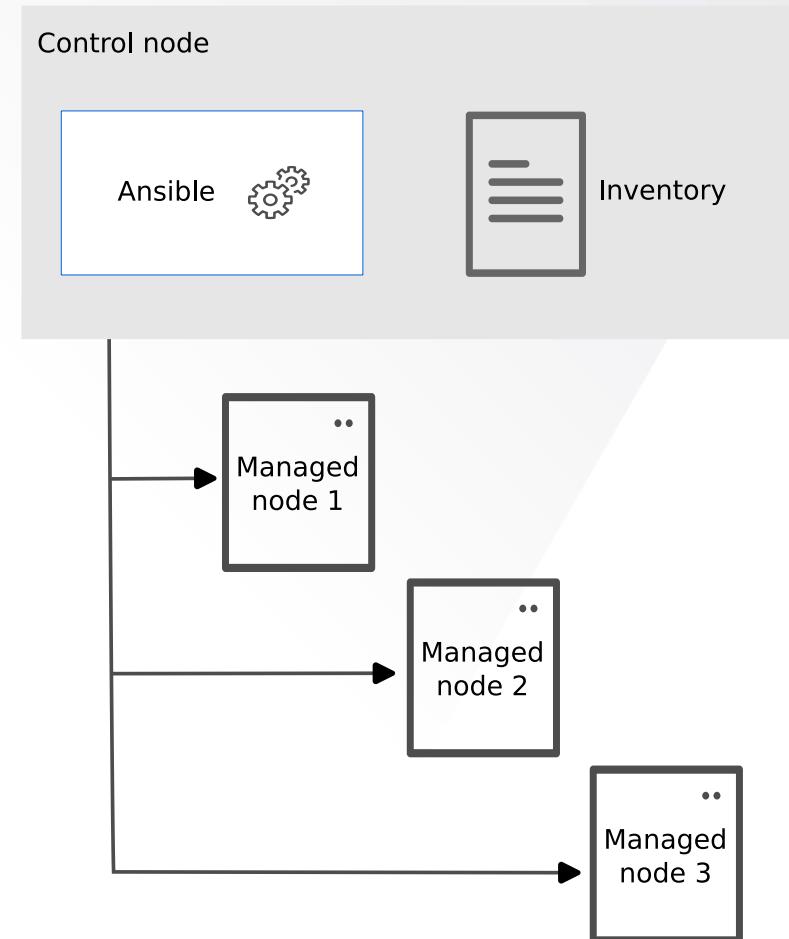
Why Ansible?

- **Simplicity:** Easy-to-read YAML format, suitable for all skill levels.
- **Ease of Use:** No complex setup; uses SSH.
- **Idempotency:** Ensures consistent results upon repeated playbook executions.
- **Community support:** Strong open-source community offering extensive modules and plugins.
- **Scalability & Flexibility:** Efficiently manages systems of any size, from a few nodes to thousands, with easy customization.

In essence, Ansible excels in simplicity, reliability, and adaptability, making it ideal for a wide range of IT automation tasks.

Ansible Overview

- **Control node** – A system on which Ansible is installed. You run Ansible commands such as *ansible* or *ansible-inventory* on a control node.
- **Managed node** – A remote system, or host, that Ansible controls.
- **Inventory** – A list of managed nodes that are logically organized. You create an inventory on the control node to describe host deployments to Ansible.
- **Playbook** – A file containing the instructions to apply to the managed nodes.



Ansible Core Terms

- Playbook
- Play
- Task
- Module
- Inventory
- Role
- Handler
- Variable
- Fact

Playbooks

- Playbooks contain Plays (which are the basic unit of Ansible execution). This is both an ‘execution concept’ and how we describe the files on which ansible-playbook operates. Playbooks are written in YAML format.
- A playbook is the file you call with the *ansible-playbook command to start the execution.*
- The playbook then includes/calls/references all other actions and parameters.

A YAML file that defines automation tasks.

```
- name: Install Apache on web servers
  hosts: webservers
  become: yes
  tasks:
    - name: Install Apache
      apt:
        name: apache2
        state: present
```

Play

- A play is one section of a playbook, mapping hosts to tasks

A YAML file that defines automation tasks.

```
- name: Install Apache on web servers
  hosts: webservers
  become: yes
  tasks:
    - name: Install Apache
      apt:
        name: apache2
        state: present
```

Tasks

- Tasks are a definition of an ‘action’ to be applied to the managed host.
- Tasks must always be contained in a Play, directly or indirectly (Role, or imported/included task list file).
- You can execute a single task once with an ad hoc command using *ansible* or *ansible-console* (both create a virtual Play).
- In short tasks are the steps you want to take in your play.

A single action in a playbook.

```
# Inside a play
- name: Install Apache
  apt:
    name: apache2
    state: present
```

Modules

- Modules are a type of plugin that **execute automation tasks** on a ‘target’ (usually a remote system). Modules work as standalone scripts that Ansible executes in their own process outside of the controller.
- Modules interface with Ansible mostly via JSON, accepting arguments and returning information by printing a JSON string to stdout before exiting. Unlike the other plugins (which must be written in Python), modules can be written in any language, although Ansible provides modules in Python and PowerShell only.
- In short modules are the scripts that get executed to do things. To extend ansible functionality you can create your own modules.
- For example: apt, file, user, copy, cron, ufw, reboot, etc

Inventory

- **Definition:** In Ansible, the inventory is a **collection of hosts** (nodes, servers) that the automation tool will manage.
- It can also include details about these hosts, like their **IP addresses** and any specific **variables** relevant to them.
- **Purpose:** The inventory file is used by Ansible to identify the hosts on which task should be executed.
- **Groups and Hosts:** In this example we have Linux and webservers **groups**. Under each group we have hosts which are the target servers.

```
inventory >  inventory.yml
 1   all:
 2     children:
 3       linux:
 4         hosts:
 5           server1.example.com:
 6           server2.example.com:
 7             ansible_user: admin
 8             ansible_ssh_private_key_file: /path/to/private/key
 9           server3.example.com:
10             ansible_user: root
11             some_custom_variable: value
12
13       webservers:
14         hosts:
15           webserver1.example.com:
16             http_port: 80
17             max_clients: 200
18           webserver2.example.com:
19             http_port: 8080
20             max_clients: 100
21
22         vars:
23           ansible_user: webadmin
24           ansible_python_interpreter: /usr/bin/python3
```

Roles

- **tasks:** Contains the main list of tasks to be executed by the role. `main.yml` inside this directory is the primary entry point.
- **handlers:** Used for tasks that should only be run when notified. Commonly used for restarting services.
- **templates:** Holds template files which can be deployed via this role. Templates allow for dynamic generation of configuration files.
- **files:** Contains files which can be deployed via this role. These are typically static files.
- **vars:** Contains variables that are high priority and **shouldn't** be overwritten by user-defined variables.
- **defaults:** Holds default variables that **can be overridden** by user-defined variables in playbooks. Lower priority than vars.
- **meta:** Contains metadata for the role, like author, support details. and dependencies on other roles.

A structured way to organize tasks, handlers, etc.

```
roles/  
  webserver/  
    tasks/  
      main.yml  
    handlers/  
      main.yml  
    templates/  
    vars/
```

Handlers

- A special task that runs only when notified by another task (commonly used for service restarts).

A special task triggered by a notify.

```
- name: restart apache
  service:
    name: apache2
    state: restarted
```

```
- name: Update Apache config
  template:
    src: apache.conf.j2
    dest: /etc/apache2/apache2.conf
  notify: restart apache
```

Variables

- Values that can be used to customize tasks and templates.

Used to customize tasks.

```
apache_port: 8080
```

```
- name: Use variable
  hosts: webservers
  tasks:
    - name: Show port
      debug:
        msg: "Apache will run on port {{ apache_port }}"
```

Facts

- Information gathered about hosts (e.g., OS, IP address) using the setup module

Automatically gathered info about hosts.

```
- name: Show OS info
  hosts: all
  tasks:
    - name: Print OS
      debug:
        var: ansible_facts['os_family']
```

Folder structure

- Ansible expects/recommends the following folder structure.
- Ansible knows where to look for files/templates/variables without explicitly referencing the folder.

```
production          # inventory file for production servers
staging            # inventory file for staging environment

group_vars/
  group1.yml      # here we assign variables to particular groups
  group2.yml

host_vars/
  hostname1.yml  # here we assign variables to particular systems
  hostname2.yml

library/           # if any custom modules, put them here (optional)
module_utils/      # if any custom module_utils to support modules, put them here (optional)
filter_plugins/    # if any custom filter plugins, put them here (optional)

site.yml          # master playbook
webservers.yml   # playbook for webserver tier
dbservers.yml     # playbook for dbserver tier

roles/
  common/
    tasks/
      main.yml      # this hierarchy represents a "role"
    handlers/
      main.yml      # <-- tasks file can include smaller files if warranted
    templates/
      ntp.conf.j2   # <-- handlers file
    files/
      bar.txt       # <-- files for use with the template resource
      foo.sh        # <-- files for use with the copy resource
    vars/
      main.yml      # <-- script files for use with the script resource
    defaults/
      main.yml      # <-- variables associated with this role
    meta/
      main.yml      # <-- default lower priority variables for this role
    library/
      module_utils/ # <-- role dependencies
      filter_plugins/ # roles can also include custom modules
    lookup_plugins/ # roles can also include custom module_utils
      # or other types of plugins, like lookup in this case

  webtier/
  monitoring/
  fooapp/         # same kind of structure as "common" was above, done for the webtier role
  # ""
  # ""
```

Collections

- **Collections** are **reusable** collection of ansible tasks, roles, plugins and modules.
- Collections are used for distributing ansible content.
- They can be stored in git and installed using the '*ansible-galaxy*' utility.

Command Line Basics

ansible:

- The base command for running Ansible tasks in an ad-hoc manner (i.e., not using playbooks).
- Useful for executing single tasks on one or more managed nodes.

ansible-galaxy:

- A tool to manage Ansible roles and collections.
- Allows you to create, share, and use community-developed roles and collections.

ansible-playbook:

- Used for running Ansible playbooks.

Variables

To utilize a variable in the playbook, use the following syntax:

- “{{ variable_name }}”
- The variable needs to be within brackets but it can contain other strings or multiple variables such as:
 - “*The value of the variable is {{ variable_name }}*”
- In case of dictionaries to address the keys there are two ways:
 - {{ dict_variable[‘level_1_key’][‘level_2_key’] }}
 - {{ dict_variable.level_1_key.level_2_key }}
- For lists and strings you can use indexing. As with python indexes start with 0. You can also use index ranges with “:” in ranges last index is excluded:
 - “*First element of the list is: {{ list[0] }}*”
 - “*First two elements of the list is: {{ list[0:2] }}*”
 - “*Last element of the list is: {{ list[-1] }}*”
 - “*Or something custom: {{ list[3:-1] }}*”

Filters

- Filters allow you to manipulate data in ansible.
- The format for adding a filter is as follows:
 - “{{ variable | filter() | filter() }}”
- The filter can do such actions as [upper(), lower(), count()]
- Most commonly used in CRP is the zfill filter for zero padding.
For example:

- “{{ (1 + slotID) |zfill(2) }}”
- *The zfill filter is custom and not available in all ansible installations.*
- *For generic zfill functionality you can also use:*
 - {{ '%02d' | format(slotID +1) }}

Variable Priority

- In Ansible you can have variables of the **same name** in **multiple locations**.
- In that case there is a **priority order** that defines which value is used:

Variable Priority Lowest to Highest

1. command line values (for example, -u my_user, these are not variables)
2. role defaults (defined in role/defaults/main.yml) [1](#)
3. inventory file or script group vars [2](#)
4. inventory group_vars/all [3](#)
5. playbook group_vars/all [3](#)
6. inventory group_vars/* [3](#)
7. playbook group_vars/* [3](#)
8. inventory file or script host vars [2](#)
9. inventory host_vars/* [3](#)
10. playbook host_vars/* [3](#)
11. host facts / cached set_facts [4](#)
12. play vars
13. play vars_prompt
14. play vars_files
15. role vars (defined in role/vars/main.yml)
16. block vars (only for tasks in block)
17. task vars (only for the task)
18. include_vars
19. set_facts / registered vars
20. role (and include_role) params
21. include params
22. extra vars (for example, -e "user=my_user") (always win precedence)

Jinja2 Template Example

index.html.j2

```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to {{ ansible_hostname }}</title>
</head>
<body>
    <h1>Hello from {{ ansible_hostname }} on port {{ install_nginx_port }}</h1>
    
    <p>This page is served by Ansible-Managed Nginx.</p>
</body>
</html>
```

Jinja2 Template Example2

```
ServerName {{ server_name }}
Listen {{ http_port }}
DocumentRoot "{{ document_root }}"

<Directory "{{ document_root }}">
    AllowOverride All
    Require all granted
</Directory>

{% if use_ssl %}
SSLEngine on
SSLCertificateFile {{ ssl_cert_file }}
SSLCertificateKeyFile {{ ssl_key_file }}
{% endif %}
```

Jinja2 notes

- For loop index addressing, there are two variables:
 - `loop.index` <- starts with 1
 - `loop.index0` <- starts with 0
- Any variables defined are only within the specific control loop. But there is a workaround using namespaces:

```
{% set ns = namespace() %}  
  {% set ns.defaultGw4set=false %}
```

- This allows you to use checks to break the loops using the `ns.defaultGw4set` variable.
- If you just set variable value without the namespace, it may not give the desired result.
- Best practise is to keep complex logic outside the template.
- For file naming use `file.ext.j2` though not necessary

Useful Commands

- `ansible-inventory - list` <- shows you the inventory.
- `ansible-galaxy collection list` <- shows installed collections.
- `ansible-doc -t filter -l` <- lists the loaded plugins.
- `ansible-doc -t filter <plugin name>` <- shows the plugin information and examples.

Ansible Documentation

For all ansible documentation use the link

- <https://docs.ansible.com/ansible/latest/index.html>

LAB03 - Ansible

- Task 1 – Install Ansible
- Task 2 – Git as a code repository
- Task 3 – Basic Ansible command and Inventory
- Task 4 – Ansible configuration file
- Task 5 – Install requirements
- Task 6 – Ansible playbooks
- Task 7 – Improve your playbook
- Task 8 – Copy files and Templates
- Task 9 – Creating a Simple Role
- Task 10 – Convert nginx and IIS tasks into roles
- Task 11 – Commit and push your changes back into the git origin
- Task 12 – (EXTRA) Add role for the Windows fileserver

Cyber Range components Details

GameNet provisioning – vLab Manager (vLM)

GAMENET provisioning challenges

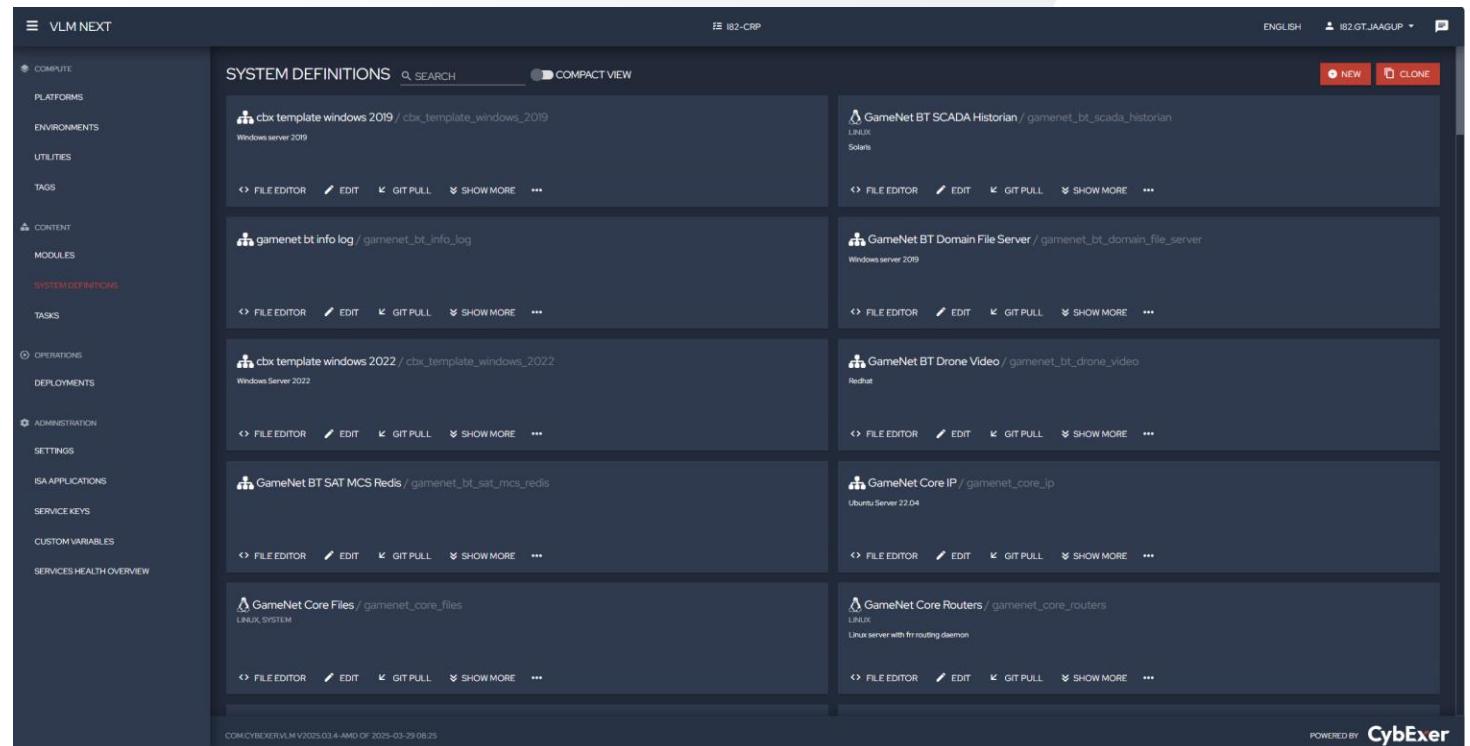
- Cyber exercises usually involve multiple teams or students.
- All teams or students must have the equal content and setup
- Each team or student requires a **private copy** of a GameNet with **unique** customizations (for example, unique network identities).
- Average GameNet contains tens of virtual machines.
- Therefore deployment, customization and post-exercise cleanups of these VMs can be very time consuming and error prone.

GAMENET provisioning options

- Successful GameNet provisioning requires **automation**.
- Along with a physical resource consolidation, automation is the key factor to use **virtualized** environments for cyber labs.
- In virtualized labs GameNet content can be provided as a set of **VM templates** and cloned using hypervisor features.
- **Customization** information can be included with each VM template.

vLab Manager (vLM)

- Your cyber lab is provided with specialized GameNet provisioning tool called **vLab Manager (vLM)**
- vLM was originally designed and built for **Estonian Defense Forces Cyber Range**.
- It has been widely used for **NATO CCDCOE** cyber exercises including the World's Largest live-fire exercise **Locked Shields**



vLM-Next

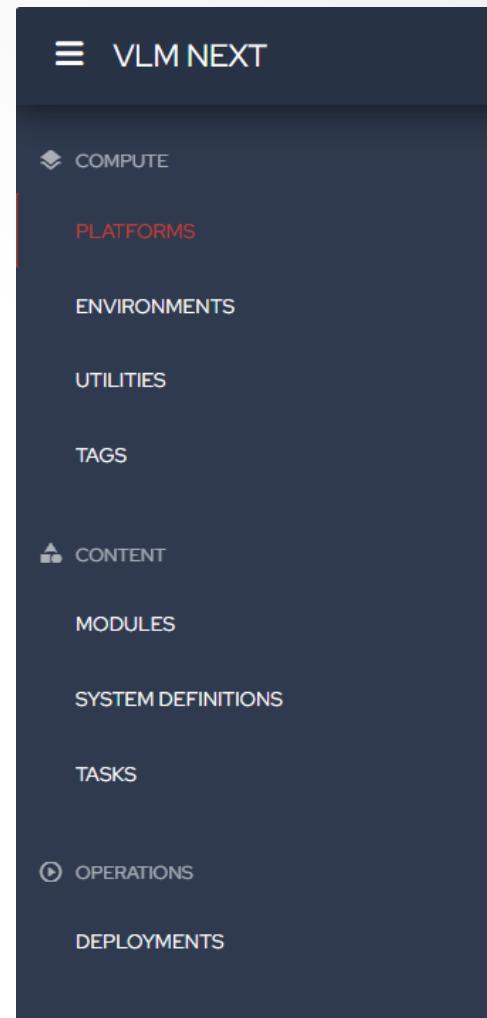
- vLM-Next is the next generation of cybexers cyber-range automation platform.
- For automation it **utilizes ansible**, allowing outside engineers and developers quick onboarding experience.
- The ansible experience is guided, **optimized for cyber-range use case** meaning there are some additional limits and features present.
- The goal for vLM-Next is to allow for creation and management of cyber-range exercises, trainings and other content, including deployment, tasks, attacks, injects and any other parts necessary.
- The vLM-Next will also provide tools for managing the hosts and the exercise environment

vLM

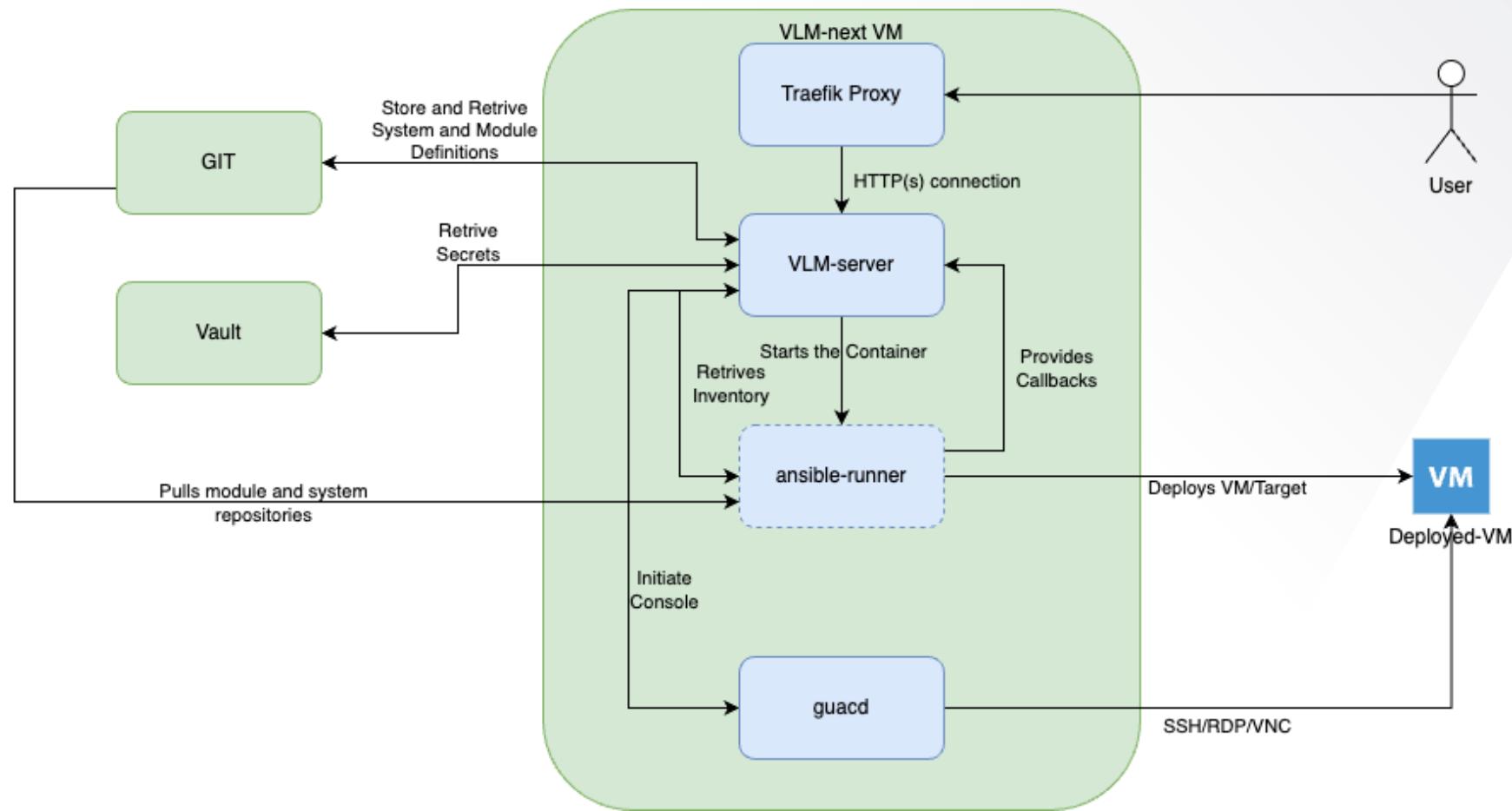
Core Concepts

Core Terms

- Platform
- Environment
- Slot
- System Definition
- Module
- Deployment
- Utilities

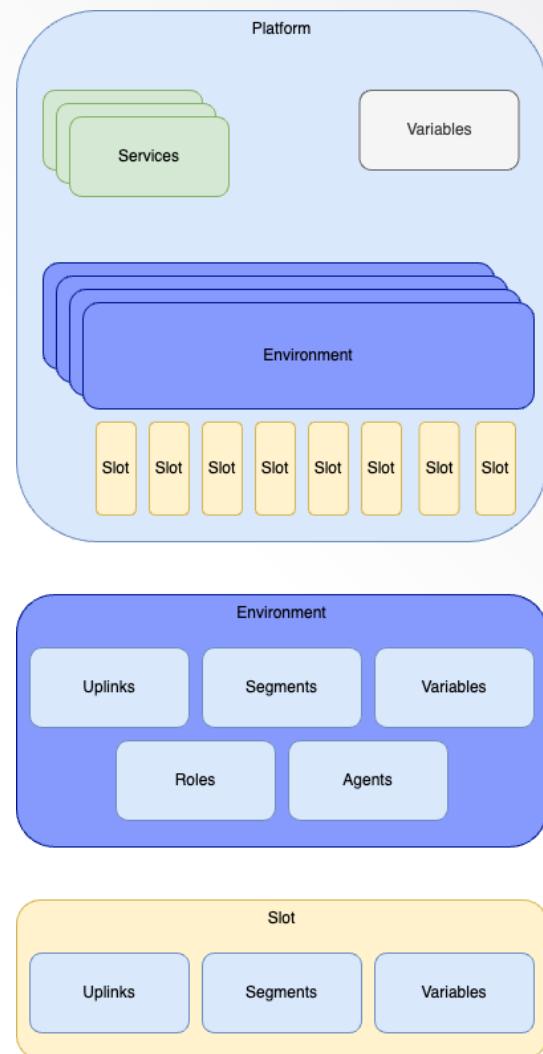


vLM-Next infrastructure



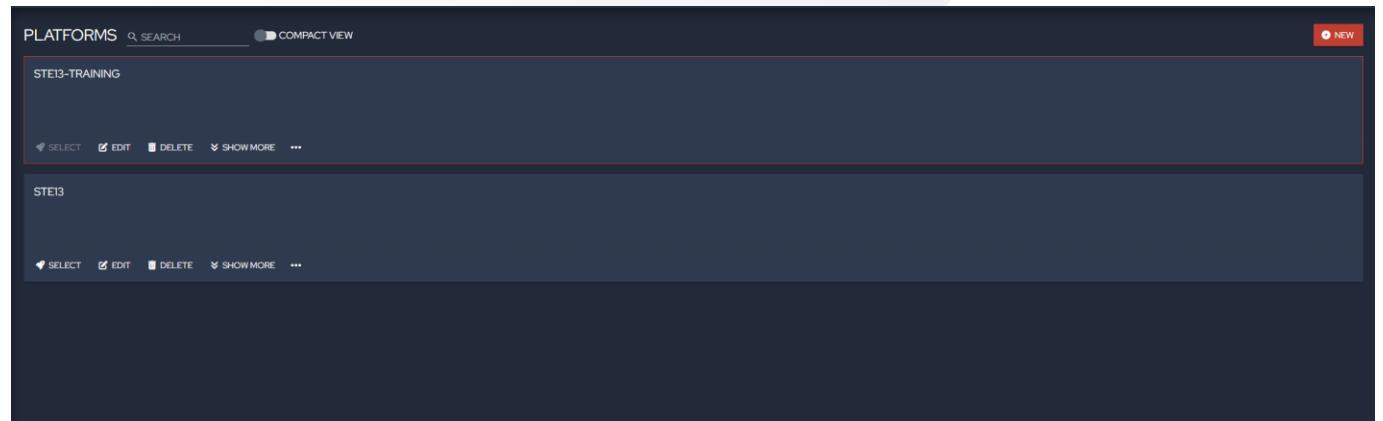
Platforms, Environments and Slots

- vLM-Next is built up in a hierarchical manner.
- **Platform** is the representation of a cyber range.
- **Environment** contains the resources needed to deploy for a single module (exercise/training/CTF)
- **Slots** contain the resources to deploy resources for a single participant/team within an environment



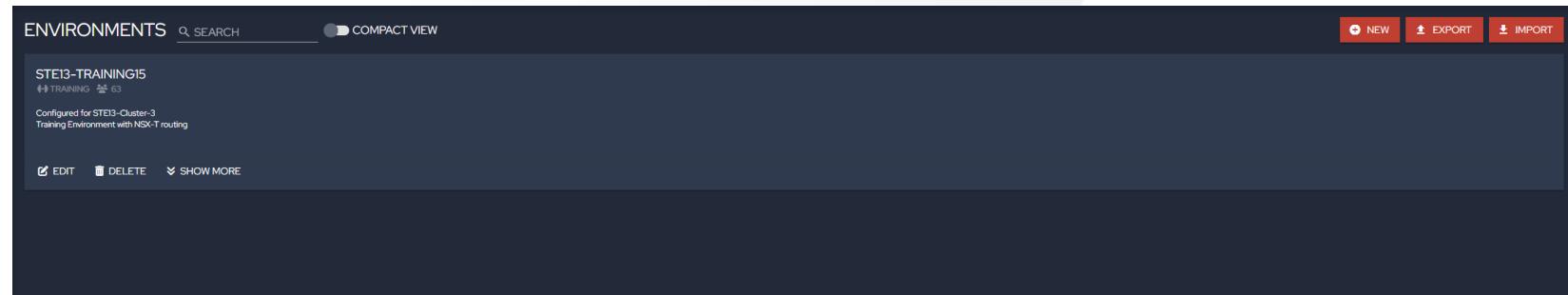
Platforms

- Platforms contain the following:
 - General information and description
 - Services such as git, vsphere, vault, portal, isa and others.
 - Platform level variables
 - Network types
 - Docker configuration (can be either local or remote)
- Platform has an identifier that can be called in playbooks:
 - `cid` and `crp_id`



Environments

- Environment contain:
 - General information including type and slot count
 - Uplinks
 - Segments
 - Environment and Slot level variables
- Environments of the same type have a unique environment identifier:
 - eid and env_id



Slots

- Slots are not directly defined but created within the environment definition.
- Slot resources are defined when setting any environment resources as “dedicated”
- Slots can be identified by unique identifier:
 - sid and slot_id

EDIT ENVIRONMENT
ENVIRONMENTS / STE13-TRAINING15

GENERAL UPLINKS SEGMENTS VARIABLES SLOT VARIABLES

NEW SEGMENT

ste13-training[...]-cmn-net[...]/SHARED
COUNT: 10 / IPv4:10.1[...].0/24 / IPv6:fd[...]:d0[...]:c[...]:[...]:/64

ste13-training[...]-slot[...]-net[...]/DEDICATED
COUNT: 4 / IPv4:10.1[...].0/24 / IPv6:fd[...]:d0[...]:c[...]:[...]:/64

SELECTED SEGMENT
ste13-training15-slot001-net01 / SLOT 1 / segment_id 1

segment_id 1

- ste13-training15-slot001-net01 / SLOT 1 ✓
- ste13-training15-slot002-net01 / SLOT 2
- ste13-training15-slot003-net01 / SLOT 3
- ste13-training15-slot004-net01 / SLOT 4
- ste13-training15-slot005-net01 / SLOT 5

DESCRIPTION

SAVE DISCARD CHANGES NETWORKING

COM.CYBEXER.VLM V2025.04.3-AMD OF 2025-04-27 15:27

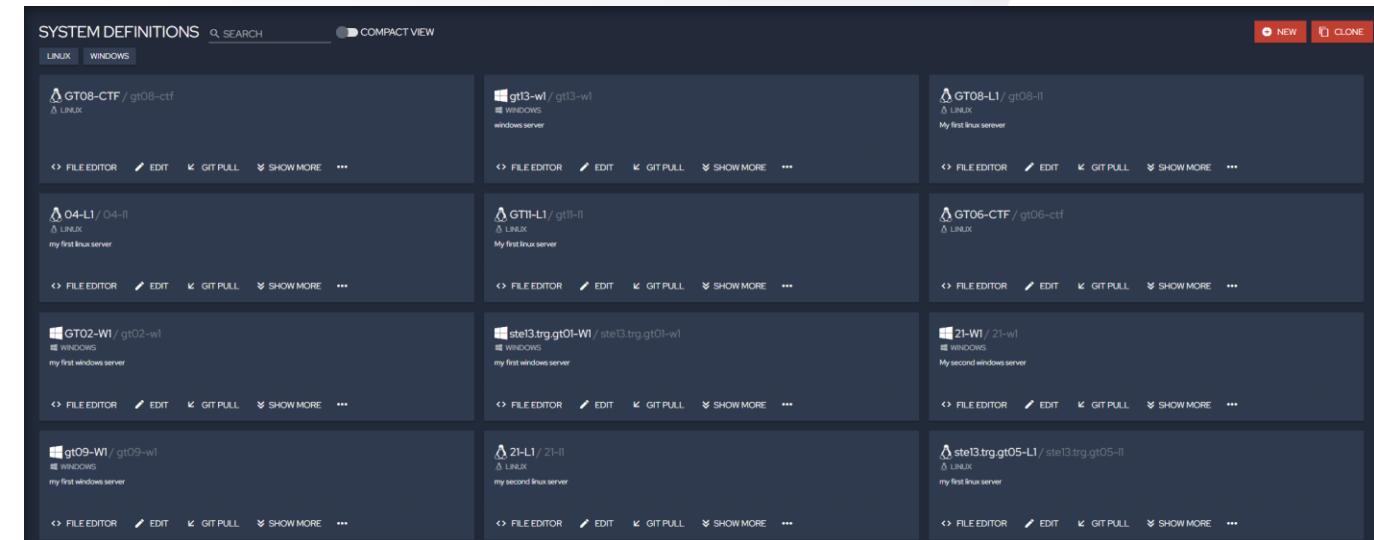
POWERED BY **CybExer**

Shared vs Dedicated resources

- Shared resources are things that usually operate at the environment level and are common across the module such as core routers, central file servers, adversary networks etc.
- Dedicated resources are used within a single slot. Such as targets and blue team networks.

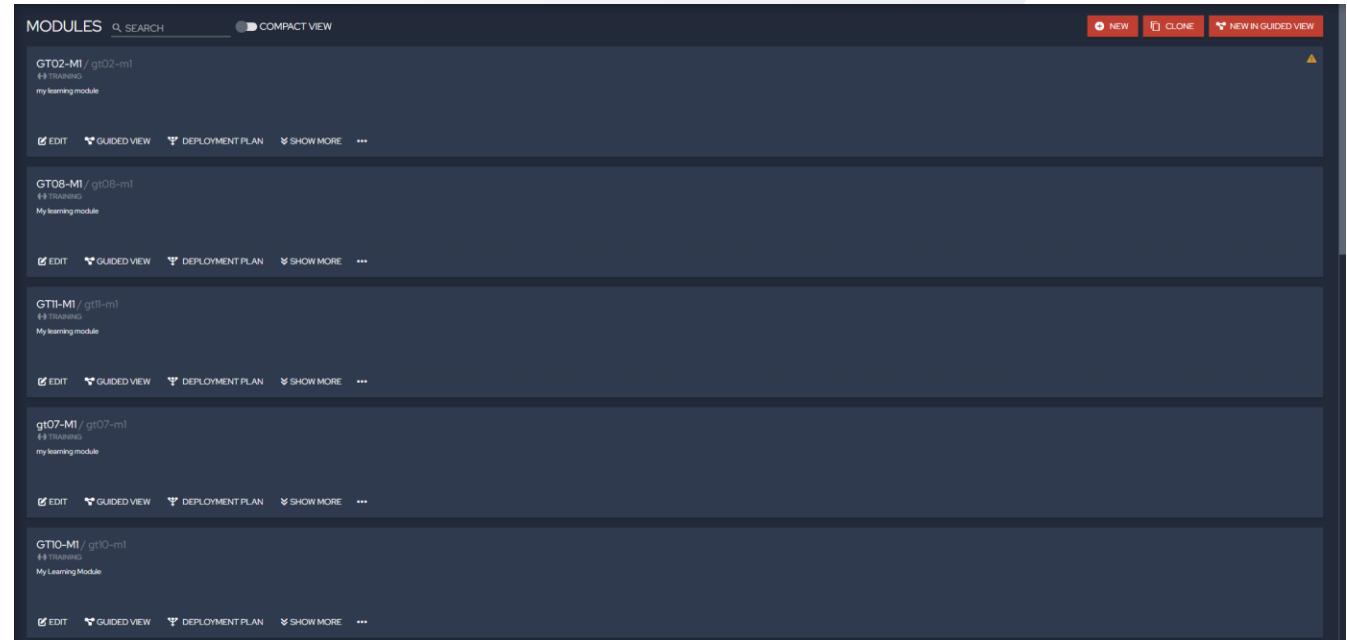
System Definitions

- System definitions, as the name says defines the system that can be used within the cyber range.
- The definition includes the metadata, ansible playbooks and task information associated with the system.



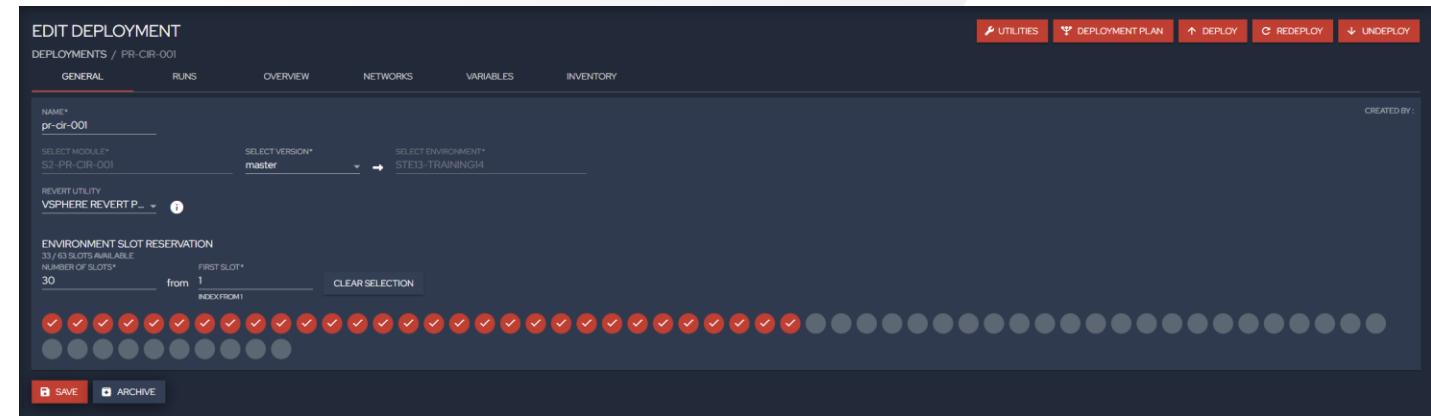
Modules

- Modules represent a template of an exercise, training, CTF or simulation environment.
- This includes the `isa.json` if needed.
- In the module you define the networks, systems and tasks in an abstract manner.
- No exact physical resources are referenced within a module



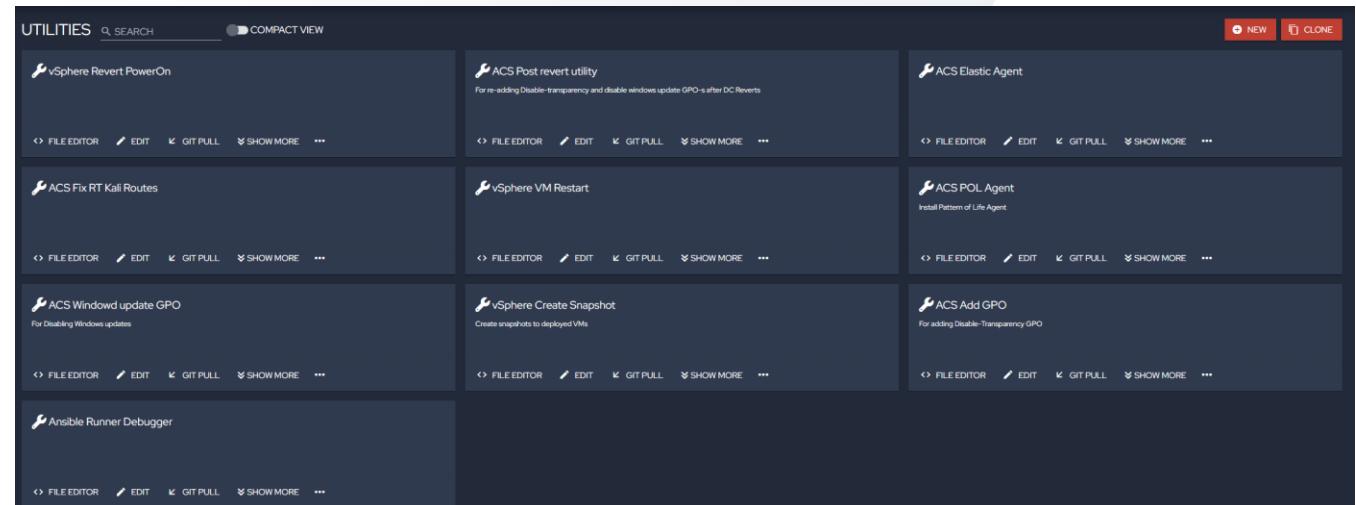
Deployments

- Deployment represents a single instance of a module that is deployed for a specific exercise.
- Example you have the CSIRT module, and you create a deployment called “Customer-X-CSIRT”.
- When creating a deployment, you map the module to a suitable environment.
- Once the deployment is created you can create the necessary resources with deployment runs that execute the ansible playbooks.



Utilities

- Utilities represent common tasks against the platform.
- Example utilities are:
 - Open / Close the GameNet
 - Power on / Power off targets
 - Revert targets to snapshot
 - Do some configuration, like preparing resource items in vSphere etc.



vLM

Defining Environments

Environments

- Environments, represent a set of resources needed to host a module.
- Environments and slots can be created in bulk using patterns and variables such as {{ env_id }} and {{ slot_id }}

Segments

- Segments represent the networks along with IP ranges available to use within the environment.

segment
SELECTED SEGMENT
TEMPLATE

COUNT*
4

NETWORK NAME ([]) *
i1-training[('02d' | format(envID + 1))]-slot[('03d' | format(slotID + 1))]-net[('02d' | format(segmentID + 1))]

NETWORK TYPE*
DEDICATED / DEDICATED

DESCRIPTION

■ NETWORKING

IP V4
IP TEMPLATE ([]) *
10.[(slotID + 1)].[(32 + envID * 4 + segmentID)].

NETMASK TEMPLATE*
255.255.255.0

IP SUFFIX ([])
24

GATEWAY TEMPLATE ([]) *
10.[(slotID + 1)].[(32 + envID * 4 + segmentID)].1

IP V6
IP TEMPLATE ([]) *
fd01:c[('02d' | format(envID + 1))]:[(slotID + 1)]:[(48 + segmentID)]:

IP SUFFIX ([])
64

GATEWAY TEMPLATE ([]) *
fd01:c[('02d' | format(envID + 1))]:[(slotID + 1)]:[(48 + segmentID)]:1

Uplinks

- Uplinks are similar to segments with the exception of IP addressing.
- Uplinks have a defined static IPs that are externally dictated.
- Uplinks are intended to be used to define connections from the environment to the CRP infra.

The screenshot shows a configuration interface for an 'uplink' under a 'SELECTED UPLINK TEMPLATE'. The 'COUNT' is set to 1. The 'NETWORK NAME' is 'IT-game-edge-training01'. The 'NETWORK TYPE' is 'SHARED / SHARED'. The 'DESCRIPTION' field is empty. The 'NETWORKING' section includes fields for 'IP V4' (IP TEMPLATE: 10.101.3.130, NETMASK TEMPLATE: 255.255.255.252), 'IP SUFFIX' (empty), and 'GATEWAY TEMPLATE' (10.101.3.129). The 'IP V6' section includes fields for 'IP TEMPLATE' (Fd01:10:101:3:128::130), 'IP SUFFIX' (empty), and 'GATEWAY TEMPLATE' (Fd01:10:101:3:128::129). The 'OTHER' section shows 'ROUTING PROTOCOL' set to 'STATIC'. At the bottom are buttons for 'DELETE', 'RECREATE FROM TEMPLATE', and 'SHOW LESS'.

Variables

- Variables tab, present in platforms, environments, slots, system definitions and modules. Allow you to define variables to be used in the ansible playbooks and other parts of the system.
- The variables as in ansible have a priority order, where higher priority takes precedence in case a variable with the same name is defined in multiple levels.

Variable Name	Description	Value (JSON)	Action
cluster	cluster	"STE13-Cluster-3"	EXPAND DELETE
folder	folder	"STE13-MODELS/STE13-TRAINING({ training_id_2d })"	EXPAND DELETE
resource_pool	resource_pool	"STE13-MODELS-C3/STE13-TRAINING({ training_id_2d })"	EXPAND DELETE
training_id	training_id	11	EXPAND DELETE
training_id_2d	training_id_2d	"11"	EXPAND DELETE

Variables priority

The variables in vLM-Next have priority as follow form highest priority to lowest:

- Host variables in module (host variable)
- System definition variables (host variable)
- Slot variables (group variable)
- Module and Environment variables (group variable)
- Platform variables (all_vars variable)

vLM

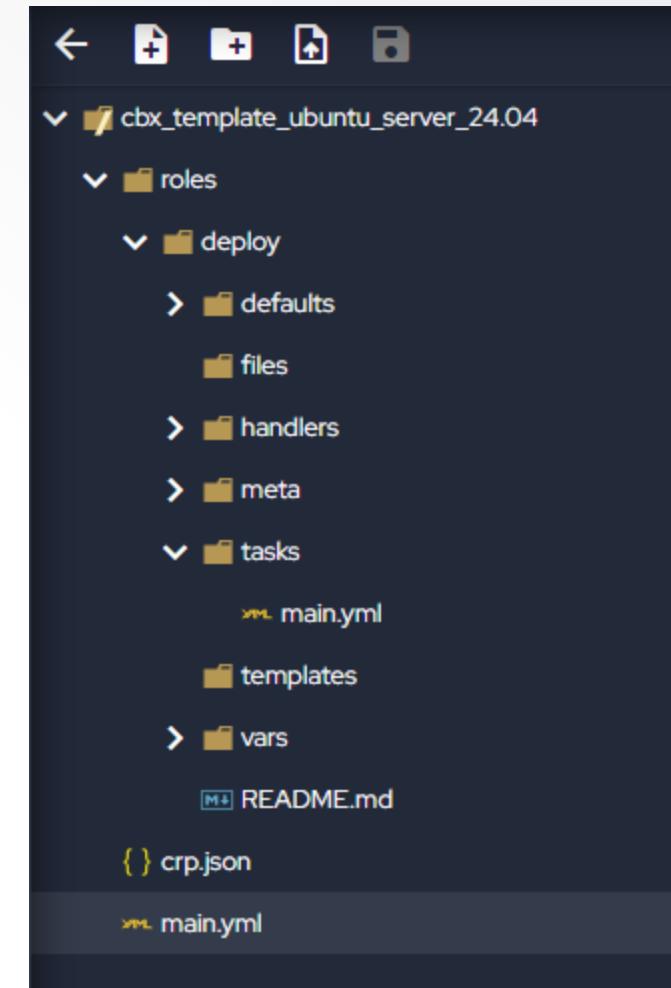
System Definitions

System Definition

- System definition is a set of files and resources used to create and manage the hosts/targets/VM-s
- It consists of a git repository containing the associated configuration files and playbooks and the golden_image VM that is used as the base for cloning.
- The system definitions are versioned using git version control.

System Definition Layout

- The system definition should follow structure similar to an ansible role with two exceptions:
 - **crp.json** <- file that contains the system definition metadata for vlm
 - **main.yml** <- ansible playbook that is executed during deployment runs.



crp.json

- Holds the metadata for vLM-Next to operate.
- Holds the information regarding:
 - General metadata such as Id-s
 - Ansible-runner image to use
 - Main playbook to execute
 - VM hardware config (cpu, memory, nics, hdd_size)
 - Console configuration
 - Variables that can be overwritten in module level

crp.json

```
{ } crp.json x
1  {
2      "name" : "cbx template ubuntu server 24.04",
3      "cloneable" : true,
4      "description" : "Ubuntu Server 24.04",
5      "nicCount" : 1,
6      "dockerImageName" : "cybexer/ansible-runner",
7      "playbookFilePath" : "main.yml",
8      "variables" : [ ],
9      "tags" : [ "LINUX" ],
10     "cpu" : 2,
11     "memory" : 2048,
12     "storage" : 25,
13     "consoleConfigs" : [ ],
14     "goldenImage" : "ste13_cbx_template_ubuntu_server_24.04",
15     "networkSetupMethod" : "netplan",
16     "deployAPI" : "",
17     "ansibleUser" : "root",
18     "ansibleHost" : "{{ connection_ip | default(omit) }}",
19     "ansiblePassword" : "Cool2Pass",
20     "ansibleShellType" : "sh",
21     "ansibleBecome" : true,
22     "ansibleBecomeUser" : "root",
23     "ansibleBecomeMethod" : "sudo",
24     "transportProtocol" : "ssh",
25     "tasks" : [ ],
26     "type" : "SYSTEM",
27     "version" : "v2024.09.3",
28     "id" : "0b0f9a2ca8265039c1542d4b"
29 }
```

AI Assistance

- vLM-Next integrates with ChatGPT to provide assistance with creating the playbooks.
- ChatGPT is context-aware: when given access to a playbook file, it automatically understands its context and can propose edits by generating a diff that you can review and accept.

The screenshot shows the vLM-Next interface with a playbook analysis and AI-generated edits. On the left, there's a tree view of a playbook structure with files like 'main.yml', 'stop_defender.ps1', 'AccessData_FTK_Imager_4.7.1.exe', and 'foobar2000.exe'. The main pane displays the 'main.yml' file content:

```
0    -> name: Install foobar.2000
1    win_shell: C:\Users\administrator\foobar_2000.exe /S
2    args:
3    executable: cmd
4
5    # name: Copy executable-bit-scanner
6    # win_copy:
7    #   src: executable-bit-scanner.exe
8    #   dest: C:\Users\administrator\executable-bit-scanner.exe
9
10   # name: Install executable-bit-scanner
11   # win_shell: C:\Users\administrator\executable-bit-scanner.exe /S
12   # args:
13   # executable: cmd
14
15   # name: Copy qsearchsetup
16   # win_copy:
17   #   src: qsearchsetup.exe
18   #   dest: C:\Users\administrator\qsearchsetup.exe
19
20   # name: Install qsearchsetup
21   # win_shell: C:\Users\administrator\qsearchsetup.exe /S
22   # args:
23   # executable: cmd
24
25   # name: Copy DAEMON_Tools_Lite
26   # win_copy:
27   #   src: DTools.exe
28   #   dest: C:\Users\administrator\DAEMON_Tools_Lite
29
30   # name: Install DAEMON_Tools_Lite
31   # win_shell: C:\Users\administrator\DAEMON_Tools_Lite /S
32   # args:
33   # executable: cmd
34
35   # name: Copy AccessData_FTK_Imager
36   # win_copy:
37   #   src: AccessData_FTK_Imager_4.7.1.exe
38   #   dest: C:\Users\administrator\AccessData_FTK_Imager_4.7.1.exe
39
40   # name: Install AccessData_FTK_Imager
41   # win_shell: C:\Users\administrator\AccessData_FTK_Imager_4.7.1.exe /S
42   # args:
43   # executable: cmd
44
45   # name: Create Volume Shadow Copy using vsadmin
46   # win_shell: "vsadmin create shadow /for=C:"
47   # args:
48   # executable: cmd
49
50   # name: Copy Wireshark-4.2.2-x64.exe
51   # win_copy:
52   #   src: Wireshark-4.2.2-x64.exe
53   #   dest: C:\Users\administrator\Wireshark-4.2.2-x64.exe
54
55   # name: Create Volume Shadow Copy using vsadmin
56   # win_shell: "vsadmin create shadow /for=C:"
57   # args:
58   # executable: cmd
59
60   # name: UnLockexe
61   # win_shell: C:\Windows\system32\unlockexe.exe
62
63   # name: USFormatToolSetup.exe
64   # win_shell: C:\Windows\system32\USFormatToolSetup.exe
65
66   # name: Wireshark-4.2.2-x64.exe
67
68
69
70
```

To the right, there's a panel titled 'GPT 4.1' with a message from ChatGPT:

What can you tell me about this playbook

Certainly! Here's an analysis of the playbook at "roles/soft3/tasks/main.yaml":

Overview

This is an *Ansible role task file* for Windows systems. It automates the deployment of several software tools and utilities, as well as the creation of a volume shadow copy. The playbook uses Ansible modules like `win_copy` and `win_shell` to perform its tasks.

--

Step-by-step Breakdown

1. **Copy and Install foobar2000**
- **Copy:** `foobar_2000.exe` is copied to the Administrator's home directory.
- **Install:** The installer is run silently ('/S'), using the Windows command shell.

2. **(Commented Out) Copy and Install executable-bit-scanner**
- These steps are commented out, so they are not executed. If enabled, they would copy and install another executable.

3. **Copy and Install qsearchsetup**
- **Copy:** `qsearchsetup.exe` is copied to the Administrator's home directory.
- **Install:** The installer is run silently.

4. **(Commented Out) DAEMON Tools Lite**
- Copy and install steps for DAEMON Tools Lite are commented out.

5. **Copy and Install AccessData FTK Imager**
- **Copy:** `AccessData_FTK_Imager_4.7.1.exe` is copied to the Administrator's home directory.
- **Install:** The installer is run silently.

6. **Copy Wireshark Installer**
- **Copy:** `Wireshark-4.2.2-x64.exe` is copied to the Administrator's home directory.
- **Note:** There is no install step for Wireshark in this file; it is only copied.

7. **Create Volume Shadow Copy**
- Uses `vsadmin` to create a shadow copy of the C drive. This is often used for backup or forensic purposes.

--

Notable Points

Type your message

222 TOKENS USED OF 104767 AVAILABLE TOKENS

POWERED BY CybExer

Cybexer ansible roles

- **cbx.core.provisioner** <- deploys the vSphere virtual machine and sets default networking.
- **cbx.core.snapshot** <- takes vm snapshot from shut-down state
- **cbx.common.docker_engine** <- adds Docker service to your Ubuntu VM.

Adding additional roles/collection

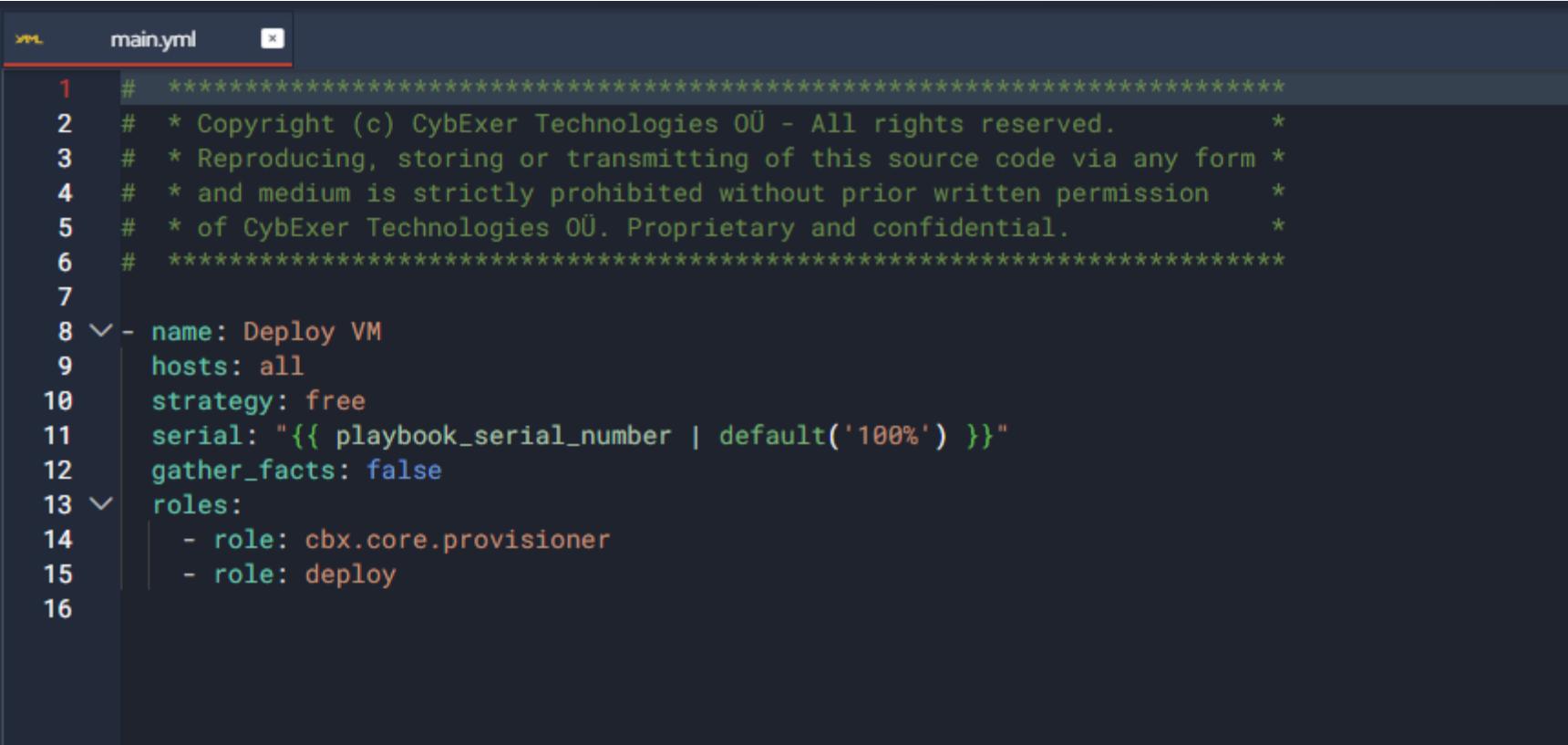
Most common roles are included in the ansible runner, but in some cases additional roles/collections are needed. For that there are two options:

- Update the ansible-runner to include the new roles (preferred)
- Create a custom ansible-runner container image (may be useful if the collection is large in size and not commonly used). For this option it is recommended to use the existing runner as base and use custom image tag.

Creating custom roles and collection

- For vSphere related roles, use the existing vSphere collection.
- For OS or other common roles, new collection should be created.

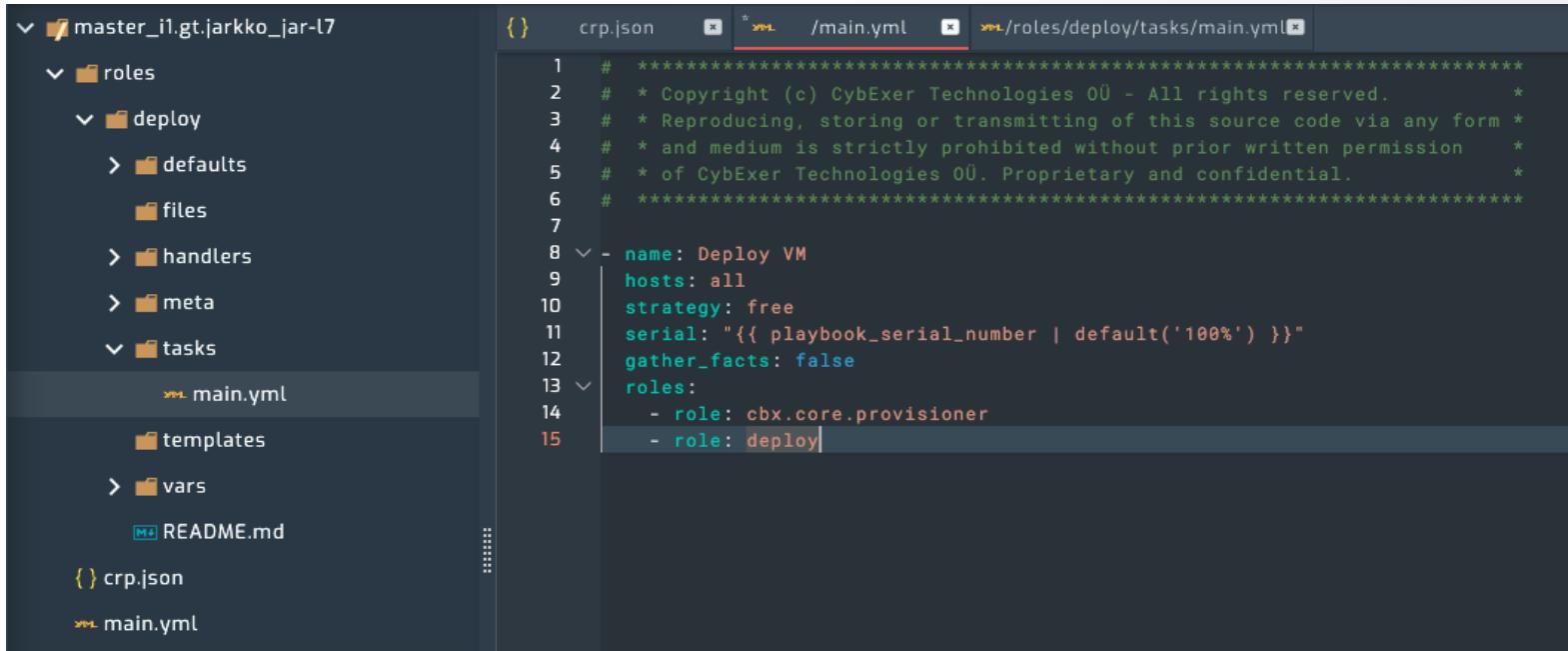
Main.yml requirements/suggestions



```
main.yml
1 # ****
2 # * Copyright (c) CybExer Technologies OÜ - All rights reserved. *
3 # * Reproducing, storing or transmitting of this source code via any form *
4 # * and medium is strictly prohibited without prior written permission *
5 # * of CybExer Technologies OÜ. Proprietary and confidential. *
6 # ****
7
8 - name: Deploy VM
9   hosts: all
10  strategy: free
11  serial: "{{ playbook_serial_number | default('100%') }}"
12  gather_facts: false
13  roles:
14    - role: cbx.core.provisioner
15    - role: deploy
16
```

Main.yml requirements/suggestions

- Add tasks for customization. Best to put them in separate role.



The screenshot shows a terminal window with a file tree on the left and a code editor on the right. The file tree shows a directory structure with 'master_i1.gt.jarkko_jar-l7' containing 'roles', 'crp.json', and 'main.yml'. The 'roles' directory contains 'defaults', 'files', 'handlers', 'meta', and 'tasks'. The 'tasks' directory contains 'main.yml' and 'templates'. The 'vars' directory contains 'README.md'. The code editor displays the content of 'main.yml':

```
# ****
# * Copyright (c) CybExer Technologies OÜ - All rights reserved.
# * Reproducing, storing or transmitting of this source code via any form *
# * and medium is strictly prohibited without prior written permission *
# * of CybExer Technologies OÜ. Proprietary and confidential.
#
# ****
-
  name: Deploy VM
  hosts: all
  strategy: free
  serial: "{{ playbook_serial_number | default('100%') }}"
  gather_facts: false
  roles:
    - role: cbx.core.provisioner
    - role: deploy
```

Helpful tasks and hints regarding playbooks

- To check if the system is dedicated or shared use the slotID check:
 - when: slot_id is defined
- To wait for system to become reachable:
 - name: Waiting for system to become reachable
wait_for_connection:
 - timeout: 240
- To perform service checks:
 - name: wait for service to come up
uri:
 - url: "https://{{ inventory_hostname }}.{{ domain }}"
 - status_code: 200
 - validate_certs: no
 - register: result
 - until: result.status == 200
 - retries: 300
 - delay: 1

Workflow controls

The following variables can/should be used to control the playbook workflows:

- **deploy_mode** <- values [deploy, undeploy, redeploy] Will be set by vlm, variables with the same name will be overwritten.
- **new_deploy** <- Boolean identifies if the VM exists, useful of 1st run only tasks.
- **network_setup_method** <- supports values [netplan, powershell] if not set no network customization on host OS will be done.
- **no_vmtools** <- if set then no vmtools connections will be set in the deploy_vm role.

Workflow controls

- Loops: used with keyword “loop:” (though there are other options) with either a list or any other iterable datatype. By default the looped objects are referred with “{{ item }}” variable.

```
- name: This is my first task
  debug:
    msg: "{{ item }}"
  loop:
    - "This is the 1st message"
    - "This is the 2nd message"
```

- Conditionals: used with keyword “when” and add any logic functions. Note that the variables don’t need to be in the “{{}}” brackets.

```
- name:This is my first task
  debug:
    msg: "this is a test variable"
  when: variable is defined
```

Main.yml requirements/suggestions

- The playbook should correctly handle the deploy/undeploy/redeploy use cases.
- In case of a deploy, remember that the deploys can be run multiple times to update the content, so add ‘when’ checks where needed and add error handling.
- This is usually needed in case some resource is created with cmd.

Secrets and Hashicorp Vault

- *Vault* secures, stores, and tightly controls access to tokens, passwords, certificates, API keys, and other secrets critical in modern computing.
- The ansible-runner is configured to authenticate with the ansible vault server for secrets management. You just need to set the secrets and then add the appropriate lookup to the playbook.

Vlm

Modules

Modules

- Modules represent a complete exercise/training/ctf/dfc with all the resources and parameters needed.
- It is stored in a git repository with the metadata stored in crp.json
- Any system definitions used are added as git submodules to the repository.
- Generally, manually modifying the module/crp.json is not required as the workflow is UI driven but its possible.

Modules

In the general overview you can provide the name and type of the module. Additionally, you can include submodules within this module.

The screenshot shows the 'EDIT MODULE' interface. At the top, there's a navigation bar with 'GUIDED VIEW' and 'DEPLOYMENT PLAN' buttons. Below the navigation, a breadcrumb path shows 'MODULES / S2-PR-CIR-001 / LATEST'. The main area has tabs for 'GENERAL', 'HOSTS', 'GROUPS', 'UPLINKS/SEGMENTS', 'DELRAP', 'ISA', 'TASKS', and 'VARIABLES'. The 'GENERAL' tab is selected. It displays fields for 'MODULE NAME*' (S2-PR-CIR-001), 'REPOSITORY NAME*' (s2-pr-cir-001), and 'CREATED BY'. A large text area for 'DESCRIPTION' is empty. A dropdown for 'ENVIRONMENT TYPE*' shows options: 'TRAINING' (selected), 'CTF', 'EXERCISE', 'TESTING', and 'TRAINING' again at the bottom. On the right side, there's a 'SUBMODULES' section with two panels: 'ADD ALL' (listing 'ACS-GT-TRAINING', 'ACS24-BT', 'ACS24-CORE-ROUTING', 'ACS24-CORE-SERVICES', 'ACS24-RT', 'CLEAN_TEMPLATES', 'GT_TRAINING_24', 'IN-INV-001', 'INFRA', 'PR-CIR-001', and 'SI-IN-INV-001') and 'REMOVE ALL' (empty).

Hosts

Hosts view allows you to add hosts to the module.

In the host adding dialog you can:

- Set the system type and version
- Provide the hostname for the system
- Change the vm hardware config of the system
- Map nics to interfaces and configure the IP addresses
- Overwrite any variables defined in the systems crp.json

< EDIT HOST
MODULES / S2-PR-CIR-001 / LATEST / SYSTEMS / PR-CIR-001_KALI / HOSTS / KALI{{ DOMAIN }}
GENERAL SYSTEM VARIABLES

GENERAL
NAME {{}}*
kali.{{ domain }}
SYSTEM DEFINITION*
PR-CIR-001_KALI / pr-cir-001_kali
VERSION*
master

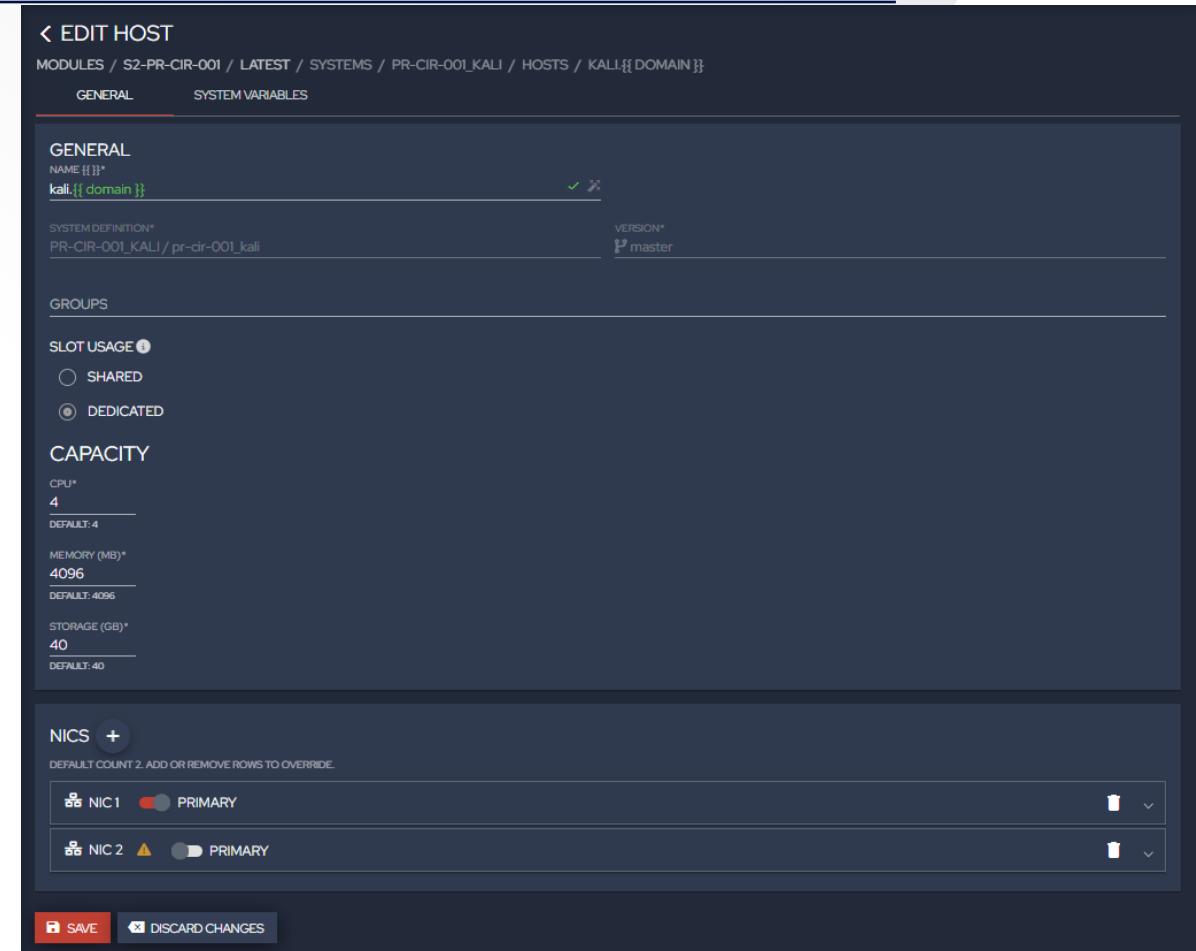
GROUPS

SLOT USAGE ⓘ
 SHARED
 DEDICATED

CAPACITY
CPU*
4
DEFAULT: 4
MEMORY (MB)*
4096
DEFAULT: 4096
STORAGE (GB)*
40
DEFAULT: 40

NICS +
DEFAULT COUNT 2. ADD OR REMOVE ROWS TO OVERRIDE.
NIC 1 PRIMARY
NIC 2 PRIMARY

SAVE **DISCARD CHANGES**



Uplinks/Segments

- Uplinks/Segments section allows you to add network resources to the module.
- This allows you to create named segments such as. DMZ, INT, ect.
- Real mapping to resources will happen during deployment.

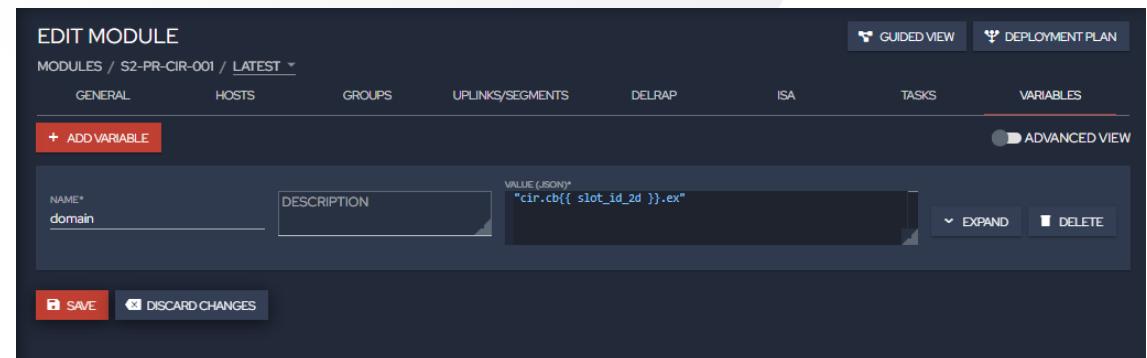
The screenshot shows the 'EDIT MODULE' screen with the 'UPLINKS/SEGMENTS' tab active. There are two entries listed:

Name	Type	Network Type
common1	SEGMENT	SHARED / SHARED
dedicated1	SEGMENT	DEDICATED / DEDICATED

At the bottom, there are 'SAVE' and 'DISCARD CHANGES' buttons.

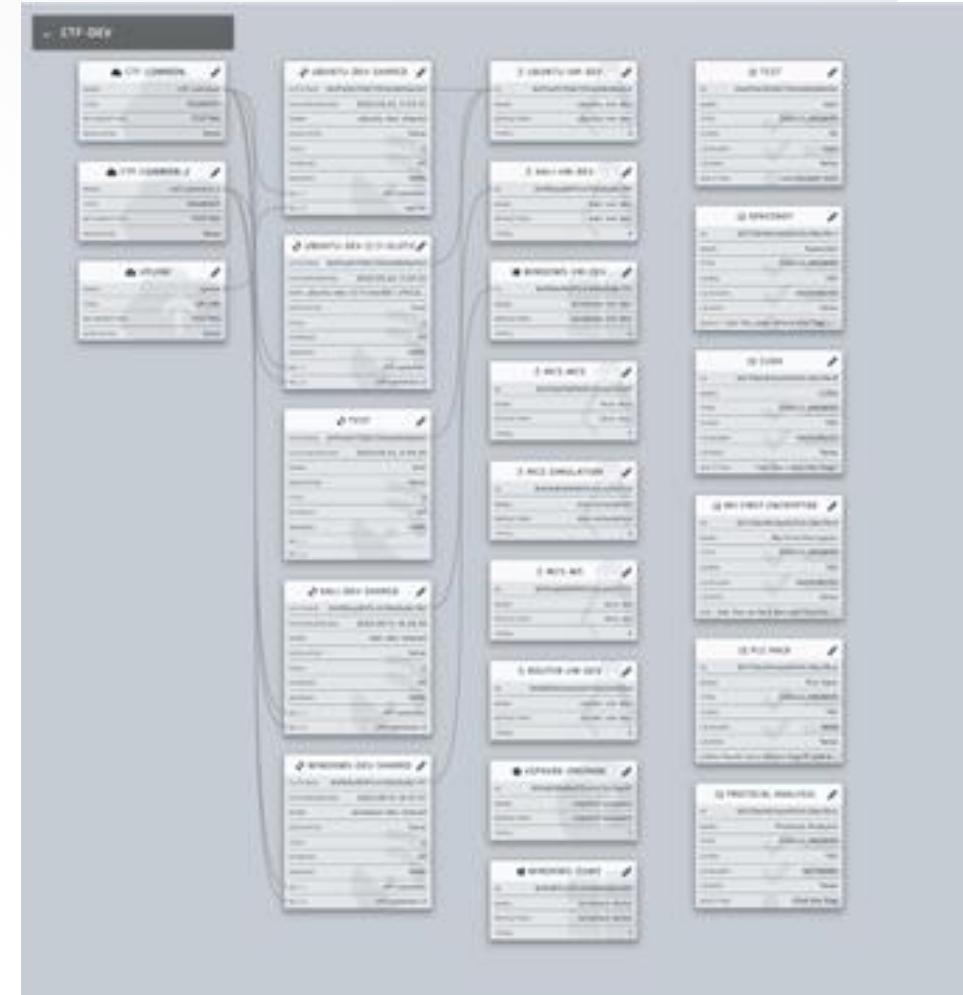
Variables

- Variables allow you to create module level variables that can be referenced in the playbooks.
- Best way to utilize is to reference these variables in the system variables section.
- Example system has variable domain, and you define module variable module_domain and then set domain = {{ module_domain }}



Guided/Canvas view

- The canvas view will be the drag-and-drop interface for designing cyber-range content.
- Currently it is still work in progress.



vLM

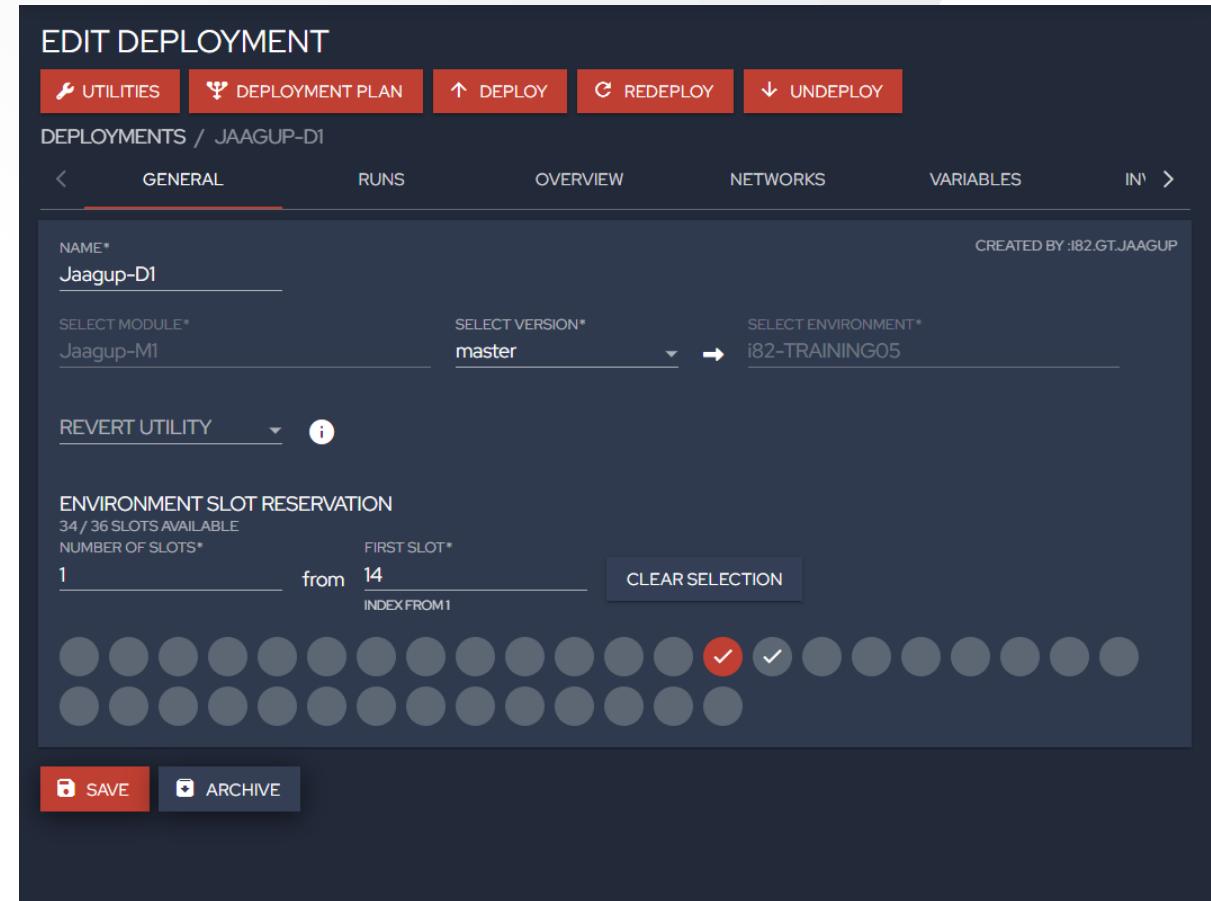
Deployments

Deployments List

DEPLOYMENTS			
NAME	MODULE	ENVIRONMENT	LAST ACTIVITY
PR-CIR-001-DEPLOY	PR-CIR-001	II-TRAINING14-EXTERNAL	14.05.2025 06:05:17
IN-INV-001-DEPLOY	IN-INV-001	II-TRAINING14-EXTERNAL	21.02.2025 04:31:51
SCENARIO5	SCENARIO5	II-TRAINING14-EXTERNAL	24.03.2025 11:26:08
DL_TEST	DL_TEST_M1	II-TRAINING14-EXTERNAL	17.01.2025 10:09:58
DL_OPENVAS_TEST	SCENARIO4	II-TRAINING14-EXTERNAL	
OPENVAS	OPENVAS	OPENVAS	02.12.2024 07:21:18
SCENARIO4	SCENARIO4	II-TRAINING14-EXTERNAL	04.04.2025 07:54:40
TEST-SCEN3	TEST_SCEN3	II-TRAINING14-EXTERNAL	21.09.2024 16:44:05
SCENARIO-3	SCENARIO_3	II-TRAINING14-EXTERNAL	02.10.2024 07:57:06
SCENARIO-02	SCENARIO 2	II-TRAINING14-EXTERNAL	24.04.2025 05:55:49
ITEMS PER PAGE		10 ▾	1 - 10 OF 11 < < > >

General tab

- In the general tab, you can view your deployment details and reserve a number of slots by specifying how many slots you need and which slot number to start from.



Runs tab

- In the Runs tab, you can review past deployments and access their details and logs.

EDIT DEPLOYMENT

DEPLOYMENTS / JAAGUP-DI

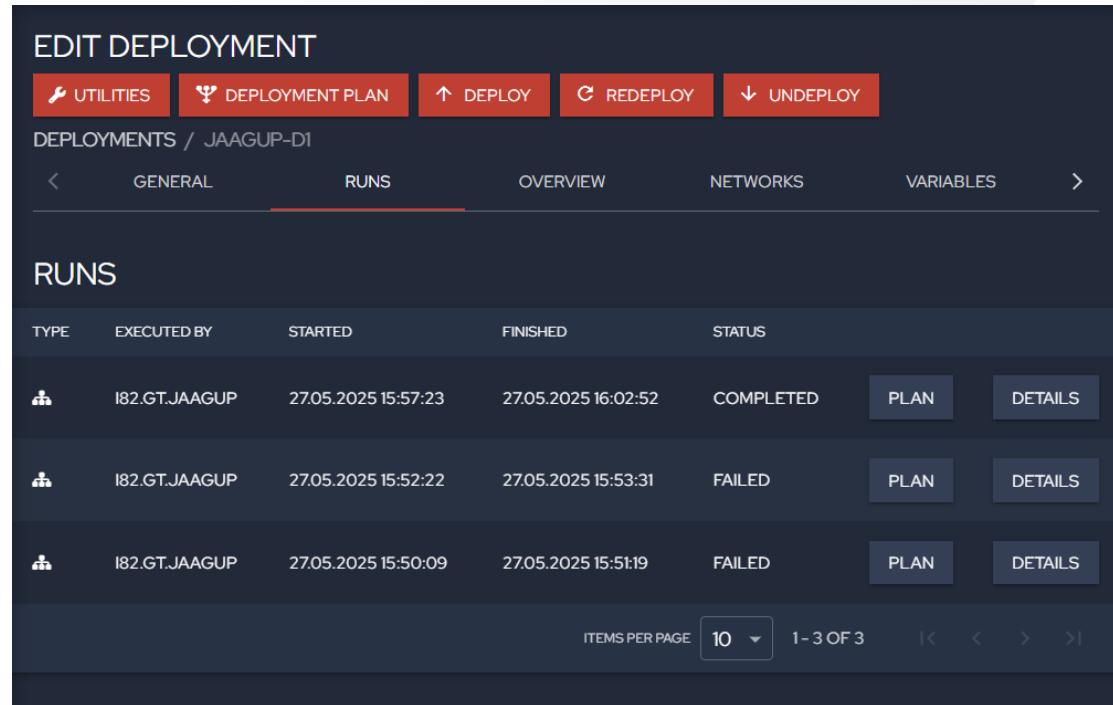
UTILITIES DEPLOYMENT PLAN DEPLOY REDEPLOY UNDEPLOY

GENERAL RUNS OVERVIEW NETWORKS VARIABLES

RUNS

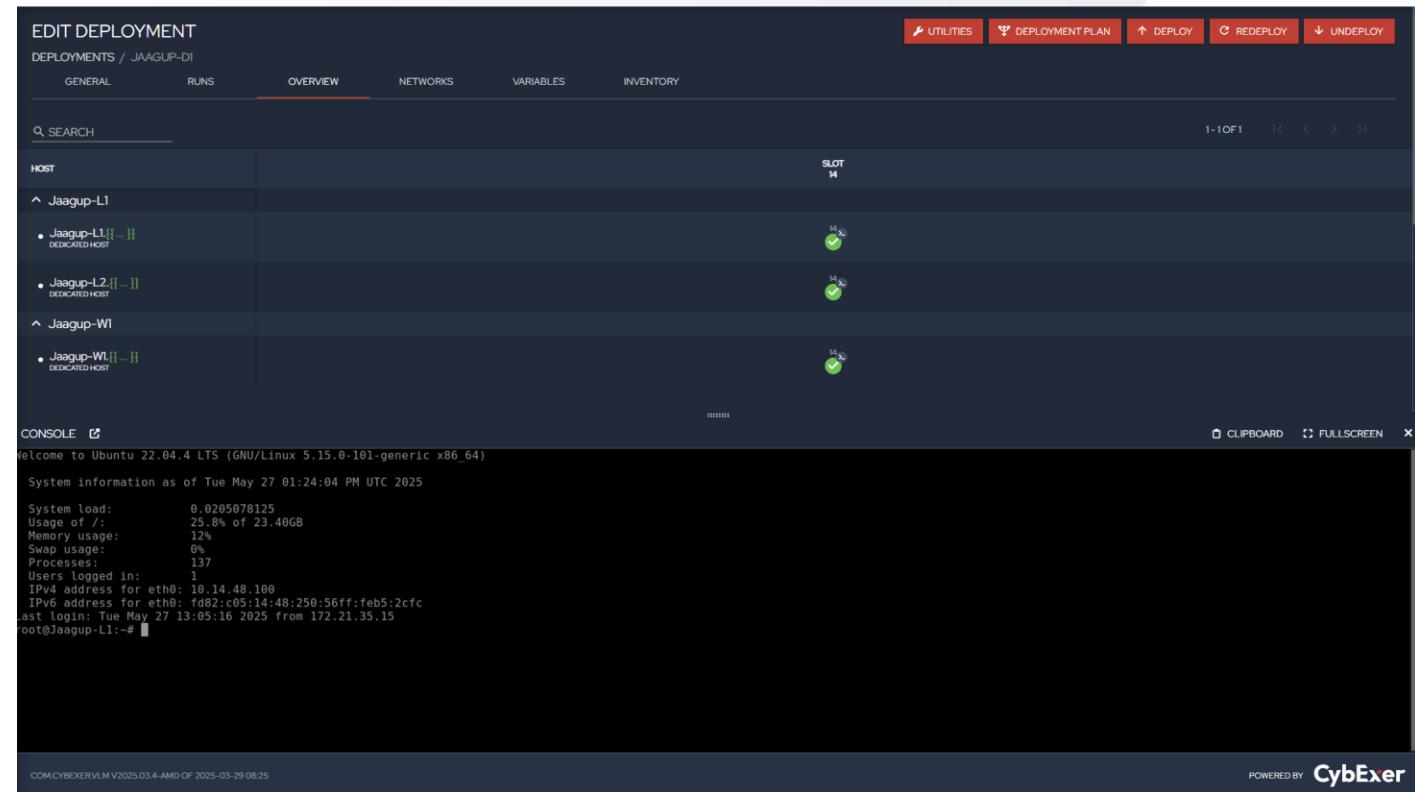
TYPE	EXECUTED BY	STARTED	FINISHED	STATUS	PLAN	DETAILS
人群	I82.GT.JAAGUP	27.05.2025 15:57:23	27.05.2025 16:02:52	COMPLETED	PLAN	DETAILS
人群	I82.GT.JAAGUP	27.05.2025 15:52:22	27.05.2025 15:53:31	FAILED	PLAN	DETAILS
人群	I82.GT.JAAGUP	27.05.2025 15:50:09	27.05.2025 15:51:19	FAILED	PLAN	DETAILS

ITEMS PER PAGE 10 1 - 3 OF 3 < >



Overview tab

- On the Overview tab, you can view all VMs in each slot, check whether they're running or stopped, and open an SSH or RDP console for any VM.



Environment Mapping

- When deploying, the module gets mapped to a specific environment.
- The mapping provides the correct hardware/networking/environment parameters specific to the environment.
- This allows for deployment without the need to know any environment or platform specifics.

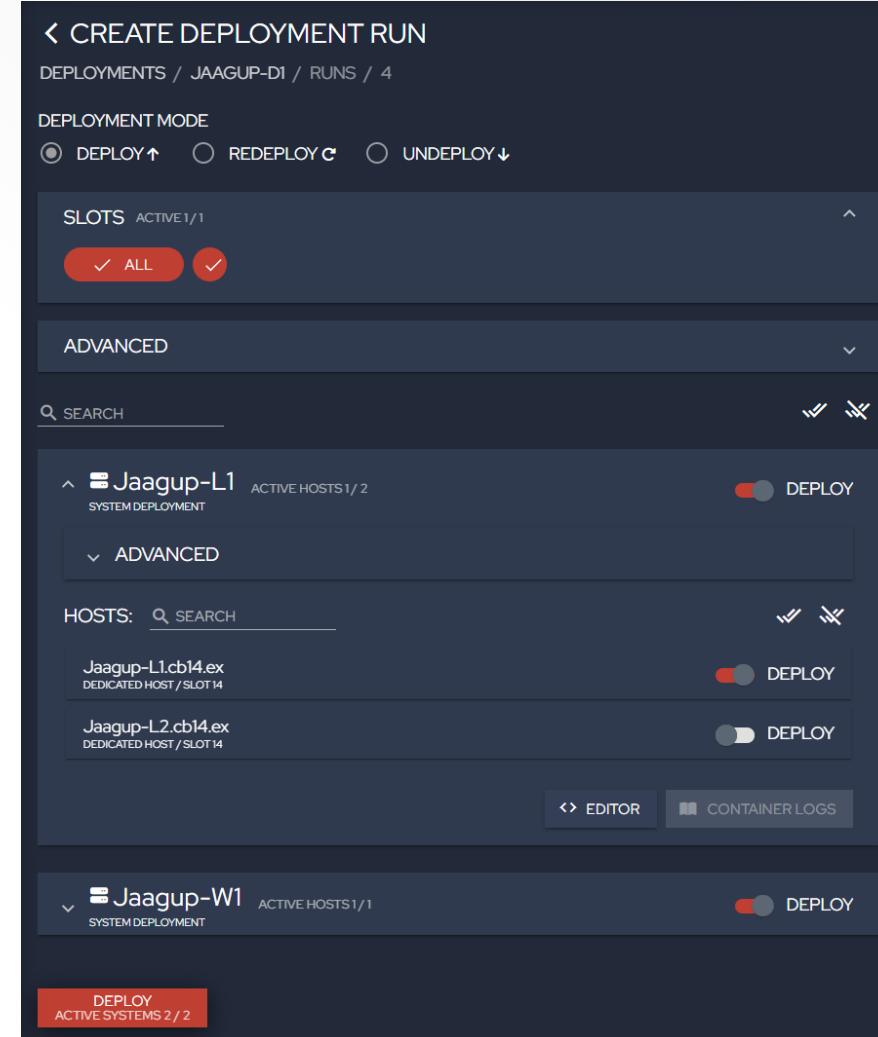
Deployment

- Deployment is a meta object representing a single event for example CBOE.
- The hosts in the deployment are created within deployment runs.
- A deployment run can contain one or more hosts from specified environments/slots.
- A deployment run can be of different types:
 - **Deploy** – Creates the host or updates the configuration if it exists.
 - **Undeploy** – Removes the host.
 - **Redeploy** – Removes and recreates the host.

Deployment Runs

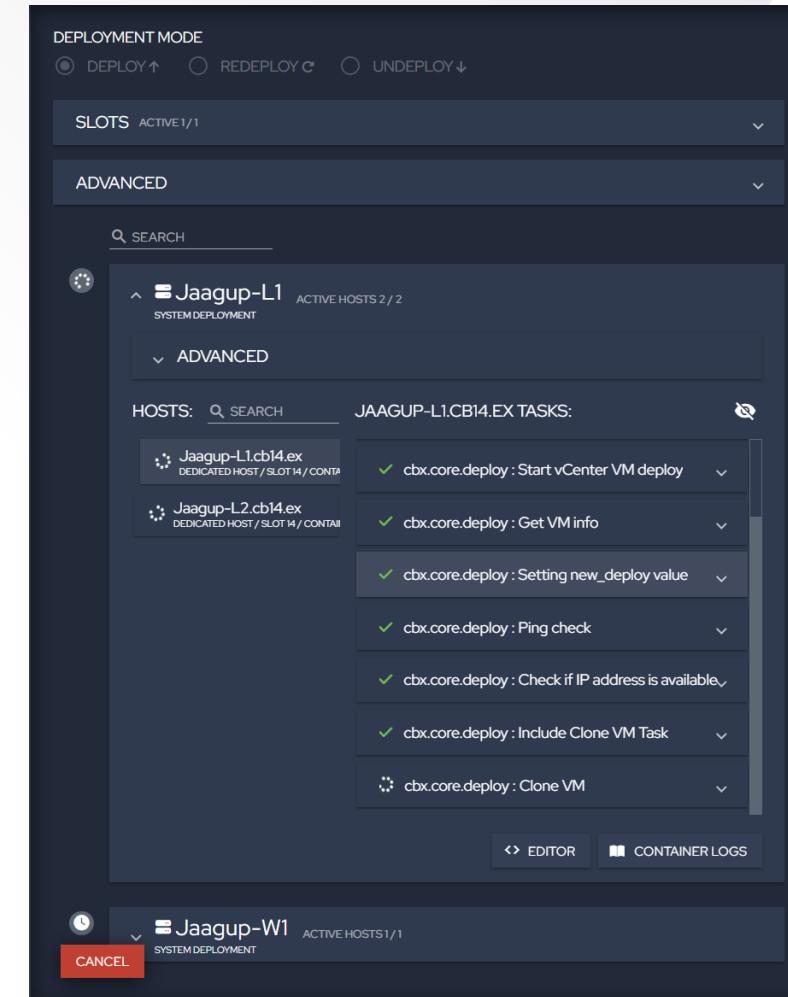
Deployment run allows you to select the following:

- Slots that should be deployed
- Systems that should be included in the deployment run
- Hosts to be included in the deployment run.



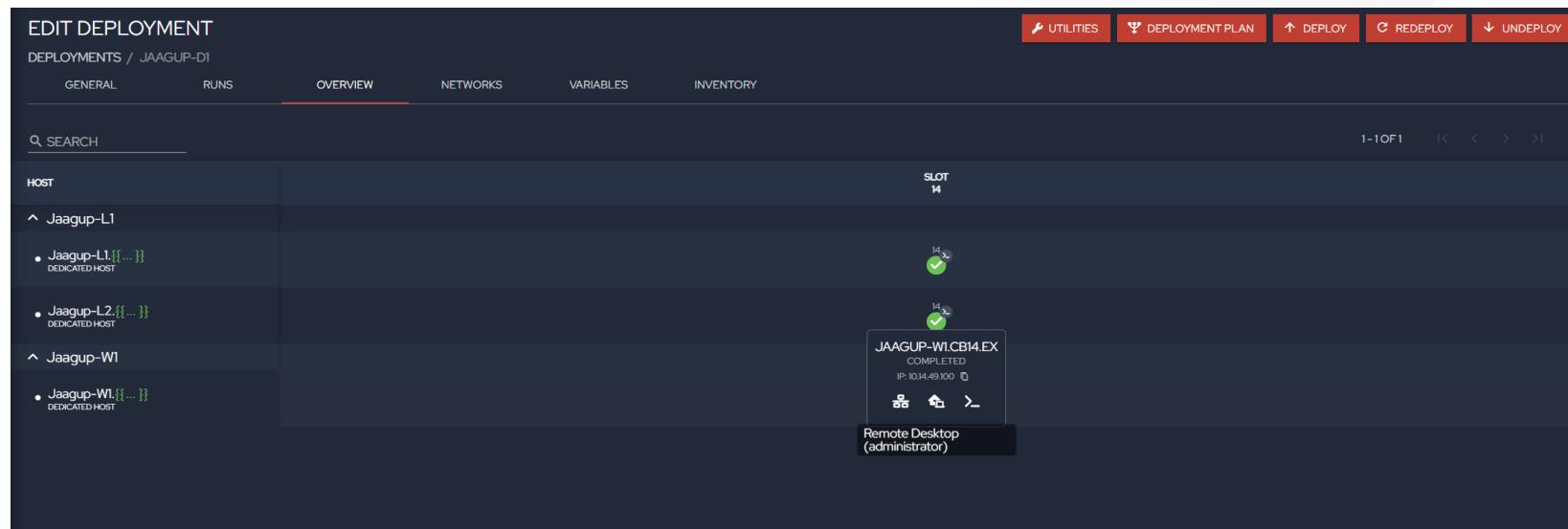
Deployment Run

- Once you start the deployment run, the systems are deployed in sequential order.
- The hosts of the system type are run serially, based on the ansible strategy.
- In the run you can expand the hosts to view the status of ansible tasks.



Console Functionality

- To open a console in the Overview tab, click the VM you want and select either RDP or SSH. A console window will then appear.
- You can also view the machine's IP address here.



Console Functionality

- Connection parameters need to be defined in the system definition.

```
"consoleConfig" : {  
    "username" : "root",  
    "password" : "Cool2Pass",  
    "protoPortPairs" : [ {  
        "protocol" : "SSH",  
        "port" : 22} ] }
```

- The connectionIP defined in the module is used as the connection target.

vSphere Locations

- The vm-s will be placed with the following logic to folders/resource_pools.
- For shared resources:
 - “STE13-WORKLOADS/{{environment_name}}
 - Such as STE13-WORKLOADS/TRAINING15
- For dedicated resources:
 - “STE13-WORKLOADS/{{environment_name}}/SLOT{{1 + slotID|zfill(2)}}”
 - Such as STE13-WORKLOADS/TRAINING15/SLOT01

LAB04

- Task 1 - Create Linux System Definition
- Task 2 - Create Windows System Definition
- Task 3 - Generate vLM learning module
- Task 4 - Generate vLM Deploy item from the Module
- Task 5 - Deploy the vLM Module
- Task 6 - Add services to your Linux target VM
- Task 7 - Add new users to your Windows target VM
- Task 8 - Redeploy your modified targets
- Task 9 - Add more features to your Web servers using AI

Cyber Range components Details

Integrated Scoring and Awareness (ISA)

ISA – Integrated Scoring and Awareness

- **Award-Winning Situational Awareness & Scoring Tool:** ISA provides near real-time visualization and scoring for cybersecurity exercises.
- **Comprehensive Data Capture:** Visualizes ongoing attacks, defense effectiveness, and overall team performance in complex exercises.
- **Enhanced Transparency:** Transforms exercises into transparent, easy-to-follow events for both technical teams and senior management.
- **Advanced Visualization:** Built on a decade of experience, it leverages cutting-edge visualization techniques to offer unmatched situational awareness.
- **Live Scoring:** Displays live team scores, timelines, and individual performance, improving engagement and training impact.

ISA – Integrated Scoring and Awareness

- ISA is the user interface for RT, BT and WT
- RT – attack campaign and RT reports
- BT – exercise tasks (Threat Hunting tasks), Incident reports, Situation reports, Gamenet status, consoles and basic VM operations (revert), scoring data, receive/generate media injects
- WT – awareness data, respond to RT and BT reports, scoring data, generate/receive media injects, control of the exercise

ISA – Modules

There are 5 different types of modules, that represent the type of exercise

- Live-Fire
 - Active Live-Fire
 - Threat Hunting
- CTF
- Hybrid (Live-Fire and CTF combined)
- VERSUS

ISA - Modules

- Some modules are designed for team-based exercise:
 - Live-Fire and Hybrid
- Some for Individual exercise:
 - CTF
- CTF modules can be team based or individual.

ISA – Module timers

- Team based CTF exercise
 - Countdown starts as soon as the module starts.
- Individual CTF exercise
 - Option 1
 - Countdown starts as soon as the module starts.
 - Option 2
 - User starts the mission whenever he is ready.
 - After user starts the mission, countdown timer starts.
- Live-Fire does not have timer

ISA Live-Fire module description

ISA Live-Fire – Targets and Objectives

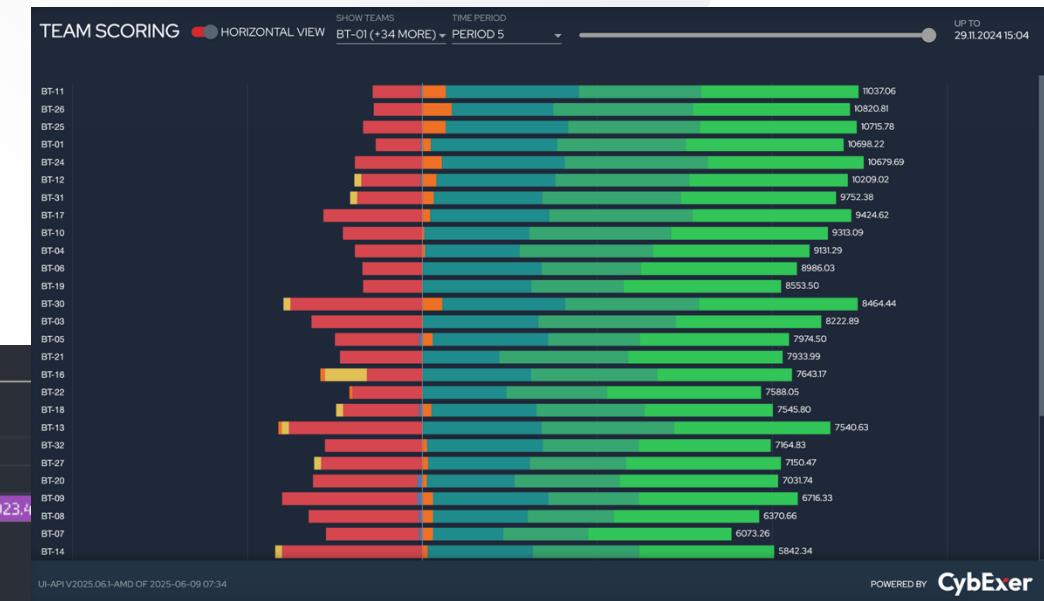
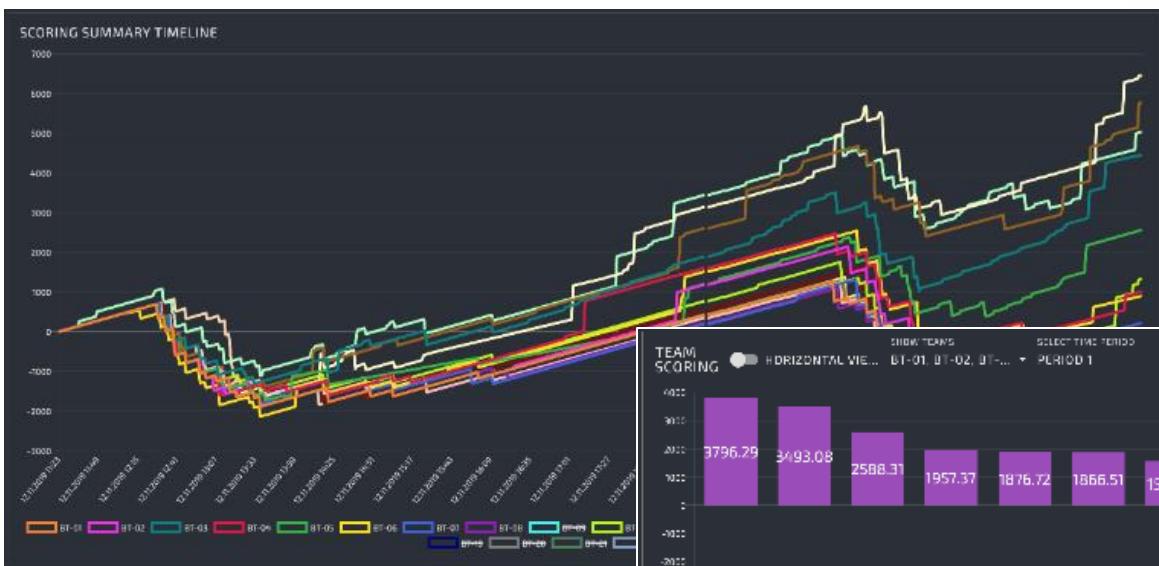
- Exercise consists of Red Team (attackers), Blue Team (defenders) and White Team (judges)
- **Blue team** defends virtual machines with predefined targets (e.g., web services, ICMP)
- The **Red Team's** mission is to breach those targets – such as defacing a website's front page.
- **Objectives** are rolled out in phases; not every objective is active at all times.
- Exercises can also include CTF tasks.

ISA – Live-Fire - Scoring

- Blue Team scores are determined by VM availability, situational awareness, confirmed Red Team attacks, quality of reports, and CTF mission results.
- Availability score is calculated automatically based on VMs availability.
- Red Team successful attacks must be confirmed by the judges.
- Judges evaluate and score the Blue Team's awareness and situation reports.
- Judges may give additional points for any reason they want.
- Mission board task scores are calculated automatically.

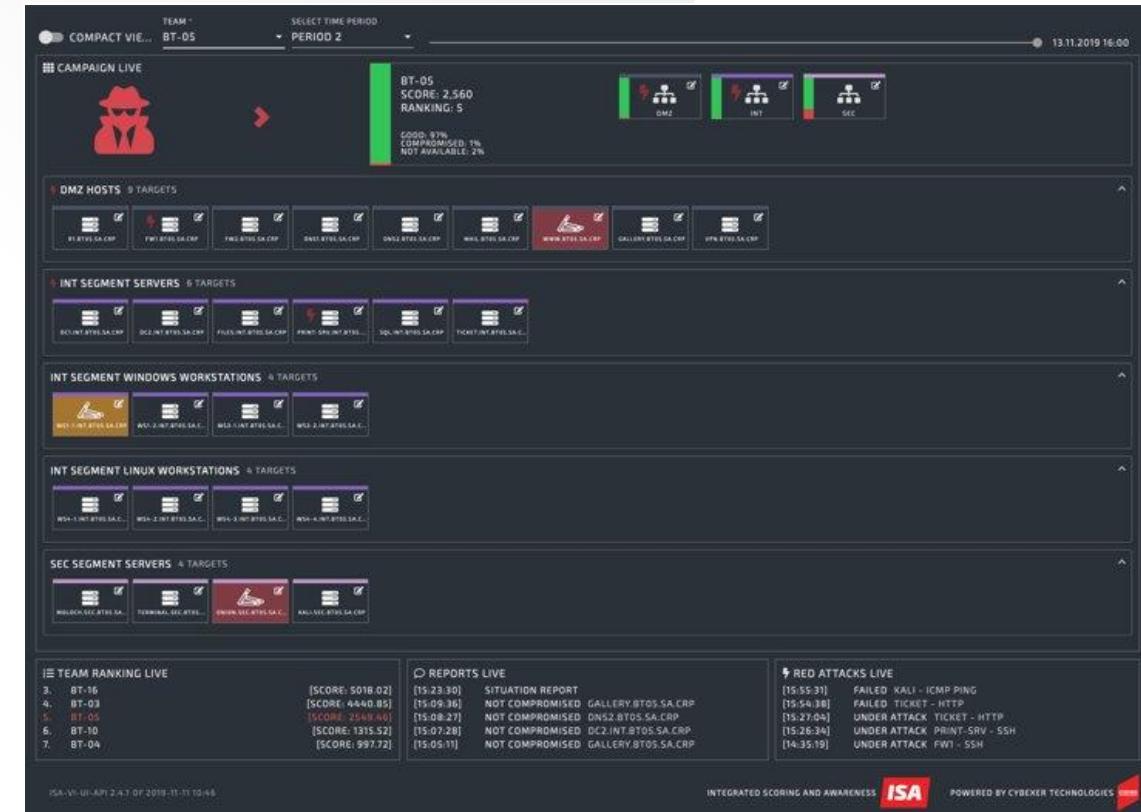
ISA – Live-Fire - Scoreboards

There are multiple views available for monitoring game scores.



ISA – Live-Fire – Awareness boards

There are multiple views available for monitoring game scores.



ISA – Live-Fire – RT views

Select phase and team and assign the attack

ATTACK CAMPAIGN

HIGHLIGHT ASSIGNMENTS BT-18 (+16 MORE) PHASE-2 FILTER BY CATE... OBJECTIVE

CLIENT SIDE NETWORKING WEB

MAIL.GOVXX.ACS.EX - SEND CREDENTIALS PHISHING LINK TO GOV EMPLOYEES.

REPORT ALL ASSIGN ALL

OVERVIEW

MITRE TAGS T1566

DESCRIPTION mail.govXX.acs.ex - Send credentials phishing link to GOV employees.

BT-18	Not Available	NOT COMPROMISED	REPORT	ASSIGN
BT-19	Not Available	NOT COMPROMISED	REPORT	ASSIGN
BT-20	Not Available	NOT COMPROMISED	REPORT	ASSIGN
BT-21	Not Available	NOT COMPROMISED	REPORT	ASSIGN
BT-23	Available	NOT COMPROMISED	REPORT	ASSIGN
BT-24	Available	NOT COMPROMISED	REPORT	ASSIGN
BT-25	Available	NOT COMPROMISED	REPORT	ASSIGN
BT-26	Available	NOT COMPROMISED	REPORT	ASSIGN
BT-27	Available	NOT COMPROMISED	REPORT	ASSIGN
RT-28	Available	NOT COMPROMISED	REPORT	ASSIGN

UI-API V2025.06.1-AMD OF 2025-06-09 07:34

POWERED BY CybExer

ISA – Live-Fire – RT views

After attack, send report, either success or unsucces

REPORT STATUS

OBJECTIVE
15:00 - www01.rndXX.acs.ex - Gain initial access public web server over webshell.php, install the Mythic C2 agent

TEAM
BT-21

REQUEST FLAG or **SUDO REQUEST FLAG**

SUCCESSFUL UNSUCCESSFUL

B I U S </> H₁ H₂ E E E E NORMAL A E = S T_x

1. Target IP/DNS name:
• www01.rndXX.acs.ex (10.X.13.120)

2. Description for access or attack vector used by Red Team:
Red Team gained access to the Blue Team webserver over webshell with 'http://10.1.1.101/webshell.php?cmd' commands.
Mythic C2 Agent was downloaded and executed
User 'www-data' was a member of sudo group, so Red Team had full administrative privileges on the system.

3. Evidence(s):
{@ screenshots @}

OK CANCEL

ISA – Live-Fire – RT views

After WT has given feedback, read the feedback for the report

RED OBJECTIVE REPORT
[22,80]
[View IPs](#)
10.19.13.119
[22,80]

STATUS	SUCCESSFUL
RED MEMBER	STE13.GT30
TIME	16.06.2025 12:34:01
OBJECTIVE	15:00 - www01.rndXX.acs.ex - Gain initial access public web server over webshell.php, install the Mythic C2 agent
TARGET CHECK/GROUP	WWW01 - HOST
CAMPAIGN PHASE	PHASE-1
TEAM	BT-19

FEEDBACK

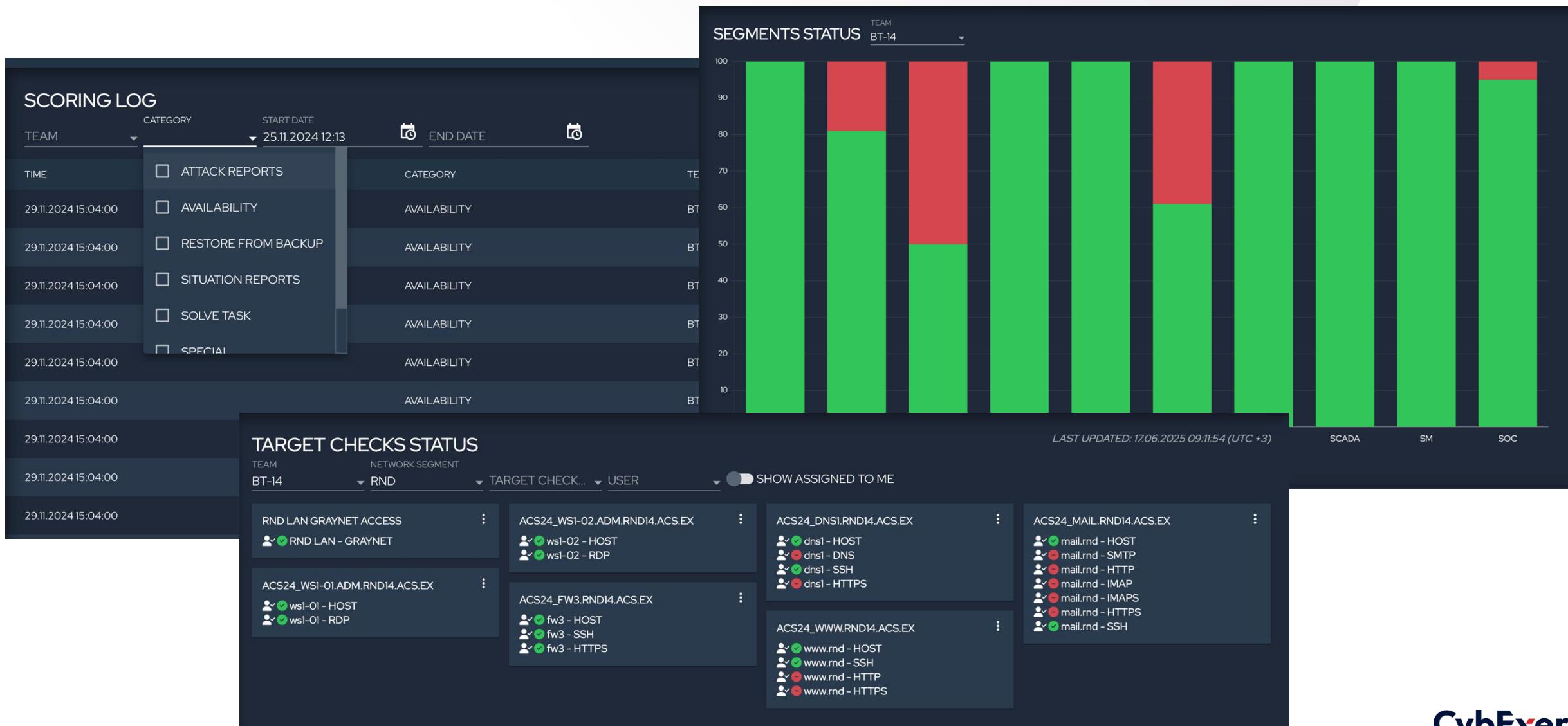
To improve this report, include specific and verifiable evidence:

- A screenshot or output of a command executed via the webshell
- Confirmation of the Mythic agent installation and check-in (e.g., screenshot of the Mythic UI showing the agent)
- Output of `id` or `groups` for the 'www-data' user to confirm sudo privileges
- A command like `hostname` or `ifconfig/ip a` to tie the IP to the hostname www01

This will make the report more credible and conclusive.

 AUTO-ASSESS CANCEL

ISA – Live-Fire – BT views – scoring and awareness



ISA – Live-Fire – BT views – reporting

Send the reports to WT

The image shows three main sections of the CybExer interface:

- INCIDENT REPORT**: A sidebar on the left containing a list of targets. The "TEAM*" dropdown is set to "BT-01". A "FILTER TARGETS" section lists items like "camera - HOST", "camera - VNC", "acs24_consul.mcs.gov01.acs.ex", "consul - HOST", "consul - SSH", "acs24_dc1.lan.gov01.acs.ex", "dc1.lan.gov - HOST" (which is selected), "dc1.lan.gov - AD DS Web", "dc1.lan.gov - DNS", and "dc1.lan.gov - Kerberos/TCP/88". A note at the bottom says "1 TARGET CHECK SELECTED". A red "SUBMIT" button is at the bottom.
- INCIDENTS**: The main content area. It has a "DETAILS" section with fields for "Chronology" (What and when happened?), "Impact" (How has the incident impacted your information systems or daily operations?), and "Source" (What caused the incident?). Each field has a list of options with checkboxes. The "Impact" field includes options like "Availability?", "Integrity?", "Confidentiality?", "No impact at all?", and "I don't know". The "Source" field includes "System failure".
- SITUATION REPORT**: A right-hand panel titled "SITUATION REPORT" with "TEAM*" set to "BT-01". It contains a "MESSAGE *" field with a rich text editor toolbar. Below the toolbar are five sections labeled I. Executive Summary, II. Event Overview, III. Adversary assessment, IV. Impact on operations, and V. Actions Taken and Results. A red "SUBMIT" button is located at the bottom of this panel.

ISA – Live-Fire – BT views – missionboard

Mission Board for LF or TH tasks with AI assistance (chat and AI agent)

The screenshot displays the CybExer Mission Board interface. On the left, a sidebar shows the team selection as "TEAM BT-14". Below it, a "FORENSICS CHALLENGE" section indicates "20% SOLVED". A list of tasks is shown with their respective points, descriptions, and status indicators:

Task	Description	Status
19 pts Initial Access - Open Files	Initial Access - Open Files	32/35 (Green checkmark)
19 pts Initial Access - Identify User	Initial Access - Identify User	30/35 (Green checkmark)
19 pts Behavior Analysis - Task 1	Behavior Analysis - Task 1	23/35 (Orange button)
19 pts Behavior Analysis - Task 2	Behavior Analysis - Task 2	27/35 (Orange button)
50 pts Behavior Analysis - Task 3	Behavior Analysis - Task 3	23/35 (Orange button)
50 pts Behavior Analysis - Task 4	Behavior Analysis - Task 4	23/35 (Orange button)
50 pts Behavior Analysis - Task 5	Behavior Analysis - Task 5	10/35 (Orange button)
50 pts Behavior Analysis - Task 6	Behavior Analysis - Task 6	18/35 (Orange button)
50 pts Behavior Analysis - Task 7	Behavior Analysis - Task 7	14/35 (Orange button)
50 pts Behavior Analysis - Task 8	Behavior Analysis - Task 8	14/35 (Orange button)

On the right, a detailed view of the "ACKNOWLEDGING THE RULES OF ENGAGEMENT" task for Day 2 (10 pts) is shown. The "DESCRIPTION" section contains a text block about the rules of engagement. The "QUESTION" section contains a text box with instructions to acknowledge the rules. The "SUBMISSIONS" tab is selected, showing a single submission from "BT-14" user "BT14.USER06" at "26.11.2024 11:01:02" with the answer "AcknowledgedWeWillNotHardenRND-GOV-MIL" and a score of "10/10" marked with a green checkmark.

UI-API V2025.06.1-AMD OF 2025-06-09 07:34

POWERED BY **CybExer**

ISA – Live-Fire – WT views – assessing reports

All reports assessments have AI assistance

The screenshot displays two reports side-by-side, illustrating AI-assisted assessment features.

Left Panel: RED OBJECTIVE REPORT

- ATTACK REPORTS
- Mythic C2 Agent was downloaded and executed
- User 'www-data' was a member of sudo group, so Red Team had full administrative privileges on the system.
- 3. Evidence(s):

STATUS	SUCCESSFUL
RED MEMBER	STE13.GT04
TIME	17.06.2025 01:27:22
OBJECTIVE	15:00 - www01.rndXX.acs.ex - Gain initial access public web server over
TARGET CHECK/GROUP	WWW01 - HOST
CAMPAIGN PHASE	PHASE-1
TEAM	BT-21

- FEEDBACK

Right Panel: BLUE SITUATION REPORT

BLUE SITUATION REPORT

AUTO-ASSESS STATUS: ASSESSED
GRADE: 9
FEEDBACK

- Clarity and Conciseness:** While the report is detailed, some sections could benefit from more concise language to enhance readability. For example, the Key Events section could be streamlined by summarizing repetitive details.
- Technical Jargon:** The report includes some technical terms and abbreviations (e.g., "SOC," "OSINT," "TTL") that should be spelled out or explained upon first use to ensure clarity for all readers.
- Impact Analysis:** While the impact on operations is discussed, it could be further enhanced by quantifying the potential risks or losses where possible, such as the expected downtime or the number of users affected.
- Adversary Assessment:** The assessment could be strengthened by providing more context or evidence supporting the potential involvement of specific threat actors, even if speculative.
- Future Plans:** The report could benefit from a brief mention of long-term strategies beyond immediate remediation efforts, such as plans for improving overall cybersecurity posture or lessons learned.

GRADE*

REPORT

26.06.2024 16:52:21 81/82 CONFIRMED W.LUSERIZ ASSESSED

AUTO-ASSESS CONFIRM DENY CANCEL

ISA – Live-Fire – WT views – awareness

Campaign live view for Live Fire and TH

TEAM BT-01 TIME PERIOD PERIOD 5 UP TO 29.11.2024 15:04

CAMPAIGN LIVE

BT-01 SCORE: 10698 RANKING: 4
GOOD: 91% COMPROMISED: 9% NOT AVAILABLE: 0%

RND GOV MIL HVAC SCADA
SATELLITE DRONE SM SOC RHQ

69 TARGETS

ACS24_FW3.RND01.A...	ACS24_DNS1.RND01.A...	ACS24_MAIL.RND01.A...	ACS24_WWW.RND01...	ACS24_WSI-01.ADM.R...	ACS24_WSI-02.ADM...	RND LAN GRAYNET A...	ACS24_FW1.GOV01.A...	ACS24_R1.GOV01.ACS...	ACS24_R2.GOV01.AC...	ACS24_DC1.LAN.GOV...
ACS24_DC2.LAN.GOV...	ACS24_FILES.LAN.GOV...	ACS24_DNS1.GOV01.A...	ACS24_DNS2.GOV01...	ACS24_MAIL.GOV01...	ACS24_TLE.GOV01.A...	ACS24_WSUS.GOV01...	ACS24_VPN.GOV01.A...	ACS24_K8S-N1.GOV0...	ACS24_K8S-N2.GOV0...	ACS24_K8S-N3.GOV0...
ACS24_PRESIDENT.G...	ACS24_WS2-01.LAN...	ACS24_WS2-02.LAN...	ACS24_WS2-03.LAN...	ACS24_WS2-04.LAN...	ACS24_WS2-05.LAN...	ACS24_WS2-06.LAN...	GOV LAN GRAYNET A...	ACS24_FW4.MILO1.AC...	ACS24_DC3.INT.MILO1...	ACS24_SPINT.MILO1...
...

TEAM RANKING LIVE

2. BT-26	[SCORE: 10821]
3. BT-25	[SCORE: 10716]
4. BT-01	[SCORE: 10698]
5. BT-24	[SCORE: 10680]
6. BT-12	[SCORE: 10199]

REPORTS LIVE

[13:52:06]	SITUATION REPORT
[16:52:44]	SITUATION REPORT
[16:59:30]	SITUATION REPORT
[16:59:27]	SITUATION REPORT
[16:59:18]	SITUATION REPORT

RED ATTACKS LIVE

[13:14:56]	UNDER ATTACK HMI.SCADA.RND - HOST
[13:11:08]	UNDER ATTACK WWW.MCS.GOV - HOST
[12:40:44]	ATTACK UNASSIGNED PILOT - HOST
[12:04:51]	ATTACK UNASSIGNED PLC - HOST
[11:27:14]	UNDER ATTACK DC1.LAN.GOV - HOST

UI-API V2025.06.1-AMD OF 2025-06-09 07:34 POWERED BY **CybExer**

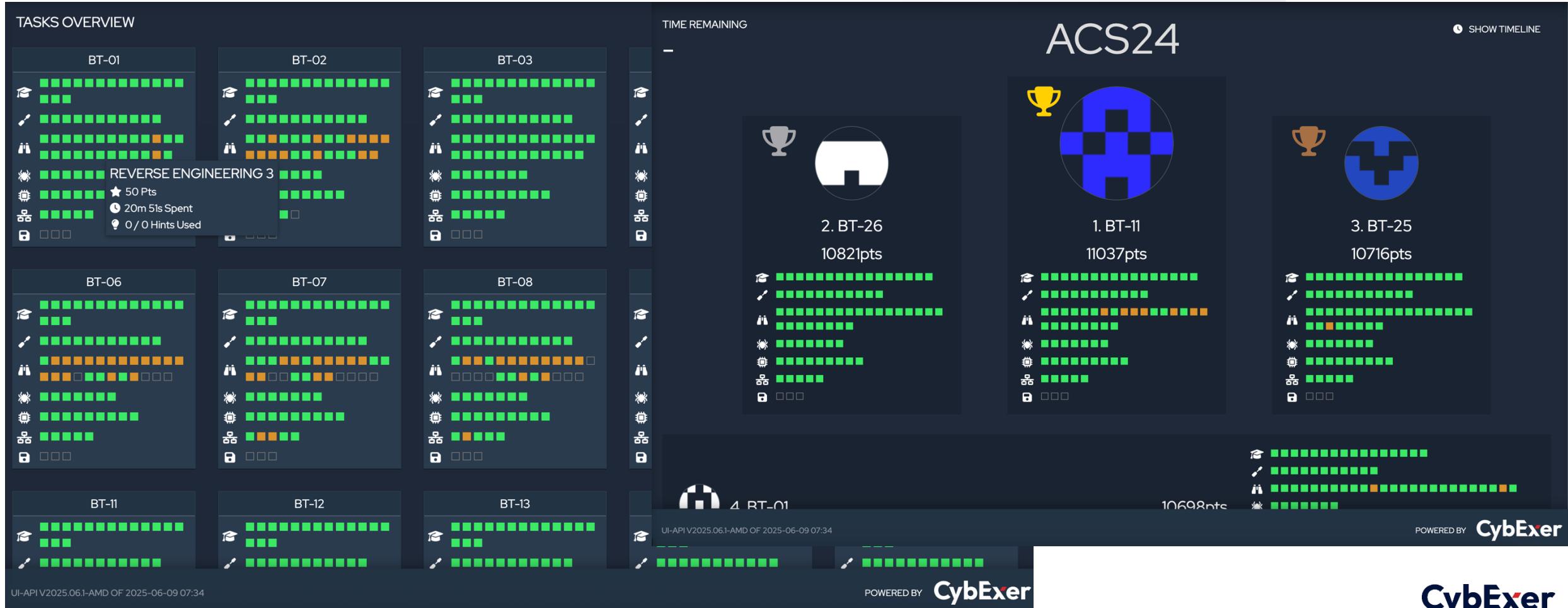
ISA – Live-Fire – WT views – awareness

Smart City view



ISA – Live-Fire – WT views – Missoin Board status

Status of the Mission Board tasks



ISA CTF module details

ISA – Capture The Flag (CTF)

- Capture the Flag tasks are special missions for Blue Team to solve.
- Tasks, hints and answers are defined in the exercise JSON.
- Answers (Flags) must be taken from live targets.
- Scores from the missions are shown in Team Scoring view.

ISA – CTF answer Types

- Single answer
 - Fixed answer common for all envs.
- Single answer array
 - Fixed answer different for every env.
- Multiple answers
 - Multiple fixed answers common for all envs.
- Multiple answer array
 - Multiple fixed answers different for every env.

ISA – Mission Board tasks answer types

- Free form
 - Participants give answer as free text which is analysed by judge. The answers will be visible in Judging -> Tasks view. Judge can use a slider to deduct score for the answer.
- Free form with auto approve
 - Participants give answers as free text which is automatically approved by system
- Script
 - ISA executes script in GMA which can login to the target and check something. Exit code 0 is considered as correct answer.

ISA – CTF – Mission Board – Team-based exercise

- In team-based CTF exercise, all mission board tasks are shared among teammates – When one member solves a task, it's completed for the entire team.
- Each team member has their own console for independent access and operations.

The screenshot shows the ISA Mission Board interface for the CYBER BATTLE OF AUSTRALIA 25 - QUALIFIER. The interface is divided into three main sections: EXPLOITING, FORENSICS, and WEB. Each section contains several mission tasks with their respective points, names, solved status, and team counts.

Category	Mission Task	Points	Description	Solved	Teams Solved	
EXPLOITING	PRINTERESTING	100 pts		OPEN	1/38 TEAMS SOLVED	
	WIN CONDITIONED	100 pts		SOLVED	12/38 TEAMS SOLVED	
	ENCRYPTED COMMUNICATION	100 pts		OPEN	0/38 TEAMS SOLVED	
	LITTLE TRICK	100 pts		SOLVED	5/38 TEAMS SOLVED	
FORENSICS	BOARDING PASS BLUNDER	100 pts		SOLVED	20/38 TEAMS SOLVED	
	FORMAT FORGIVENESS	100 pts		SOLVED	4/38 TEAMS SOLVED	
	ESSAY	100 pts		OPEN	0/38 TEAMS SOLVED	
	PHP SHELLCODE	100 pts		OPEN	0/38 TEAMS SOLVED	
	SPIDERMAN	100 pts		SOLVED	20/38 TEAMS SOLVED	
	AUTHORS NOTES 1	100 pts		SOLVED	34/38 TEAMS SOLVED	
	AUTHORS NOTES 2	100 pts		SOLVED	29/38 TEAMS SOLVED	
	AUTHORS NOTES 3	100 pts		SOLVED	12/38 TEAMS SOLVED	
	WEB	HISTORICAL	100 pts		SOLVED	7/38 TEAMS SOLVED
		SHIPMENTSIO	100 pts		SOLVED	17/38 TEAMS SOLVED
		SOLAR STORM	100 pts		OPEN	2/38 TEAMS SOLVED
		SOLAR POWER	100 pts		OPEN	2/38 TEAMS SOLVED
DEFACED		100 pts		SOLVED	9/38 TEAMS SOLVED	

UI-API V2025.041-AMD OF 2025-04-07 08:01

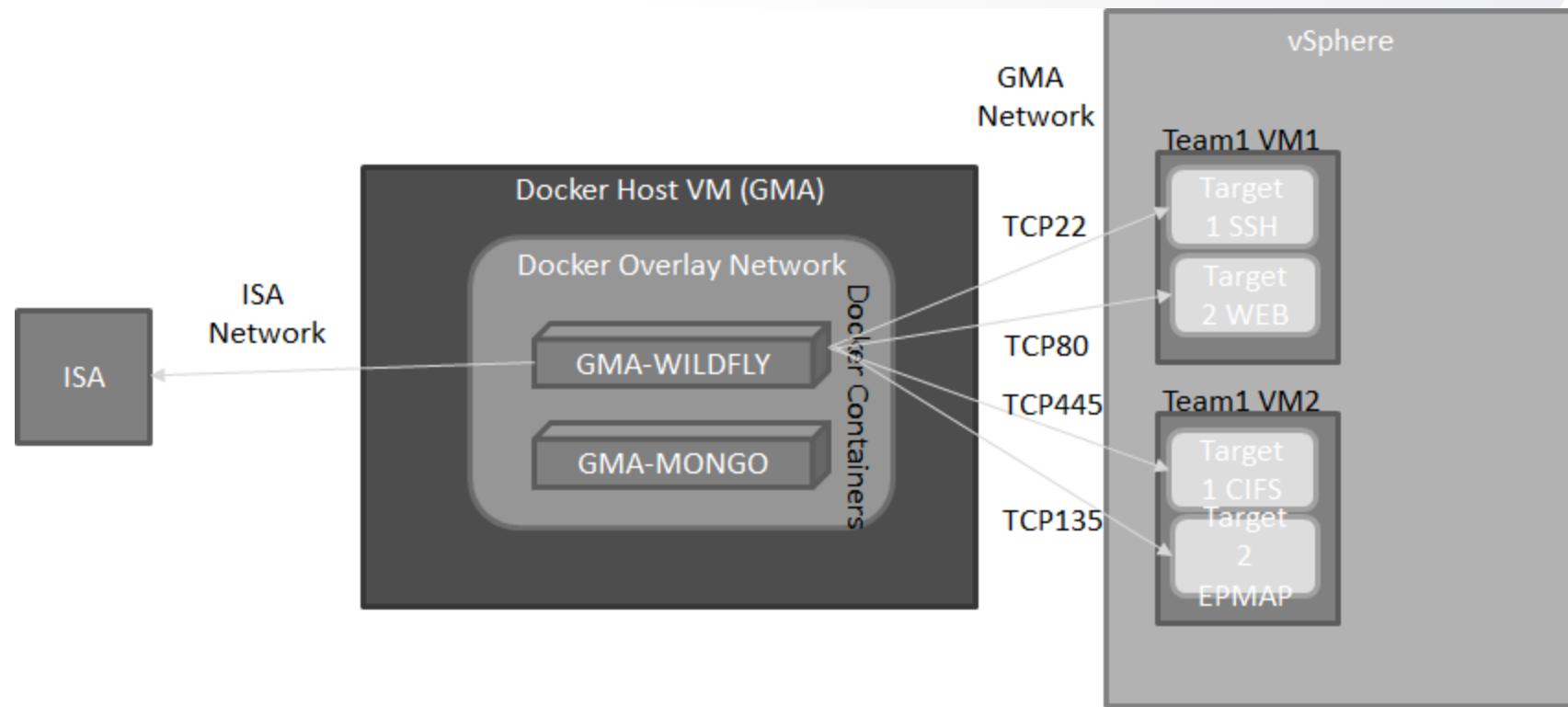
POWERED BY CybExer

ISA General Data

GameNet Monitoring Agent (GMA)

- The GameNet Monitoring Agent (GMA) is a standalone service which will be installed into each Blue-Team's network (or network segment) as a dedicated virtual appliance.
- GameNet Monitoring Agent polls the Blue Team's defined services (targets) and sends availability (and consistency) information to the ISA.
- This information is used for scoring and situational awareness visualizations.
- GMA can also execute scripts. which can be used for:
 - Availability checks (ex. check for website defacement)
 - Tasks: Check target configuration (ex. check if website is correctly configured)

GMA Architecture



Exercise definition

- Exercise definition contains the following data:
 - Exercise name
 - Exercise description
 - Red Team LDAP group
 - White Team LDAP group
 - Blue Team LDAP groups per team
 - Network segments
 - Campaign phases
 - Blue Team names
 - Blue Team GameNet Monitoring Agent info
 - Blue Team virtual machines
 - Blue Team targets and objectives
 - Points budget
 - CTF/cSTR Tasks
- The whole exercise is defined with one .json file.

Exercise definition

Exercise creation contains two steps:

- First exercise definition template needs to be created
 - Contains initial data necessary for creating the exercise
 - Has patterns inside
 - Mostly manual work to get all the data correctly entered
- Second step is submitting the exercise definition template. This results in generating full exercise definition in ISA.

Exercise generation example - template

```
{  
  "name": "POST Threat Hunting 2023",  
  "numberOfTeams": 3,  
  "blueTeamName": "BT-{[%+1]:2}",  
  "blueTeamLdapGroup": "CN=bt{[%+1]:2},OU=Groups,OU=INT,DC=int,DC=ste13,DC=com",  
  "targetGroups": [  
    {  
      "name": "DMZ Servers",  
      "targets": [  
        {  
          "name": "r1.ic{[%+1]:2}.jc.crp",  
          "hostname": "r1.ic{[%+1]:2}.jc.crp",  
          "networkSegmentName": "DMZ",  
          "targetChecks": [  
            {  
              "name": "r1 - HOST",  
              "description": "r1.ic{[%+1]:2}.jc.crp responds to icmp echo requests",  
              "type": "ICMP_RESPONDER_PING",  
              "ipAddress": "10.{%+1}.0.1",  
            }  
            .  
            .  
            .  
          ]  
        ]  
      ]  
    }  
  ]  
}
```

Exercise generation example – exercise

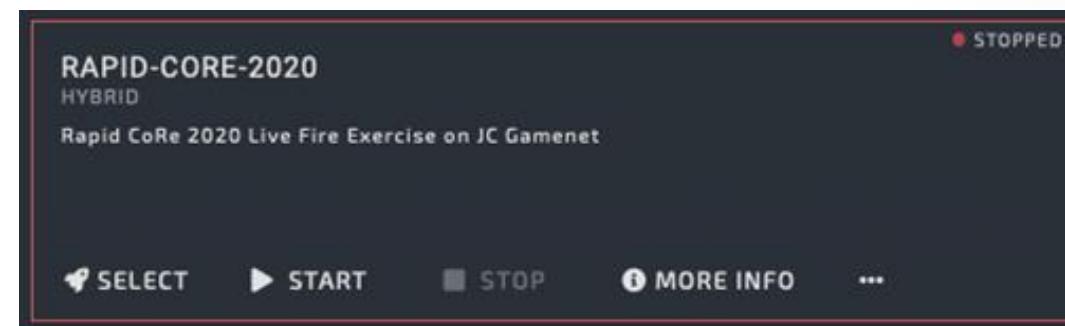
```
{  
  "id": "3f609650-f9ee-4f74-99b0-c457edc9706b",  
  "name": "POST Threat Hunting 2023",  
  "blueTeams": [  
    {  
      "id": "178be989-d3ac-43ee-b002-292ae429a457",  
      "name": "BT-01",  
      "ldapGroup": "CN=bt01,OU=Groups,OU=INT,DC=int,DC=ste13,DC=com-hidden",  
      "targetGroups": [  
        {  
          "id": "677fcda5-f073-421a-afdb-ad0fddad762c",  
          "name": "DMZ Servers",  
          "targets": [  
            {  
              "id": "7e131c65-c943-4e30-9861-862d0257e0da",  
              "name": "r1.ic01.jc.crp",  
              "hostname": "r1.ic01.jc.crp",  
              "networkSegmentId": "70cb334a-c504-4b29-9b88-fb307c2e71d0",  
              "targetChecks": [  
                {  
                  "id": "3cc06d63-72df-4c8d-ab4f-f476e83a69c8",  
                  "name": "r1 - HOST",  
                  "description": "r1.ic01.jc.crp responds to icmp echo requests",  
                  "type": "ICMP_RESPONDER_PING",  
                  "ipAddress": "10.1.0.1",  
                },  
              ],  
            },  
          ],  
        },  
      ],  
    },  
    ...  
  ],  
  "id": "3a2f014f-0892-4a4d-ada2-4d029dac9e10",  
  "name": "BT-02",  
  "ldapGroup": "CN=bt02,OU=Groups,OU=INT,DC=int,DC=ste13,DC=com",  
  "targetGroups": [  
    {  
      "id": "3bddfba5-a9f5-40d3-91c3-e888d8e6c446",  
      "name": "DMZ Servers",  
      "targets": [  
        {  
          "id": "3281862-3779-48c2-89c1-820313ce4588",  
          "name": "r1.ic02.jc.crp",  
          "hostname": "r1.ic02.jc.crp",  
          "networkSegmentId": "70cb334a-c504-4b29-9b88-fb307c2e71d0",  
          "targetChecks": [  
            {  
              "id": "d679fb48-3f6b-48ce-9aa0-55e061d0fdb4",  
              "name": "r1 - HOST",  
              "description": "r1.ic02.jc.crp responds to icmp echo requests",  
              "type": "ICMP_RESPONDER_PING",  
              "ipAddress": "10.2.0.1",  
            },  
          ],  
        },  
      ],  
    },  
  ],  
},  
...  
}
```

Exercise definition edit

- If exercise is already created, there are two ways to edit the existing data:
 - Edit the full .json and submit the changes.
 - This is the only way to edit an exercise that's already running.
 - This will keep all the existing IDs intact.
 - Changes must be applied to every environment's data.
 - Recreate the exercise from template
 - Recommended option for editing an exercise that is not running.
 - All object IDs, except the exercise ID, will be regenerated.
 - Exercise data must be deleted (reset exercise).

STARTEX

- Exercise start is notified in ISA by running STARTEX command.
- STARTEX command is sent by clicking on “Play” button on exercise module.
- Scoring will only work between STARTEX and ENDEX commands. All other scoring messages will be ignored, but stored in the database.



ENDEX

- Similarly, like STARTEX, exercise finish is notified in ISA by running ENDEX command.
- ENDEX command is sent by clicking on “Stop” button on exercise module

LAB05 - ISA

- Task 1 – Create your own ISA exercise
- Task 2 – Use your ISA exercise

Cyber Range components Details

DeLRAP (Portal)

Dynamic e-Learning and Risk Assessment Platform (portal)

- DeLRAP platform is an ICT environment to carry out learning and testing activities which enable consequent assessment of the results within the same platform.
- Portal can act as a primary access solution for individual end-users.
- Portal tracks the user progress and stores the user history for passed learnings and exercises.
- Portal can also act as a self-service solution for users to enroll and reset their credentials.

Dynamic e-Learning and Risk Assessment Platform

The screenshot displays the CybExer platform interface. At the top, the navigation bar includes the logo "CybExer", "Dashboard", "Admin", "Analyst", "Leaderboard", "Support", a user icon, "English", and a dropdown for "i13.portal.jaagup". Below the navigation is a dark blue header section titled "Admin board" with three cards: "Manage Groups" (Map Active Directory (AD)/OpenLDAP groups to platform groups), "Manage Product Keys" (Add or watch information about application product key), and "Edit Settings" (Change display settings, integrated account password and other information). The main content area features a search bar and tabs for "Video", "Self-Study", "Skills Testing" (which is highlighted in red), "Live Fire", "CTF", "Live-fire & CTF", and "uni". Two skill testing modules are shown: "Skills Testing - Network Operations Specialist - OPEN-SOURCE" (CUSTOM, HARD) and "Skills Testing - System Administrator" (CUSTOM, HARD). Both modules include descriptions, "Go to exercise page", "Configure", and "More info" buttons.