A Scalable Cyber Security Framework for the Experimentation of DDoS Attacks of Things

Davi D. Gemmer[*], **Bruno H. Meyer**[†], **Emerson R. de Mello**[‡],
Marcos Schwarz[¶], **Michelle S. Wangham**[§], **Michele Nogueira**[*]
[*]Department of Computer Science - Federal University of Minas Gerais, Brazil
[†]Department of Informatics - Federal University of Paraná, Brazil
[‡]Federal Institute of Santa Catarina, Brazil
[§]University of Vale do Itajaí, Brazil
[¶]National Research and Education Network, Brazil
Emails: ddgemmer@gemmer.com.br, bruno.meyer@ufpr.br, mello@ifsc.edu.br,
marcos.schwarz@rnp.br, wangham@univali.br, michele@dcc.ufmg.br

Abstract—**The Internet of Things (IoT) has amplified cyber security challenges for governments, businesses, and individuals. IoT is a straightforward attack target once it comprises resource-constrained and heterogeneous devices that often present security vulnerabilities easily exploited in different attack vectors. Recent Distributed Denial of Service (DDoS) attacks leverage thousands of IoT devices connected to the Internet called DDoS of things (a.k.a. DoT). DoT requires systematic cyber security research, but advancing the state-of-the-art depends on methods and tools that jointly manage scalability and performance. Experimentation is an essential and well-known tool for scientific research. However, experimental environments for investigating DoT are challenging, given limitations in scale and IoT heterogeneity. Hence, the main contribution of this work lies in presenting a cyber security framework for DoT experimentation that manages scalability and performance in scenarios under attack. It is the first initiative to create a framework of reference to assist in implementing cyber security testbeds. Hence, this work also presents an instantiation of this framework, called the MENTORED testbed, and the results of a case study using it.**

Index Terms—**Cyber security framework, DDoS of Things, Experimentation, Testbed, Kubernetes.**

I. INTRODUCTION

The Internet of Things (IoT) has amplified cyber security challenges for governments, businesses, and individuals. IoT comprises many heterogeneous and resource-constrained connected devices, significantly increasing the volume of generated and transmitted data [1]. The manufacture of IoT devices rarely handles security and privacy issues, and there are no security standards for IoT [2]. These aspects make IoT devices easy targets for attackers, growing the number and volume of attacks leveraging IoT devices, such as Distributed Denial of Service of Things (a.k.a. DoT attacks or DDoS of Things) [3]. DoT has reached critical mass, i.e., each attack relies on hundreds of thousands of devices connected to the Internet [4]. This behavior highlights the urgent necessity of systematic cybersecurity research in this context.

Investigating and designing robust solutions to prevent, detect and mitigate DoT attacks require appropriate tools and methods to test and validate them [5]–[11]. Today, researchers rely on datasets or controlled environments to study DDoS attacks by simulations. Experimental environments allow the investigation of DDoS attacks close to real conditions. However, there is a lack of experimental environments that meet specific requirements for IoT cyber security. Also, it urges a framework to support and guide the development of environments to test solutions and study DoT [12]. Existing experimental environments encounter performance issues when the scale grows. But, scalability is a genuine feature of IoT and DDoS attacks, requiring serious consideration [13], [14]. Also, managing the diversity of devices, protocols, and communication technologies is a challenging task for experimental environments [15], which depend on dedicated hardware and customized software [16].

In a nutshell, there are two groups of cyber security testbeds for DoT: (i) those focusing on security but not on IoT; (ii) those focusing on IoT but either ignore security or present minimal functionalities related to it. These two groups stand out, led by two prominent testbeds: DETERLab [17] and FIT IoT-LAB [18]. The first has been a pioneer in cyber security large-scale experimental environment, whereas the second plays an essential role in IoT experimentation. Unfortunately, DETERLab presents limitations to wireless network experimentation, an important property for IoT experiments. Further, FIT IoT-LAB does not consider security as its primary focus and lacks traffic isolation between experiments of different users. Recently, Cámara *et al.* [19] proposed a network security testbed for IoT scenarios. However, scalability is still an issue. A common observation in these examples lies in the absence of a general reference to guide and design testbeds for cyber security, mainly regarding DoT attacks.

Hence, the main contribution of this work is a cyber security framework for the experimentation of DoT attacks. The framework manages scalability and performance in experimentation scenarios. It serves as reference for implementing cyber security testbeds concerned with DoT. This work also presents the MENTORED testbed, an instantiation of the framework implemented in the scope of the Brazilian MENTORED project[1].

---

[1] The MENTORED project website: https://www.mentoredproject.org/

The testbed considers features as scalability and performance, as expected, and user experience.

The MENTORED testbed implementation occurs over the Software-Defined Infrastructure of the National Research and Education Network (RNP), part of the academic national-wide network infrastructure. The implementation follows Kubernetes, a widespread technology that provides flexibility in networking technologies and makes it easy to create, execute, manage and delete experiments spread over multiple operating systems. The isolation and control of permissions from Kubernetes bring the possibility of multiple users to perform experiments sharing resources simultaneously. The testbed follows a distributed deployment among the Points of Presence (PoPs) in the Brazilian academic network. Available resources are dynamic and require management. The description of an experiment follows YAML format.

This paper presents the results of a case study performed in the MENTORED testbed. The results express the network traffic throughput considering two evaluation scenarios under DDoS attacks. This work also analyzes the viability of defining and executing experiments in the testbed. Results from an evaluation scenario demonstrate the capacity of a high-performance processing node in the testbed infrastructure to emulate several small devices. The results show a DDoS attack experiment designed and performed in the testbed. The results show that the DDoS attack scales, although using a unique node as the attack target and attackers.

This paper proceeds as follows. Section II presents the related works. Section III details the proposed framework. Section IV describes the MENTORED testbed, including software and hardware available to users. Section V shows the performance evaluation of a case study using the testbed. Finally, Section VI concludes the paper.

II. RELATED WORK

This section overviews the related works on experimental environments from the literature. Testbeds focusing simultaneously on cyber security and IoT are rare. Hence, the next paragraphs overview works from two separate groups: (1) testbeds focusing on cyber security, but not IoT; and (2) IoT testbeds. The two representative works for these two groups are DETERLab [17] and FIT IoT-LAB [18], respectively.

DETERLab [17] is a pioneer cyber security testbed designed for large-scale emulation and experimentation. It provides a set of tools for the creation, manipulation, and observation of experiments. DETERLab offers a controlled and secure environment, i.e., experiments do not threaten other testbed users or the Internet. FIT IoT-LAB plays a vital role in IoT experimentation. It offers a platform for researchers to build, evaluate and optimize protocols, applications, and services. It comprises various hardware boards, communications technologies, and different physical topologies. Despite its importance, DETERLab presents limitations related to the use of different virtualized topologies [20] and ignores the context of wireless network, which is necessary for IoT experimentation. In contrast, FIT-IoT Lab lacks traffic isolation between experiments of different users, which can result in hazards for all.

Recently, Cámara et al. presented the Gotham [19], a network security testbed for IoT. It is based on the GNS3 network emulator and provides a set of tools for experimenters to carry out DoS attacks. Although the testbed offers means for emulating scenarios with many devices, scalability is still an issue. Takeoglu and Tosun [21] proposed a low-cost testbed based on off-the-shelf hardware and open-source software. It investigates security and privacy on IoT devices connected by WiFi and Bluetooth. Although the testbed considers heterogeneity, it does not address scalability. Particular devices require specific software and configurations for capturing packets and analyzing data, and there is no standard or reference for performing a large-scale experiment.

EdgeNet [16] is a distributed system testbed from the PlanetLab family. Its infrastructure is software-only, and the environment comprises virtual machines (VM) interconnected by Kubernetes-based implementation. The project has 40 nodes distributed around the world. The work in [16] analyzes the benefits and challenges of building a testbed based on Kubernetes and highlights high performance with low overhead, being suitable for all types of experiments and systems. However, an analysis is not performed with heavy network loads or experiments that cause network stress, such as DDoS.

Sarirekha *et al.* [22] investigated the challenges and requirements of developing an IoT-based testbed. They highlight challenges such as the heterogeneity of protocols, operating systems, and types of attacks as challenges. The authors also defined requirements for an IoT testbed as (i) flexibility, (ii) handling a high volume of data and heterogeneous devices, networks, and communication protocols, (iii) having as basis open source software and firmware, and (iv) support for multiple use cases. The work analyzes neither aspects related to dense network traffic flows as DDoS attacks nor presents a framework of reference for IoT testbed design.

Each testbed provides specific features to its context [15], [23], and all these testbeds offer contributions to the academic community. However, there is still a place for improvement. A relevant observation is a need for well-defined references to assist in designing testbeds for cyber security, mainly concerned with DoT attacks and their genuine scalable nature. There is still a gap in defining the properties and requirements for such environments to guide the implementation of testbeds that encompass characteristics such as realistic and large-scale geographically distributed environments, considering IoT devices in their infrastructure, flexibility, and scalability.

III. THE CYBER SECURITY FRAMEWORK FOR DDOS OF THINGS

This section details 'the MENTORED framework', a cyber security framework conceived as a reference for designing and implementing scalable experimental environments to DoT attacks. The framework considers existing terminologies among cyber security testbeds [23] and presents three main actors: managers, clients, and users. Managers are an abstract concept representing those responsible for implementing, maintaining, and managing the environment and network. Clients are non-necessary natural persons (e.g., institutions) that require access to an experimental environment. In contrast, users are natural persons associated with clients and use resources.

Fig. 1 overviews the main characteristics of the framework. It encourages collaborations between several partnerships, each providing different resources, benefiting a research community composed of teams, projects, and institutions (clients). The framework considers security and safety issues, including authorization, authentication, accountability, and isolation of experiments. It follows a distributed infrastructure, i.e., it manages resources over different physical locations. The infrastructure is essential, considering the need for scalable experiments involving several real or emulated devices.

Fig. 1. An Overview of the MENTORED framework

The MENTORED framework is founded on a set of requirements: fidelity, validity, scale, reproducibility, transparency, user-centric perspective and real-time access. These requirements guide the framework as follows.

- Fidelity: capability to obtain sufficient precision in reproducing a phenomenon under study in a experiment.
- Validity: the results of experiment must be agnostic to the limitations of the environment and those limitations should not accidentally distort them. The environment must identify and report violations, alerting the user to possible failures.
- Scale: support experiments of representative size to capture complex effects of attacks related to massive data traffic at Internet scale.
- Safety: no code or malicious users can gain unauthorized access or harm other network infrastructures, information, or code of the environment itself or the Internet.
- Reproducibility: ensures that an experiment, once runs, can be exported and then run in an identical environment later to produce comparable results.
- Transparency: enable real-time and non-intrusive monitoring of network traffic and computing resources, and employ tools to visualize these resources graphically and via the command line.
- User-Centric Perspective: give to users the possibility to develop tools that facilitate experimental research and to use the traditional experimental research functions, such as setting up experiments and monitoring traffic.
- Real-time Access: provide real-time access to devices. Then, a user can reset, reschedule, and monitor the state of each device while experiments are running.

The framework defines a set of entities as a general manager, master, IoT resource provider, federated identity provider, and resource communication channels. Similarly, it presents a set of modules such as portal, team management, scenario management, data storage, resource management, and run-time environment. Fig. 2 shows the entities, modules, and their relations. Each element in this framework can be implemented using different technologies. The framework serves as a reference for implementation specification with the primary goal of jointly managing scalability and performance. The description of these entities follows.

- **General Management**: has a general and complete view of the available resources, preferably being able to manage their use. It provides technologies and infrastructure to implement each entity and module of the framework.
- **Master**: responsible for connecting clients with the resources and managing their authorization for each action allowed in the experimental environment.
- **IoT Resource Provider**: bare-metal IoT devices that can communicate with other IoT devices, processing servers, and the master. It should provide an API to allocate, manage and control each device.
- **Processing Resource Provider**: servers with processing capabilities that can communicate with other servers, IoT devices, and the master. It provides an API to allocate resources in each device through simulation or emulation.
- **Resources Communication Channel**: directs resource requests to resource providers. This entity is optional if devices are directly connected to the Internet, which is not recommended concerning isolation.
- **Federated Identity Provider**: manages authentication and user information in a domain (e.g., an university).

The description of the modules follows.

- **Portal**: an interface to the user. The master offers the initiation of the federated authentication, interacts with Team Management module, and manages the user environment to define access and store data related to the experiments. Different options must be considered for different types of users.
- **Team Management**: module triggered by users associated to a collaborative project to create and manage virtual teams (identify their users, organize them in groups, assign them roles) and share common resources (defining access rights) using federated identities. This module is responsible for generating authorization tokens for a project user to conduct experiments.
- **Scenario (Experiment) Management**: the master entity implements policies to restrict which resources can be

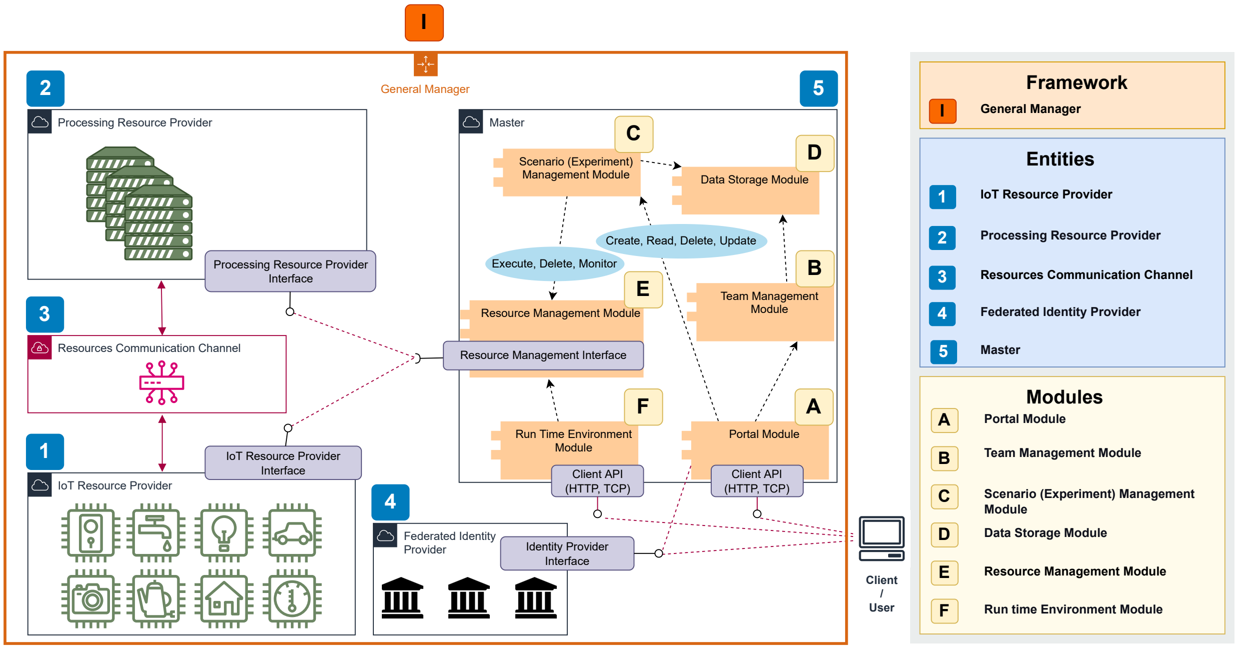

Fig. 2. The MENTORED framework

accessed by users. Syntaxes define experiments, indicating how the resources can be used and if the Data Storage Module must save any resulted data. This module orchestrates user requests and makes available resources.

- **Data Storage**: datasets record two main types of data: 1) data related to experiments, such as descriptions, network traffic, and log files; 2) user environment and authorization data, used for controlling tasks and access.
- **Resource Management**: implements the operations required to consume Resource Providers APIs and enables the master to control different actions of the Resource Providers, like the run-time access to experiments or the deployment and destruction of experiment definitions.
- **Run-Time Environment**: the definition of processes that enables direct access to the experiments in execution.

Users access the federated authentication interface, the portal and the run-time environment interface. The federated authentication interface depends on the Team Management Module and the Federated Identity Provider. The portal enables actions to define, execute and manage the environment. Then, the run-time environment interface monitors experiments in execution. With these interfaces, it is possible to trigger modules in the master, which acts as an intermediate entity managing users and computation resources. The framework focuses on DoT attacks, which enable several IoT devices and robust processing servers to simulate or emulate nodes. Hence, experiments that try to define large topologies take advantage of all assets offered by the resource providers because they will be able to communicate according to the Resources Communication Channel entity.

Fig. 3 overviews the lifecycle of a given experiment as experiment definition, resource management, execution and monitoring. First, the user needs to define what topology, resources, and software will be used in an experiment by a description language with a syntax. Users may define this by a GUI interface in the portal, simplifying user experience. Users specify the resources and settings required by his/her experiment, which can be crucial to guarantee validity, scalability, and safety requirements.

Assuming a multi-use of resources simultaneously by multiple users, an experimental environment must address security issues concerning the experiments and provide individualized settings for all users. Hence, the general management predefines policies to divide assets available on the resource providers for different clients and verify if a user experiment definition is valid. Also, an optional verification can be implemented to isolate the resource providers from the Internet. Then users access them only through the master entity. If an experiment is validated, the user requests its execution. The master guarantees that different experiments running in the environment do not have access to each other, preserving all requirements.

A consistent analysis of DoT attacks depends on faithful representations of network traffic in real-world scenarios. A key point to implement a testbed following this framework lies in defining the technology to deploy experiments and the communication methods among the topology nodes. The MENTORED framework assumes the existence of an infrastructure able to support the network communication of several real (or simulated) IoT devices. The infrastructure management is performed by the resource communication channel entity. Hence, an user monitors the experiment by the Run-Time Environment Module, logs, and saved data.

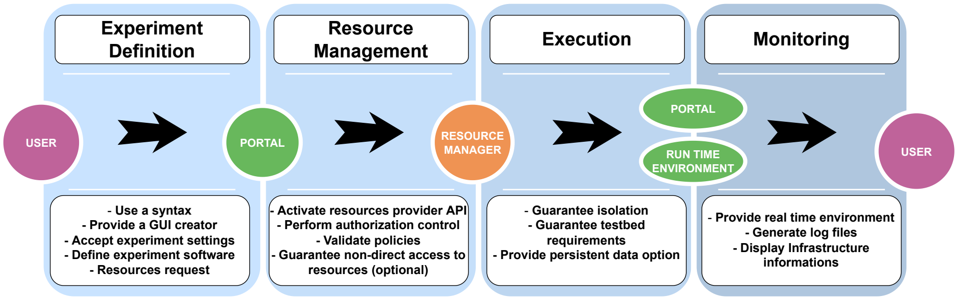

Fig. 3. Experiment lifecycle in the MENTORED framework

IV. THE MENTORED TESTBED

This section details the MENTORED testbed, an instantiation of the proposed framework, conceived for the experimentation of DoT attacks. First, it introduces the testbed architecture, its entities, and modules correlating them with the framework. Second, it describes user experience when defining, executing, and monitoring an experiment in the testbed (experiment life cycle).

A. Overview

Fig. 4 shows the architecture of the testbed, highlighting the MENTORED Master (in the blue external frame) and the Software-Defined Infrastructure of the National Education and Research Network (IDS-RNP) (white external frame). The MENTORED Master implements the MENTORED portal, the orchestrator (backend), REST API, and the command-line interface (CLI). The MENTORED Master mediates the interaction of users with the resources present in IDS-RNP, authenticating, controlling, and managing user permissions.

IDS-RNP incorporates functionalities from the Processing Resource Provider and the Resource Communication Channel defined in the framework (Fig. 2). RNP and its administrators play the role of general managers defined by the framework. Using small and high-performance nodes as the resources offered by the Processing Resource Provider, it is possible to use nodes to emulate IoT devices in IDS-RNP. Also, Kubernetes allows the insertion of IoT devices like Raspberry in the cluster. This feature enables an IoT Resource Provider (from the framework) and increases fidelity and scalability in the testbed. The MENTORED master implements the Data Storage Module, which uses a database to save all experiment-related data. The scenario management and resource management modules are implemented in the MENTORED testbed through the Orchestrator features. The next sections detail IDS-RNP and the components of the MENTORED testbed architecture in Fig. 4.

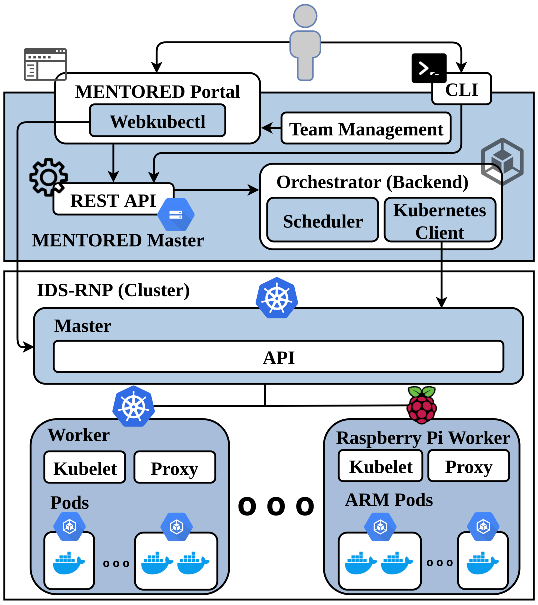

Fig. 4. The MENTORED Testbed

B. The RNP Infrastructure

IDS-RNP is a core component of RNP's Testbed Service. Fig. 5 shows the 15 locations of its dedicated physical servers hosted at RNP's Points of Presence and interconnected by Rede Ipê, the Brazilian national-wide academic network, encompassing all five regions of Brazil. IDS-RNP accommodates two types of servers called worker nodes and small nodes. It has 13 high-performance worker nodes with 48 CPUs, 192GB of RAM, 20TB of storage and up to 100G networking; and 15 small nodes, 13 of which are co-located with the high-performance nodes, with 8 CPUs, 16 GB of RAM, 300GB of storage and up to 10G networking. Additionally, a central node of the cluster, called Master (Fig. 4), creates the topology and starts the experiment. The topology includes an arbitrary number of nodes, instantiated on the worker nodes. IDS-RNP relies on different network virtualization technologies.

Kubernetes orchestrates IDS-RNP and provides a centrally programmable distributed edge cloud in which a researcher requests a slice and runs an experiment using 'namespaces' and 'deployments'. A 'namespace' is a pool of resources that a user or group can use. It can be unlimited and allow the use of available resources, or have a quota of CPU, memory, disk, and network. 'Deployment' means a description of the resources allocated for a given experiment. It indicates the applications or container images, quantities, and locations.

IDS-RNP offers KNetLab, an extension of Kubernetes implemented to introduce network topology as a resource and enhance the experience in IDS-RNP. For KNetLab, a network topology consists of ‘devices’ and ‘links’. A ‘device’ is an application with the location attribute that explicitly defines in which node it is deployed. A ‘link’ is an adjacency declaration between two devices to provide a layer-2 circuit. Each container instance deployed in a given ‘namespace’ is interconnected by a private network employed as the experiment control plane and a set of user-defined links that provides the data plane topology. KNetLab implements the user-defined topology by Open vSwitch (OVS) [24] through virtual Ethernet connections (veth). These connections implement a link between devices on the same node and a VXLAN tunnel over Rede Ipê for devices on different nodes. KNetLab integrates two additional mechanisms: DPDK acceleration on OVS and offload tunnels between nodes to EVPNs, dynamically provisioned on Rede Ipê. This supports wire-rate packet processing up to 100 Gbps.

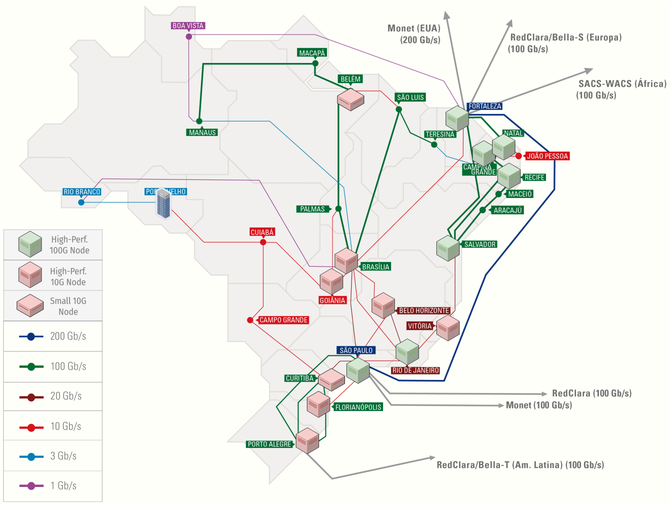

Fig. 5. The IDS-RNP Infrastructure

C. The MENTORED Master

The MENTORED Master operates as a software layer on top of the Kubernetes API, which handles the iterations between modules, as shown in Fig. 4. It controls the MENTORED Portal, CLI, REST API, Team Management and the orchestrator. The MENTORED Portal is a Federated Service Provider that manages user interactions, delegates user authentication to federated identity providers, interacts with the team management service, and establishes a secure remote connection with the REST API. The portal offers several GUI options to the users in their web browsers once they are authenticated. These options include actions as the creation, visualization, update and exclusions of accounts and experiments. Team management service provides flexible enrollment flows to bring experimenters and their federated identities in the testbed and creates a virtual team for the collaborative experimentation project. This service manages users, projects, groups, and access rights for the resources in IDS-RNP.

The Webkubectl[2] is a technology incorporated by the MENTORED Portal, and provides specific commands for reaching the requirements of the Run-Time Environment module. This module enables users to access a UNIX-based environment in their web browser, in which it directly accesses the IDS-RNP Master. This tool goes inside the MENTORED Portal and is based on the same authorization context implemented by the Team Management module using the Namespaces elements of Kubernetes and KNetLab.

All operations allowed to the user are implemented using a REST API, enabling the implementation of several interfaces that can be operated using HTTP requests. The REST API offers interaction between the MENTORED Portal and the orchestrator. The orchestrator interacts with the other modules and the master to provide specific functions for creating experiments, controlling and monitoring resources. It simplifies the configuration of network scenarios and employs the necessary tools to fulfill the defined requirements. Thus, a legitimate user configures a container to create, submit and run a script describing the experiment according to the REST API commands. The orchestrator interprets the script and interacts with the IDS-RNP Master to configure the Kubernetes-based infrastructure to make the necessary resources available and allocate modules to each namespace. When starting an experiment, a namespace is automatically generated for the user with the authorization hierarchy and isolation.

D. Experiment lifecycle

One of the essential keys to a functional testbed is enabling users to define what should happen in experiments. Fig. 6 shows the lifecycle for running an experiment in the MENTORED Testbed. A legitimate user describes an experiment following an object syntax that extends the Kubernetes Pods. Pods are the smallest deployable units of computing composed of one or more containers in Kubernetes.

Fig. 6 shows two groups of parameters in the experiment definition script: node actors and type of topology. **Node actors** comprise a list of objects with attributes as *containers*, *number of replicas*, *pods environment variables*, and *worker*. The *containers* attribute is a pod definition (list of containers). The structure of this definition must follow the exact specification of the pod definition in the Kubernetes documentation. With this specification, an user configures the containers using several parameters like those that control disk usage and data persistence, or network behavior options. The *number of replicas* attribute defines how many replicas of this pod the topology will have. For instance, an user defines a pod that simulates the behavior of a DDoS attacker following a topology and easily replicates the attacker presence in the topology by increasing the number of replicas. The *pods environment variables* are information accessed by each pod during their execution. Beyond these user-defined variables, the MENTORED testbed initiates each container with different information such as the IPv4 of each pod on the topology,

---

[2] https://github.com/KubeOperator/webkubectl

the pod identifier, and other. Finally, the *worker* attribute is the name of a worker (node) of the Kubernetes cluster implemented on the IDS-RNP. All replicas of a given node actor will be instantiated in a worker.

The **type of topology** parameter defines how node actors connect. The user specifies a topology and informs it to the testbed, or s/her uses the built-in topology framework implemented on the MENTORED Testbed Orchestrator. One type of built-in topology is a mesh-based pattern where all node actors are connected. Another type is an OVS-based (Open Virtual Switch) framework, where each node actor instanced in the same worker of the IDS-RNP cluster will be connected to an OVS, and all OVS are connected. The connections created in these topologies are built upon the KNetLab technology, which enables communication between nodes without generating network traffic to the Internet.

After the definition of the node actors and the type of topology, the generated YAML script is sent to the REST API server, through a HTTP request with the name of the experiment definition and the YAML data. The script will be validated, considering the MENTORED syntax. If the request is accepted, then the description is saved into the database, allowing its reuse. Next, the execution of the experiment can be initiated by another HTTP request to the REST API.

When any user is trying to execute an experiment, it must associate an experiment definition and a project that this user is related to. It is important to mention that in the MENTORED Testbed, different projects can access different types of resources, such as processing nodes, disk usage quota, and others. An important parameter that the user must inform is the maximum time of the experiment execution, which is limited to a value defined by the testbed admins. If there is any docker container running until this time, all remaining containers are abruptly stopped.

Finally, as in step 3 of Fig. 6, the legitimate user accesses Webkubectl and connect to the experiment namespace in execution. Using the Kubernetes commands, the user performs several actions, such as obtaining the list of nodes, downloading or uploading files from nodes, and managing containers in real time. Using the Portal, the user accesses a status code that indicates the progress, step of the execution, and if there is any error. An experimenter, during the execution, can also access the KNetLab interface to visualize the defined topology and inspect if it corresponds to what it is supposed to be. Each KNetLab update will be automatically deployed on this interface, improving the user experience of the testbed.

In the future, a user-friendly feature in the MENTORED portal will allow experiment definitions that enables the creation of YAML files following the MENTORED specification, using a drag-and-drop GUI. The MENTORED orchestrator produces several log files during each experiment execution to ensure validity and transparency. These files are saved and can be accessed by the user by the REST API. Therefore, the same experiment definition can be executed several times, producing multiple log files to be compared.

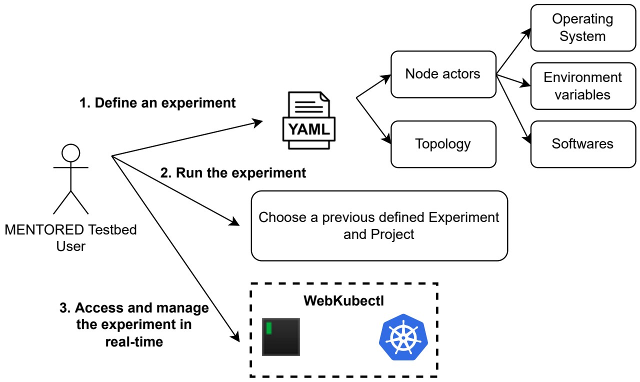

Fig. 6. Experiment Execution Flow from an User Perspective

V. THE MENTORED TESTBED - CASE STUDY

This section presents a case study for the MENTORED testbed. This case study considers different scenarios to evaluate their viability in enabling users to easily perform DDoS experiments. Subsection V-A shows a simple experiment scenario and its deployment on the MENTORED testbed. Subsection V-B describes an experiment that aims to identify the capacity to extend the previously presented experiment scenario to be executed with more devices.

A. Evaluation Scenarios

The DDoS attack was carried out using IDS-RNP nodes at Vitória, Salvador and São Paulo (see Fig. 5 for all locations of IDS-RNP nodes). The victim node implements a standard NGINX server at Vitória. There are two attackers configured with hping3 to make 100 requests per second in Salvador's node, and three clients responsible for requests every 0.5 seconds in São Paulo's node. The attack has a duration of 300s comprising the phases: pre-attack (0-59s), where there is only clients traffic; attack (60-240s) composed of the traffic of the clients and the attackers; and post-attack (241-300s) with clients traffic only.

The MENTORED testbed implements a predefined policy for authentication and authorization. Considering as previous step the federated authentication, an user can access the REST API through any type of interface using HTTP. A experiment description code[3] describes the scenario. The container definitions follow the exact syntax of container definitions in Kubernetes. The rest of the YAML code contains information about the MENTORED context, and topology (e.g., replicas, connections). This demonstrative example employs a default topology architecture named "ovs_fully_connected" where all nodes of the same worker will be connected with other worker nodes through an OVS.

The experiment implements the NGINX software and a program that monitors CNI (Container Networking Interface). Since each experiment execution produces different results, log data is stored in the server and accessed at the end of

---

[3] Experiment description: https://github.com/mentoredtestbed/MENTORED-Testbed/blob/main/doc/demo-noms2023-topology.yaml

the experiment. With data, the average network throughput per second was measured to detect the DDoS performed by attackers. In Fig. 7, the effects of the DDoS attack is between instants 60s and 240s, the exact time the attackers execute their activity. Although it is a simple experiment scenario, this evaluation serves as proof-of-concept for using the MENTORED framework and the proposed testbed to define and execute DDoS research experiments. The interaction with the testbed is made by a simple syntax definition and is based on an infrastructure that scales with several high-performance servers and IoT devices.

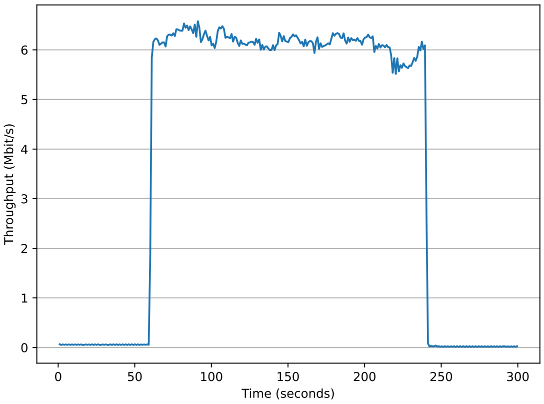

Fig. 7. Throughput on NGINX during a DDoS attack

B. Stress Testing Evaluation

Real DDoS attacks usually consider thousands or millions of devices attack and generate traffic to one or few targets. Testbeds struggle to reproduce this level of infrastructure due to computational resource constraints. In order to identify these limits on the MENTORED testbed, a stress test experiment was performed with the goal of analyzing the computational capacity of one IDS-RNP worker to simulate several small devices. The stress testing evaluation scenario used the Vitória IDS-RNP node. The Pod resource management enabled by Kubernetes limits CPU and memory employed to the definition of attacker pods. Each attacker pod uses the equivalent of half CPU and 128M RAM. These limitations influence the hping3 tool to perform DDoS attacks and generate network traffic. The main purpose of the stress test was to identify what is the maximum number of devices that can be added to a DDoS attack simulation, then the size of the DDoS attack keeps growing. The network traffic throughput in the target identifies the size of the DDoS attack. When the increase in the number of device attacks does not impact the throughput in the server, it means that adding new devices does not contribute to the increase the attack size.

The stress scenario experiment follows the previous scenario as base. A NGINX server runs as target, and different number of attackers from a range between 1 and 30 perform different DDoS attacks. The duration of the experiment was in total of 300s and the average throughput in Fig. 8 refers to the attack period from instants 59 to 240s. Each point of the graphic is the average of five executions for each number of attackers.

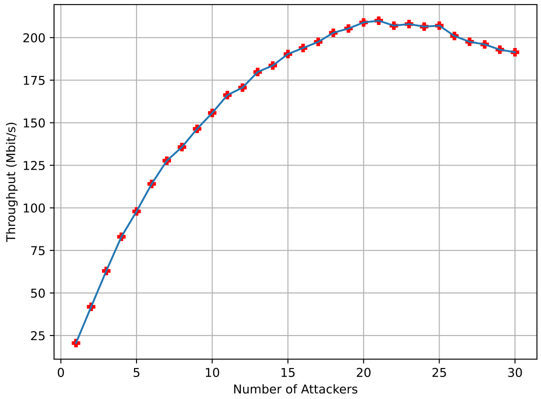

Fig. 8. Average throughput on NGINX during DDoS vs. # of attackers

Although using a unique node to implement the target of the DDoS attacks and attackers, results show that the DDoS attack scales up to 21 device simulations using a single node and considering the capacity to increase DDoS networking traffic generation. The throughput average in Fig. 8 serves as a basis for using multiple nodes in a DDoS attack that simulates attackers in different regions of the MENTORED testbed. The thoughput peak has been reached in the presence of 21 pods employed to simulate attackers and the throughput average is close to 210 Mbps. Higher than 21 pods, a reduction in throughput is observed even under an increase in the number of attackers. The MENTORED testbed users should consider this limit when defining their experiments. Similar results should be expected to other nodes of the testbed. The MENTORED testbed documentation will include the information about these limits to guide users in the experiment definition and benefit from the full capacity of the testbed infrastructure to scale attackers in DDoS attacks.

VI. CONCLUSION

Experimental environments are essential in investigating cybersecurity issues, such as Distributed Denial of Service of Things, i.e., DoT attacks. However, designing and implementing such environments are challenging given limitations in scale, particularly for this specific type of attack. This work presented the MENTORED framework, a reference to designing scalable testbeds for DoT investigation. The framework defines requirements, actors, entities, and modules composing a cybersecutity testbed for DoT, their relations, and scalability and performance management. This work also presented an instantiation of the framework, called the MENTORED testbed, deployed over the national-wide Academic Brazilian network. Evaluations of the MENTORED testbed followed a DDoS attack scenario composed of a web server and attackers using NGINX and hping3 software. Experiments could be easily defined and sent through the Portal using a simple and flexible syntax. A unique high-performance server could effectively reproduce the traffic generation of up to 21 simulated small devices in a controlled scenario where the attackers and server were in the same Kubernetes worker.

REFERENCES

[1] K. Doshi, Y. Yilmaz, and S. Uludag, "Timely detection and mitigation of stealthy DDoS attacks via IoT networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2164–2176, 2021.

[2] F. Alsubaei, A. Abuhussein, and S. Shiva, "Quantifying security and privacy in internet of things solutions," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*. IEEE, 2018, pp. 1–6.

[3] A. A. Santos, M. Nogueira, and J. M. F. Moura, "A stochastic adaptive model to explore mobile botnet dynamics," *IEEE Communications Letters*, vol. 21, no. 4, pp. 753–756, 2017.

[4] A. Khalimonenko and O. Kupreev, "DDoS attacks in Q1 2017," A10 Networks, Tech. Rep., 2017.

[5] B. H. Schwengber, A. Vergütz, N. G. Prates, and M. Nogueira, "Learning from network data changes for unsupervised botnet detection," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 601–613, 2022.

[6] F. Nakayama, P. Lenz, and M. Nogueira, "A resilience management architecture for communication on portable assisted living," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2536–2548, 2022.

[7] F. Nakayama, P. Lenz, A. LeFloch, A.-L. Beylot, A. Santos, and M. Nogueira, "Performance management on multiple communication paths for portable assisted living," in *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021, pp. 340–348.

[8] J. Steinberger, B. Kuhnert, C. Dietz, L. Ball, A. Sperotto, H. Baier, A. Pras, and G. Dreo, "DDoS defense using MTD and SDN," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2018, pp. 1–9.

[9] A. L. Santos, C. A. V. Cervantes, M. Nogueira, and B. Kantarci, "Clustering and reliability-driven mitigation of routing attacks in massive iot systems," *Journal of Internet Services and Applications*, 2019.

[10] S. Hameed and U. Ali, "Efficacy of live ddos detection with hadoop," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2016, pp. 488–494.

[11] B. M. Rahal, A. Santos, and M. Nogueira, "A distributed architecture for ddos prediction and bot detection," *IEEE Access*, vol. 8, pp. 159 756–159 772, 2020.

[12] S. Siboni, V. Sachidananda, Y. Meidan, M. Bohadana, Y. Mathov, S. Bhairav, A. Shabtai, and Y. Elovici, "Security testbed for internet-of-things devices," *IEEE transactions on reliability*, vol. 68, no. 1, pp. 23–44, 2019.

[13] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *USENIX security symposium (USENIX Security)*, 2017, pp. 1093–1110.

[14] P. Schwaiger, D. Simopoulos, and A. Wolf, "Automated iot security testing with seclab," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*. IEEE, 2022, pp. 1–6.

[15] A. Gluhak, S. Krco, M. Nati, D. Pfisterer, N. Mitton, and T. Razafindralambo, "A survey on facilities for experimental internet of things research," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 58–67, 2011.

[16] B. C. Şenel, M. Mouchet, J. Cappos, O. Fourmaux, T. Friedman, and R. Mcgeer, "Edgenet: a multi-tenant and multi-provider edge cloud," in *Proceedings of the 4th International Workshop on Edge Systems, Analytics and Networking*, 2021, pp. 49–54.

[17] J. Wroclawski, T. Benzel, J. Blythe, T. Faber, A. Hussain, J. Mirkovic, and S. Schwab, "Deterlab and the deter project," in *The GENI Book*. Springer, 2016, pp. 35–62.

[18] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele *et al.*, "Fit iot-lab: A large scale open experimental iot testbed," in *IEEE World Forum on Internet of Things (WF-IoT)*. IEEE, 2015, pp. 459–464.

[19] X. Sáez-de Cámara, J. L. Flores, C. Arellano, A. Urbieta, and U. Zurutuza, "Gotham testbed: a reproducible IoT testbed for security experiments and dataset generation," *arXiv preprint arXiv:2207.13981*, 2022.

[20] C. Dietz, M. Antzek, G. Dreo, A. Sperotto, and A. Pras, "Dmef: Dynamic malware evaluation framework," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2022, pp. 1–7.

[21] A. Tekeoglu and A. Ş. Tosun, "A testbed for security and privacy analysis of IoT devices," in *IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, 2016, pp. 343–348.

[22] G. Sasirekha, T. Adhisaya, P. Aswini, J. Bapat, and D. Das, "Challenges in the design of an iot testbed," in *2019 2nd international conference on intelligent Communication and computational techniques (ICCT)*. IEEE, 2019, pp. 14–19.

[23] M. M. Yamin, B. Katt, and V. Gkioulos, "Cyber ranges and security testbeds: Scenarios, functions, tools and architecture," *Computers & Security*, vol. 88, p. 101636, 2020.

[24] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar *et al.*, "The design and implementation of open {vSwitch}," in *USENIX symposium on networked systems design and implementation (NSDI)*, 2015, pp. 117–130.