RIO: A Denial of Service Experimentation Platform in a Future Internet Testbed

Igor Drummond Alvarenga* and Otto Carlos M. B. Duarte*
*Universidade Federal do Rio de Janeiro – GTA/COPPE/UFRJ
Email: {alvarenga,otto}@gta.ufrj.br

***Abstract*—Denial of service attacks (DoS) present a serious threat to the security of communication network infrastructure. Whereas the dissemination of new DoS attacks is fast and dynamic, the development of countermeasures to these attacks requires the construction and validation of a test environment that accurately reproduces Internet behavioral patterns. The construction of this environment often requires more time investment from security professionals than the time spent with the development of associated attack countermeasures. This work introduces RIO (Resource Orchestration Infrastructure) platform. RIO platform enables security professional a realistic and flexible approach to DoS experimentation creation and setup. This platform automates the creation and setup of virtual machines, virtual switches, network links, attackers, experimental data collection and experiment execution, based on a single document defining the experiment in a descriptive language. The proposal efficacy is supported by RIO platform ease of use and time reduction towards the acquisition of experimental results.**

I. Introduction

A Denial of Service (DoS) attack is defined as any explicit attempt to deny the legitimate use of a service or a resource. DoS attacks and it's distributed variant (DDoS) are a serious threat to the security of communication networks infrastructures, as the interval between detection and reaction to an attack may result in serious damage to a service, even resulting in financial losses. Moreover, the number of DoS attacks is increasing consistently in the last decade. This increase is influenced by the crescent availability of DDoS for hire services, that offers DDoS in a cheap commodity available to a wider audience [1]. Furthermore, the technical improvement of DoS attacker technology is facilitated by renovation and increase of Internet transmission capacity, coupled with the easy testing of new attack mechanisms directly on the Internet, without the requirement of a dedicated test environment [2].

The development of effective DDoS and DoS countermeasures to reduce or eliminate attack effects requires the construction of an effective testing environment. This environment must portray a realistic behavior akin to the Internet and the complete set of devices and systems involved in the offer of specific service to be protected. The construction of this environment often demands more time investment from security professionals than the development of the targeted attack countermeasures [3]. Furthermore, a proposed DoS defense mechanism requires validation against the possible attack scenarios and conditions ranges and, then, an accurate measurement of the mechanism efficiency against its target attacks, its computational cost, and its probability of disrupting legitimate traffic through false positives may be evaluated. These validations steps take further time of security professionals. Therefore, there is an evident disparity between the difficulty in developing a new DoS attack, and in the researching and devising an effective countermeasure for said attack. Hence, the development of effective denial of service countermeasures is slower and more expensive then the development of DoS attacks. This makes the development of tools to facilitate research and testing of DoS countermeasures a necessity.

In this paper, we propose the Resource Infrastructure Orchestration (RIO) platform as tool to facilitate and speed up the development of effective denial of service countermeasures. The main objective of the platform is to confer agility to the security professional in performing the multiple steps necessary for DoS research and new DoS countermeasure proposals evaluation, ultimately reducing the time spent in repetitive configuration tasks and aiding DoS countermeasure development as a whole. In order to accomplish this objective, through a single input configuration file written in descriptive language, the platform offers a automation of: (i) environment creation and setup; (ii) network configuration; (iii) experimental data measurement; (iv) attacker configuration; (v) legitimate user configuration; (vi) experiment execution and control; and (vii) data pre-processing and preliminary analysis. Furthermore, the tools provide make it easy to re-calibrate and repeat similar scenarios. In this way, the security professional may focus his efforts in the specification of the correct test environment, and in the development of his proposed DoS countermeasures. Moreover, our proposed platform was designed with virtualized testbed environments in mind, as we consider ways of providing realism to the experimental scenarios without the usually cost-prohibitive approach of replication parts of internet topology. The RIO prototype was developed and tested in the Future Internet Testbed with Security (FITS) [4].

In the development of this platform, we considered the management and orchestration facilities of many research testbeds. When compared to PlanetLab facilities [5], RIO offers a similar network virtualization approach, but confers more flexibility to test scenarios by employing machine virtualization technologies. Furthermore, RIO provides a centralized configuration file, while PLanetLab needs separate manual configuration of all resources and experiment subsystems. Ofelia [6], Emulab [7] and GENI [8] testbeds also offers vir-

978–1–5090–4671–3/16/$31.00

tualization and somewhat centralized configuration, but lacks experiment automation. Emulab approach is further developed by DETER [9], a security focused testbed for medium scale experiments. DETERS offers experiment resource provisioning and automation using the Montage AGent Infrastructure (MAGI) [1], which inspired the development of RIO orchestration platform. RIO differs from MAGI in centralizing all experimental configuration in a single file, and in providing a state machine based approach to experiment execution, which provides conditional execution based on event triggers, while MAGI approach to experimentation is procedural. RIO also aims to simplify the definition of large topologies allowing a high-level network definition when desired, interchanged with specific low-level network interface configuration when needed. Furthermore, resource provisioning and environment setup in RIO is faster when compared to DETER, because it does not involve physical machine provisioning, physical network slicing, and base image copies for each experiment node.

The rest of this paper is structured as follows. We present denial of service research considerations and challenges in Section II. In Section III we discuss FITS testbed environment. RIO platform architecture is presented in Section IV. Finally, we conclude and present future work in Section V.

II. DoS Testbed Challenges

The evaluation methods employed in denial of service evaluation directly define how useful test results can be in predicting DoS defense mechanism behavior in a real scenario. The main objective in a DoS defense mechanism evaluation is proving that it is effective. Thus, security professionals must prove that: (i) service is denied by a DoS attack in the absence of the proposal; (ii) denial of service is significantly reduced or eliminated in presence of the proposal; (iii) legitimate user traffic is not significantly hindered by the presence of the proposal at all times [3]. Therefore, an evaluation process is defined by: (i) a test methodology, which could be a theoretical model, a simulation, an emulation or a real environment deployment; (ii) test-case scenarios that model legitimate traffic and behavior, malicious traffic and behavior, and network topology; (iii) one or more success metrics that can prove a proposal effectiveness in mitigating a target attack.

Theoretical model approaches are based in the construction of a behavioral model of arbitrary complexity, in conjunction with a set of mathematical, probabilistic and logic rules. While theoretical modules are useful in elucidation specific problem characteristics, there are no theoretical tools powerful enough to accurately model the complexity of traffic mixes [3]. Simulation based approaches are a popular evaluation mechanism for communication networks, however, problems arise when this methodology is applied to DoS and DDoS research. Denial of service relies in borderline stressful conditions of real hardware and software, in a way that simulated facilities cannot cope because of the computational time trade-off between scalability and fidelity, therefore producing non-realistic results. Emulation, coupled with virtualization, is the preferred approach when experimenting with DoS, as the attacks and defense mechanisms are employed over real systems. The main challenge when employing an emulation-based methodology is the associated configuration time, data gathering and analysis. Real environment deployment is usually not an option, as it requires further configuration complexity and it is cost prohibitive for most researchers and security professionals to mimic Internet scale topologies.

The construction of a realistic testbed environment for DoS attack and countermeasure experimentation must answer five significant challenges: (i) sufficient realism and associated configuration time; (ii) scaled down topology significance; (iii) diversity; (iv) hardware intricacies and interaction; (v) complexity, fine adjustment and evaluation. While the use of emulation and virtualization technologies can cope with the first and fourth challenges, it further increase the other challenges due to emulation characteristics.

III. FITS Testbed as a Test Environment

The Future Internet Testbed with Security was selected as the environment for RIO prototype development as this testbed features can be employed in order to answer the enunciated DoS testbed challenges. FITS allows the creation of arbitrary isolated layer two networks, each one with its own set of characteristics and control environment. Moreover, FITS allows the interconnection of geographically separated physical nodes though the Internet using the concept of Islands, which are defined as a set of physical machines and resources locally interconnected and affiliated with testbed. FITS architectural diagram is presented on Figure 1. Testbed control is centralized in a main island, and all other islands are connected to the main island and to each other using Ethernet over Generic Routing Encapsulation (GRE) tunnels. FITS is a virtualization testbed, based on OpenFlow [10] and Open vSwitch[2], for network virtualization, and Xen hipervisor[3], for machine virtualization.

[2]http://openvswitch.org/
[3]https://www.xenproject.org/

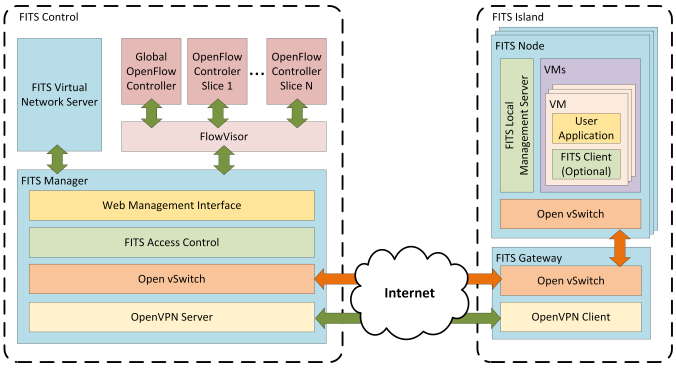

Fig. 1. FITS architectural diagram. Multiple nodes in a FITS island are connected to the centralized controller through a gateway, which performs traffic tunneling to interconnect islands over the Internet. This interconnection behavior is transparent to virtual machines, as they can only perceive the topology intended by the security professional.

[1]http://montage.deterlab.net/magi/magi.html

Thise characteristics establish FITS as an ideal environment for deployment of the RIO platform, as this features can be used to provide sufficient realism for DoS experimentation.

The primitives offered by FITS services were employed by RIO in order provide the full management cycle of a testing environment: provision, instantiate, configure, monitor, and clean. Furthermore, FITS intrinsic characteristics allow RIO to cope with the DoS testbed challenges. The scaled down topology significance challenge is solved for a wide range of testing scenarios when the islands available on FITS are employed to accurately model critical systems topology, while the Internet is used as an intermediary network, as exemplified on Figure 2. This approach is based on a trade-off between control and realism. When using the Internet as the intermediary network, the security professional forfeits control in favor of background traffic and delay realism. At the same time, modeling target attacking and legitimate systems using virtualization provides the necessary level of control for a target proposal evaluation. FITS cope with the diversity challenge as it's based on non-uniform commercial off-the-shelf (COTS) hardware, offering a wide array of systems and deployment configurations possible.

There are still challenges associated with virtual machine and virtual switch homogeneity that require special care of security professional during result evaluation. One of the main challenges in RIO platform lies in the execution of experimentation related to distributed denial of service attacks. As FITS testbed is built upon virtualization, and thus virtual nodes are built upon a shared resource pool, the capability of traffic absorption by virtual nodes is limited when compared to dedicated systems. Furthermore, a large-scale distributed attack may cause denial of service at some point in the intermediary network and pose a challenge of result evaluation if the experiment is not cautiously planned, taking into account island geographic distribution and interconnection limitations. As a countermeasure, FITS traffic shaping and isolation primitives may be employed by the security professional for the creation an scaled-down version of the real scenario, which may prove sufficient for the evaluation of certain DoS mitigation mechanisms. RIO provides configuration parameters to control outbound network interface bandwidth allocation.

IV. Proposed RIO Architecture

The Resource Infrastructure Orchestration platform is designed as set of tools that provide configuration, orchestration and monitoring facilities for experiments on testbeds based on network and machine virtualization. The main objective of RIO is the automation of DoS related experiments, handling experimental and environmental parameters setup, calibration and reconfiguration, thus, reducing the time associate with these tasks from the perspective of security professionals. Moreover, RIO architecture aims to face the main challenges associated with DoS experimentation in virtualized environments. Although the platform may be adapted to experimentation in other areas, it falls out of the scope of this paper.

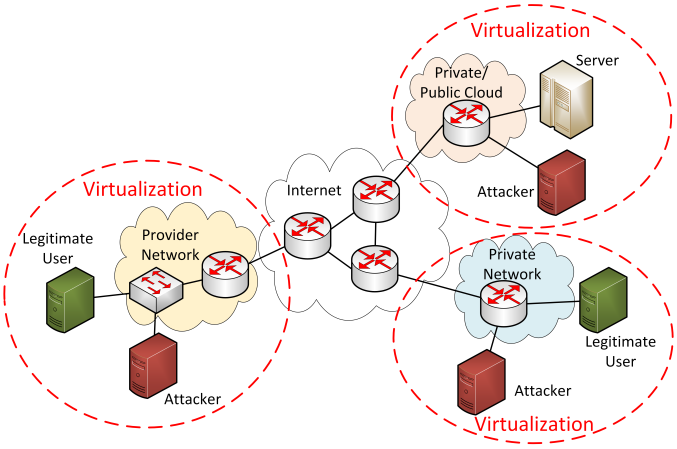

Fig. 2. RIO critical topology virtualization approach. The critical topology pertinent to a DoS experiment is virtualized using testbed facilities, while the Internet is employed as the Intermediary network, as it would be in the real scenario.

A. Modules

Each module of the proposed arqchitecture is designed as an independent service, and each service communicate with its peers using a remote procedure call (RPC) message interface, as depicted in Figure 3. The security professional creates and manages the experiment using a unified interface, which centralize all information pertaining to the experiment and module configuration. The main modules that integrate RIO platform are control, orchestration, data collection and analysis. The orchestration module is aided by a set of experimentation modules according to demand.

Control Module: The control module is the main architectural module, and it is responsible for the coordination of the other modules. The control module takes a unified descriptive document (UDD) as input. This document contains: (i) the experimental topology; (ii) each network and network component description; (iii) desired data collection points and post collection analysis; and (iv) experimental event sequence. Based on this document, the control module decides and commands resource provisioning for correct experiment execution, as well as setup facilities to be employed by the monitoring modules. Next, the control module delegate experiment execution to the orchestration module. After experiment completion, the control module collects execution metrics and

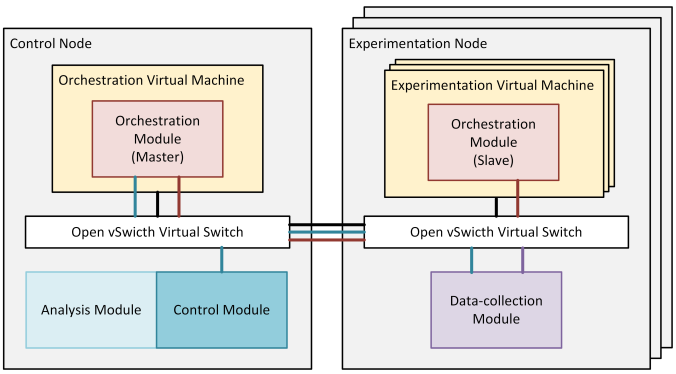

Fig. 3. RIO architecture and communication diagram. Different line colors indicate independent and isolated communication channels between platform modules.

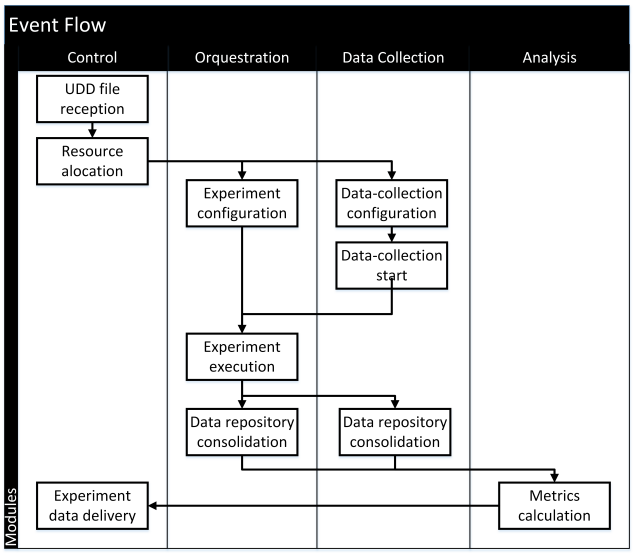

Fig. 4. RIO experimental flow diagram. The experimental flow starts and ends at the control module, while the bulk of experiment execution is delegated to orchestration and data collection modules.

generated data associated with all platform modules, and sends the gathered data to the analysis module. The complete set of generated experimental data and supplemental analysis is compressed to an output file, and this file is made available to the requesting security professional at the platform data repository. At last, the control module frees all allocated resources tied to the specific experiment. The experimental flow diagram is presented on Figure 4.

Orchestration Module: The orchestration module is inspired in DETER and GENI control mechanisms. This module is deployed as a distributed service, with one designated component in each virtual machine tied to the experiment, in addition to a management component tied to a specialized virtual machine to be used exclusively for a single experiment orchestration purposes. This centralized architecture simplifies experiment orchestration an synchronization tasks, moreover, it provides strong consistency during experiment orchestration [10], which is essential for the conservation of event parallel chronology. All communication pertaining to orchestration module components is executed as remote procedure calls in a dedicated isolated network, thus, it does not cause interference in experimental communication performance. The orchestration module previously calculates the necessary network configuration of the orchestration network tied to the experiment. Then this information is codified in the last four octets of each virtual network interface MAC address, and conveyed to the orchestration component located in every virtual machine at boot time. This enables the platform to avoid delays associated with network configuration, as well as using the same disk image for all experimental nodes. Based on the configuration derived from the UDD, the orchestration module verifies and configures each network element on all nodes, and then proceed to collect initial performance data on each virtual network link. Next, the orchestration module will employ experimentation modules in each node to conduct experiment execution following the events defined in the UDD.

Data Collection Module: A data collection module is deployed in each physical machine of the testbed. These modules are activated by the control module at the beginning of the experiment, and work in parallel to the orchestration module during an experiment execution. This module main attribution is performing network packet capture on all designated virtual interfaces, based on experiment configuration. This packet data is complemented by virtual machine resource usage data, such as CPU load, memory utilization and other available metrics, when requested by the security professional in the metrics section of the UDD. The experimental data is kept at a local repository in each physical node during the experiment execution, and is transferred to a centralized location only after experiment completion, to avoid interference with experimental packet flows.

Analysis Module: The analysis module is only activated after experiment completion and operates over the complete experimental data set. Its main attribution is data pre-processing and calculation of a supplemental set of metrics derived from experimental bulk data. This metrics are defined at the analysis section of the UDD, and are attached to the data set after calculation. After completion of all requested operations, the analysis module informs the control module of experiment completion.

Experimentation Modules: Experimentation modules are the core of an experiment, performing every action described in the experimental steps of the UDD. Each experimentation module is responsible for managing the execution of a well-defined experimental function according to the UDD parameters specification. All these modules are located inside the experiment-associated virtual machines, tied to the local orchestration module. Example of those functions may include configuration and deployment of DoS and DDoS attacks; configuration and deployment of DoS defense mechanisms; configuration and deployment of legitimate user behavior; and assorted traffic generation and measurement tools. Some of these modules are to be supplied with the initial version of the platform, while it is expected that the security professionals will provide modules associated with their respective defense mechanism proposal.

B. Experimental Disk Image

A specially constructed disk image was built to host the orchestration module. This image contains all software related to the orchestration and experimentation modules, thus, any virtual node can use it during an experiment. Nevertheless, in order for this image to be used, it must attain different behavior at boot time depending on the role expected from it. The distinct behaviors associated with experiment management and experiment execution associated with the orchestration module are enabled based on virtual machine configuration at boot time, which enables communication in a dedicated network and allows the configuration of remaining experimental parameters This image is based on Debian Linux as the operational system, but it was altered in a way to keep all modifications to hard disk content in memory, preserving the original data. This

design decision enables all virtual nodes to be created with the same boot disk, avoiding a disk copy to each node necessary in an experiment, and thus greatly speeding up the setup of large experimental virtual networks, as the only file needed to start a new virtual machine is the configuration file. Other associated benefit of this strategy is the fixed requirement of storage space associated with virtual node provisioning for physical nodes. This enables the physical node to avoid using shared storage, which affects virtual machine performance directly, as it ties virtual disk response times the network transmission delay. Furthermore, once all physical nodes install this disk image on the same path, live virtual machine migration is possible without any constraints. Nevertheless, the memory requirements of virtual nodes grows to accommodate storing file system changes. This need to be estimated and provisioned according prior to experiment execution.

C. Descriptive Language

In order to manage the complete experiment life cycle, RIO platform relies on a unified descriptive document to infer all the necessary resource provisioning, allocation and configuration, as well as to derive the experiment execution model. The JavaSctipt Object Notation (JSON) standard was selected to be used as the descriptive document syntax in RIO platform as it is an human readable format, and also supported almost all programming languages and major platforms. Using JSON syntax and RIO defined semantics, the security professional must specify four key elements of an experiment: nodes, network, behavior and measurement. The nodes key has semantic elements to define virtual machine resources as CPU, memory, base image and placement. The network key has semantic elements to define network topology, as well as network interfaces configuration for every node and bandwidth constraints of virtual elements. It also employ semantic facilitators to build known topologies with minimal user input. The behavior has semantic elements to describe an experiment as a state machine, where events are caused by, and may cause, triggers. Each behavior is associate with an agent group and experimentation module, and then nodes are included in one or more agent groups. The measurement keys has semantic elements to identify and configure every data collection point, as well as post processing and analysis.

V. Conclusion

RIO platform allows the security professional the realistic experimentation with denial of service scenarios in order to investigate attack patterns and validate proposed DoS mitigation mechanisms. RIO simplifies and significantly reduces the configuration overhead associated with the environment and experiment setup by employing automation techniques and a centralized configuration file for complete experiment lifecycle automation. The platform focuses on providing agility to the security professional in performing the necessary steps for DoS experiment facilitation, ultimately reducing the time spent in repetitive configuration tasks, thus allowing the security professional to focus on his DoS mitigation mechanism proposal and evaluation.

One of the main advantages in employing a centralized configuration document is associated with the repeatability of designed experimental scenarios, in a way that simplifies comparing different DoS mitigation proposals under the same conditions. Moreover, the variation of experimental parameters is simplified and it becomes easy to repeat the experiment varying specific parameters to obtain insight on how those parameters affects a specific DoS countermeasure proposal. Another advantage of centralizing experiment configuration lies in the facilitation of information sharing and exchanging in the research community, enabling easier collaboration and knowledge construction, thus speeding up DoS mitigation research consequently.

For future work, we will quantify the impact of each step of experiment automation on overall research time, as well as further investigate the possibility of automation and experimentation within the platform constraints.

Acknowledgment

Work supported by CT-Mon/RNP, CAPES and CNPq.

References

[1] M. Karami and D. McCoy, "Understanding the emerging threat of ddos-as-a-service," in *6th USENIX Workshop on Large-Scale Exploits and Emergent Threats*. Berkeley, CA: USENIX, 2013. [Online]. Available: https://www.usenix.org/conference/leet13/workshop-program/presentation/Karami

[2] M. Geva, A. Herzberg, and Y. Gev, "Bandwidth distributed denial of service: Attacks and defenses," *Security and Privacy, IEEE*, vol. 12, no. 1, pp. 54–61, janeiro de 2014.

[3] J. Mirkovic, S. Fahmy, P. Reiher, and R. Thomas, "How to test dos defenses," in *Conference For Homeland Security, 2009. CATCH '09. Cybersecurity Applications Technology*, março de 2009, pp. 103–117.

[4] I. M. Moraes, D. M. Mattos, L. H. G. Ferraz, M. E. M. Campista, M. G. Rubinstein, L. H. M. Costa, M. D. de Amorim, P. B. Velloso, O. C. M. Duarte, and G. Pujolle, "Fits: A flexible virtual network testbed architecture," *Computer Networks*, vol. 63, no. 0, pp. 221–237, 2014, special issue on Future Internet Testbeds - Part II. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128614000036

[5] A. Bavier, M. Bowman, B. Chun, D. Culler, S. Karlin, S. Muir, L. Peterson, T. Roscoe, T. Spalink, and M. Wawrzoniak, "Operating system support for planetary-scale network services," in *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*, ser. NSDI'04. Berkeley, CA, USA: USENIX Association, 2004, pp. 19–19. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251175.1251194

[6] OFELIA FP7 Consortium. (2010) Ofelia fp7 project. Disponível em http://www.fp7-ofelia.eu/ (Acessado em 20/10/2015).

[7] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *OSDI02*. Boston, MA: USENIXASSOC, dezembro de 2002, pp. 255–270.

[8] M. Berman, J. S. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, and I. Seskar, "Geni: A federated testbed for innovative network experiments," *Comput. Netw.*, vol. 61, pp. 5–23, março de 2014. [Online]. Available: http://dx.doi.org/10.1016/j.bjp.2013.12.037

[9] J. Mirkovic, T. Benzel, T. Faber, R. Braden, J. Wroclawski, and S. Schwab, "The deter project: Advancing the science of cyber security experimentation and test," in *Technologies for Homeland Security (HST), 2010 IEEE International Conference on*, novembro de 2010, pp. 1–7.

[10] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, março de 2008. [Online]. Available: http://doi.acm.org/10.1145/1355734.1355746