IEICE TRANS. INF. & SYST., VOL.E101–D, NO.3 MARCH 2018

PAPER

# Cybersecurity Education and Training Support System: CyRIS

**Razvan BEURAN**[†a)], **Cuong PHAM**[†*], **Dat TANG**[†**], ***Nonmembers***, **Ken-ichi CHINEN**[†], **Yasuo TAN**[†], ***Members***, ***and*** **Yoichi SHINODA**[†], ***Nonmember***

**SUMMARY** Given the worldwide proliferation of cyberattacks, it is imperative that cybersecurity education and training are addressed in a timely manner. These activities typically require trainees to do hands-on practice in order to consolidate and improve their skills, for which purpose training environments called *cyber ranges* are used. In this paper we present an open-source system named CyRIS (Cyber Range Instantiation System) that supports this endeavor by fully automating the training environment setup, thus making it possible for any organization to conduct more numerous and variate training activities. CyRIS uses a text-based representation in YAML format to describe the characteristics of the training environment, including both environment setup and security content generation. Based on this description, CyRIS automatically creates the corresponding cyber range instances on a computer and network infrastructure, for simultaneous use by multiple trainees. We have evaluated CyRIS in various realistic scenarios, and our results demonstrate its suitability for cybersecurity education and training, both in terms of features and performance, including for large-scale training sessions with hundreds of participants.

***key words:*** *cybersecurity training, cyber range, education support system*

## 1. Introduction

The online availability of malware and cyberattack tools lead to the fact that it is nowadays possible even for non-experts to wage successful cyberattack campaigns. This has lead to an escalation in the number and scale of recent cyberattacks, with significant negative effects on society as a whole. Examples include the DDoS attack on the Dyn DNS provider in the U.S. using the Mirai IoT botnet that occured in October 2016—the largest DDoS attack to date, with traffic exceeding 1.2 Tbps, that resulted in the inaccessibility of several high-profile websites; and the WannaCry ransomware campaign in May 2017—the largest ransomware attack to date, with over 300,000 computers infected in 150 countries.

In such circumstances it is mandatory to intensify the cybersecurity training and education efforts, so that organizations are able to defend themselves against such large-scale attacks. Training activities typically use hands-on practice to improve the skills of participants, performed in training environments called *cyber ranges*. Currently, these environments are created mostly manually by security experts, and their content is generally static. Moreover, many of the cyber range systems are proprietary, creating vendor lock-in and offering little freedom to the organizations that deploy them, while also being very expensive.

In this paper we introduce CyRIS (Cyber Range Instantiation System), an open-source tool that tackles the aforementioned issues by providing a flexible, scalable and low-cost mechanism for managing security training environments. The key insight of our approach is to employ an easy-to-understand text-based representation of the cyber range, that is used by CyRIS to automatically create the training environment on a computer and network infrastructure.

The main contributions of the paper are as follows:

- Propose a text-based representation of cyber range environments that is both human and machine readable, and also easily extensible;
- Present the design and implementation of the training support system CyRIS that is able to automatically deploy cyber ranges corresponding to this representation;
- Discuss the evaluation of the said instantiation system in various scenarios, both from feature and performance characteristics perspectives.

A preliminary version of this work was presented in [1]. The novelty of the current paper arises from three main aspects: (i) a more thorough description of CyRIS; (ii) new performance evaluation results, including large-scale training setup for up to 600 participants; (iii) details on user feedback following the use of our system by third parties.

The remainder of this paper is organized as follows. In Sect. 2 we provide more context by introducing CyTrONE, a training framework that is powered by CyRIS, and some related work. Then, in Sect. 3 we present in detail the design and architecture of CyRIS, including the proposed cyber range description format. In Sect. 4 we discuss the methodology and results from a thorough evaluation of CyRIS, both in terms of features and performance. The paper ends with conclusions, acknowledgments and references.

## 2. Background

In April 2015, the Cyber Range Organization and Design NEC-endowed chair was created at the Japan Advanced Institute of Science and Technology with the mission of advancing cyber range creation technologies. In this context

Manuscript received June 30, 2017.
Manuscript revised October 13, 2017.
Manuscript publicized November 24, 2017.
[†]The authors are with Japan Advanced Institute of Science and Technology, Nomi-shi, 923–1292 Japan.
[*]Presently, with Institut Mines-Télécom SudParis, France.
[**]Presently, with Delivion GmbH, Germany.
a) E-mail: razvan@jaist.ac.jp
DOI: 10.1587/transinf.2017EDP7207

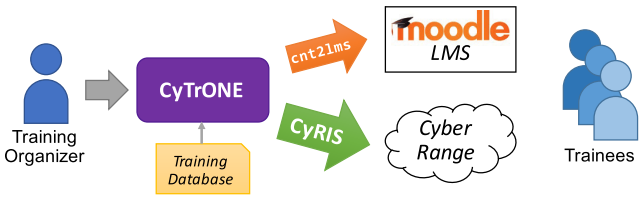


**Fig. 1** Overview of the CyTrONE integrated cybersecurity training framework that is powered by CyRIS.

we are developing the CyTrONE training framework [2], for which CyRIS serves as a core component.

Figure 1 shows the overall architecture of CyTrONE. Based on input from the training organizer and a training database, CyTrONE uploads the training content to a Learning Management System (LMS) via the helper tool called `cnt2lms`, and also creates the associated training environment via CyRIS. Trainees can then access the LMS to consult the training content, connect to the cyber range to conduct the necessary investigation, and subsequently provide answers to questions via the LMS.

The training support system presented in this paper, CyRIS, is addressed to training organizers and instructors. However, CyRIS by itself does not provide a mechanism for interacting with the trainees. For this purpose, we have also developed the framework discussed above, which employs CyRIS for the creation and management of the training environment, but also relies on the LMS interface to address the needs of question display, trainee score management, etc. Basically, CyTrONE is a kind of meta-system, that is able to manage higher-level training operations, and delegates some lower-level tasks to other tools, in particular the cyber range instantiation task to CyRIS.

However, CyTrONE is only a use case for CyRIS, and any training program could use CyRIS as a back end, while having its own custom user interface (UI) for the participants. To use some examples from Japan, Hardening Project [3] is a security contest organized by the Web Application Security Forum, in which teams of IT professionals compete in terms of the service level they can provide for a virtual e-commerce company. Consequently, Hardening Project has no interface for learning, and uses a web UI for presenting contest results, such as the amount of virtual sales, or technical score. Another popular security contest in Japan, named SECCON [4], is a Capture The Flag (CTF) type of competition, and its custom UI presents users with challenges they have to solve in order to obtain points, shows their scores, etc. Although both these programs have different mechanisms for interaction with the participants, they could both use CyRIS as a back end to simplify the environment setup task.

### 2.1 Related Work

As for existing tools that are related to CyRIS, they can be divided into three categories, as discussed next:

1. *Configuration of individual nodes*: Installing and configuring programs, setting up the system, etc., but lacking features such as network service and topology configuration: Ansible [5], Vagrant [6], Chef [7];
2. *Configuration of node sets (cloud)*: Creating and managing either physical hosts or virtual machines (VMs) and the network service amongst them, however lacking features of configuring the settings and content of individual nodes: OpenStack [8], VMware vSphere [9] for VMs, SpringOS [10] for physical hosts;
3. *Full node and network configuration*: To the best of our knowledge, Alfons [11] is the only tool in this category, in particular related to security training, albeit not having security content generation functionality *per se*.

Given that Alfons is the system most similar to CyRIS, we shall briefly introduce it here. According to [11], Alfons was designed to address four main requirements: (i) facility management and virtual/physical node support; (ii) network parameter configuration; (iii) installation of content; (iv) environment separation on a network testbed. We conclude that Alfons features are mainly oriented towards environment setup; moreover, it is only able to install content that has been prepared in advance, but not to generate any new security content, e.g., through malware or cyberattacks.

As it will be described in the next section, CyRIS is a system that aims to combine the features of the above tools to provide the same usage convenience, but also provisions specific security features and training content generation, so as to offer additional flexibility to the organizers in the context of security training.

## 3. CyRIS Overview

The role of CyRIS is essential in the context of cybersecurity education and training, as it enables the automatic creation of cyber range training environments in a dynamic manner, and in a relatively short time, based on a cyber range description. Since it requires no significant effort and no advanced technical knowledge on the side of the organizers, the use of CyRIS makes it possible to conduct training sessions in various circumstances, and for a large number of trainees.

To facilitate the training activities in various organizations, CyRIS was released as open-source software via GitHub (`https://github.com/crond-jaist/cyris`). The only necessary resources for using CyRIS are one or more base VM images to be used for the cyber range guests—for which a sample is also available via GitHub—and an infrastructure for the cyber range, such as off-the-shelf servers.

Figure 2 presents an overview of the CyRIS architecture. The input to CyRIS are the cyber range description in YAML format, as well as VM disk images from the *VM Image Pool*, as specified in the cyber range description. The first processing stage of CyRIS is called *Base VM Preparation*, in which the base VM images to be employed are

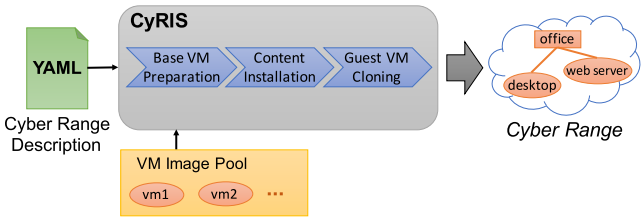


**Fig. 2** Overview of the CyRIS input, output, and internal architecture.

copied from the pool and their basic setup is conducted. The second stage, called *Content Installation*, is the main processing step, when all the additional settings are applied, such as account creation, software installation, attack emulation, etc. The third and final stage is called *Guest VM Cloning*, and it refers to the actual deployment of multiple cyber range guest VMs based on the prepared base VM images.

To run CyRIS one has to deploy it on the servers to be used for cyber range creation, which should all use the Ubuntu OS. One of the servers plays the role of *master*, and manages the cyber range creation on all the other servers. Currently, CyRIS supports KVM virtualization technology, and the CentOS operating system for the cyber range guests; nevertheless, training environment setup features are also available for Ubuntu OS guests.

### 3.1 Cyber Range Description

The input format for CyRIS is based on the YAML format [12], with a series of custom tags used to define the characteristics of the cyber range to be created. Each file must contain the following sections:

- *Host settings*: Describe the host(s) on which the cyber range is to be created, such as the IP address used to access the host, the virtual bridge IP address, and the user account; a user-defined id is used to refer to the host elsewhere in the file;
- *Guest settings*: Describe the guest(s) the cyber range is composed of, with two classes of information:
  - *Base VM information*: Details about the base VM image that will be used to create the guest, such as the location of the base VM image file and its properties; a user-defined id is used to refer to the guest elsewhere in the file;
  - *Setup tasks*: Details about the setup operations that are to be performed on the guest during the cyber range instantiation process, such as user account operations, software installation, security incident emulation, etc.;
- *Clone settings*: Describe the manner in which the cyber range composed of the above guests is to be created, such as the total number of cyber range instances, the number of clones for each guest type, the interconnecting network topology, etc.

**Table 1** Summary of the training environment setup and security content generation features of CyRIS, along with the related description tags.

**Training Environment Setup**

| *Function* | *Description tag(s)* |
|---|---|
| Account Management | `add_account`; `modify_account` |
| Tool Installation | `install_package` |
| File Copy | `copy_content` |
| Program Execution | `execute_program` |
| Network Configuration | `topology`; `networks`; `members` |

**Security Content Generation**

| *Function* | *Description tag(s)* |
|---|---|
| Firewall Configuration | `firewall_rule` |
| Malware Emulation | `emulate_malware` |
| Attack Emulation | `emulate_attack` |
| Traffic Capture | `emulate_traffic_capture_file` |

We summarize the most essential features of CyRIS in Table 1, together with the tags that are to be used in the cyber range description file in order to activate them. Since training environment setup functionality is easier to understand, we'll detail only the security content generation functions, which are specific to CyRIS. Thus, Firewall Configuration can be used to add predefined rules to the guest OS firewall, so that trainees can learn how to analyze such settings from a security point of view. Malware Emulation functionality can be used to run malware-like software; currently we include two dummy malware modules, one for creating an artificial CPU load, and one for listening to a given port, both providing the "symptoms" of malware without being actually malicious. Attack Emulation functionality includes three types of emulated attacks with configurable parameters, such as target or source IP, and so on: (i) SSH dictionary attack; (ii) DoS attack; (iii) DDoS attack; this functionality is further extensible via the program execution function, which allows running any real malware or attack tools. Finally, Traffic Capture makes it possible to store a network packet trace in a file for trainees to conduct forensics analysis on, for instance containing the traces of an attack conducted using the attack emulation features.

Figure 3 shows an example of a CyRIS cyber range description file. Its sections have the following content:

- In the section `host_settings`, the file defines a host with the id "host_1" and certain IP settings, that is to be accessed via the account "cyuser".
- Then, in the section `guest_settings`, it defines a guest with the id "dektop" to be created based on the configuration file "basevm_desktop.xml" located on "host_1". The tasks that are to be carried out to prepare the guest are: account creation for user "daniel"; package installation for "wireshark"; attack emulation via a number of 123 ssh login attempts; and malware emulation by means of a dummy process called "DAEMON" that will use 45% of CPU. An additional guest with the id "webserver" is also defined, which uses the base image configuration file "basevm_webserver.xml". Due to space limitations we only show one sample task, namely the execution of the

script "install_wordpress.sh" via the interpreter "bash";
- Finally, in the section `clone_settings`, the file describes that 30 instances of this cyber range will be created on "host_1". Each cyber range will include 1 "desktop" and 1 "webserver" guest, with the former being used as entry point for logging into the cyber range. Both guests will be connected to the network segment named "office" via the interface "eth0".

The example provided demonstrates that the cyber range description can be easily understood and modified if needed, either manually or through automated processing. The use of ids to refer to elements makes it easy to define the relationships between the logical components of the CyRIS cyber range model, such as hosts, guests, or networks. Moreover, users do not have to be concerned with low-level details, such as IP address settings, as these are carried out automatically by CyRIS depending on the composition and topology of the cyber range. We expect that, in the future, a GUI could be used to generate such description files in an even more convenient manner, thus eliminating the need to remember the keywords and file structure.

### 3.2 Workflow

To facilitate the understanding of CyRIS, we present its detailed workflow in Fig. 4. The program starts by checking whether the input description file is syntactically and semantically correct. Then the *Base VM Preparation* stage is initiated, with steps such as copying the disk image(s) from the base VM pool, and starting them. If the start is successful, some basic setup operations are conducted, such as setting up ssh access, hostname, network connectivity, etc.

The second stage, *Content Installation*, consists first of all of performing all the setup tasks, such as installing content into the base VMs or emulating attacks; these steps are performed sequentially as described in the input file. The phase called "Post-cloning setup" however is carried out only *after* the cloning process ends; this is essential for performing configurations that depend on the properties of each cloned VM, such as its IP address, etc.

The third stage, *Guest VM Cloning*, refers to creating multiple cyber range instances that include guests based on the prepared based VMs—for use by trainees to conduct the same training simultaneously—and configuring their network topology. For this purpose, the configured base images are copied in parallel to all the hosts on which the cyber range instances are to be instantiated using the `parallel-scp` command; if a cyber range instance contains multiple base images, then they are also copied in parallel. After the cloned VMs are started, the user accounts and passwords for accessing the cyber range are randomly generated and the settings are applied. Note that network topology configuration between cloned VMs takes place at Layer 3, via IP address based routing through firewall rules. In the future, Layer 2 topology could also be configured via VLAN mechanisms.

```
---
- host_settings:
  - id: host_1
    mgmt_addr: 172.16.123.234
    virbr_addr: 192.168.122.1
    account: cyuser
- guest_settings:
  - id: desktop
    basevm_host: host_1
    basevm_config_file: /kvm/basevm_desktop.xml
    basevm_type: kvm
    tasks:
    - add_account:
      - account: daniel
        passwd: daniel_passwd
    - install_package:
      - package_manager: yum
        name: wireshark
    - emulate_attack:
      - attack_type: ssh_attack
        target_account: daniel
        attempt_number: 123
    - emulate_malware:
      - name: DAEMON
        cpu_utilization: 45
        mode: dummy_calculation
  - id: webserver
    basevm_host: host_1
    basevm_config_file: /kvm/basevm_webserver.xml
    basevm_type: kvm
    tasks:
    - execute_program:
      - program: /database/install_wordpress.sh
        interpreter: bash
- clone_settings:
  - range_id: 123
    hosts:
    - host_id: host_1
      instance_number: 30
      guests:
      - guest_id: desktop
        number: 1
        entry_point: yes
      - guest_id: webserver
        number: 1
      topology:
      - type: custom
        networks:
        - name: office
          members: desktop.eth0, webserver.eth0
```

**Fig. 3** Example of a CyRIS cyber range description file.

During the entire setup process, CyRIS records the return values for all internal and external operations, and logs all the commands and their output. In case an error is detected from an invalid return value, an appropriate message is displayed. Otherwise, the cyber range creation is successful, and cyber range login information and internal details are provided to the user for reference.

**Table 2** Functionality coverage of the NIST Technical Guide to Information Security Testing and Assessment [13] via CyRIS training environment setup and security content generation features.

| Information Security Testing and Assessment Techniques | Training Environment Setup | | | | | Security Content Generation | | | |
|---|---|---|---|---|---|---|---|---|---|
| | *Account Management* | *Tool Installation* | *File Copy* | *Program Execution* | *Network Configuration* | *Firewall Configuration* | *Malware Emulation* | *Attack Emulation* | *Traffic Capture* |
| Documentation Review | | | ○ | | | | | | |
| Log Review | | ○ | | ○ | | | | ○ | |
| Ruleset Review | | ○ | | ○ | | ○ | | | |
| System Configuration Review | ○ | ○ | ○ | ○ | | ○ | | | |
| Network Sniffing | | ○ | ○ | | ○ | | | ○ | ○ |
| File Integrity Checking | | ○ | ○ | ○ | | | | | |
| Network Discovery | | ○ | | | ○ | | | ○ | |
| Port and Service Identification | | | | ○ | ○ | ○ | ○ | | |
| Vulnerability Scanning | | ○ | ○ | | ○ | ○ | ○ | ○ | ○ |
| Wireless Scanning | | ○ | ○ | | | | | ○ | ○ |
| Password Cracking | ○ | ○ | | | | | | | |
| Penetration Testing | ○ | ○ | ○ | ○ | ○ | ○ | ○ | | |
| Social Engineering | ○ | | ○ | | | | | | |

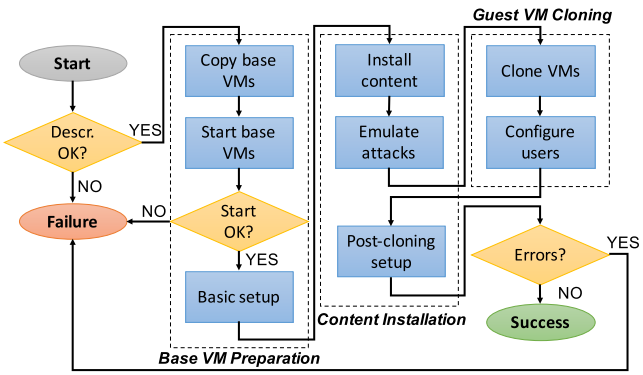


**Fig. 4** Detailed processing workflow of CyRIS.

## 4. Evaluation

The evaluation of CyRIS was conducted as follows:

- *Functionality evaluation*: We compared the functionality of CyRIS with the review techniques described in the U.S. NIST Technical Guide to Information Security Testing and Assessment [13], as well as with some of the related tools;
- *Performance evaluation*: We assessed the performance of CyRIS in different training scenarios through experiments conducted on the large-scale network testbed StarBED [14];
- *User evaluation*: We summarized feedback about CyRIS from external users who have employed it in practice.

### 4.1 Functionality Evaluation

Next we evaluate the functionality of CyRIS, both in terms of its features, and also in comparison with other tools.

(1) Feature Overview

The first type of evaluation we present refers to the functionality of CyRIS, more precisely its features in relation with the types of training that we envisage it being used for. Our reference is the U.S. NIST Technical Guide to Information Security Testing and Assessment [13], which includes three classes of technical assessment techniques:

1. *Review techniques:* Documentation review, log review, ruleset review, system configuration review, network sniffing, and file integrity checking;
2. *Target identification and analysis techniques:* Network discovery, network port and service identification, vulnerability scanning, and wireless scanning;
3. *Target vulnerability validation techniques:* Password cracking, penetration testing, and social engineering.

In Table 2 we show the manner in which the cyber range creation features of CyRIS, both related to training environment setup and security content generation, can be combined in order to set up environments that cover the requirements for conducting training related to the techniques above.

To construct the table, for each technique we assess which CyRIS features are necessary in case an instructor intends to teach and evaluate the given technique. In the *Review techniques* category, for teaching how to conduct Documentation Review, the file copy feature can be used to copy the relevant documents into the cyber range, so that the trainees can review them. For Log Review purposes, instructors could use the attack emulation feature to conduct an attack that will leave traces in the OS log files, or optionally use tool installation and program execution to install and execute any other software that may be necessary for logs not related to attacks (e.g., mail or database server logs), logs which can then be investigated by the trainees. For Ruleset Review, firewall configuration functionality can be used to modify firewall settings, or other tools could be installed and executed as above to modify the rulesets for other components (router access control, IDS/IPS, etc.), so that trainees can learn how to review these types of rulesets. For System Configuration Review, in addition to the

features mentioned above, account management may be required to create users, and file copy may be required to install predefined configuration files in the cyber range for services that are not supported through other methods; trainees can then review these configuration files and decide whether they are safe or not. Network Sniffing requires first of all network configuration functionality for network setup, and traffic capture for saving packet traces; in addition, attack emulation could be used to generate attack traces, or custom tools for any other kind of communication that is to be captured and analyzed (command & control, etc.). Finally, File Integrity Checking training requires to prepare appropriate files in the cyber range if necessary, either through the file copy functionality if predefined files exist, or through generation via custom tools.

For the *Target identification and analysis techniques* category, Network Discovery can only be conducted once the network was configured, and the tools needed by trainees, such as `nmap` or `wireshark` were set up via tool installation. For Port and Service Identification training, one also needs to set up the network, and also execute specific programs (services) that will use the ports to be identified; depending on the training scenario, firewall configuration and/or malware emulation may be necessary in order to prevent/allow communication with outside systems or have a service running on a designated port. Vulnerability Scanning is one of the most complex set of techniques, and all security content generation functions of CyRIS can be used for it; required tools and files may also need to be installed, and network needs to be configured. Wireless Scanning training has similar requirements, however, since CyRIS does not currently support real wireless communication, several of the modules mentioned above are not necessary/usable.

Finally, for the *Target vulnerability validation techniques* category, Password Cracking training requires account management functionality for creating the account for which the password is to be discovered, and tool installation to set up a password cracking software such as "John the Ripper". Penetration Testing requires the use of all the environment setup features to prepare the environment with appropriate vulnerabilities, as well as firewall configuration to open the necessary ports; malware emulation may also be used to introduce vulnerabilities via malware execution. As for Social Engineering, although currently it has no extended support in CyRIS, account management should be used to create the target users, and file copy can be used to place sensitive information in their accounts.

Based on Table 2, we conclude that CyRIS provides functionality that is broad enough to support all the training topics related to security testing and assessment, at least as they were envisaged when the said guideline was released.

Training instructors can use the rich set of features of CyRIS in order to create environments appropriate for learning the corresponding skills. The actual learning methodology, however, depends on the characteristics of each particular training program, and is outside the scope of CyRIS. Nevertheless, the CyTrONE framework that we discussed in Sect. 2 represents a valuable use case in this respect.

**Table 3** Comparison between CyRIS and related tools.

| Tool Names | Node Content | | Network Topology | |
|---|---|---|---|---|
| | *Basic Setup* | *Security Content* | *Physical Host* | *Virtual Machine* |
| Ansible, Chef, Vagrant | ○ | | | |
| OpenStack, SpringOS | | | ○ | ○ |
| Alfons | ○ | | ○ | ○ |
| CyRIS | ○ | ○ | | ○ |

(2) Comparative Analysis

We have also conducted a comparative analysis with respect to several of the related tools presented in Sect. 2.1, and its summary is provided in Table 3. Tools such as Ansible, Chef, or Vagrant are only useful for setting up virtual machines, but have no features related to security content, nor for network topology. OpenStack and SpringOS, on the other hand, are focused on network environment setup, and have no intrinsic features related to the setup of the virtual machines themselves.

The closest tool to CyRIS in terms of features is Alfons, which however can only install content prepared in advance, but cannot generate security content (see Sect. 2.1). Therefore, the main advantages of CyRIS with respect to Alfons are its various security generation features (firewall configuration, malware emulation, attack emulation, traffic capture) that make it possible to conduct attacks on demand, capture attack traces, generate log files, change firewall rules, all done dynamically, without the need for security experts to prepare such content in advance. Moreover, we note that Alfons is a closed-source software, hence it does not provide the freedom coming from the open-source nature of CyRIS.

The only currently not supported feature in CyRIS is the ability to setup the physical host network topology, for instance, via VLAN mechanisms. We decided not to tackle this aspect yet as it would create a dependency on the physical network infrastructure on which CyRIS is used, since different switches have different interfaces for VLAN setup. An alternative would be to leverage existing tools for performing such configurations on a particular environment, such as SpringOS or Alfons on StarBED.

In the context of this comparative analysis, we would also like to discuss the reasons why CyRIS is a better alternative than the simple aggregation of other tools. It is possible to imagine that by combining software such as Chef and OpenStack with some security penetration tools, such as those included in the Kali Linux distribution [15], one could configure an environment appropriate for cybersecurity education and training. However, the orchestration of a large number of heterogeneous tools would require managing many different configuration files, as well as the interaction and synchronization between these tools, hence it would require advanced technical knowledge and skills from the instructors, and it would be error-prone.

On the other hand, CyRIS provides only one configu-

ration file for managing the entire process, hence it can be used even without advanced skills, hence one can easily deploy a training environment in a reliable manner. Here are just a few examples of the convenience of CyRIS: (i) attacks can be emulated simply by specifying the type of the attack and its key characteristics; (ii) the traces of the attack will appear in various system logs without any further action from the instructor; (iii) by enabling traffic capture, packet traces become available as files in the cyber range for trainees to analyze. To achieve the same results without CyRIS, one would have to somehow set all the parameters for the attack tools, synchronize the attacks with the packet capture tools, then copy the resulting packet traces in the environment via complex scripts. Doing all this in a multi-user environment with multiple training activities going on simultaneously would be very challenging indeed.

In addition, CyRIS also provides some original functionality, for instance by means of the malware emulation module. In order to make possible to use CyRIS even for participants with limited security abilities, we decided to provide a solution that avoids the use of real malware, so as to prevent inadvertent leaks, given that malware would have dangerous consequences outside the cyber range. Consequently, we have implemented a dummy program that only listens to a given port, or creates an artificial CPU load, thus recreating malware symptoms without any potential negative effects. Of course, actual malware can also be used if desired via the program execution functionality of CyRIS.

### 4.2 Performance Evaluation

The second kind of evaluation we present refers to the cyber range instantiation performance characteristics of CyRIS.

(1) Large-scale Experiments

As we are having discussions with technical colleges in Japan for integrating our system with their cybersecurity program to be initiated nationwide, we envisage the need to set up many parallel training sessions for hundreds of students.

To assess the CyRIS execution performance in such scenarios, we have conducted experiments on the large-scale network testbed StarBED [14]. The performance evaluation was done using two training scenarios, as follows:

**Level 1** A basic training which includes topics such as log and system configuration review, network sniffing, and vulnerability scanning; one guest VM, playing the role of a desktop PC, is needed for each trainee;

**Level 2** A training of medium difficulty on topics such as network discovery, password cracking, and penetration testing; two guest VMs, playing the roles of a desktop and a web server, are required for this scenario.

To study the scalability of CyRIS we decided to keep the number of VMs per host constant (namely, 20), and assess the execution performance for a large-scale scenario with up to 600 VMs on 30 hosts (representing 600 cyber range instances for Level 1, or 300 cyber range instances for Level 2). The results are shown in Figs. 5 and 6, for Level 1 and 2, respectively. For the purpose of our evaluation we have divided the cyber range instantiation process into its three stages, as discussed in Sect. 3: preparation of base VMs, installation of content, and cloning of the guest VMs.

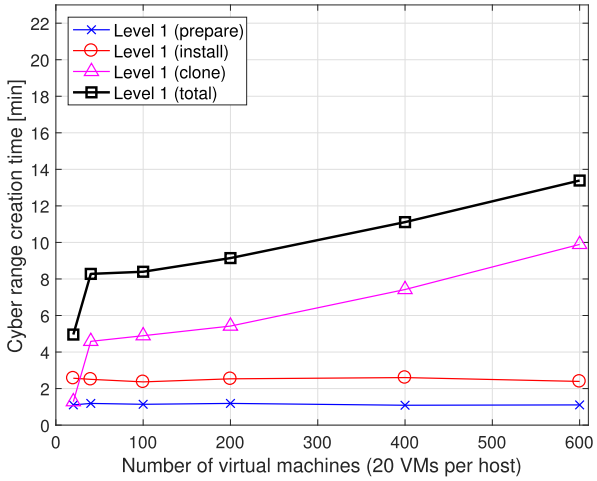


**Fig. 5** Cyber range creation time versus the total number of VMs for training scenario Level 1 (up to 30 instantiation hosts, and 600 trainees).

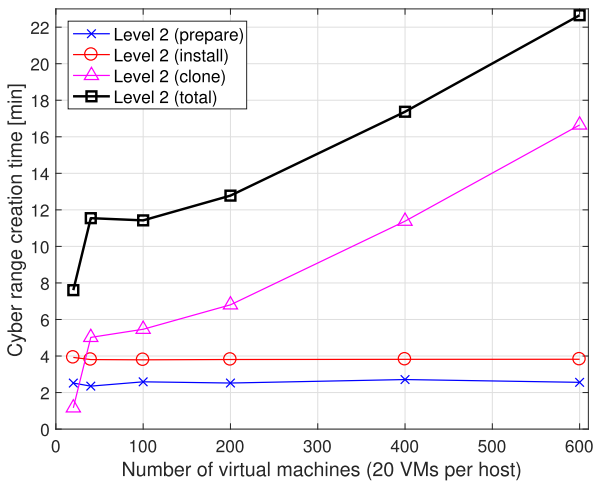


**Fig. 6** Cyber range creation time versus the number of VMs for training scenario Level 2 (up to 30 instantiation hosts, and 300 trainees).

The experimental results for Level 1 (Fig. 5) show that preparation and installation times are relatively constant and small: about 1 minute for preparation, and 2.5 minutes for installation; this is expected given that these operations are performed locally on the master CyRIS host. The cloning time becomes non-negligible when more than one host is used, due to the need to copy the base VM image files to the other CyRIS hosts. The cloning time has a slightly exponential increase with a maximum of about 10 minutes for 30 hosts (600 trainees). The total time for the above three operations is dominated by the cloning time, but it is under 14 minutes even for the maximum number of hosts.

The experimental results for Level 2 (Fig. 6) indicate

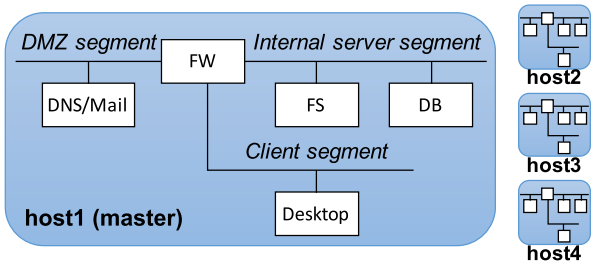


**Fig. 7** Setup for evaluation experiments based on the topology in [11].

**Table 4** Performance comparison between Alfons and CyRIS.

| Guest Name | Alfons | CyRIS |
|---|---|---|
| | *Disk Image Size [GB]* | |
| Firewall (FW) | 1.4 | 2.0 |
| DNS/Mail | 3.3 | 3.4 |
| File Server (FS) | 5.0 | 5.1 |
| Database Server (DB) | 4.7 | 4.8 |
| Desktop Client | 10.0 | 10.2 |
| Total Size | 24.4 | 25.6 |
| PER INSTANCE TIME [min] | 27.1 | N/A |
| TOTAL TIME [min] | 112.6 | 13.2 |

again flat preparation and installation times: about 2.5 minutes for preparation, and 4 minutes for installation. The increases with respect to Level 1 are expected given the higher complexity of the scenario (two VMs instead of one, more complex content). Cloning time again dominates, and the exponential behavior is more obvious, as two files have to be copied in this case instead of one, hence increasing the network congestion; still, the cloning time is under 17 minutes even for 30 instantiation hosts (300 trainees). The total instantiation time for Level 2 is of about 22 minutes.

We consider the total instantiation time results reasonable given the typical 10 minute breaks between classes, and the time needed to explain the activity to students. Thus, if we consider a reasonable and relatively large number of 100 trainees, our results demonstrate that creation can be finished in under 10 minutes for Level 1 (100 VMs), and in under 15 minutes for Level 2 (200 VMs).

(2) Comparison Experiments with Alfons

As Alfons is the most similar tool in terms of features to CyRIS, we have also conducted a comparison with the experiment results reported by the authors in [11]. We use the same network topology and 4 deployment hosts for the cyber range, as reported in the said paper (see Fig. 7).

A summary of the results is presented in Table 4. For the purposes of this comparison, we tried to match the disk image size used when evaluating Alfons, but there was not enough information available to exactly match the content to be installed in each VM, so we could only approximate it. The upper part of the table shows the disk sizes for the base VM image used for each type of guest, and the bottom part shows the average execution times, per instance and in total.

One important note about the results is that Alfons is creating the cyber range instances sequentially, whereas CyRIS is conducting this operation in parallel—hence we don't measure the instance creation time. Therefore, it is more fair to compare the total CyRIS execution time (about 13 min) to the Alfons execution time per instance (about 27 min), not to the total time (about 112 min). Even in this case tough, the execution of CyRIS is twice as fast as that of Alfons, mostly thanks to its increased parallelism.

**Table 5** Comparison of cyber range creation times for different VM image file formats and number of deployment hosts.

| Measured Time | VM Image File Format & Size | | | |
|---|---|---|---|---|
| | *RAW, 8.1 GB* | | *QCOW2, 2.1 GB* | |
| | *1 host* | *2 hosts* | 1 host | 2 hosts |
| Prepare Time [s] | 111.6 | 113.6 | 37.4 | 37.5 |
| Install Time [s] | 82.3 | 84.3 | 88.4 | 89.0 |
| Clone Time [s] | 34.6 | 363.9 | 38.6 | 208.2 |
| TOTAL TIME [s] | 228.5 | 561.7 | 164.4 | 334.7 |

(3) Performance Optimization

We have started considering various alternatives for optimizing even more the performance of CyRIS. Given that the base VM copying time—reflected in the preparation and cloning times—is significant, we focused on this issue first. In order to reduce the copy time, we decided to change from the RAW image file format for the base VM image to the alternative QCOW2 format. With the QCOW2 format, the VM image size on disk matches the actual used data size in the guest VM, not the logical disk size, as with the RAW format. Therefore, the VM image file size is reduced when using QCOW2 compared to the RAW format. On the other hand, write operations are slower when using the QCOW2 format if the image file size on disk needs to grow due to an increase in the used data size, since the operating system needs to allocate new blocks on the disk, thus leading to a potential execution performance penalty.

To investigate the performance characteristics of this optimization approach, we have conducted experiments on 2 servers in JAIST, using either one or both of them as deployment hosts, and a representative sample description file. The RAW image file we used so far had a 8.1 GB size, and was reduced to 2.1 GB when using the QCOW2 format.

In Table 5 we present experimental results that illustrate the influence of VM image file format and size on cyber range creation time (averages over 5 runs). For the case of 1 host, the preparation time is reduced by 66% when switching from RAW to QCOW2 format, due to a shorter time for copying the VM image file locally, whereas the install and clone times increased by 7% and 11%, respectively, due to the slower write operation issue mentioned above. Overall, the total creation time is reduced by 28%.

For the case of 2 hosts, the preparation time is reduced by 67% when changing from RAW to QCOW2 format, whereas the install time increased by 7%, for the same reasons mentioned for the 1 host case. As for the cloning time, in this case it decreased by 43%. The explanation is that in the case of 2 hosts the base image prepared on the master host needs to be copied to the other host via network;

hence, the cloning time for multi-host setups is also affected by the change in file size. The total creation time for 2 hosts decreased by 40%, demonstrating the positive effect of file format change on execution performance.

We believe that through additional optimization we can reduce even more the total cyber range creation time. While local copy time could be further reduced through the use of SSD storage—albeit an expensive solution—we are currently investigating the possibility to tackle the long cloning time due to the remote copying of base VM image files from the master to the other hosts by allowing each host to prepare its own base VM (after an initial replication of the VM image pool on all hosts, which only needs to be done once before the very first training). This approach would eliminate the need to copy the VMs from the master host during cloning, as it is done now, and would eliminate the exponential increase seen in Fig. 6; thus, we expect that the total creation time results would become relatively flat for any number of hosts. However, an inconvenient in this case would be that all the hosts need to be provided access to the repositories containing the required packages and tools to be installed, which may pose a security risk in some circumstances.

### 4.3 User Evaluation

In addition to the internal users of CyRIS (about a dozen members of our group in JAIST who do research on related topics), we have also had the opportunity to conduct a user evaluation with external users.

In particular, four students of the Tokyo Metropolitan Technical College came to JAIST for an internship in March 2017. They have used CyRIS in order to create security training environments inspired by AppGoat, a public web application security training tool created by the Information-technology Promotion Agency (IPA), Japan [16].

Each of the four students attempted to build a cyber range containing a website with security vulnerabilities that can be accessed by trainees in order to validate their security skills, especially for penetration testing. Two of the students only had a webserver in their environment (based on Apache `httpd`), but the other two students decided to also include a desktop guest, on which they could install several tools needed to conduct the penetration testing (e.g., Wireshark, John the Ripper, etc.).

All the four students succeeded in creating their target cyber ranges using CyRIS, and their feedback reassured us that our system is relatively easy to use. Students did encounter some difficulties in setting up the environment inside the cyber range, in particular the webserver, but these issues were mainly caused by differences in versions of OS, webserver software and browser compared to those their were used to, or otherwise lack of a deeper technical knowledge regarding the tools their were using (e.g., how to configure `httpd` settings, etc.).

Overall, the students considered the internship very useful, as the use of CyRIS freed them from the need to address low-level environment setup aspects, thus letting them focus more on training content creation. Moreover, one of them decided to continue using CyRIS for his research activity. In addition, some minor issues that were discovered on that occasion regarding multi-user support helped us improve CyRIS before its public release in April 2017.

We would also like to mention two other milestones:

- CyRIS was used in a public demonstration conducted at Interop Tokyo 2017, where it was nominated Best of Show Award Finalist. The reception was hugely positive, with many visitors recognizing the importance of our endeavor; several of them also acknowledged the difficulties they had in setting up training events, and informed us they shall try our system in the future;
- A visualizer named “CyRIS-vis” was created to display the state of the cyber range as it is being created, such as guests and the network topology, including a real-time update of their status—whether the guests are running or not, the virtual network interfaces are active or not, etc. This visualizer greatly improves the usability of CyRIS, and will be the topic of a future paper.

## 5. Conclusions

In this paper we have presented CyRIS, a system that we have designed and implemented in order to support cybersecurity education and training. CyRIS makes it possible to automatically setup the training environment necessary for hands-on activities related to cybersecurity education and training using only a text-based cyber range description file.

CyRIS was implemented using the NIST Technical Guide for Information Security Testing and Assessment as a reference, and it supports all the techniques described in the guideline. In term of performance, CyRIS can create training sessions for hundreds of participants in around 20 minutes, and our extensive use of parallelism makes it superior to similar tools. Moreover, a user evaluation conducted via internships demonstrated the usability of CyRIS, and helped us fix some minor issues before its public release, which took place in April 2017.

CyRIS is still under development and we shall keep making various improvements in short term. One topic we plan to focus on is support for cloud infrastructures such as Amazon Web Services (AWS), so that users can deploy cyber ranges on demand in the cloud, thus eliminating the need to purchase and maintain deployment servers. One other important future direction that we plan to pursue is the standardization of the cyber range representation format that we proposed with CyRIS, so that security training vendors can use it as a common format; for this purpose we are planning to establish a consortium with our existing partners.

## Acknowledgments

We wish to acknowledge the assistance of Yoshio Niimi in performing some of the experiments presented in this paper.

## References

[1] C. Pham, D. Tang, K. Chinen, and R. Beuran, "CyRIS: A Cyber Range Instantiation System for Facilitating Security Training," Proceedings of the International Symposium on Information and Communication Technology (SoICT), pp.251–258, Dec. 2016.
[2] R. Beuran, C. Pham, D. Tang, K. Chinen, Y. Tan, and Y. Shinoda, "CyTrONE: An Integrated Cybersecurity Training Framework," Proceedings of the International Conference on Information Systems Security and Privacy (ICISSP 2017), pp.157–166, 2017.
[3] Web Application Security Forum, "Hardening Project (in Japanese)," 2017. http://wasforum.jp/hardening-project/
[4] Japan Network Security Association (JNSA), "Security Contest (SECCON) (in Japanese)." http://seccon.jp/
[5] Red Hat, Inc., "Ansible is Simple IT Automation," 2017. https://www.ansible.com/
[6] HashiCorp, "Vagrant – Development Environments Made Easy." https://www.vagrantup.com
[7] Chef, "Chef – Automate Your Infrastructure," 2017. https://www.chef.io/chef/
[8] The OpenStack Foundation, "OpenStack – Open Source Cloud Computing Software," 2017. https://www.openstack.org/
[9] VMware, "Server Virtualization with VMware vSphere." http://www.vmware.com/products/vsphere.html
[10] T. Miyachi, K. Chinen, and Y. Shinoda, "StarBED and SpringOS: Large-scale General Purpose Network Testbed and Supporting Software," Proceedings of the International Conference on Performance Evaluation Methodologies and Tools (Valuetools 2006), 2006.
[11] S. Yasuda, R. Miura, S. Ohta, Y. Takano, and T. Miyachi, "Alfons: A Mimetic Network Environment Construction System," Proceedings of the 11th EAI International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities (TridentCom 2016), vol.177, pp.59–69, 2016.
[12] C. Evans, "The Official YAML Website," 2017. http://www.yaml.org/
[13] K. Scarfone, M. Souppaya, A. Cody, and A. Orebaugh, "National Institute of Standards and Technology – Technical Guide to Information Security Testing and Assessment," 2008.
[14] National Institute of Information and Communications Technology, Japan, "Hokuriku StarBED Technology Center," 2017. http://starbed.nict.go.jp/en/index.html
[15] Offensive Security, "Kali Linux – Penetration Testing and Ethical Hacking Linux Distribution," 2017. https://www.kali.org/
[16] Information-technology Promotion Agency, Japan, "Vulnerability hands-on learning tool AppGoat (in Japanese)." https://www.ipa.go.jp/security/vuln/appgoat/

**Razvan Beuran** received the B.Sc., M.Sc. and Ph.D. degrees from Politehnica University, Bucharest, Romania in 1999, 2000 and 2004, respectively; the Ph.D. degree was delivered jointly with Jean Monnet University, Saint Etienne, France. From 2001 to 2005 he was with CERN, Geneva, Switzerland, and from 2006 to 2015 he was with the National Institute of Information and Communications Technology, Hokuriku StarBED Technology Center, Japan. Since 2015 he is Research Associate Professor with the Japan Advanced Institute of Science and Technology, Ishikawa, Japan. His current research topics are cybersecurity education and training, and network experiment methodologies. He is a member of IEEE.

**Cuong Pham** received his B.Sc. in Software Engineering from Post and Telecommunication Institute of Technology, Vietnam, and his M.Sc. in Information Science from Japan Advanced Institute of Science and Technology in 2015 and 2017, respectively. He is currently a research engineer with Institute Mines-Télécom SudParis, France. His research interests include distributed computing, network security, and network traffic engineering.

**Dat Tang** received the B.E. from Vietnam National University, University of Engineering and Technology, Hanoi in 2014, and the M.Sc. in Information Science from Japan Advanced Institute of Science and Technology in 2017. From 2014 to 2015, he worked as a Linux system engineer. Dat is currently cloud platform engineer with Delivion GmbH, Germany. He is interested in Linux, computer networks, network security, and distributed systems.

**Ken-ichi Chinen** received B.E. degree from Ryukyu University, Okinawa, Japan in 1992. He received M.E. and D.E. degrees from Nara Institute of Science and Technology in 1995 and 1998, respectively. From 1998 to 2003, he was an assistant professor in the Graduate School of Information Science, Nara Institute of Science and Technology. Since 2003 he has been an assistant professor, and from 2009 associate professor in School of Information Science, Japan Advanced Institute of Science and Technology. His research interests include performance evaluation of computer network services, and simulation/emulation technology for network testbeds. He is a member of the IPSJ, and IEICE.

**Yasuo Tan** was born in 1965. He received his Ph.D. from Tokyo Institute of Technology in 1993. He joined Japan Advanced Institute of Science and Technology (JAIST) as an assistant professor of the School of Information Science in 1993. He has been a professor since 1997. He is interested in IoT Systems, especially Smart Home Systems. He is chairman of the Technology and Standardization Subcommittee of the Smart IoT Acceleration Forum, expert committee member of the Information and Communications Council, chairman of the Green Grid Platform at Home Alliance, an advisory fellow of the ECHONET Consortium, and a member of IEEE, ACM, IPSJ, IEICE, IEEJ, JSSST, and JNNS.

**Yoichi Shinoda** received his B.E., M.E. and Ph.D. from Tokyo Institute of Technology in 1983, 1985 and 1989, respectively. He joined Japan Advanced Institute of Science and Technology in 1991 as a professor of the School of Information Science. His research interests include distributed and parallel computing, networking systems, operating systems, and information environments.