Master's Thesis

Cloud Deployment Support for Cybersecurity Training

1810427 ZHANG ZHE

Supervisor	Assoc. Prof. Razvan Beuran
Main Examiner	Assoc. Prof. Razvan Beuran
Examiners	Prof. Yasuo Tan
	Assoc. Prof. Shinobu Hasegawa
	Assoc. Prof. Yuto Lim

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

August, 2020

Abstract

In recent years, cloud computing technology has received considerable attention from various fields, playing a vital role in all walks of life. Cybersecurity is always a hot research topic. With the introduction of the cyber range concept, more and more cybersecurity education and training systems have been developed. But most of these systems are implemented on the local computer and server. Although there are many existing cloud-based cybersecurity systems developed in recent years, there are limited researches on how to transform those classic cybersecurity education and training systems into cloud deployment.

Thanks to the current market competition in cloud computing, various cloud vendors have introduced different welfare policies to attract users. This makes it easier for everyone to get the opportunity to use cloud computing services. After comparing the use threshold of some cloud computing platforms (e.g. AWS, Azure, Google Cloud), we have found that the AWS educate account jointly launched by AWS and Vocareum (a third-party service as a manager) is very suitable for research, development, and testing. Thus, we applied for an AWS educate account and used it in our research.

For the research, we proposed an approach to introduce AWS cloud computing into an open-source system named CyRIS (Cyber Range Instantiation system), so that cyber range can be deployed in the AWS cloud platform. CyRIS generates cyber range based on the description file in YAML format automatically on the computer and network infrastructure, the description file includes environment settings and security content. This CyRIS process can be divided into three parts, base VM preparation, content installation, and guest VM cloning. We used EC2(Amazon Elastic Compute Cloud) services to replace the KVM-based phase of base VM preparation and guest VM cloning through the Python SDK Boto3 of AWS (CyRIS is developed based on Python).

In our experiment, firstly, we completed the installation of the original CyRIS to ensure that there are no problems with the environment and programs, and then we prepared the AWS development environment, which includes account registration, import of credentials information, and SDK installation. On the method proposed, we used EC2 service to create instances as base VMs. Then we proceed to the content installation phase. Since the target machine is in the cloud, we have made adjustments to the content installation part of the source program, such as modifying the SSH

login method, modifying the commands of individual tasks, and so on. During the cloning step, we stop the instances that the content has been installed before, and create AMIs (amazon-based image) based on the instances. According to the specified number of clone settings in the description file, launch the instances with the newly created AMIs. How to run the improved CyRIS is basically the same as before. After ensuring that the original CyRIS can run normally, log in the credentials information and run the improved CyRIS program. Then users can get the range notification from the specified directory, according to the information in the range notification, users can access the cyber range.

To evaluate the performance of our improved system, we first conduct a review of the operating status of AWS services. We created a uniform specification cyber range several times per hour of the day and collected all creation time data. Then we calculated the average and standard deviation of each hour's creation time to compare the efficiency and stability of cyber range creation on AWS for finding a suitable time period to create cyber ranges. We create cyber range of different specifications during these times and compared the total cyber range creation time with the original CyRIS. Besides, we compare the details of the time for both kinds of CyRIS. Finally, the performance is analyzed and evaluated through line charts and bar plots with error bars.

Based on the comparison results, we proved that cloud-based CyRIS has higher efficiency while creating multiple base VMs. In addition, the improved CyRIS can use more operating systems for the base VMs because of the convenience that the instance can be launched only through AMI id on the AWS platform. Through the improvement of the source program, the original installation contents are supported in more operating systems (e.g., Red Hat 8, Ubuntu 20.04, Amazon Linux) and almost all of the contents work well. We adjusted the source program to make some tasks that were originally only available on CentOS7 available on all of our AMI operating systems in AWS cloud, such as emulate attacks, emulate malware, and modify firewall rules.

However, we have also discovered some limitations in our experiment. The AWS educate account has some restrictions. It can not keep more than nine instances running at the same time, so we are not able to evaluate the situation of creating big scale cyber range. And due to account restrictions, we unable to confirm how much the used cloud services cost, etc. Besides, because of some security policies of the cloud platform, the real attack emulation task in the original system cannot be inherited by the improved system.

In summary, our research has successfully improved CyRIS so that it can deploy cyber range on the AWS cloud platform based on the description file

in the YAML format. And the method we proposed can also provide some reference value to other classic cybersecurity education and training tools or systems for cloud deployment and conversion.

In future work, we plan to add more installation content to enrich CyRIS. We will also apply for a general AWS account, which has lifted the previous educate account restrictions so that we can make more improvements and tests. For example, we can evaluate the situation of cyber range creation in more kinds of scales. And we can use more AMI from the AWS marketplace. For more secure and reliable user login, we plan to create a separate key pair for each cloned instance in future improvements.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter is a brief introduction to this research, including the related background, research motivations, the goals to be achieved, and the outline of the remaining part.

1.1 Background

With the rapid development of computer network technology and its wide application in various fields, more and more attention has been paid to cybersecurity. Cybersecurity has become one of the most important fields for society, military, government and enterprise. In recent years, various government agencies have invested heavily in cybersecurity education and training. The demand for cybersecurity-related professionals is growing rapidly, so education and training are becoming more and more important in the cybersecurity field, which play a huge role in solving the problem of cybersecurity labor shortage.

Effective education and training require not only theoretical research results, but also practical and hands-on experience. For instance, the trainees are expected to think about the theories behind cybersecurity problems, as well as to have the ability to apply these theories to design, develop and implement innovative related solutions after training. Usually to achieve the goals, some guidelines mentioned in [1] are followed for developing cybersecurity education and training systems or tools as summarized in [2]:

1. The laboratory machines must have an Internet connection in order to download the necessary tools and access online information;

2. The laboratory machines must be isolated from the campus network.

3. The laboratory network environment should be as realistic as possible and be able to perform most popular known cybersecurity training exercises

in the literature

4. The laboratory should be established in the way which is easy to manage, allocate and expand resources for different tasks.

5. The laboratory should be equipped with a sufficient number of IT personnel and technicians to provide adequate maintenance and timely troubleshooting.

6. The laboratory should be distributed naturally, and equally accessible to on- and off-campus users.

Besides, the concept of cyber range has led to a new pattern of cybersecurity education and training. Cyber ranges are interactive, simulated representations of an organization's local network, system, tools, and applications that are connected to a simulated Internet level environment [3]. Tools developed based on cyber range standards are always proven to be effective (e.g., [4] [5]).

Cloud computing is a new innovation in the information age after the Internet and computers. It can be considered that now is the era of cloud computing. Although there are many definitions of cloud computing, in general, the basic meaning of cloud computing is the same. That is, cloud computing has strong scalability and demand, and can provide users with a new experience. The core of cloud computing is to coordinate among many computer resources. Therefore, users can obtain resources on-demand through the network, and the obtained resources are not limited by time and space.

1.2 Motivation

Thanks to the improvement of cloud computing technology, more and more systems based on cloud deployment are applied in various fields. Many of them are geared towards cybersecurity education and training, such as [6] [7] [4].

Various technologies are progressing with cloud, and so can cyber ranges. Clouds provide users a flexible, reconfigurable, and elastic computing infrastructure at an acceptable price. Cyber ranges in cloud can provide a safe, controlled, and isolated environment. In addition, the ranges can scale in size based on task scenarios.

The application of cloud solutions can improve the efficiency of development, otherwise, it will consume a lot of time and cost to purchase, design, build and test computing and network hardware, not to mention the time, energy, and cost required to maintain the server.

As cloud is a new trend, cloud service companies will provide various

benefits to encourage users to use cloud computing platforms. Especially Amazon Web Service (AWS), as the industry leader, AWS has a project named 'AWS Educate', which has credits, training, content, and collaboration for students and educators. The AWS Educate provides four important resources [8]:

1. Grants of AWS credits for use in courses and projects.
2. Free content to embed in courses or to use as-is.
3. Access to free and discounted AWS Training resources.
4. Online and in-person collaboration and networking opportunities.

Using AWS education services is completely free by default, and does not need to bind any payment information. For beginners, learning and using cloud services without any cost is the best choice.

1.3 Objective

In this study, the objective is to use the cloud computing platform to deploy content that can support cybersecurity education and training. At present, there are many mature cybersecurity education and training tools such as [5], [9], [10], among them CyRIS is an open-source KVM-based cybersecurity education and training support system. This research aims to improve CyRIS to support cloud computing platform deployment.

Since the improvement is based on the cloud, the original cybersecurity education and training tools that are based on virtual machine or server deployment, which should also be compatible with the improved system. In addition, cloud-deployed content will be provided to users through cloud computing platforms. Thus our goal is to establish an effective method to automatically generate cybersecurity education and training content on the cloud.

Due to the difference in the overall structure of cloud-based and locally deployed cybersecurity tools when generating education and training contents, integrating cloud computing services into the original system is the primary task, which is the most important in this research. Finally, the improved system would be tested and compared with the original system, and a specific discussion would be conducted.

1.4 Contributions

The following contributions can be provided through this research:

- By deploying cyber range in the cloud, users will no longer have to consider the purchase and maintenance of deployment servers.
- CyRIS based on cloud deployment will not need to prepare images for the base VMs, which improves the efficiency of development. And in the cyber range created in the cloud, machines of various specifications can be configured, which does not like the original CyRIS need to prepare the configuration file of the basic VM. In theory, such an improved method of deploying classic cybersecurity systems to the cloud is not only suitable for CyRIS, but can also provide a certain reference for other systems.
- Finally, we will summarize the issues encountered when using cloud services to improve CyRIS and discuss them.

1.5 Outline

This section is an outline of the remaining parts of the study, organized as the following:

- Chapter 2, reviews of cyber range, cloud computing and cloud based cybersecurity training system.
- Chapter 3, problem formulation and similarity assessment approach presentation.
- Chapter 4, experimental details and specific discussion around the results.
- Chapter 5, summary of the whole work and prospects for future perspective.

Chapter 2

Related Work

In this chapter, we focus on the reviews of the cyber range and cloud computing. Firstly, the overview of cyber range. Then, we will introduce a mature cyber range system CyRIS. Next, brief introduction of cloud platforms. Finally, introduce some cybersecurity training systems.

2.1 Cyber Range Overview

Cyber ranges are interactive, simulated representations of an organization's local network, system, tools, and applications that are connected to a simulated Internet level environment. [3]

A cyber range provides an environment to practice computer network operations skills such as penetration testing, defending networks, hardening critical infrastructure, and responding to attacks. It should represent real-world scenarios such as emulating large-scale, complex networks. It should offer isolation from other networks to contain malicious activity. The same environment should also support experimentation and testing for cybersecurity products. [11]

2.2 Cyber Range Instantiation System: CyRIS

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, the related personnel started to develop the CyTrONE training framework [12], for which CyRIS serves as a core component.

CyRIS is an open-source tool for facilitating cybersecurity training by

automating the creation and management of the corresponding training environments (a.k.a, cyber ranges) based on a description in YAML format. [5]

The role of CyRIS is essential in cybersecurity education and training, as it enables the automatic creation of cyber range training environments dynamically, and in a relatively quick 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 diverse 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 2.1 presents the overview of CyRIS architecture. The input to CyRIS is the cyber range description of YAML format, as well as VM 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 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.

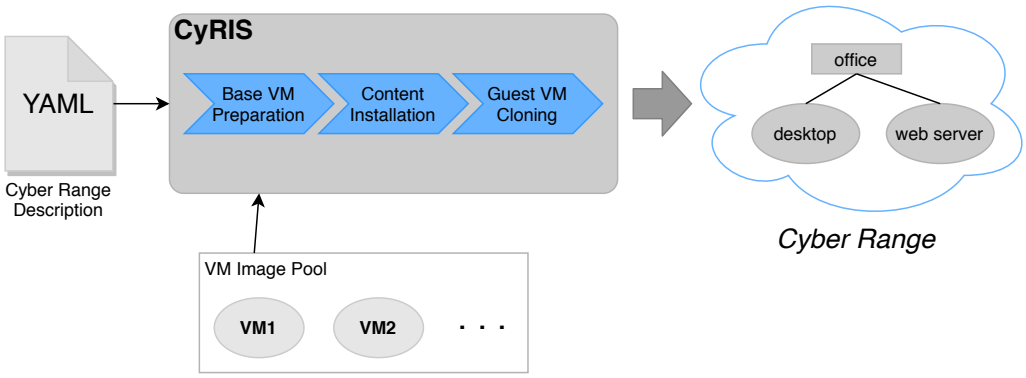

Figure 2.1: Overview of CyRIS

Figure 2.2 presents the detailed workflow of CyRIS. 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.

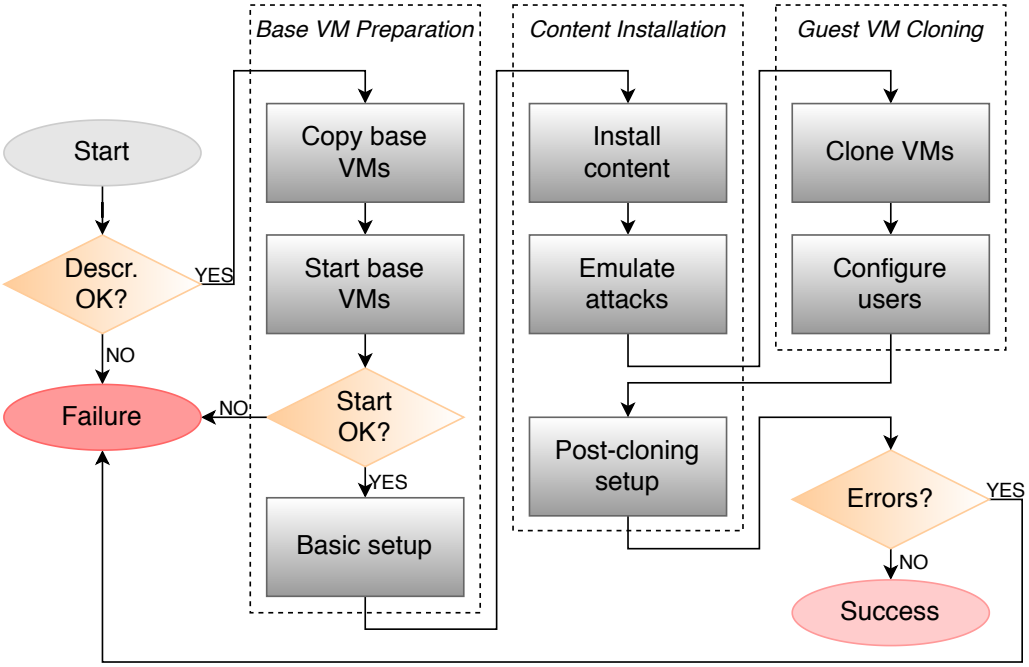

Figure 2.2: Detailed processing workflow of CyRIS

2.3 Cloud Platforms

In this section, we will introduce some popular cloud computing platforms in the market, and further introduce AWS, which is used in our research.

2.3.1 Popular Cloud Platforms

Amazon Web Service

Creative cloud computing from one of the big names.

AWS was founded in 2006. It provides on-demand cloud computing to individuals and organizations.

AWS offers extensive admin controls available via its secure Web client. Users can access several features from here including encryption key creation and auditing. AWS lets you customize infrastructure requirements. This costs far less than if you were set up on your premises. Users can also access EC2 services. This permits users to run and acquire servers as necessary. AWS has three different pricing models; 'Pay as you Go', 'Save when you reserve', and 'Pay-less using more'. For more information about these, users must contact the sale directly.

AWS also offers a free 12-month tier. Once your trial period has expired, you must either choose a paid plan or cancel your AWS subscription.

Microsoft Azure

A wide array of services from a tech giant.

Microsoft Azure was released nearly a decade ago, in 2010. Users can run any service on the cloud or combine it with any existing applications, data centers, or infrastructure. Microsoft Azure provides a wide array of solutions suitable for all types of industries. All your business needs will be taken into consideration. This results in a package better suited for needs. Azure means there is no need to have physical servers on-site. This reduces the usual costs, such as an onsite server support team. The Azure Migration Centre makes cloud transfers faster and easier. The solution is also compatible with Linux.

Microsoft Azure offers a 12-month free tier which includes access to all popular services, $200 (£153.74) credit, and over 25 'Always Free' services. All of Microsoft Azure's prices and plans are laid out in great detail on their site. The page includes a cost calculator and a 'Pay as you go' service. Each plan can be tailored to your specific needs.

Google Cloud

Elastic and inexpensive cloud computing from the genii of Google.

Google Cloud Platform is Google's cloud service provider. The platform enables users to create business solutions using Google-provided, modular web services. It offers a wide array of services including IaaS and PaaS solutions. With Google Cloud's multilayered secure infrastructure, users can rest assured that anything you build, create, code or store will be protected. This is done through a commitment to transparency and a highly trained team of engineers. Google Cloud has a variety of tools to ensure consistent performance and management. These include Compute Engine, App Engine, Container Engine, Cloud Storage and Big Query. Google also offers smooth migration to virtual machines with flexible pricing.

There is a free 12-month trial, which includes $300 (£230.62) towards all services and products offered by Google Cloud Platform.

2.3.2 Amazon Web Service: AWS

Amazon Web Services (AWS) is the world's most comprehensive and broadly adopted cloud platform, offering over 175 fully featured services from data

centers globally. Millions of customers—including the fastest-growing startups, largest enterprises, and leading government agencies—are using AWS to lower costs, become more agile, and innovate faster.

Most functionality

AWS has significantly more services, and more features within those services, than any other cloud provider–from infrastructure technologies like compute, storage, and databases–to emerging technologies, such as machine learning and artificial intelligence, data lakes and analytic, and Internet of Things. This makes it faster, easier, and more cost-effective to move your existing applications to the cloud and build nearly anything you can imagine.

AWS also has the deepest functionality within those services. For example, AWS offers the widest variety of databases that are purpose-built for different types of applications so you can choose the right tool for the job to get the best cost and performance.

Most secure

AWS is architected to be the most flexible and secure cloud computing environment available today. Our core infrastructure is built to satisfy the security requirements for the military, global banks, and other high-sensitivity organizations. This is backed by a deep set of cloud security tools, with 230 security, compliance, and governance services and features. AWS supports 90 security standards and compliance certifications, and all 117 AWS services that store customer data offer the ability to encrypt that data.

Cloud computing leadership

Gartner Research positions AWS in the Leaders Quadrant of the new Magic Quadrant for Cloud Infrastructure as a Service (IaaS), Worldwide. Cloud IaaS, in the context of this Magic Quadrant, is defined as a standardized, highly automated offering, where compute resources, complemented by storage and networking capabilities, are owned by a service provider and offered to the customer on-demand."(Figure 2.3)

Figure 2.3: Gartner: Magic Quadrant for cloud infrastructure as a service, worldwide (2019)

2.4 Related Research

With the literature review of cybersecurity training system. Recent years have witnessed great development of virtualization technologies to cybersecurity training. Many tools and systems related to cybersecurity training have been proposed. Based on the structural model, they can be roughly divided into three categories: hosted hypervisor-based (e.g., [10] [13]), bare metal hypervisor-based (e.g., [5] [9] [14]), and cloud-based (e.g., [6] [7] [4]). [15]

Hosted hypervisor-based refers to a hypervisor program running on top of an existing OS, for example, Linux and Windows, or Virtual Box, VMware Workstation, and Parallels Desktop. Tools relying on this type of hypervisor such as SEED Labs [10]. The SEED environment consists of Minix, an instructional operating system, and Linux, a production OS; it takes advantage of the simplicity of Minix and the completeness of Linux, and provides a unified platform to support a rich set of laboratories for computer security education. SEED Labs are a collection of security labs designed primarily for security courses and individual practices. SEED presents a mature,

well-documented set of experiments, but the experiments are not always interactive or dynamic. Set up and run these experiments require a lot of work.

Bare metal hypervisor-based refers to a hypervisor that directly controls physical hardware, e.g., VMware ESXi, Xen and KVM. Tools relying on this type of hypervisor include CyRIS [5], Alfons [16], V-NetLab [9] and Platoon [14]. CyRIS has been introduced above. Alfons is a mimetic environment construction system for the dynamic analysis and cyber range. Alfons allows for the configuration of applications and has an installation mechanism for content to reduce these costs. V-NetLab is a virtual network lab platform designed for networking courses and network related labs. Platoon is a team-oriented cybersecurity exercise platform for both security labs and competitions.

Cloud-based cybersecurity training programs are the new trend, which cybersecurity labs can be deployed using public or private cloud resources. This kind of systems include EDURange [17], DETER Lab [6], V-Lab [7] and KYPO Cyber Range [4]. EDURange is a framework for creating exercises and games in a variety of environments including remotely hosted web services, i.e. cloud computing. The DETER Lab (Cyber- Defense Technology Experimental Research Laboratory) is a research testbed mainly for cyber defense research on large-scale network attacks such as DDoS and botnet. V-Lab is a cloud-based virtual lab education platform for hands-on course experiments. KYPO cyber range provides a research environment for simulation, detection, and mitigation of cyber threats against critical infrastructure.

Since the purpose of our research is to improve CyRIS to support cloud deployment, in the relevant literature collected, we focused on reviewing EDURange and KYPO Cyber Range. The two are based on public and private clouds respectively. The AWS cloud computing platform is used in the development and design of EDURange, while KYPO is currently running on the OpenNebula [18] cloud and is developing an adaptation to OpenStack [19]. Through EDURange, we can effectively understand how the AWS cloud platform plays a role in the design of cybersecurity education and training tool [20]. KYPO Cyber Range is designed by the KYPO team based on a private cloud, so physical servers, computer rooms, maintenance services, etc. will still be needed. However, its success is undeniable. Although we have no plans to apply private clouds in our research, KYPO still provides a very valuable reference. It summarizes the design requirements of some cloud computing-based cybersecurity education and training tools: 1) Flexibility, 2) Scalability, 3) Isolation vs. Interoperability, 4) Cost-Effectiveness, 5) Built-In Monitoring, 6) Easy Access, 7) Service-Based Access, 8) Open Source [21].

Chapter 3

Proposed Method

In this chapter, we will introduce our proposed method on how to use the cloud-based deployment of cybersecurity training, the implementation details and how to run the improved program.

3.1 Problem Formulation

In [5], the authors mentioned letting CyRIS support for cloud infrastructures, so that users can deploy cyber ranges on-demand in the cloud, thus eliminating the need to purchase and maintain deployment servers.

Firstly, the details of the task problem would be illustrated. The original CyRIS completely relies on the local entity server. Therefore, in order for CyRIS to support cloud infrastructure, the cloud interface needs to be added to the system.

According to the original deployment of guest virtual machines in CyRIS, support for cloud deployment can be defined that these guest virtual machines can be replaced with virtual servers on the cloud. So that user can access to the cyber range through the cloud but not the entity server. The task of this research is to merge the cloud infrastructures into CyRIS, keep the process of creating cyber range feasible. In addition, after the merge cloud infrastructure into CyRIS, the original process should be also compatible.

3.2 Approach

In this section, we introduce the proposed approach of merging the cloud into CyRIS system.

Whether through vertical or horizontal comparison, AWS is the first choice for cloud services because of its popularity and comprehensiveness. The following approach and experiments will be centered on AWS.

The overall system on which the original CyRIS is running has the architecture presented in Figure 3.1. The execution infrastructure is represented by a collection of hosts, each of them equipped with a virtualization platform (currently QEMU/KVM), which are connected to a LAN network.

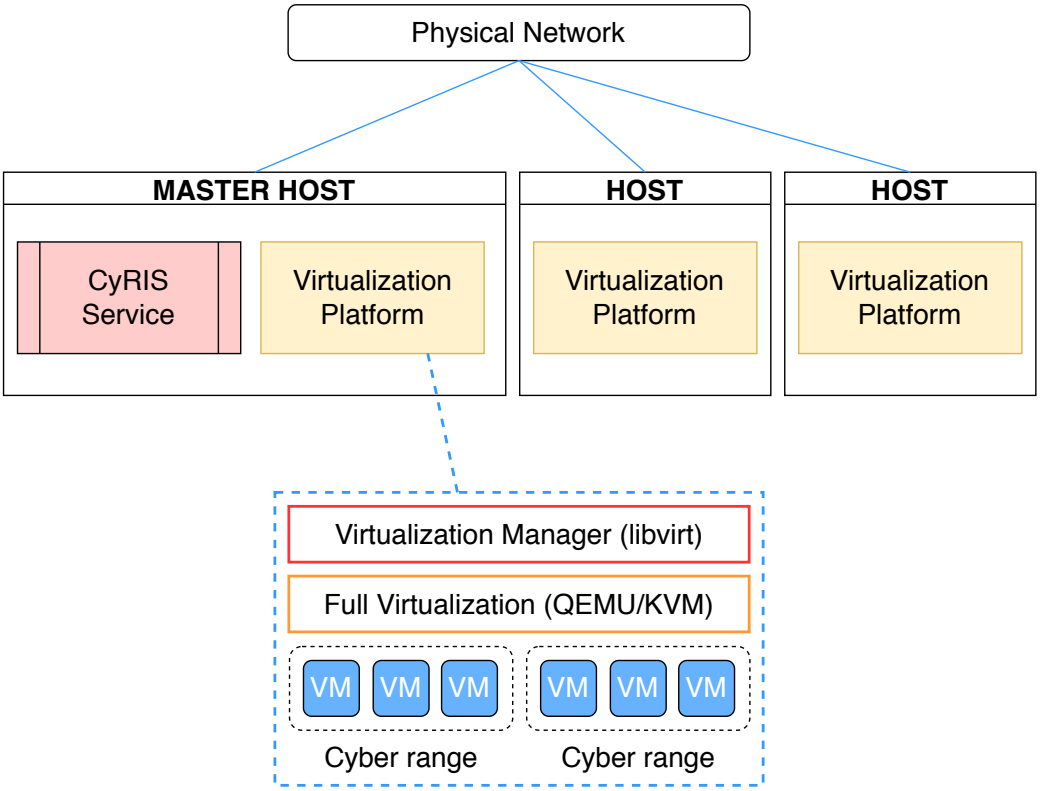

Figure 3.1: Overall architecture of the system on which original CyRIS is running.

If cloud servers support virtualization functions the same as physical servers, CyRIS may be deployed directly on the cloud, but the actual situation does not permit us to do so given most cloud servers do not support virtualization. So we consider replacing the original virtualization part with cloud services. The original process creates virtual machines on local server to generate cyber range. Now in our proposed method, the virtual machines will be created on the cloud platform to generate cyber range. The other parts will be kept as the original features consistent, such as description file check, content installation, etc.

As shown in chapter 2, in Figure 2.2, it can be found that the process of creating cyber range be divided into three parts, Base VM Preparation, Content Installation, and Guest VM Cloning. The first consideration is how to replace Base VM Preparation and Guest VM Cloning with AWS services. Therefore we proposed the approach as shown in Figure 3.2, rounded rectangles with orange filling are the functions running on the AWS, and the octagons are the original functions we will keep them workable on the improved system.

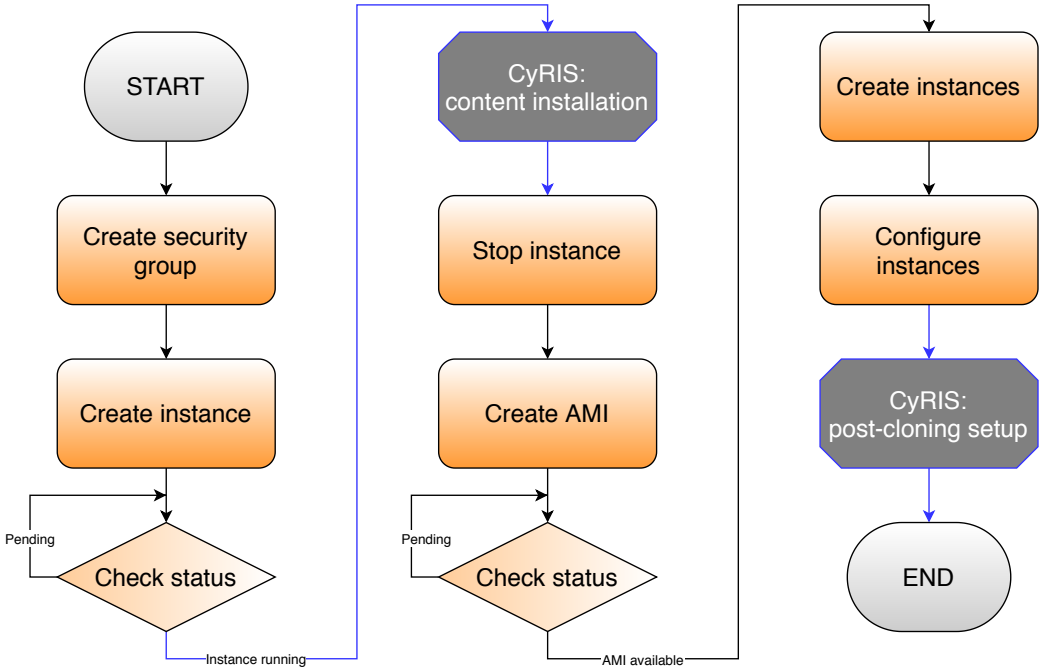

Figure 3.2: Framework of the proposed approach

About the framework of our proposed approach, it mainly consists of the following three parts next.

3.2.1 Create EC2 Instance

As in the original Base VM Preparation, the first step is to create a virtual machine. Here, we use AWS EC2 service instead.

Amazon EC2 provides such a feature, Virtual computing environments, known as instances. In order for these instances to connect to the network, we need to create security group for these instances. A security group acts as a virtual firewall for instance to control incoming and outgoing traffic.

Inbound rules control the incoming traffic to the instance, and outbound rules control the outgoing traffic from the instance.

3.2.2 Content Installation

Although the effect of this function what we need is the same as before, it does not mean that no adjustment is necessary.

In order to make the improved system compatible with the original CyRIS, we need to make a 2-layer branch. The first layer branch decides whether the original process based on KVM will be carried out, or the improved process we proposed, which is using AWS will be carried out. The second layer branch is about the AMI we chose when creating the EC2 instance. Because the newly created EC2 instance cannot be directly logged in as root, and the default user name will be different depending on the AMI. For example, the user name of Amazon Linux is 'ec2-user' and the user name of ubuntu is 'ubuntu', so some command-line statements in the content installation part have to be modified.

The Amazon Machine Image (AMI) is a template that contains a software configuration (e.g., operating system, application servers, and applications). From AMI, we can use it to launch an instance, which is a copy of the AMI running as a virtual machine in the cloud. Also, we can launch multiple instances with one AMI, as shown in the following Figure 3.3.

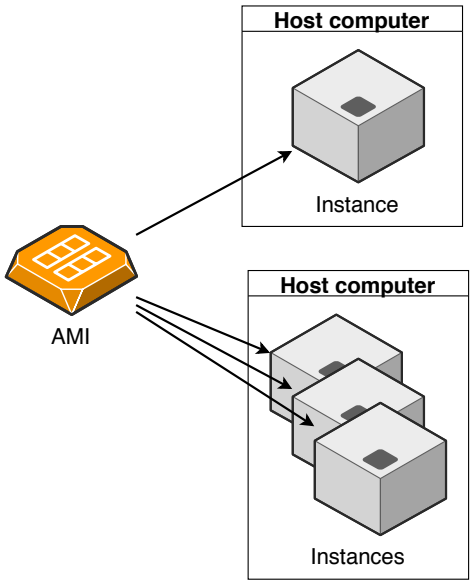

Figure 3.3: Launch instances of any type from AMI

Adding these two branches requires not only modification in the source program, but also adjustment of the original cyber range description file. We added a new option corresponding to 'basevm_type' in the cyber range description file, which is the cloud platform AWS. The original option for the parameter is only KVM. There another new parameter is the operating system corresponding to the AMI, which named 'basevm_ostype'.

3.2.3 Launch Instances from New AMI

To meet the original Guest VM Cloning part, what we will do is to package the EC2 instance that complete the content installation into a new AMI, and then launch multiple instances based on this AMI to achieve the effect of cloning, as shown in the following Figure 3.4.

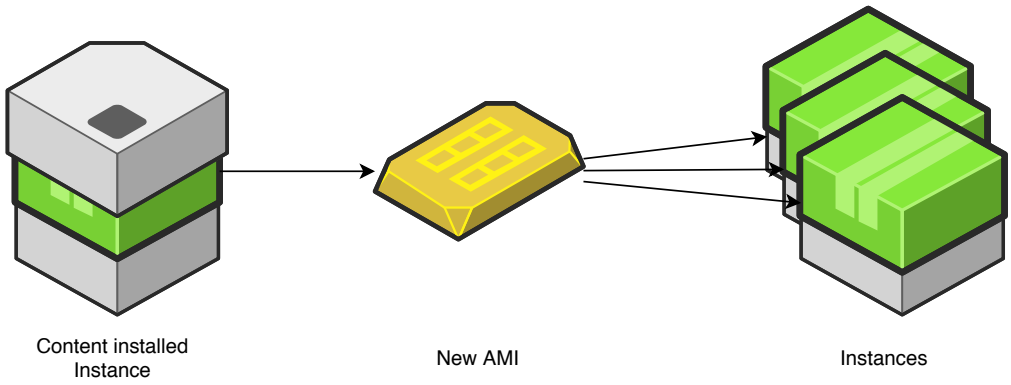

Figure 3.4: Cloning on AWS

3.3 Implementation Details

In this section, we will introduce the preparation of AWS development environment. And then introduce the details of each phase of the implementation. Taking Content Installation as the node, divide into two parts, before and after Content Installation, and introduce the adjustment of Content Installation.

3.3.1 AWS Development Environment Preparation

AWS allows users to access and manage AWS services with their preferred development language or platform. For Python, AWS provides the SDK

named Boto3. It enables Python developers to create, configure, and manage AWS services, such as EC2 and S3. Boto3 provides an easy to use, object-oriented API, as well as low-level access to AWS services. [22]

It's easy to start using Boto 3, it takes a few steps.

First, we should install the Boto3 into our host machine. We can install the latest Boto3 release via pip:

```
pip install boto3
```

After completing the installation of Boto3, there is one more step to be done before using it. Authentication credentials need to be set up. Credentials for AWS account can be found in the IAM Console. (AWS Identity and Access Management (IAM) is a web service for securely controlling access to AWS services. With IAM, you can centrally manage users, security credentials such as access keys, and permissions that control which AWS resources users and applications can access.)

However, there is a problem here. Because the AWS Educate account is used in this experiment, IAM is restricted, and we cannot find the authentication credentials under the IAM in the console. For the AWS Educate account, we can find authentication credentials through 'account details' in the interface before logging into AWS console. As shown in Figure 3.5, it is recorded under AWS CLI. (The AWS Command Line Interface (AWS CLI) is a unified tool that provides a consistent interface for interacting with all parts of AWS.)

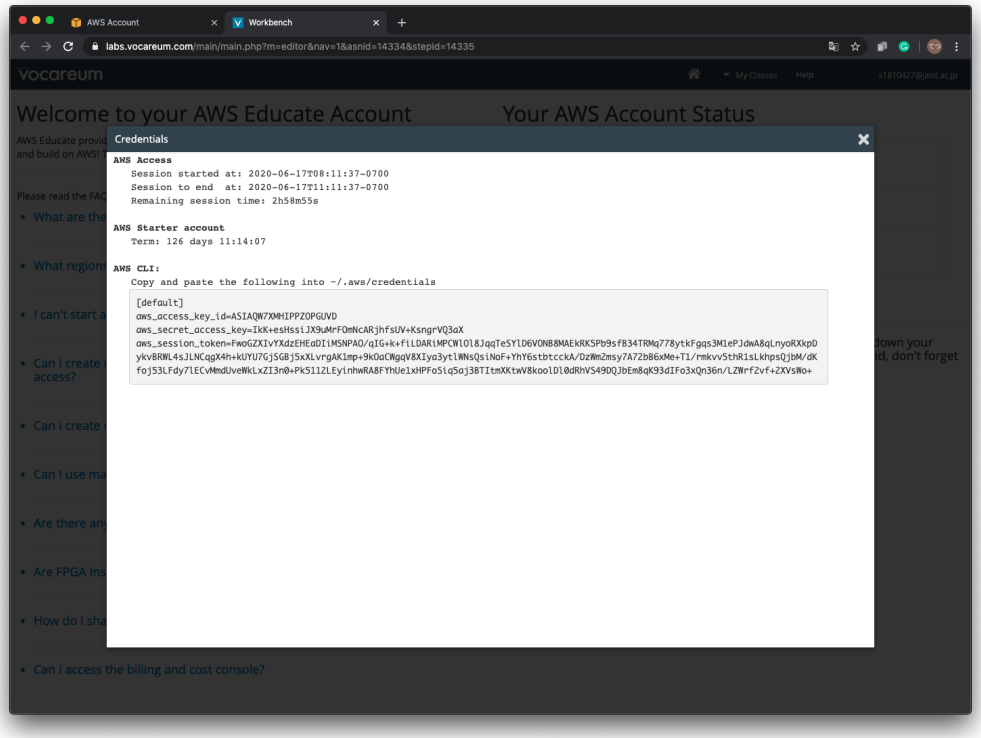

Figure 3.5: The authentication credentials in AWS educate account

So in our implementation, we create the credential file by self. The location is at ∼/.aws/credentials by default. The content of credentials is as follow

```
[default]
aws_access_key_id = YOUR_ACCESS_KEY
aws_secret_access_key = YOUR_SECRET_KEY
```

3.3.2 Base VM Preparation on AWS

The first is about the phase before content installation.

In the original system, this part is called Base VM Preparation. In our experiment, we can still describe it like this, but now we use the AWS EC2 instance on the cloud instead of the virtual machine in the previous physical server. As shown in Figure 3.6, the phase before content installation divided into 3 steps.

Figure 3.6: Workflow before content installation

In Figure 3.7, we list the used methods in every steps.

Figure 3.7: Boto3 methods of the part before content installation

Step 1: Create Security Group

create_security_group(): This method is used to create a security group.

As mentioned above, a security group acts as a virtual firewall for the instance to control inbound and outbound traffic. When we launch instances, we must assign security group to them. So we need to create a security group first. In this method, we should define the name of the security group.

authorize_security_group_egress(): Adds the specified egress rules to a security group.

In this method, we should specify a protocol for each rule, for example, TCP, ICMP, etc. In our experiment, we use the TCP protocol, we must also specify the destination port.

describe_security_groups(): Describes the specified security groups or all of the security groups.

We use this method to check the status of the previously created security group. If the security group is not created successfully, we cannot call it when we create an instance.

Step2: Create Instance

run_instances(): This method is used to launch the specified number of instances with the AMI for which we have permissions.

This phase is equivalent to the 'start base vms' phase of the original CyRIS. In this part, we don't need to perform 'copy base vms' as in the

original CyRIS, we only need to specify an AMI Id to create an instance of the desired operating system.

Therefore, we associate this part with the cyber range description file, configure the AMI IDs of some free operating systems in this method, and create a new parameter named 'base_vm_ostype' in the cyber range description file. In this way, we can launch the corresponding instances with the specified operating system in the cyber range description file.

describe_instance_status(): Describes the status of the specified instances or all of the instances.

In this method, we check the status by specifying the instance ID generated by the instance created before to ensure that the instance is in a running state because subsequent connection checks and installation content require the instance to be in the running state to proceed normally.

3.3.3 Adjustment for Content Installation

Then is about the content installation. Here we need to make the installed content compatible with the AWS EC2 instance

In the phase of content installation, the original program logs in as root user directly every time through SSH command and performs relevant operations. However, for security reasons, the EC2 instance of AWS does not allow SSH login as root by default, so we need to modify the login part of the program, and some operations after login should also consider the current non-root identity.

Generally, the default user name of EC2 instance for SSH login will be different according to the AMI selected at the time of creation. In the optional AMI we added into cyber range description file, Amazon Linux 1, Amazon Linux2, and Red Hat, the default user names of these AMI are ec2-user. The user name of the Ubuntu system is ubuntu.

By default, AWS does not allow users to access instances with a password, users need to use key pair for SSH login. Therefore, for example, when we modify such an original command:

```
ssh -o UserKnownHostsFile=/dev/null -o
StrictHostKeyChecking=no root@${image_addr}
```

We will revise it as follows:

```
ssh -i my-key-pair.pem -o StrictHostKeyChecking=no
ec2-user@${image_addr}
```

3.3.4 Guest VM Cloning on AWS

Next is the phase after content installation. As the meaning of cloning, at this phase, we use the created image to generate a specified number of EC2 instances. As shown in Figure 3.8, the phase is divided into 4 steps.

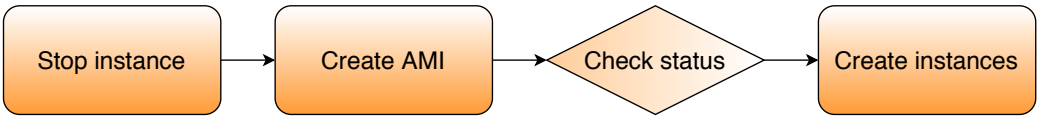

Figure 3.8: Workflow after content installation

And the used methods in every step have been listed, as shown in Figure 3.9.

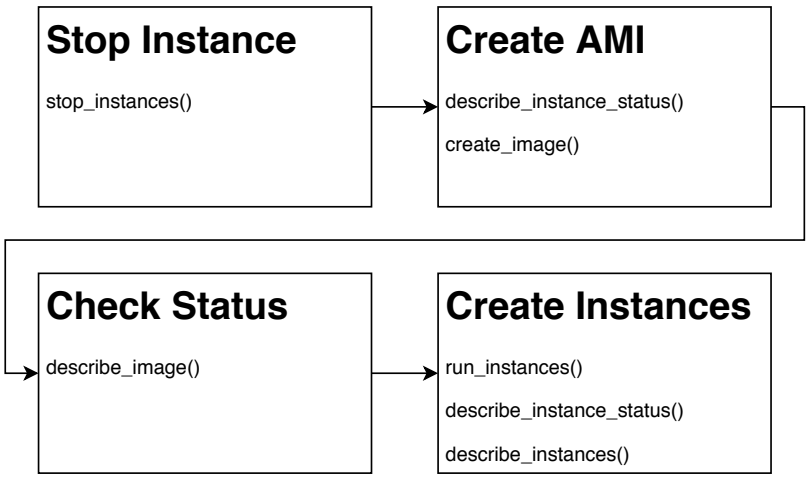

Figure 3.9: Boto3 methods of the part after content installation

Step1: Stop Instance stop_instances(): Stop the specified instances.

The stop step here is similar to the shutdown of an ordinary computer, which can be restarted at any time. In order to ensure the completeness of the image generated later without problems, we use this method to stop the instance.

Step2: Create AMI describe_instance_status(): Describes the status of the specified instances or all of the instances.

While the instance is in a running state, the AMI can also be created, so here we use this method again to check the status of the instances, but this time we use it to confirm whether the instances are completely stopped.

create_image(): Creates an Amazon AMI from an instance that is either running or stopped.

In this method, we specify the instance IDs that have completed the content installation and have been stopped, create new AMIs based on these instances. Then we can get the ImageIds, which can be used to launch new instances. The content of the newly created instance is exactly the same as the previously stopped instance.

Step3: Check Status describe_image(): Describes the specified images which are available.

Here we use the previously obtained ImageIds to check the status of the created AMIs through this method. When the status becoming 'available', we can launch instances with the AMIs.

Step4: Create Instances run_instances(): Launch the specified number of instances with the AMI for which we have permissions.

Here we use this method again to perform the step of cloning in the original CyRIS. We specify the number of how many EC2 instances will be launched through the 'clone_settings' in the YAML format range description file.

describe_instance_status(): Describes the status of the specified instances.

The method used here is to check whether the cloned instances are in running status. In order to make sure that we can accurately obtain the information of the instances and conduct connectivity checking.

describe_instances(): Describes the specified instances.

This method can get almost all the information about the created instances. Here we mainly use this method to get the IP addresses of the cloned instances and give them to the source program's 'check_connectivity_to_cr' function for connectivity checking.

3.4 Running AWS CyRIS

In this section, we will introduce how to use our improved CyRIS.

The following three conditions are required to run AWS CyRIS:

1) The environment where KVM CyRIS can operate normally.

2) AWS development environment, as we introduced in 3.3.2.

3) Cyber range description file corresponding to AWS CyRIS.

Based on the premise that the environment is prepared already, we divide the process into 3 steps, prepare the description file, run AWS CyRIS, and clear the cyber range. The following is an example of running AWS CyRIS:

Step 1: Cyber range description file preparation

As shown in Figure 3.10, here we should pay attention to the parameter settings of basevm_type and basevm_os_type.

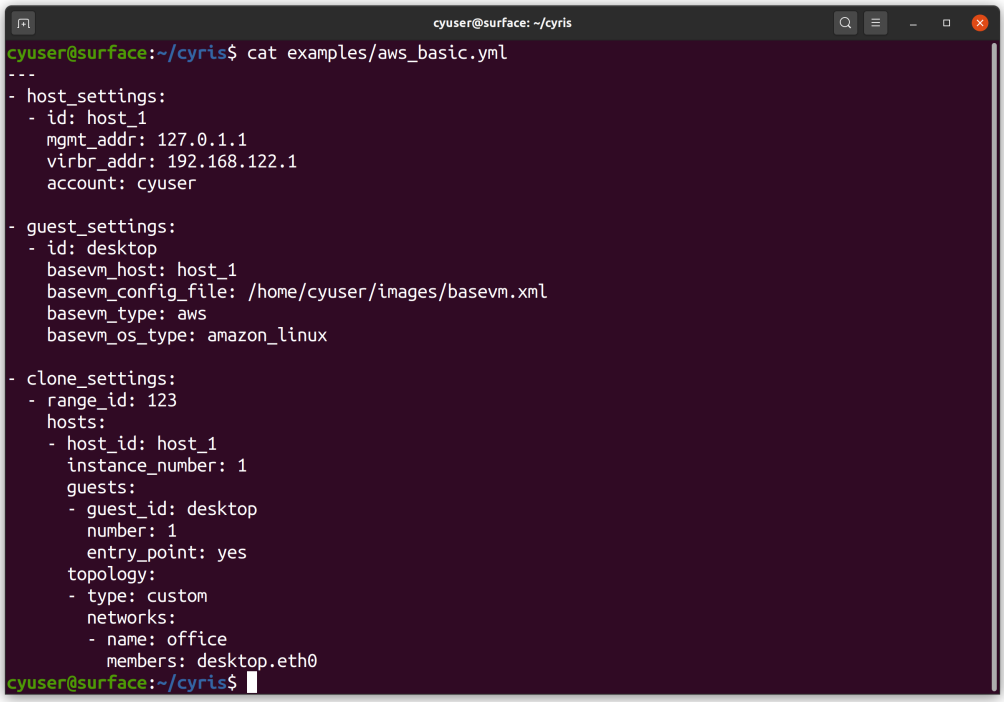

Figure 3.10: An example of YAML cyber range description file

Step 2: Running AWS CyRIS

This step is almost the same as KVM CyRIS. We enter the sh command in the terminal and specify the prepared YAML description file, as bellow:

```
python main/cyris.py -v examples/aws_basic.yml CONFIG
```

When the program ends normally, we will receive a message indicating that cyber range was successfully created, as shown in the Figure 3.11 below.

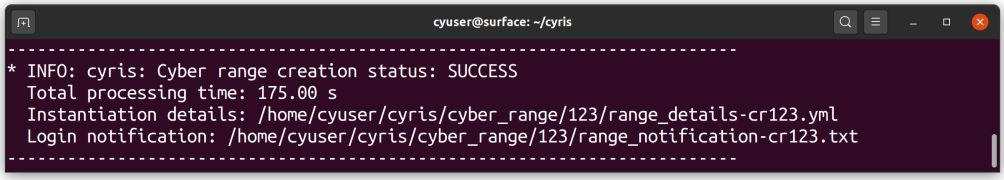

Figure 3.11: Feedback of running CyRIS program

Step 3: Cyber range cleanup

In addition to running the range_cleanup script of the original resource, we also added a new script about AWS_cleanup, through which we can delete the AWS services generated by running AWS CyRIS. As shown in Figure 3.12.

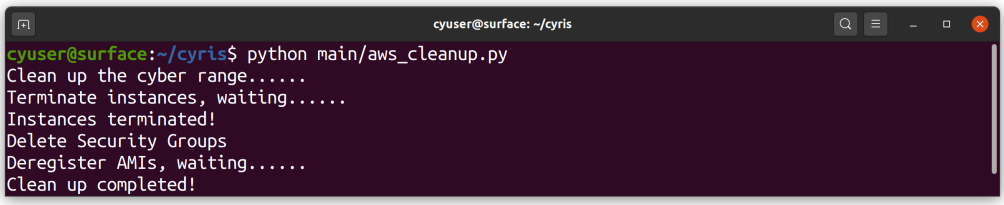

Figure 3.12: Feedback of AWS_cleanup script

Chapter 4

Experimental Evaluation

In this chapter, we first introduce our environmental requirements and preparations in the experiment. Then elaborate on the details of the experiment. Finally, the results of the experiment will be discussed in detail.

4.1 Preparation of Experiment

In this section, we will introduce the preparations in detail.

4.1.1 Original CyRIS Preparation

First of all, we have to make sure that the original CyRIS can run normally on the host. For this part, please refer to the CyRIS user manual (`https://github.com/crond-jaist/cyris/releases`). In the manual, two points are mentioned about the hardware requirements. One is the operating system ubuntu 14.04 LTS or 16.04 LTS required by the physical machine, and the other is whether the CPU supports virtualization. Regarding the first point, we have installed the ubuntu 20.04 LTS operating system as a basic development facility on an ordinary computer. In the cloud-based case, almost all operations for creating and cloning virtual machines are performed on the cloud, so there is no high requirement for the performance of the local machine, therefore physical server is not required. For the second point, because we are mainly developing the cloud part, the requirement of whether the local machine supports virtualization can be temporarily ignored. We only need the system to run successfully to run all the steps before creating the virtual machine.

The operating system ubuntu 20.04 LTS is not directly compatible with CyRIS like the old versions of 16.04 LTS and 18.04 LTS, so we need to

make some adjustments to the local system manually. First, the default version of Python after ubuntu 19.04 LTS is 3, but the locale corresponding to CyRIS is Python2, so we need to install Python2 and modify the default Python version of the system to Python2. In addition, due to the change of the operating system, some installation packages in the script 'HOST PREPARE.sh' will be failed to install, and these issues are necessarily fixed manually. (Several external tools need to be installed on the hosts. For automating this task, a bash script called HOST PREPARE.sh is given)

4.1.2 AWS Account Sign Up

When registering an Amazon web service account, the personal credit card information is required to log in. Although the successfully registered account will have a 12-month free trial, these trials do not contain all the content, so if the customer has enabled some non-free trial services or functions due to misuse, unnecessary expenses will be incurred. In order to avoid such a situation, we chose to use AWS Educate account in the experiment. Signing up for AWS Educate account does not require a credit card, so there is no need to worry about unexpected expenses. In addition, AWS Educate account also has a free trial and will provide a certain amount of USD in the account, so that users do not have to worry about using some charged items.

Although AWS Educate account can use most of the services and functions of AWS general account, there are still some limitations, which will be further discussed in the subsequent part.

4.1.3 Others

Create key pair

AWS EC2 instance uses public key cryptography to encrypt and decrypt login information.

When we need to log in to an EC2 instance, we must create a key pair, specify the name of the key pair while starting the instance, and then use the private key to connect the instance.

There are many ways to create key pairs in the user manual [23]. We can choose the most intuitive way to create key pairs in the EC2 console:

1. Log in to EC2 console interface
2. Find the KEY PAIR tag and create the key pair
3. Set the key pair's name when creating
4. At the same time of creation, the key pair will be downloaded to the local automatically in the format of .pem

5. Set permissions for the key, as following command:

```
chmod 400 my-key-pair.pem
```

Then we can log in to the EC2 instance by specifying the key pair through SSH.

How to use AWS SDK Boto3

To use Boto3, we must first import it and tell it what service we are going to use. In our experiment, we use the service EC2, so we should add some code like bellow into the main function.

```
import boto3

client = boto3.client('ec2', region_name='us-east-1')
```

This is a low-level client representing Amazon Elastic Compute Cloud (EC2)

Here should be noted that the argument region_name doesn't appear in the user manual [22], because we use the AWS Educate Account in this experiment, and the AWS Educate Account only supports the region us-east-1, so we need to provide the argument.

4.2 Test of AWS Platform

When developing and testing AWS CyRIS, we found that there is a certain deviation in the total time of cyber range creation even with the same specification. The creation time is not presented in a stable state. Therefore, it is not suitable for evaluation to comparing with KVM CyRIS. When using AWS cloud service, the running speed is affected by many aspects, such as network speed, background load, etc.

In order to do better evaluation and subsequent experiments, we decided to conduct a detailed measurement and evaluation of the cyber range creation time of AWS CyRIS. To make the measurement progress efficiently, we configured the cyber range description file with one base VM and one cloned VM, and without any tasks. We write a simple script to automatically run AWS CyRIS ten times every hour of the day. The original CyRIS contains a functional module to record the creation time, so we can directly obtain relevant data from the log file. Here we use the Python tool 'pandas' to create a table base on the data of the creation time, and then calculate the average and standard deviation of the cyber range creation time for each hour. Next, we make a bar plot with error bars, as shown in Figure 4.1. The dotted line

in the figure represents the average of the creation time for each hour of the day. (When collecting data in this experiment, the reference time is Tokyo time.)

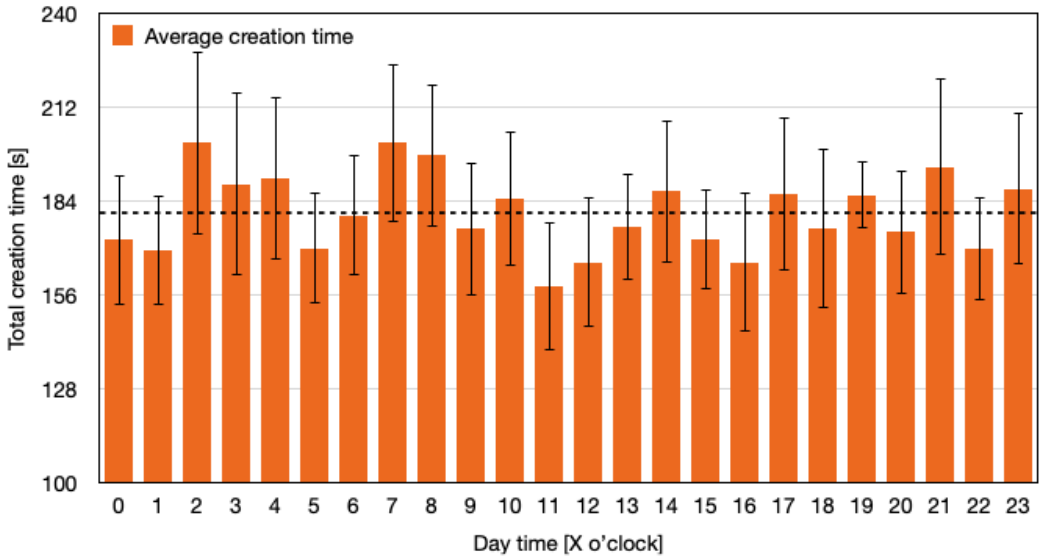

Figure 4.1: Cyber range total creation time every hour of the day

As can be seen from Figure 4.1, the cyber range creation time of each hour is different, and the error bar of each column is also different. (The error bar here is added according to the standard deviation.) The whole is not a stable state. The standard deviation here represents the dispersion degree of the total cyber range creation time in the corresponding hours. To observe the standard deviation more clearly, we draw a bar plot of the standard deviation separately as shown in Figure 4.2. The dotted line in the figure represents the average of the creation time standard deviation for each hour of the day. The larger the standard deviation, the greater the dispersion of the time when cyber range is created in the hour.

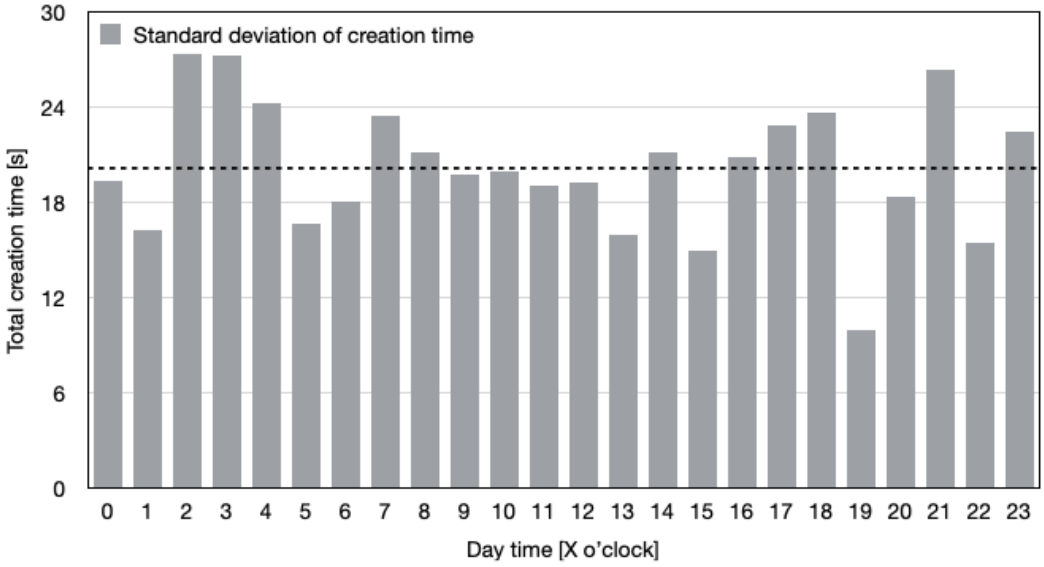

Figure 4.2: Standard deviation of cyber range total creation time every hour of the day

By observing the above two figures, it can be found that the hour for the creation time below the average and the standard deviation below the average are mostly the same. Therefore, we can analyze and conclude that the background of the AWS cloud is operating well during 0, 1, 2, 4, 9, 11, 13, 14, 18, 19, 22 o'clock of Tokyo.JP time. AWS CyRIS should be run to test or make experiments in these times as much as possible.

4.3 Overall Evaluation

In our experiments, the results are positive. In the end, we can clone the specified number of instances on the AWS cloud platform to provide users with login to use, and meet the content installation and configuration of the cyber range description YAML file.

Figure 4.3 below shows the cyber range we created under the AWS console. The running state instances are the cloned instances belong to the cyber range, and the stopped state instances are the original base instances.

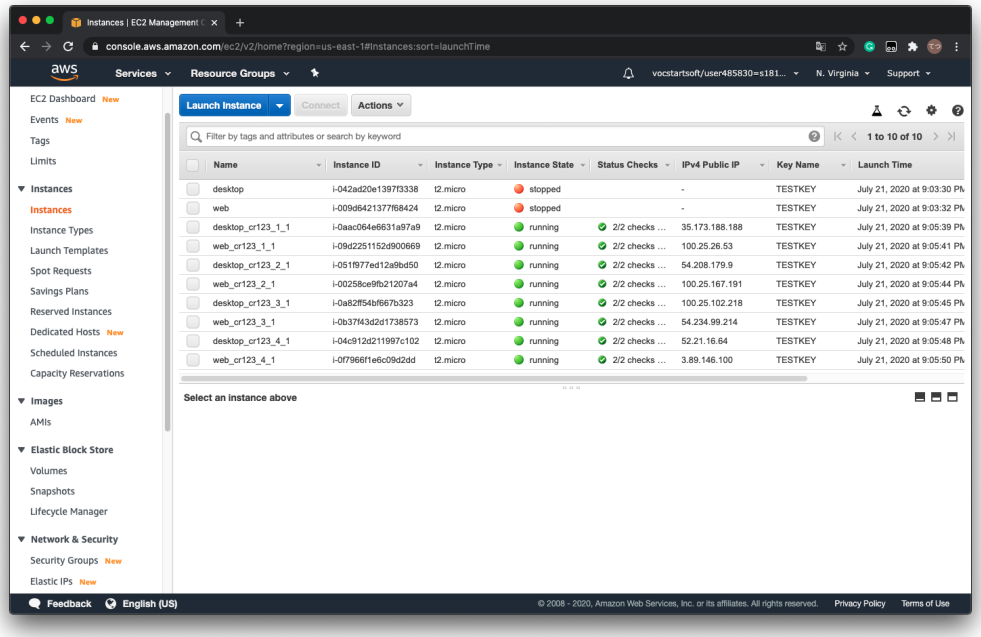

Figure 4.3: Cyber range in AWS management console

In our improved AWS CyRIS, we adjusted the content installation part of the source program to make it compatible with more operating systems on the AWS platform. And almost all original KVM CyRIS content can run well on AWS CyRIS.

In addition, after the AWS CyRIS create the cyber range, it can also generate some logs and files like the original KVM CyRIS, such as cr_creation_time, range_details, and range_notification. It should be emphasized that this experiment has made some adjustments in range_notification, as shown in the Figure 4.4, the description of the login information is not to log in to cyber range through the host machine, but to directly access the machines in cyber range, and the login method is also changed from password to key pair login.

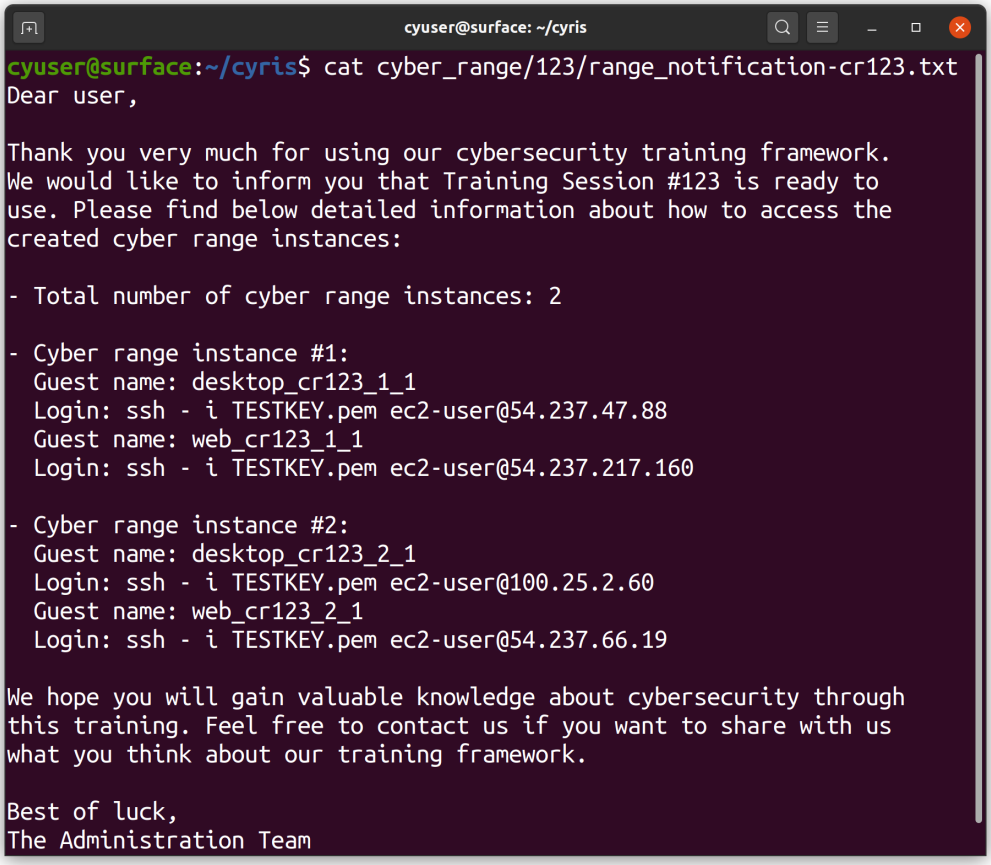

Figure 4.4: The range notification of cyber range

There are also some shortcomings in our experiment. These shortcomings are mainly due to the limitation of using AWS educate account. With the conservative attitude, we didn't register AWS general account for login credit card information. However, these shortcomings did not fave such a very serious impact, which can be summarized as follows.

Dynamic credentials: The credentials information contains aws access key id, aws secret access key and aws session token, when we use the boto3 sdk, these information need to be configured on the host.

Limitation for running instances: According to the official data [24], different types of instances will be limited in different numbers. In theory, the t2.micro instance we used in our experiment can run 256 at the same time, but the actual experimental result is only 9. Although there is a service for requesting to increase the limit, it cannot be done due to the permission of the AWS educate account.

AMI usage restrictions: The official AMI of some operating systems in AWS marketplace is prohibited from being used in educational accounts, such as CentOS 7.

Billing and cost management limitation: In AWS educate account, we don't have permission to access billing information. Therefore, we cannot make a complete cost estimate for creating a cyber range directly.

In general, AWS educate account is worthy of promotion. The AWS services are almost free, the official will also provide a certain amount of US dollars in the account, if complete some simple training, more US dollars can be gained in the account. While improving self-skills, ensure that the AWS cloud services can be used continuously. The function of AWS education account is almost the same as that of general account. Except for some restrictions mentioned above (for this study only), it can be said that in some tests before developing projects that require cloud services, using AWS educate account is conservative feasible.

4.4 Performance Evaluation

AWS EC2 service can run instances based on 256 CPUs by default, which means that we can create cyber range with a scale of 256 cloned VMs with one account. Unfortunately, we cannot implement cyber range creation of such a big scale. Because we are using an AWS educate account, the account is subject to some restrictions from the AWS educate team in order to prevent excessive spending and fraud. The number of concurrent running instances allowed in an account is one such restriction.

There are only 9 instances that can be running at the same time, so we create small-scale cyber range for performance evaluation. We define two scenarios based on single and double Guest VMs to create cyber ranges with a different number of clone machines.

Scenario 1: One guest VM, playing the role of a desktop PC, is needed for each trainee;

Scenario 2: Two guest VMs, playing the roles of a desktop and a web server, are required for this scenario.

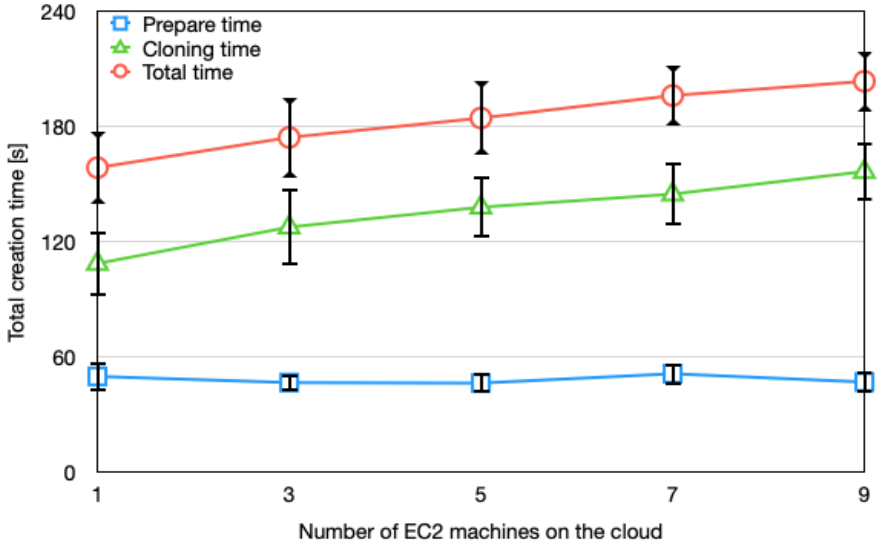

Figure 4.5: Cyber range creation time versus the total number of EC2 machines for training Scenario 1

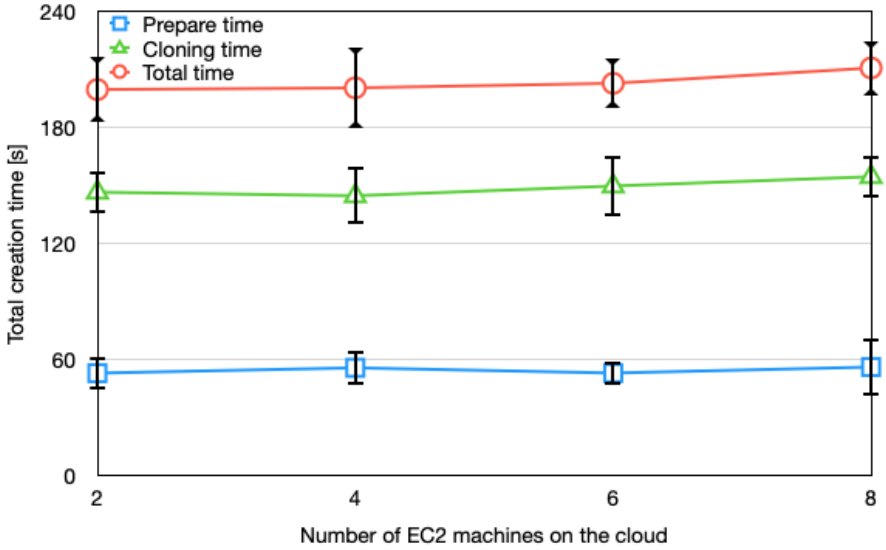

Figure 4.6: Cyber range creation time versus the total number of EC2 machines for training Scenario 2

We create each scale cyber range ten times based on Scenario 1 and Scenario 2, calculate the average of the creation time and draw line charts with error bars (The error bar here is added according to the standard

deviation.). Similar to testing AWS, for efficiency, we also did not configure any tasks for cyber range in this data collection, and only implemented the base VM preparation and cloning phase.

The results of Scenario 1 (Figure 4.5) show that the prepare time is relatively constant, less than one minute. The increase in total time is similar to the clone time. The clone time tends to increase according to the number of cloned machines. During the cloning time, AWS CyRIS mainly conducts the creation of AMI and the operation of starting cloned machines. Since launching the EC2 instances process is parallel, the time difference is very small. The total time tends to be a smooth growth.

The results of Scenario 2 (Figure 4.6) show that the preparation time is also relatively constant, also less than one minute. The increase in total time is similar to the clone time. The total time tends to be a smooth growth.

Most of the prepare time comes from base VM preparation, from the above two figures, we can found that the standard deviation of prepare time is relatively smaller. Base VM preparation and cloning phase are all based on the operation of launch EC2 instances, so the standard deviation of cloning time is mainly from creating AMIs.

From the results, we can judge that the main time increase comes from cloning because the other times are relatively constant. Since almost operations on the cloud are parallel, these increased time are mainly due to the running status check and SSH connection check of each cloned machine.

4.5 Performance Comparison

4.5.1 Single Cyber Range Creation

In order to better compare the difference between AWS CyRIS and KVM CyRIS, we extracted detailed data on the creation time of single cloned cyber range based on the above Scenario 1 and Scenario 2.

The detailed information on cyber range creation time includes prepare time, install time, cloning time, parallel AMI/scp time, parallel cloning time, and total time. The cloning time is composed of parallel AMI/scp and parallel cloning. Because there is no configuration task, the install time is zero. So we use prepare time, parallel AMI/scp time, parallel cloning time and total time, calculate their average and standard deviation to draw bar plots with error bars to compare AWS CyRIS and KVM CyRIS in detail.

The result is shown in Figure 4.7 and Figure 4.8, we can see that prepare time of AWS CyRIS is quicker. This is because it is not necessary to start a base instance with a copyed image when the whole process is on the cloud.

In the meantime, the 'Parallel AMI' of AWS CyRIS is much slower, because the creation of AMI takes some time. In addition, from the error bars, we can find that the standard deviation of 'Parallel AMI' is relatively bigger, so it can be judged that the instability of cyber range creation time mainly comes from the function of creating AMI in AWS cloud.

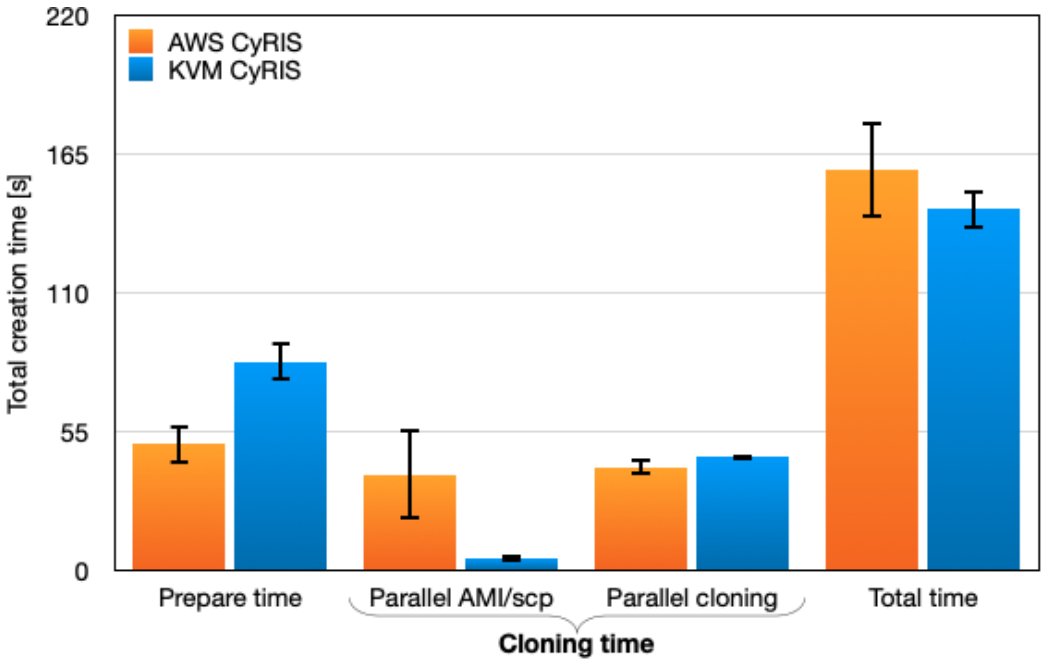

Figure 4.7: Comparison of creation time details for a single cyber range when using Scenario 1

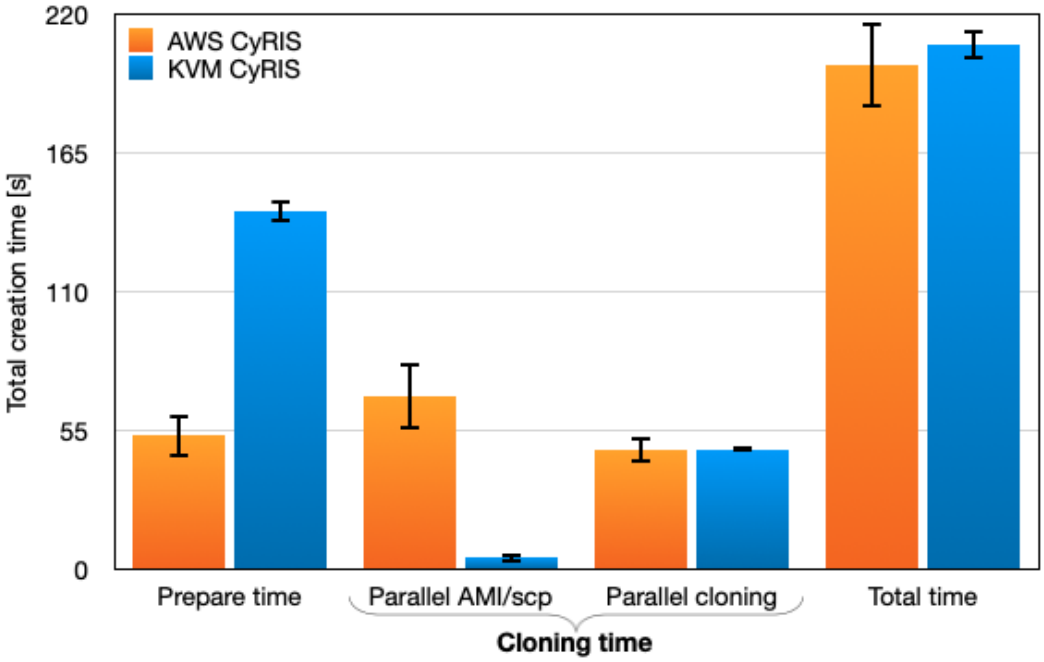

Figure 4.8: Comparison of creation time details for a single cyber range when using Scenario 2

4.5.2 Multiple Cyber Range Creation

To compare the performance of AWS CyRIS and the original CyRIS, we also used KVM CyRIS to create cyber ranges of different scale ten times each based on Scenario 1 and Scenario 2. We select the total creation time of both CyRIS, calculate the average and standard deviation, draw bar plots with error bars.

In the case of Scenario 1, as shown in Figure 4.9, the total creation time of AWS CyRIS is slower than KVM CyRIS. And the error bar on the AWS column is larger, so based on a single base VM cyber range, AWS CyRIS has no advantage in performance. However, as the scale increases, the overall gap is up to about 30 seconds. We believe it can be further improved in the subsequent optimization.

In the case of Scenario 2, as shown in Figure 4.10, the time of AWS CyRIS is faster. Because in KVM, 2 files need to be copied which increasing the network congestion. Although the stability of creation time is still not as good as KVM CyRIS, with the number of base VMs increases, the overall cyber range creation time of AWS CyRIS will be faster.

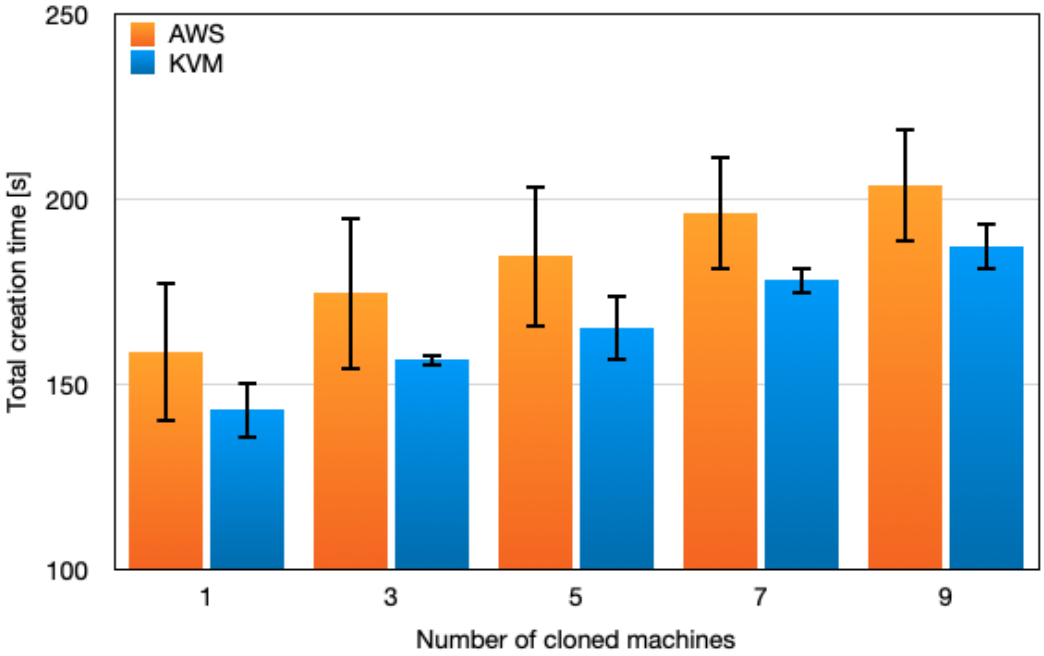

Figure 4.9: Comparison of cyber range creation times versus the total number of cloned machines for Scenario 1

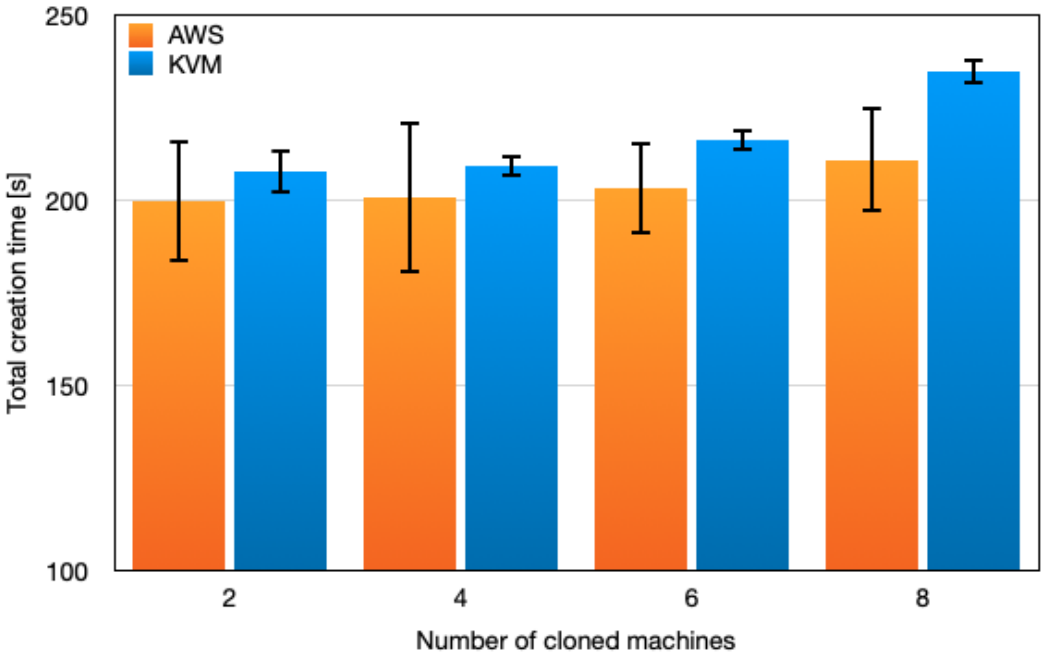

Figure 4.10: Comparison of cyber range creation times versus the total number of cloned machines for Scenario 2

4.6 Compatibility Comparison

There are more choices for the base image in AWS CyRIS.

Since we used the AWS educate account in this study, we only introduced some AMIs for free trial. If we use the AWS general account, we can get more free AMIs.

The original CyRIS only supports four base images: CentOS 7, Ubuntu 16.04, Ubuntu18.04 and Windows7. Now the number of tested base images is six, there are Amazon Linux1, Amazon Linux2, red hat 8, Ubuntu 16.04, Ubuntu 18.4, and Ubuntu 20.04. And compared with the original need to prepare the base image, only the AMI ID of the corresponding operating system needs to be specified for AWS CyRIS to launch instances, which effectively saves the cost of testing and improves the efficiency while developing.

For the content installation part, AWS CyRIS basically inherits all the original content, as shown in Table 4.1. In AWS CyRIS, tasks Emulate attacks, Emulate malware and Modify firewall rules are supported on all AMIs used, unlike the original CyRIS, which only supports the three tasks with CentOS 7.

But the downside is that task Emulate traffic capture files cannot be supported on AWS CyRIS. This function is for emulating network sniffing activities, and capturing traffic in PCAP format, e.g., to include attack traces. In the current stage, CyRIS supports generating traffic traces for SSH dictionary, DoS, and DDoS attacks. The task causes real malicious attacks between the host and guest VMs. For EC2 instances on AWS, there are some security constraints. Therefore, malicious operations are forbidden, whether the outside goes to the cloud, or the cloud goes out, or between cloud servers. If some malicious attacks are carried out, the official will give a warning or even directly disable the account.

Table 4.1: Improved CyRIS support depending on guest OS type

Content Installation	KVM CyRIS			AWS CyRIS		
	CentOS 7	Ubuntu 16.04 Ubuntu 18.04	Windows 7	AmazonLinux 1 AmazonLinux 2	Red Hat 8	Ubuntu 16.04 Ubuntu 18.04 Ubuntu 20.04
Add accounts	✓	✓	✓	✓	✓	✓
Modify accounts	✓	✓	✓	✓	✓	✓
Install packages	✓	✓	✓	✓	✓	✓
Emulate attacks	✓			✓	✓	✓
Emulate traffic capture files	✓					
Emulate malware	✓			✓	✓	✓
Copy content	✓	✓	✓	✓	✓	✓
Execute programs	✓	✓	✓	✓	✓	✓
Modify firewall rules	✓			✓	✓	✓

Chapter 5

Conclusion

5.1 Summary

In this thesis, we introduce the improved CyRIS based on AWS cloud platform,, which is an improved solution proposed to support users to deploy cyber range in the cloud on-demand. Consistent with the original CyRIS, the new CyRIS can also use the text-based cyber range description file to automatically configure the environment related to cybersecurity education and training.

In the preparation process, we registered an AWS educate account and obtained some credits for using various services on the AWS cloud. Through finishing some cloud computing courses provided in the education account, AWS provided an additional 100 US dollars in credits. In addition, we completed the review of CyRIS and made it run successfully for subsequent improvements.

In the process of improving the system, the proposed approach is mainly to replace the original KVM part with AWS. Change the cyber range deployed on the local server to the cyber range on the cloud. In this research, using Boto3 provided by AWS, base VMs preparation, and clone VMs are converted to cloud solutions. Modify the programs of installation content to support machines on the cloud.

In the evaluation process, we first measure the cyber range creation time at each hour of the day to find a suitable time for further testing. And then we measure the creation time of different size cyber ranges in these times and compare the date with the original CyRIS. At last, we discussed the functionality support of AWS CyRIS compares to the KVM CyRIS.

5.2 Contributions

In this experiment and study, we regard the following aspects as the main contributions.

First, in this research, the improvement of cyber range deployment based on cloud computing platform for CyRIS is successful. Trainees can easily access cyber range on the cloud. For training managers, creating cyber range on the cloud only needs to prepare the AWS account and cyber range description file, and then the cyber range can be automatically created on the cloud. In terms of performance, AWS CyRIS has the advantage of being faster when creating multiple base VMs, and more compatible operating systems.

Secondly, the AWS educate account used in our experiment has been proved to be quiet effective. AWS educate is completely free to use, and almost all AWS services can be used. Although there are some limitations, as we have summarized above, these limitations are only needed in later test improvements. For the part of simply deploying cyber range in the cloud, the AWS educate account is completely sufficient.

In addition, the script we made can automatically generate and destroy cyber range for a specified number of times, and record the cyber range creation. This method can effectively evaluate what time of day is suitable for cyber range creation to improve the overall efficiency.

5.3 Future Work

Create key pair for each EC2 instance separately

Because of the limitation of the cloud server, we cannot directly log in to the instance using the username and password. In this experiment, we used a public key as the only license to log in to all the instances. Obviously, it is unreliable to access all instances with only one key pair, even though users do not know each other's IP addresses. For future improvements, we can use the 'create_key_pair' in boto3 to add a unique key to each cloned instance, so that each instance in the cyber range can be independent of each other.

Clone phase without creating AMI

During testing, we found that the creation time of the cyber range is similar for AWS and KVM.

To further improve the runtime, we need to adjust the process of AWS CyRIS.

Since AWS CyRIS creates AMIs before cloning, which takes some time, if it is possible to start new instances directly based on the configured current instances, but not through creating AMIs, the entire cloning time can be significantly reduced. Among the currently available operations on AWS, the most similar one is to do 'Launch More Like This' on some existing instances. After testing, we found that although we can inherit and start instances of the same specification, the installed package could not be inherited. This function cannot replace the current cloning method when the cyber range includes some installation tools. However, we envision that this functionality will be available soon with the rapid growth of the cloud computing domain.

Using a normal AWS account

As summarized above, there are many restrictions on an AWS educate account.

We can potentially improve the integrity of the entire research by using the AWS general account through research funding or the opportunity of an official collaboration with AWS. First, we don't need to use dynamic credentials, which can make it more convenient for us to continue using the SDK, especially when modifying the program for testing. Otherwise, we need to reconfigure the credentials every two hours. Secondly, we will not meet the current limit of only 9 running instances by using general accounts, and if necessary, we can also propose to increase the upper limit, so that we can simulate the creation of a larger cyber range and conduct various kinds of tests. The last is billing management. After all, the use of cloud computing is charged on-demand, we need to evaluate how much the use of each specification will cost. Under general accounts, we can confirm the details on what services we use and how much consumption in advance.

Bibliography

[1] Randal T Abler, Didier Contis, Julian B Grizzard, and Henry L Owen. Georgia tech information security center hands-on network security laboratory. *IEEE Transactions on Education*, 49(1):82–87, 2006.

[2] Khaled Salah, Mohammad Hammoud, and Sherali Zeadally. Teaching cybersecurity using the cloud. *IEEE Transactions on Learning Technologies*, 8(4):383–392, 2015.

[3] The National Institute of Standards and Technology. Cyber ranges. `https://www.nist.gov/system/files/documents/2018/02/13/cyber_ranges.pdf` Accessed Feb 13, 2018.

[4] Jan Vykopal, Radek Ošlejšek, Pavel Čeleda, Martin Vizvary, and Daniel Tovarňák. Kypo cyber range: Design and use cases. *SciTePress*, 2017.

[5] Razvan Beuran, Cuong Pham, Dat Tang, Ken-ichi Chinen, Yasuo Tan, and Yoichi Shinoda. Cybersecurity education and training support system: Cyris. *IEICE TRANSACTIONS on Information and Systems*, 101(3):740–749, 2018.

[6] Terry Benzel. The science of cyber security experimentation: the deter project. In *Proceedings of the 27th Annual Computer Security Applications Conference*, pages 137–148, 2011.

[7] Le Xu, Dijiang Huang, and Wei-Tek Tsai. Cloud-based virtual laboratory for network security education. *IEEE Transactions on Education*, 57(3):145–150, 2013.

[8] Inc. Amazon Web Services. Aws educate. `https://aws.amazon.com/education/awseducate/?nc1=h_ls` Accessed 2015.

[9] Weiqing Sun, Varun Katta, Kumar Krishna, and R Sekar. V-netlab: An approach for realizing logically isolated networks for security experiments. *CSET*, 8:1–6, 2008.

[10] Wenliang Du and Ronghua Wang. Seed: A suite of instructional laboratories for computer security education. *Journal on Educational Resources in Computing (JERIC)*, 8(1):1–24, 2008.

[11] Jon Davis and Shane Magrath. A survey of cyber ranges and testbeds. Technical report, DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION EDINBURGH (AUSTRALIA) CYBER AND ..., 2013.

[12] Razvan Beuran, Cuong Pham, Dat Tang, Ken-ichi Chinen, Yasuo Tan, and Yoichi Shinoda. Cytrone: An integrated cybersecurity training framework. *SCITEPRESS–Science and Technology Publications*, 2017.

[13] Richard H Wagner. Designing a network defense scenario using the open cyber challenge platform. -, 2013.

[14] Yanyan Li and Mengjun Xie. Platoon: A virtual platform for team-oriented cybersecurity training and exercises. In *Proceedings of the 17th Annual Conference on Information Technology Education*, pages 20–25, 2016.

[15] Yanyan Li, Dung Nguyen, and Mengjun Xie. Ezsetup: A novel tool for cybersecurity practices utilizing cloud resources. In *Proceedings of the 18th Annual Conference on Information Technology Education*, pages 53–58, 2017.

[16] Shingo Yasuda, Ryosuke Miura, Satoshi Ohta, Yuuki Takano, and Toshiyuki Miyachi. Alfons: A mimetic network environment construction system. In *International Conference on Testbeds and Research Infrastructures*, pages 59–69. Springer, 2016.

[17] Stefan Boesen, Richard Weiss, James Sullivan, Michael E Locasto, Jens Mache, and Erik Nilsen. Edurange: meeting the pedagogical challenges of student participation in cybertraining environments. In *7th Workshop on Cyber Security Experimentation and Test ({CSET} 14)*, 2014.

[18] Dejan Milojičić, Ignacio M Llorente, and Ruben S Montero. Opennebula: A cloud management tool. *IEEE Internet Computing*, 15(2):11–14, 2011.

[19] Omar Sefraoui, Mohammed Aissaoui, and Mohsine Eleuldj. Openstack: toward an open-source solution for cloud computing. *International Journal of Computer Applications*, 55(3):38–42, 2012.

[20] Richard Weiss, Jens Mache, and Michael Locasto. Edurange: hands-on cybersecurity exercises in the cloud. *Journal of Computing Sciences in Colleges*, 30(1):178–180, 2014.

[21] Zdenek Eichler, Radek Ošlejšek, and Dalibor Toth. Kypo: A tool for collaborative study of cyberattacks in safe cloud environment. In *International Conference on Human Aspects of Information Security, Privacy, and Trust*, pages 190–199. Springer, 2015.

[22] Inc. Amazon Web Services. Boto3 docs 1.14.4 documentation. `https://boto3.amazonaws.com/v1/documentation/api/latest/index.html` Accessed 2020.

[23] Amazon elastic compute cloud user guide for linux instances. `https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-ug.pdf#ec2-key-pairs`.

[24] Amazon elastic compute cloud user guide for linux instances. `https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-ug.pdf#ec2-resource-limits`.