Building a Cyber Range for UVA Computer Science Students

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Chase Hildebrand

Spring, 2024

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Briana Morrison, Department of Computer Science

Building a Cyber Range for UVA Computer Science Students

CS4991 Capstone Report, 2024

Chase Hildebrand
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
chaseh@virginia.edu

ABSTRACT

In recent years, the cybersecurity skill gap has increased dramatically. Our team is combating this problem by building a cyber range, designed to simulate realistic networks, for UVA students to get hands-on experience with cybersecurity. To build this system, we chose OpenStack as our virtualization platform, and Ceph as our storage solution. We designed our network to support 2000 concurrent virtual machines and to be flexible, resilient, secure, and scalable. This project is expected to bring computational resources to over 200 UVA students seeking to learn cybersecurity. Going forward, we want to improve automation and explore integrating this system with course curriculum to enhance cybersecurity education at UVA.

1. INTRODUCTION

As the frequency and complexity of cyber attacks reaches unprecedented levels, the need for highly skilled cybersecurity professionals has never been more pressing. A study conducted in 2023 revealed a global shortage of approximately 4 million skilled cybersecurity professionals [2]. However, despite this demand, students often encounter significant barriers to gaining practical experience in cybersecurity. These barriers in part stem from the formidable technical expertise required to set up learning environments and the need for substantial computing infrastructure to simulate larger networks. These hurdles limit students' access to hands-on learning opportunities, leaving many to rely solely on classroom instruction, which, in addition to being sparse at UVA, often falls short at providing the realistic applications needed to prepare them for the demands of the field.

To address these problems, our team plans to develop a platform designed to empower UVA students to easily engage in hands-on learning, giving them the freedom to explore within reasonable boundaries while preventing inadvertent harm to adjacent systems and the internet at large. Additionally, it will equip professors, club leaders, and competition organizers with the resources to provide virtual environments to many students at scale. Moreover, this initiative will inspire and serve as a starting point for other educational institutions who want to build similar systems for their cybersecurity students, promoting accessibility throughout the field.

2. RELATED WORKS

This project builds on the work done by Emil Baggs in 2023 [1]. He presents a system for a small-scale cyber range with emphasis on automation to build training networks for students. The system was transformative within its niche, but struggled with building and running large networks. This project builds upon his work and addresses its shortcomings by redesigning the cyber range from the ground up, emphasizing scalability, reliability, and user experience.

3. PROJECT DESIGN

We completed this project in four phases: requirements elicitation, software selection, system design, and implementation.

3.1 Stakeholders

While its application is not imminently planned, to account for predicted future needs, this project needs to support a wide range of stakeholders and varying use cases. These stakeholders include Computer and Network Security (CNS, the cybersecurity club at UVA) members, CNS presenters, CS students doing independent research or personal practice, competitive cyber teams, and professors. Each of these stakeholders has a specific use case that we foresee. CNS presenters need a way to host workshops for CNS members. CS students and CNS members need to be able to participate in workshops and have a ground to build networks to experiment with. Competitive teams need an arena to practice live cybersecurity engagements. Professors need a way to host assignments and labs. Each of these stakeholders needs their environment segmented from others, but also need a way to collaborate in some cases.

3.2 Design Requirements

Accounting for the needs of the stakeholders, we curated a list of design requirements. We separated this list into three sections: critical, important, and nice to have.

3.2.1 Critical

- Users should be able to create and delete VMs, networks, and routers without admin intervention
- User-created VMs should be able to reach the internet, if desired
- Networks created by users should be completely segmented from others' networks
- Simulated systems and networking should emulate real environments as close as possible
- Users should be able to share projects (groups of VMs) between each other
- Physical aspects of computing environments should be abstracted from the user
- System should support 100 concurrent users and 1000 running VMs
- System should have a web interface to control the system
- System should be designed to easily scale

3.2.2. Important

- Users should be able to log in with NetBadge
- System should log user activity
- Admins should be able to set resource quotas for users
- Users should be able to share projects without admin intervention
- System should support 300 concurrent users and 3000 running VMs
- User interface should be easy to use for the average CS student
- Automation support

3.2.3. Nice to Have

- Users should be able to connect to their networks with a VPN
- System should automatically or offer a self-service option to configure VPNs for user networks
- System should automatically deallocate resources that are not being used to save compute power

3.3 Hardware Requirements

Based on the requirement of having 3000 running VMs, we estimate that each Linux server (no GUI) VM will use around 2 GB of RAM and 1 CPU core and each Windows or Linux Desktop VM will use 8 GB of RAM and 4 CPU cores. We also allocate 32 GB of disk space per instance on average. We expect about a 50/50 split GUI to non GUI machines, which puts the average resources per instance at 5 GB of RAM and 2.5 CPUs. We can allow 4x overprovisioning on CPU. This adds up to 3000 * 2.5 / 4 = 1875 vCPU cores, 3000 * 5 = 15,000 GB of RAM, and 96TB of disk space total, which we rounded to 100 TB. In practice, due to budget constraints, we had to alter these numbers down to 10,000 GB of RAM and 1250 vCPU cores, keeping the 100 TB of disk space. These specifications will support a minimum of 2000 concurrent VMs. While this is less than the ideal 3000, it is not too great a problem because our system is designed to be scalable, so adding compute later is easy and 2000 VMs is within the 1000 to 3000 range outlined in the design requirements.

3.4 Software Selection

To determine which software and system architectures best fit our requirements, we first searched online for software that advertised support for all of our critical requirements and most of our important requirements. After building a short list of potential solutions, we evaluated each one. To evaluate each of these systems, we built virtualized mock-ups of each of them in our private cloud using a technique that lets us run hypervisors inside of virtual machines, called nested virtualization. We then evaluated each software's effectiveness of fulfilling our design requirements using the following criteria: presence of important features, presence of nice-to-have features, user interface design, ease of setup and maintenance, availability of automation tooling, availability of community support, scalability, extensibility. The software we chose to evaluate are Apache CloudStack, OpenStack, OpenNebula, and Proxmox, because they are well-maintained, featureful, and have community support. We chose not to use any enterprise or paid software because of vendor lock-in and funding considerations in the future.

We ultimately decided to build OpenStack because of its scalability, extensibility, availability and presence of automation software, and the availability of decent documentation and community support. The main downside to this option is its complexity and its user interface design, which is somewhat confusing, ugly and sluggish.

3.5 System Design

Our system consists of three control servers, twelve compute servers, and three storage servers. The control servers are responsible for managing the cluster. The compute servers host the virtual machines and the storage servers store data. Due to the extensive design requirements and sheer scale, the in-depth implementation of this cluster is outside the scope of this report. This section will provide a high level overview of each of the components that we implemented and what they are used for.

3.5.1 Provisioning and Deployment

Before we can install a server into the cluster, we need to install an operating system and install and set up the OpenStack software on it. The process of getting servers ready to use is called *provisioning*. The process of taking a provisioned server and installing software on it is called *deploying*. Rather than manually doing this for each server in our cluster, we decided to automate the process. This also makes it easy to add, remove, or repurpose servers as needed, making our system rapidly scalable.

To implement this, we used two technologies: metal as a service (MaaS) for provisioning and Kolla Ansible for deployment. MaaS has the capability of automatically provisioning servers. To register a device in MaaS all we have to do is plug it in to the network and turn it on. MaaS also has an interface that tracks resources, allowing us to remotely manage bare-metal resources with ease. Kolla Ansible takes a configuration state and deploys services on the network to match that state. Kolla Ansible provides additional flexibility to our cluster design by allowing us to quickly move or redeploy services as needed.

3.5.2 Storage

For storage, we run Ceph on the three storage servers. Ceph is a distributed storage system and supports the three different types of storage that we need: block, object, and file storage. We decided to use Ceph because it is widely supported in the community and well-supported by OpenStack. Because Ceph is distributed, it provides redundant storage so if one node goes offline unexpectedly or is taken down for maintenance, the cluster will continue to be fully operational. Additionally, Ceph provides triple replication across disks. This means that there are three copies of data stored in case of disk or system failure.

3.5.3 Networks

The networking is the most sophisticated part of the cluster design. To build this cluster, we split our network into seven sub networks, called VLANs, or “virtual local area networks.” VLANs segment a physical network into multiple logical networks without having to add

additional network equipment or cables. These VLANs are outlined below:

- VLAN 28 iDRAC
  Network for managing physical hardware. This network is used by MaaS to manage hardware.
- VLAN 29 PXE Boot
  Used by MaaS to provision servers using PXE boot, which is a way to network boot servers.
- VLAN 30 Openstack management network
  Used for internal OpenStack communication and management by administrators.
- VLAN 31 API Network
  Used by users accessing the cluster
- VLAN 32 External storage network
  Used by Ceph to share data with OpenStack.
- VLAN 33 External network
  Provider network used for VM traffic going to the internet.
- VLAN 40 Internal storage network
  Used by Ceph to replicate data between nodes. Unlike the other networks, which are 10Gbps, this network is 40Gbps.
- VLAN 999 Public
  This is UVA's public network. We have a Palo Alto firewall between VLAN 30, 31, 33 and the 999 (the public internet). This firewall is used to regulate traffic and log requests for accountability.

Our network is fully redundant, so if any cable or networking appliance fails, the cluster will continue to operate as normal. To implement this, we have two switches running virtual chassis. This allows the second switch to take over if the first one fails. Each host is connected to both switches and uses a protocol called LACP that enables redundancy.

4. EXPECTED RESULTS

This project is expected to bring the resources to build realistic cybersecurity environments to 200 students. It will enable CNS to host larger competitions internally that have historically been limited in size due to the lack of computing resources. This project will also enable running larger scale intercollegiate competitions.

5. CONCLUSION

Building a cyber range is a big step towards training a strong cybersecurity workforce. As the frequency of cyber attacks increases, this is becoming critically important. The cyber range will add to the list of resources that distinguish UVA as a leader in cybersecurity.

6. FUTURE WORK

There are still many ways this project can be expanded upon. Currently, there is no automation to build networks implemented outside of what OpenStack provides. In the future, we hope to integrate the automation that we developed last year [1] with this project.

We would like to open this project for professors to use in their classes. While students can use it in its current state, going through the process of getting the cyber range approved as an official educational tool and ensuring it follows education laws and university guidelines is a subject for future work. Additionally, because allowing professors to use it for their classes would attract a larger user base, we may need to scale up the system to account for the additional load.

7. ACKNOWLEDGMENTS

I would like to thank the Jefferson Trust for graciously providing us with a $71,548.79 grant to build this project. I would also like to thank members of the CNS systems team, especially Vincent Zhang, Shreyas Mayya, and Lulu Han, for their hard work on this project.

REFERENCES

[1] Emil Baggs. 2023. Designing and Building a Virtual Cyber Security Range; Analyzing the Competition Between Internet Protocol Versions 4 and 6. Retrieved April 29, 2024 from https://doi.org/10.18130/p5e0-xz56

[2] ISC2 Cybersecurity Workforce Study: Looking Deeper into the Workforce Gap. Retrieved May 8, 2024 from https://www.isc2.org/Insights/2023/11/ISC2-Cybersecurity-Workforce-Study-Looking-Deeper-into-the-Workforce-Gap