Evaluation of Vulnerability Reproducibility in Container-based Cyber Range

Ryotaro Nakata[a] and Akira Otsuka[b]
Institute of Information Security, Yokohama, Kanagawa, Japan

Keywords: Information Security Education, Cyber Range, Container-based Virtualization, Docker, Vulnerability.

Abstract: The cyber range is a practical and highly educational information security exercise system, but it has not been widely used due to its high introduction and maintenance costs. Therefore, there is a need for a cyber range that can be adopted and maintained at a low cost. Recently, container type virtualization is gaining attention as it can create a high-speed and high-density exercise environment. However, existing researches have not clearly shown the advantages of container virtualization for building exercise environments. Moreover, it is not clear whether sufficient vulnerabilities are reproducible, required to conduct incident scenarios in the cyber range. In this paper, we compare container virtualization with existing virtualization type and confirm that the amount of memory, CPU, and storage consumption can be reduced to less than 1/10 of the conventional virtualization methods. We also compare and verify the reproducibility of the vulnerabilities used in common exercise scenarios and confirm that 99.3% of the vulnerabilities are reproducible. The container-based cyber range can be used as a new standard to replace existing methods.

1 INTRODUCTION

With the development of ICT technology, the scale and impact of cyber-attacks continue to increase worldwide.On the other hand, the shortage of human resources for security is pointed out, and the government and higher education institutions are making various efforts for human resource development. Still, the quantitative and qualitative shortage has not been solved (Maki et al., 2020).

Information security skills can be effectively learned through education using cyber ranges. The cyber range is a large scale exercise system for learning by experiencing real security incidents in an organization in a virtual environment that simulates a real-world system. Although the educational effect is high, the cyber range's introduction and maintenance are millions of dollars (Razvan et al., 2017).

Although the effectiveness and necessity of cyber ranges are recognized, it is difficult for educational institutions to implement and maintain them independently. Therefore, a low-cost training environment is needed, and the use of container type virtualization is attracting attention (Irvine et al., 2017).

However, previous research does not show the advantages of container type virtualization in concrete terms. Containers are not suitable for recreating realistic environments and are generally considered to be limited to specific applications. As a result, most existing cyber range products and educational research use other virtualization types (Razvan et al., 2017).

This paper uses a container-based cyber range environment to examine the differences between virtualization types and confirm the performance benefits of container type virtualization in cyber range. We also describe the results of our comparison with other virtualization types by applying vulnerability checking tools and attack programs to see how well the vulnerabilities and incidents in commonly used exercise scenarios can be reproduced in cyber range exercises.

2 THE CYBER RANGE

2.1 Definition of Cyber Range

The cyber range is a system to build and provide an environment for information security exercises. For smooth implementation of the exercises, with the following functions.

[a] https://orcid.org/0000-0001-8885-848X
[b] https://orcid.org/0000-0001-6862-2576

Nakata, R. and Otsuka, A.
Evaluation of Vulnerability Reproducibility in Container-based Cyber Range.
DOI: 10.5220/0010324606350642
In *Proceedings of the 7th International Conference on Information Systems Security and Privacy (ICISSP 2021)*, pages 635-642
ISBN: 978-989-758-491-6

- Creating a realistic system environment
  The system environment can be faithfully reproduced as it is used in the real world.
- Duplicate or replace the environment
  Exercise environments can be reset, replicated, or replaced flexibly and quickly.
- Reproduction of vulnerabilities and incidents
  Vulnerable environments and actual malware can be used to recreate security incidents and execute attack and defense scenarios.

These functions are made possible by virtualization technology (Costa et al., 2020).

2.2 Typical Cyber Range Scenarios

In the cyber range exercise, various scenarios will be developed depending on the learning objectives and the student's level. Figure 1 shows a network environment that reproduces a typical attack scenario.

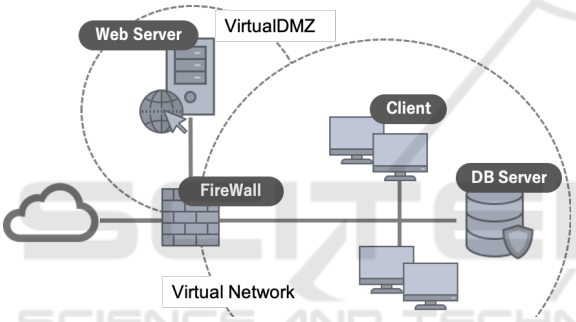

Figure 1: Example of a virtual network for cyber range.

In the cyber range, reproduce vulnerabilities in web servers, DB servers, client devices, etc., to carry out the scenario. Also, security devices such as firewalls, IDS/IPS, etc. will be installed, depending on the exercise's nature. To execute scenarios in the cyber range that replicate real security incidents and responses, prepare a virtual environment equivalent to the real system environment (Stout et al., 2018).

3 VIRTUALIZATION TECHNOLOGY

3.1 Types of Virtualization Technologies

Virtualization technology efficiently utilizes hardware by sharing and dividing the resources required for operating systems and applications among multiple environments (VM: Virtual Machine) (Ameen and Hamo, 2013). Figure 2 shows an overview of each virtualization technology type.

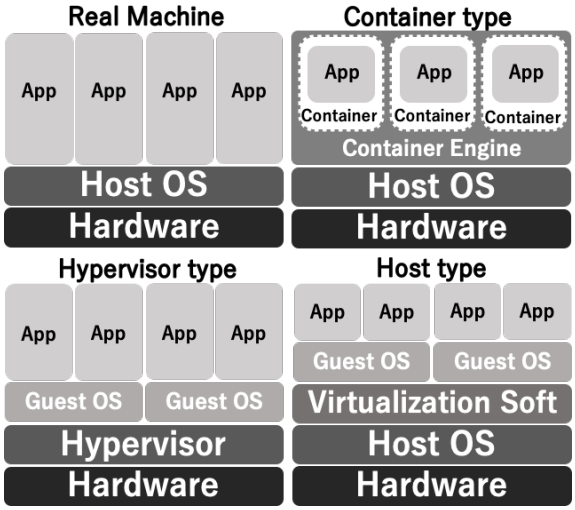

Figure 2: Overview of each virtualization types.

In the hypervisor type, a program called hypervisor builds a VM on specially prepared hardware, and it operates by occupying resources such as memory and CPU. In the host type, a VM is built by running dedicated software that plays a hypervisor role on an operating system (host OS) running on the actual machine. A portion of the resources recognized by the host OS is allocated and operated.

Unlike other virtualization types, the container type operates like a VM by creating a separate namespace, called a container, on a running host OS that operates only the processes required for the functions to be used. Containers do not occupy physical resources and run as a single process on the host OS. Table 1 shows the characteristics of each virtualization types.

Table 1: Characteristics of each virtualization types.

Virtualization type	isolation level	over head	guest OS
Hypervisor	max	high	require
Host	high	max	require
Container	low	low	unrequire

The isolation level indicates independence from the host OS and other VMs running on the same hardware. If the isolation level is low, there is a high probability that the host OS and other VMs will be affected if processing on the VM is slow or troubles occur. Overhead refers to the decrease in processing performance that occurs in a virtual environment. Since the hardware is accessed through the mechanism used to run the VM, there is a high probability that processing performance will decrease compared to the actual environment.

Figure 3 shows the operating architecture of each virtualization types In the hypervisor and host type,

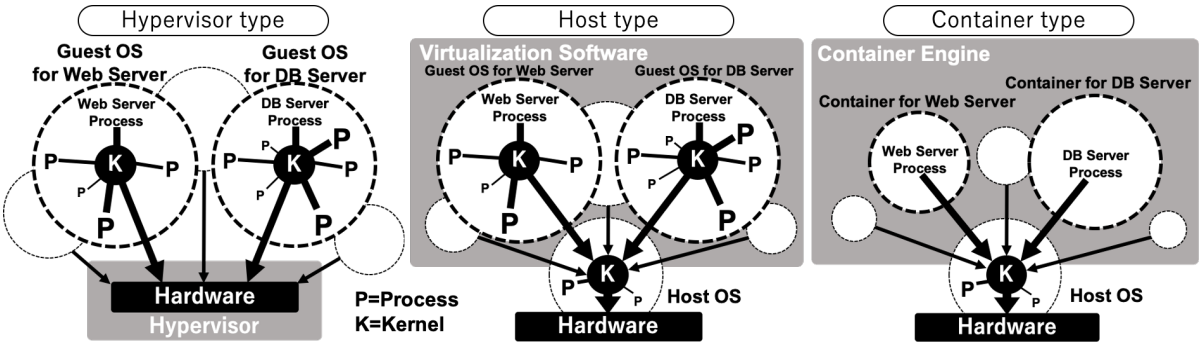

Figure 3: Operating architecture of each virtualization type.

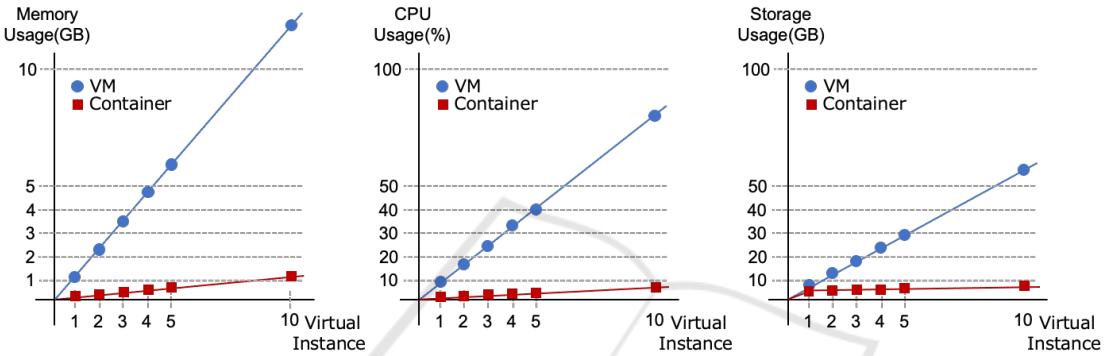

Figure 4: Comparison of resource consumption as the number of virtual instances increases.

the VMs run independently and replicate the real machine's equivalent environment. Still, they require installing a guest OS, which consumes a large number of physical resources (Li et al., 2017). The guest OS does not need to be started or stopped in the container type, and only the necessary functions can be run with minimal configuration. Containerized systems share kernels and resources. They can be fast and efficient. Still, they are affected by interactions with the host OS and other containers and may behave differently from the real machine (Preeth E N et al., 2015).

3.2 Virtualization Type Used in the Cyber Range

Existing cyber ranges use HV and hosted virtualization types. In particular, virtualization solutions such as VMWare and Citrix are being used in the commercial cyber range for their stability and VM management capabilities (VMWare, 2019). However, depending on the number of participants and the number of exercise groups, there are more than 100 virtual instances running at the same time, requiring high load operations such as environment replication and rapid startup/termination. Therefore, high-performance hardware is required, causing increased costs (Maki et al., 2020).

3.3 Advantages of Container-based Cyber Range

Cyber range environments using containerized virtualization are faster and less resource-intensive than other virtualization types. Figure 4 shows a comparison of the resources consumed by VMs and containers as the number of virtual instances increases.

In the example of the figure 4, the OS was installed on a VM with 1GB memory allocated and prepared as a device intended to be a client for the cyber range environment. We also installed the same OS and desktop package as the VM in a container running on the same host. The resource consumption was compared by preparing the container as a container that allowed the same operations as the VM.

VMs always consume host resources without any particular actions, such as running a guest OS and various services. Many processes that are not necessary to execute cyber range scenarios are also running, and even virtual machines that are not explicitly running consume a certain amount of resources, which is inefficient.

Containers, on the other hand, consume very few resources per instance because they run on a minimal number of processes. Therefore, even if the cyber range environment is built and the number of virtual

instances increases, the physical resource consumption can be significantly reduced compared to a VM environment. By using containers to build a cyber range environment, the required specifications of the host machine can be significantly reduced.

3.4 Concerns of Container

Using container type virtualization can build a cyber range environment at a lower cost. However, container type virtualization has different characteristics than VMs used in the existing cyber range and may have different states and behaviors in executing scenarios. As a result, individual vulnerabilities and incidents cannot be reproduced correctly and may not work as envisioned. The biggest concern with the container-based cyber range is whether the vulnerabilities required to execute a scenario can be reproduced on a container similar to on a VM.

4 EXPERIMENTS ON VULNERABILITY REPRODUCIBILITY

4.1 Reproducibilty Metrics

To assess the reproducibility of vulnerabilities on a container-based cyber range, we model the OS/HW environment in which the programs as an oracle O and every system calls invoked by a program A is sent to the oracle O. If a program is run in the environment over Real, VM, container, we write the output of the programs as $A^{O_{\text{Real}}}$,$A^{O_{\text{VM}}}$ and $A^{O_{\text{Container}}}$ respectively. Where "Real" means the physical environment where no virtualization technology is used.

If all programs' execution results are the same, there is no problem executing any exercise scenario, and it eliminates concerns of container-based cyber range. Theoretically, the identification algorithm ϕ can be defined as the inability to identify which environment the program A was executed.

We write the set of results of running the program in each environment as A^{Real}, A^{VM}, and $A^{Container}$, respectively, as shown in the following equation.

$$A^{Real} = \left\{ A \mid \phi\left(A^{O_{\text{Real}}}\right) = 1 \right\}$$
$$A^{VM} = \left\{ A \mid \phi\left(A^{O_{\text{VM}}}\right) = 1 \right\}$$
$$A^{Container} = \left\{ A \mid \phi\left(A^{O_{\text{Container}}}\right) = 1 \right\}$$

By measuring these sets' similarity, we confirm the reproducibility of vulnerabilities in the container-based cyber range. The relationship between each set is shown in Figure 5.

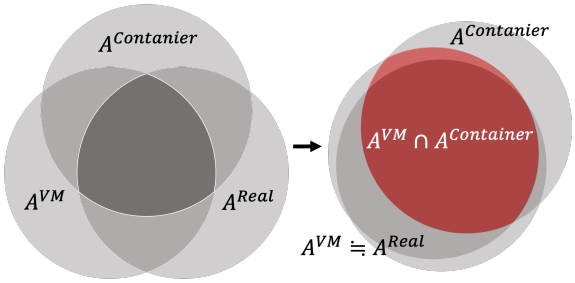

Figure 5: Reproducibility Comparison Model.

Considering the current use of VMs in the cyber range, we will consider Real and VM's nearly identical. Thus, J, which represents the similarity between VMs and containers, be the metric for using container type virtualization in the cyber range.

$$J\left(A^{VM}, A^{\text{container}}\right) = \frac{\left|A^{VM} \cap A^{\text{Container}}\right|}{\left|A^{VM} \cup A^{\text{Container}}\right|}$$

The higher the value of J, the less concern there is about the container-based cyber range. In reality, some programs that are difficult to reproduce by containers, such as those related to physical vulnerabilities. These should be excluded from the container-based cyber range and will be discussed in Chapter 6.

4.2 Experimental Method

We compared the container and VM environment through an exhaustive experiment with programs used in cyber range exercise scenarios. We built an equivalent environment with VMs and containers to confirm a difference between the vulnerability assessment tool's scan results and the results of attacks against the detected vulnerabilities. Table 2 shows the vulnerability assessment tools used in the experiment.

Table 2: Vulnerability inspection tool used for verification.

Name	Target			
	Web App	Middle ware	OS	Net work
OpenVAS		○	○	
Nmap				○
Owasp ZAP	○			

We have selected tools that can be used in cyber range exercises and divided the target areas into four areas: web applications, middleware, OS, and networks. Using tools capable of scanning for each area,

we thought we could perform a comprehensive experiment for various vulnerabilities.

4.3 Experimental Settings

We experimented with a Docker container and a VirtualBox VM. Docker is a platform for container type virtualization, becoming more popular for various applications such as cloud services. VirtualBox is a free hosted virtualization software widely used in verification environments, education, and research because of its ease of installation and many supported operating systems. The environments used for verification are shown in Table 3.

Table 3: Experimental Settings detail.

env	Hardware VM software and Image	Spec
Host	**MacBookPro-13inch** Ubuntu 18.04 LTS	2.3GHz Corei5 16GB RAM
VM	**VirtualBox6.0** Metasploitable2/3 Kalilinux2019.1	1CPU 2GB RAM
container	**DockerCE18.09** Metasploitable2/3(created from VM) kalilinux/kali-linux-docker(official)	

We used Kali-linux, a Linux distribution for penetration testing, and Metasploitable2/3, which are used as a verification environment for deliberately adding vulnerabilities to containers and VMs, and compared the vulnerability assessment results and exploit tests (Kali.org, 2020; Rapid7, 2020).

Since official container images of Metasploitable 2 and 3 are not available to the public, we created container images from each VM machine. The original container image was created using the docker import command, which collects files other than /boot, /dev, /mnt, /proc, /sys, /tmp that are unnecessary for container operation using the tar command creates the base container image from the archive file. We have created an environment that works with the same configuration, although the startup process is different.

5 EXPERIMENTAL RESULTS

5.1 Measuring Reproducibility with Vulnerability Assesment Tools

5.1.1 OpenVAS

OpenVAS is an open-source vulnerability testing tool with rich testing capabilities. It can detect a wide

Table 4: Comparison of the number of vulnerabilities detected by OpenVAS.

CWE	vulnerability classification	VM	container
16	Configuration	2	2
17	Code	2	2
18	Source Code	1	1
20	Improper Input Validation	82	82
22	Pass Traversal	7	7
59	Link Following	6	6
74	Injection	1	1
79	Cross Site Scripting	33	33
89	SQL Injection	7	7
93	CRLF Injection	4	4
94	Code Injection	13	13
113	HTTP Response Splitting	1	1
119	Buffer Error	66	66
125	Out-of-bounds Read	2	2
134	Use of Externally-Controlled Format String	5	5
189	Numeric Errors	31	31
190	Integer Overflow or Wraparound	3	3
200	Exposure of Sensitive Information to an Unauthorized Actor	42	42
254	7PK - Security Features	2	2
255	Credentials Management Errors	2	2
264	Permissions, Privileges, and Access Controls	54	54
275	Permission Issues	1	1
284	Improper Access Control	5	5
287	Improper Authentication	8	8
295	Improper Certificate Validation	2	2
310	Cryptographic Issues	22	22
311	Missing Encryption of Sensitive Data	22	22
320	Key Management Errors	2	2
327	Use of a Broken or Risky Crypto-graphic Algorithm	1	1
345	Insufficient Verification of Data Authenticity	1	1
352	Cross Site Request Forgery	5	5
362	Race Condition’	9	9
384	Session Fixation	1	1
399	Resource Management Errors	51	51
400	Uncontrolled Resource Consumption	2	2
415	Double Free	1	1
416	Use After Free	2	2
476	NULL Pointer Dereference	8	8
502	Deserialization of Untrusted Data	2	2
552	Files or Directories Accessible to External Parties	1	1
601	Open Redirect	5	5
732	Incorrect Permission Assignment for Critical Resource	1	1
772	Missing Release of Resource after Effective Lifetime	1	1
787	Out-of-bounds Write	1	1
835	Infinite Loop	8	8
Design	Design errors	4	4
Other	Other errors	50	50
noinfo	Lack of information	123	123

range of vulnerabilities such as software bugs, usage flaws, and configurations and check for corresponding CVEs (Common Vulnerabilities and Exposures) (MITRE, 2020a) using the latest database (GreenboneNetworks, 2020). Table 4 shows the results of an experiment by OpenVAS. Since the number of CVE detections is very high, it is aggregated by CWE (Common Weakness Enumeration) (MITRE, 2020b).

For the items confirmed in vulnerability scans for Metasplotable 2 and 3 by OpenVAS, we could confirm the same results for the VM and container. There was no difference in the functions and services running in both environments, and the vulnerabilities detected were consistent.

5.1.2 Nmap

Nmap is an open source port scanner with extensive OS and service version detection capabilities (Nmap.org, 2020). In cyber-range exercises, they are often used in the early stages of exercise scenarios, for example, to respond to incidents after port scan detection. Therefore, experimental results from Nmap are also important in a container-based Cyber-range environment. Table 5 shows the results of detection by NMAP.

Table 5: Detection results by Nmap, and CWE classification.

CWE	Port	Service	Version
189,339	21	ftp	vsftpd 2.3.4 ProFTPD 1.3.5
119,200	22	ssh	OpenSSH 4.7p1 OpenSSH 6.6 1p1
254,416	53	dns	ISC BIND 9.4.2
79,287	80	http	Apache 2.2.8 Apache 2.4.7
399	111	rpcbind	2 (RPC #100000)
22,275	139	samba	Samba smbd 3.X-4.X
264,290	631	ipp	CUPS 1.7
94,119	1099	java-rmi	Java Rmi Registry
264	2049	nfs	2-4(RPC #100003)
22,399	2121	ftp	ProFTPD 1.3.1
134,189	3306	MySQL	MySQL 5.0.51a MySQL 5.5.62
20	8181	http	Webrick httpd 1.3.1
other	3632	distccd	distccd v1
264,284	5432	postgresql	PostgreSQL DB8.3.0-8.3.7
other	5900	vnc	VNC protocol 3.3
20,189	6667	irc	UnrealIRCd
16	8009	ajp13	Apache Jserv (Protocol v1.3)
20,119	8180	http	Tomcat/Coyte JSP engine 1.1
189,399	8787	drb	Ruby DRb RMI

Table 6: Comparison of the number of detected vulnerabilities by ZAP.

CWE	Vulnerability	Metasploitable2		Metasploitable3	
		VM	container	VM	container
89	SQL Injection	358	422	2	2
97	Server Side Include	1	1	0	0
79	XSS(refrected)	1075	1000	1	1
79	XSS(stored)	5	5	0	0
22	Pass Traversal	21	21	0	0
78	Command Injection	361	342	0	0
98	File Inclusion	209	206	0	0
200	Application error disclosure	242	246	1	1
548	Directory Browsing	14	15	21	21
472	Parameter tampering	13	14	1	0
200	Buffer Error disclosure	291	287	1	1
200	Private IP disclosure	136	139	1	1

Experiments of port scanning by Nmap gave the same detection results in VM and container. As shown in Table 5, we checked the vulnerabilities in the detected contents and confirmed the corresponding CWE.

5.1.3 OWASP ZAP

ZAP is an open-source web application vulnerability assessment tool that can detect many vulnerabilities and check how to deal with them (Makino and Klyuev, 2015). Metasploitable 2 and 3, prepared as vulnerable environments, have several web application environments, and web pages with various vulnerabilities, can be checked. Table 6 shows a comparison of the results of the ZAP vulnerability check.

In some cases, more vulnerabilities were detected by containers. This result was unexpected. It could not be determined whether this was due to the characteristics of the container or the ZAP characteristics. Since many vulnerabilities have been detected, it is possible to utilize them in a cyber range environment fully, but we think that clarifying the causes of the lack of matches will help to define the possible use of containers better.

5.2 Measuring Reproducibility with Exploit Modules

The Metasploit framework included in Kali-linux can use a variety of exploit modules. We compared the results of the attack experiments against Metasploitable 2 and 3 in the VM and container environments to see if the attacks were successful and if the display and behavior were identical. The exploit modules were

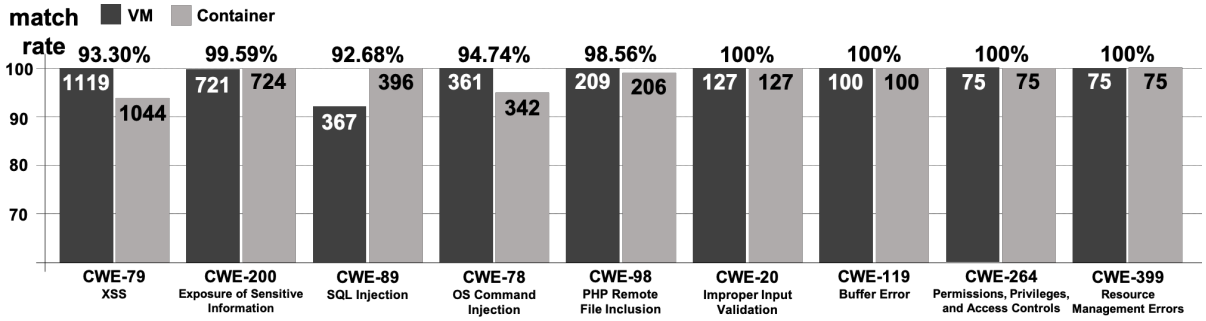

Figure 6: Match rate of vulnerability reproducibility about VM and container.

Table 7: Modules used for attack testing and CWE classification.

CWE	Metasploit Module
119	auxiliary/scanner/ssl/openssl_heartbleed
200	auxiliary/scanner/ssh/ssh_enumusers auxiliary/scanner/http/options auxiliary/scanner/http/trace auxiliary/scanner/http/tomcat_enum
310	auxiliary/scanner/http/ssl_version auxiliary/scanner/ssl/openssl_ccs
416	auxiliary/scanner/http/apache_optionsbleed
other	auxiliary/scanner/rservices/rexec_login auxiliary/scanner/rservices/rlogin_login auxiliary/scanner/rservices/rsh_login
119	auxiliary/server/openssl_heartbeat_client_memory
94	exploit/linux/samba/is_known_pipename
189	exploit/linux/samba/setinfopolicy_heap
16	exploit/unix/misc/distcc_exec
20	exploit/unix/irc/unreal_ircd_3281_backdoor
284	exploit/unix/ftp/proftpd_modcopy_exec
other	exploit/unix/webapp/twiki_history
20	exploit/multi/browser/java_storeimagearray
other	exploit/multi/http/php_cgi_arg_injection
noinfo	exploit/multi/samba/usermap_script

selected from those capable of attacking vulnerabilities detected by the vulnerability assessment tool used in the 5.1 experiment.

Table 7 shows the modules used in the attack experiments. All the exploit modules used worked as expected. We confirmed the reproducibility of the attacks when used in the cyber range exercise, including the detection of account information using vulnerabilities, unauthorized login, and successful privilege escalation.

5.3 Overall Results

We checked the percentage of coincidence between VMs and containers for all experiments performed. Figure 6 shows the total number of detections for each CWE and the percentage of identical results between VMs and containers for the vulnerabilities and detection items identified in each experiment. Because the items are displayed in order of number, there are many CWE items that were detected but are not shown or that do not correspond to CWEs.

The similarity J calculated from the results of the experiment was 0.993. This value is higher than expected, and we believe it is an acceptable value for the construction of a suitable cyber range environment and the use of exercise scenarios. This experiment largely eliminates the cyber range concerns of container-based cyber range.

6 LIMITATIONS OF REPRODUCIBILITY

Vulnerabilities related to physical resources are difficult to reproduce due to the characteristics of containerized virtualization. These are not suitable for cyber range environments and should be excluded. In reality, we checked what differences occur between containers and VMs. Table 8 shows the results of an attack on a VM and a container.

Table 8: Physical resource consumption and behavior of virtual environment.

resource	type	results
memory	VM	Program is delayed at VM memory allocation limit, but other VMs are not affected.
	container	When physical memory usage reaches a limit, the entire operation, including the host OS, is delayed.
strage	VM	Machine stops at VM storage allocation limit, but other VMs are not affected.
	container	When physical storage usage reaches a limit, the entire operation, including the host OS, is stopped.

A VM occupies a portion of the resources on the host OS and operates as an independent machine. Therefore, even if an attack slows down or stops the VM, it does not interfere with the host OS's operation. However, the standard configuration of the container shares resources with the host OS and other contain-

ers. An attack on the container can affect the entire system, including the host OS, which is a clear difference from the VM. Against such attacks, it was confirmed that container type virtualized environments are unsuitable for cyber range exercises. Similarly, vulnerabilities such as kernel vulnerabilities, host OS configuration, and files required for container execution, and attacks against them, should be excluded from container-based cyber range exercises.

7 CONCLUSIONS

With the shortage of information security personnel, the cyber range is expected to be highly effective in education. However, the cost of cyber range is high, making it difficult for educational institutions to implement them independently.

Therefore, to disseminate an inexpensive and deployable cyber range using container-based virtualization, we confirmed the superiority of containers and the performance of reproducing vulnerabilities through comprehensive experiments. Comparing the performance of containers and VMs in a typical cyber-range environment, the containers consumed less than 1/10th of the resources of the VMs. Containers can run more virtual instances than VMs on the same host, building a lower cost cyber range.

We also compared the reproducibility of vulnerabilities between containers and VMs in an exhaustive experiment using the vulnerability assessment tool and the exploit module. We can confirmed a very high similarity J of 0.993. Although content derived from container characteristics must be excluded, containers have a very high vulnerability reproduction performance, confirming that container-based virtualization can be fully used in the cyber range.

These contents can be used as a benchmark for developing scenarios and conducting exercises for container-based cyber ranges. In the future, we will experiment with more situations and conduct more detailed research, including the ability to carry out attack and defense scenarios. We will also promote research and studies to increase the container-based cyber range's effectiveness, such as examining the effectiveness of education and refining exercise scenarios based on behavioral analysis, to broaden the base of security personnel training.

ACKNOWLEDGEMENTS

This work is supported in part by the Telecommunication Advancement Foundation.

REFERENCES

Ameen, R. Y. and Hamo, A. Y. (2013). Survey of server virtualization.

Costa, G., Russo, E., and Armando, A. (2020). Automating the generation of cyber range virtual scenarios with vsdl.

GreenboneNetworks (2020). Openvas - open vulnerability assessment scanner. https://www.openvas.org.

Irvine, Cynthia, E., Thompson, Michael, F., and Khosalim, J. (2017). Labtainers: a framework for parameterized cybersecurity labs using containers.

Kali.org (2020). Official kali linux docker images. https://www.kali.org/docs/containers/official-kalilinux-docker-images/.

Li, Z., Kihl, M., Lu, Q., and Andersson, J. A. (2017). Performance overhead comparison between hypervisor and container based virtualization. In *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, pages 955–962.

Maki, N., Nakata, R., Toyoda, S., Kasai, Y., Shin, S., and Seto, Y. (2020). An effective cybersecurity exercises platform cyexec and its training contents. In *2020 International Conference on Advances in Education and Information Technology(AEIT'20)*.

Makino, Y. and Klyuev, V. (2015). Evaluation of web vulnerability scanners. In *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, volume 1, pages 399–402.

MITRE (2020a). Common vulnerabilities and exposures. https://cve.mitre.org/.

MITRE (2020b). Common weakness enumeration. https://cwe.mitre.org/index.html.

Nmap.org (2020). Port scanning techniques. https://nmap.org/book/man-port-scanning-techniques.html.

Preeth E N, Mulerickal, F. J. P., Paul, B., and Sastri, Y. (2015). Evaluation of docker containers based on hardware utilization. In *2015 International Conference on Control Communication Computing India (ICCC)*, pages 697–700.

Rapid7 (2020). Metasploitable2. https://docs.rapid7.com/metasploit/metasploitable-2/.

Razvan, B., Cuong, P., Dat, T., Ken-ichi, C., Yasuo, T., and Yoichi, S. (2017). Cytrone: An integrated cybersecurity training framework. In *Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP 2017): 157-166*.

Stout, W. M., Urias, V., Van Leeuwen, B. P., and Lin, H. W. (2018). An effective cybersecurity exercises platform cyexec and its training contents. In *Proposed for presentation at the IEEE International Carnahan Conference on Security Technology (ICCST)*.

VMWare, I. (2019). Vmware digital learning platform - cyber range solution. https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/solutions/industry/vmware-cyber range-datasheet.pdf.