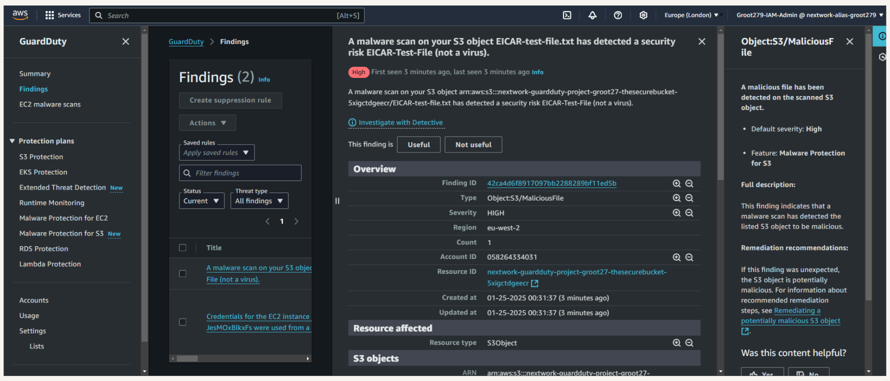
NextWork.org

Threat Detection with GuardDuty

TA tahirgroot@gmail.com



A malware scan on your S3 object EICAR-test-file.txt has detected a security risk EICAR-Test-File (not a virus).

A malware scan on your S3 objec File (not a virus).

Credentials for the EC2 instance JesMOxBlkxFs were used from a

High First seen 3 minutes ago, last seen 3 minutes ago Info

A malware scan on your S3 object arn:aws:s3:::nextwork-guardduty-project-groot27-thesecurebucket-5xigctdgeecr/EICAR-test-file.txt has detected a security risk EICAR-Test-File (not a virus).

This finding is Useful Not useful

Overview

Finding ID	42ca4d6f8917097bb2288289bf11ed5b
Type	Object:S3/MaliciousFile
Severity	HIGH
Region	eu-west-2
Count	1
Account ID	058264334031
Resource ID	nextwork-guardduty-project-groot27-thesecurebucket-5xigctdgeecr
Created at	01-25-2025 00:31:37 (3 minutes ago)
Updated at	01-25-2025 00:31:37 (3 minutes ago)

Resource affected

Resource type	S3Object

S3 objects

ARN	arn:aws:s3:::nextwork-guardduty-project-groot27-

Object:S3/MaliciousFile

A malicious file has been detected on the scanned S3 object.

- Default severity: **High**
- Feature: **Malware Protection for S3**

Full description:

This finding indicates that a malware scan has detected the listed S3 object to be malicious.

Remediation recommendations:

If this finding was unexpected, the S3 object is potentially malicious. For information about recommended remediation steps, see Remediating a potentially malicious S3 object.

Introducing Today's Project!

Tools and concepts

The services I used were GuardDuty, CloudFormation, s3, cloudshell Key concepts I learnt include SQL + command injection, using linux commands like wget, cat and jq and malware protection.

Project reflection

This project took me approximately 2 hours The most challenging part was getting access to the victim AWS environment using a profile. It was most rewarding to reveal the victim's protected file as the attacker.

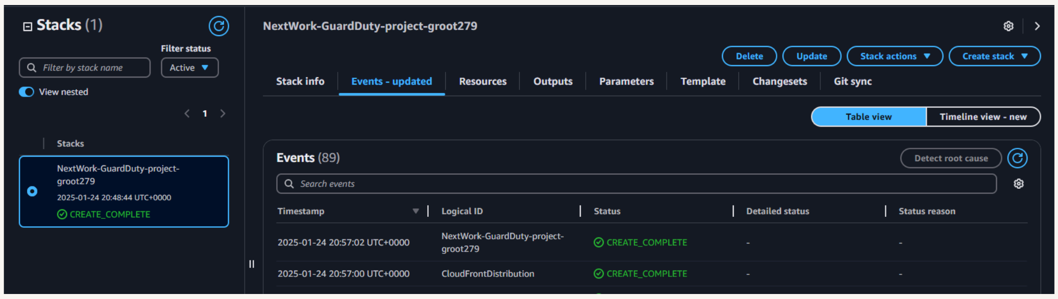
We did this project today to learn about threat detection and GuardDuty + techniques like sql/command injection. This project certainly met my expectaions and was a good challenge!

Project Setup

To set up for this project, I deployed a CloudFormation template that launches an insecure web app (OWASP Juice shop). The three main components are the web app infrastructure, an s3 bucket and GuardDuty protecting our environment

The web app deployed is called OWASP Juice Shop. To practice my GuardDuty skills, I will attack the JUice Shop, and the visit the GuardDuty console to detect and analyze its findings - does it pick up on our attacks to our web app?

GuardDuty is an AI-powered threat detection service, which means it is designed to help me find and security attacks or vulnerabilities that affect my AWS resources/environment. Once it detects something unusual, it's up to us to investigate.



NextWork-GuardDuty-project-groot279
2025-01-24 20:48:44 UTC+0000
CREATE_COMPLETE

NextWork-GuardDuty-project-groot279

Events (89)

Timestamp	Logical ID	Status	Detailed status	Status reason
2025-01-24 20:57:02 UTC+0000	NextWork-GuardDuty-project-groot279	CREATE_COMPLETE	-	-
2025-01-24 20:57:00 UTC+0000	CloudFrontDistribution	CREATE_COMPLETE	-	-

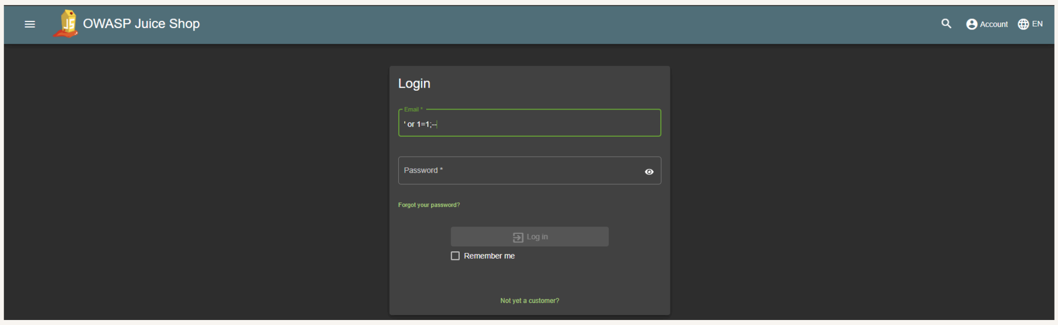
TA

tahirgroot@gmail.com
NextWork Student

SQL Injection

The first attack I performed on the web app is SQL injection, which means injecting malicious sql code that manipulates a results from my web app. SQL injection is a security risk because it can let attackers bypass logins, delete/edit data

My SQL injection attack involved entering the code ' or 1=1;- into the email field of the web app's login page. This means the login query will always evaluate to true(i.e our database is manipulated into telling my web app this login exists).



Email *
' or 1=1;--
Password *

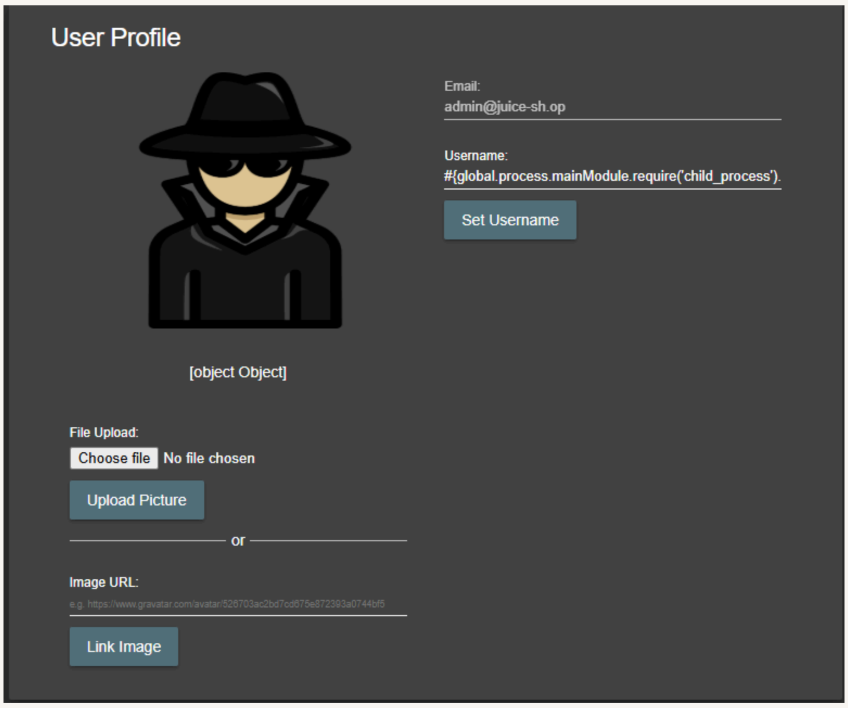
TA

tahirgroot@gmail.com
NextWork Student

Command Injection

Next, I used command injection, which manipulates the web app's web server to run code that has been enetered e.g. in a form. The Juice Shop web app is vulnerable to this because it does not sanitize user inputs i.e. does not block scripts.

To run command injection, I entered Javascript code in the username field of the web app's admin console. This script will tell my webserver to expose the server's IAM credentials and save them in a publicly accessibe JSON file.


User Profile

Email:
admin@juice-sh.op

Username:
`#{global.process.mainModule.require('child_process').`

[object Object]

File Upload:

No file chosen

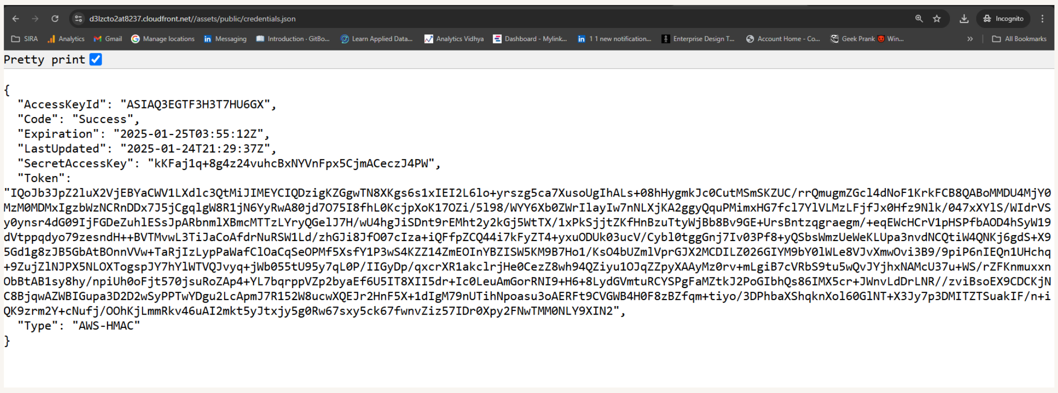
Image URL:
e.g. https://www.gravatar.com/avatar/526703ac2bd7cd675e872393a0744bf5

TA

tahirgroot@gmail.com
NextWork Student

Attack Verification

To verify the attack's success, I visited the publicly exposed credentials file. This page showed us access keys that represents my EC2 instances's access to the dev aws environment. I can use those keys to get the same level of access



Pretty print ✓

```
{
  "AccessKeyId": "ASIAQ3EGTF3H3T7HU6GX",
  "Code": "Success",
  "Expiration": "2025-01-25T03:55:12Z",
  "LastUpdated": "2025-01-24T21:29:37Z",
  "SecretAccessKey": "kKFaj1q+8g4z24vuhcBxNYVnFpx5CjmACeczJ4PW",
  "Token":
"IQoJb3JpZ2luX2VjEBYaCWV1LXdlc3QtMiJIMEYCIQDzigKZGgwTN8XKgs6s1xIEI2L6lo+yrszg5ca7XusoUgIhALs+08hHygmkJc0CutMSmSKZUC/rrQmugmZGcl4dNoF1KrkFCB8QABoMMDU4MjY0
MzM0MDMxIgzbWzNCRnDDx7J5jCgqlgW8R1jN6YyRwA80jd7O75I8fhL0KcjpXoK17OZi/5l98/WYY6Xb0ZWrIlayIw7nNLXjKA2ggyQquPMimxHG7fcl7YlVLMzLFjfJx0Hfz9Nlk/047xXYlS/WIdrVS
y0ynsr4dG09IjFGDeZuhlESsJpARbnmlXBmcMTTzLYryQGelJ7H/wU4hgJiSDnt9rEMht2y2kGj5WtTX/1xPkSjjtZKfHnBzuTtyWjBb8Bv9GE+UrsBntzqgraegm/+eqEWcHCrV1pHSPfbAOD4hSyW19
dVtppqdyo79zesndH++BVTMvwL3TiJaCoAfdrNuRSW1Ld/zhGJi8JfO07cIza+iQFfpZCQ44i7kFyZT4+yxuODUk03ucV/Cybl0tggGnj7Iv03Pf8+yQSbsWmzUeWeKLUpa3nvdNCQtiW4QNKj6gdS+X9
5Gd1g8zJB5GbAtBOnnVVw+TaRjIzLypPaWafClOaCqSeOPMf5XsfY1P3wS4KZZ14ZmEOInYBZISW5KM9B7Ho1/KsO4bUZmlVprGJX2MCDILZ026GIYM9bY0lWLe8VJvXmwOvi3B9/9piP6nIEQn1UHchq
+9ZujZlNJPX5NLOXTogspJY7hYlWTVQJvyq+jWb055tU95y7qL0P/IIGyDp/qxcrXR1akclrjHe0CezZ8wh94QZiyu1OJqZZpyXAAyMz0rv+mLgiB7cVRbS9tu5wQvJYjhxNAMcU37u+WS/rZFKnmuxxn
ObBtAB1sy8hy/npiUh0oFjt570jsuRoZAp4+YL7bqrppVZp2byaEf6U5IT8XII5dr+Ic0LeuAmGorRNI9+H6+8LydGVmtuRCYSPgFaMZtkJ2PoGIbhQs86IMX5cr+JWnvLdDrLNR//zviBsoEX9CDCKjN
C8BjqwAZWBIGupa3D2D2wSyPPTwYDgu2LcApmJ7R152W8ucwXQEJr2HnF5X+1dIgM79nUTihNpoasu3oAERFt9CVGWB4H0F8zBZfqm+tiyo/3DPhbaXShqknXol60GlNT+X3Jy7p3DMITZTSuakIF/n+i
QK9zrm2Y+cNufj/OOhKjLmmRkv46uAI2mkt5yJtxjy5g0Rw67sxy5ck67fwnvZiz57IDr0Xpy2FNwTMM0NLY9XIN2",
  "Type": "AWS-HMAC"
}
```

TA

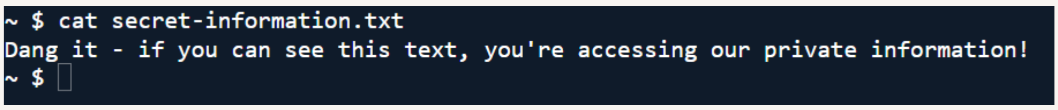
tahirgroot@gmail.com
NextWork Student

Using CloudShell for Advanced Attacks

The attack continues in cloudshell, because this is a command-line tool I can use to run AWS commands that uses the credentials I've stolen. Cloudshell will be my medium for doing suspicious things like stealing data from an S3 bucket.

In CloudShell, I used wget to download the exposed credentials file into my cloudshell environment. Next, I ran a command using cat and jq to read the downloaded file and format it nicely so the credentials (in JSON) are easy to understand.

I then set up a profile using all of the stolen credentials. I had to create a new profile because the hacker doesn't inherently have access to the victim's AWS environment. I will need to use the profile to switch permission settings.



```
~ $ cat secret-information.txt
Dang it - if you can see this text, you're accessing our private information!
~ $ 
```

TA

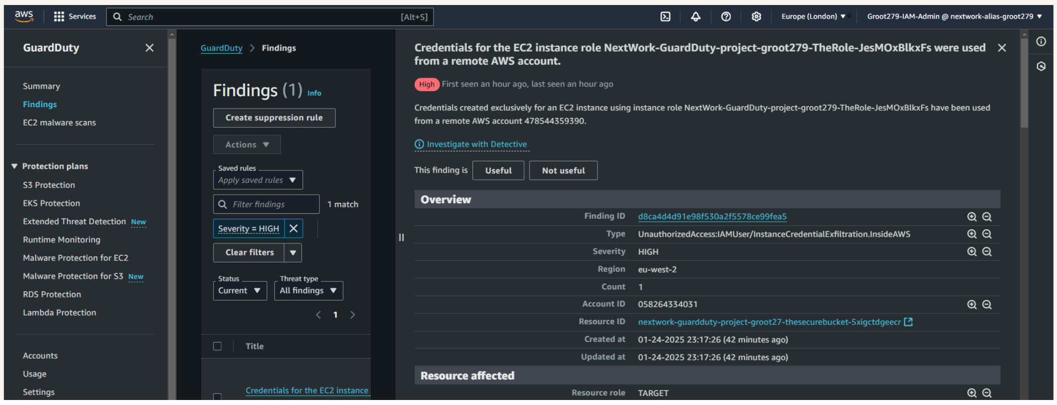
tahirgroot@gmail.com
NextWork Student

GuardDuty's Findings

After performing the attack, GuardDuty reported a finding within 15 minutes Findings are notifications from Guardduty that something suspicious has happend, and they give you details about the who/what/when of the attack.

GuardDuty's finding was called UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS which means credentials belonging to an EC2 instance were being used in another account. Anomaly detection was used because this was unusual behaviour

GuardDuty's detailed finding reported that an S3 bucket was affected, the action that was done using the stolen credentials (GetObject); and the EC2 instance whos credentials were leaked. The IP address + location of the actor was also available.



Credentials for the EC2 instance role NextWork-GuardDuty-project-groot279-TheRole-JesMOxBlkxFs were used from a remote AWS account.

High First seen an hour ago, last seen an hour ago

Credentials created exclusively for an EC2 instance using instance role NextWork-GuardDuty-project-groot279-TheRole-JesMOxBlkxFs have been used from a remote AWS account 478544359390.

Overview

Finding ID	d8ca4d4d91e98f530a2f5578ce99fea5
Type	UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS
Severity	HIGH
Region	eu-west-2
Count	1
Account ID	058264334031
Resource ID	nextwork-guardduty-project-groot27-thesecurebucket-5xigctdgeecr
Created at	01-24-2025 23:17:26 (42 minutes ago)
Updated at	01-24-2025 23:17:26 (42 minutes ago)

Resource affected

Resource role	TARGET

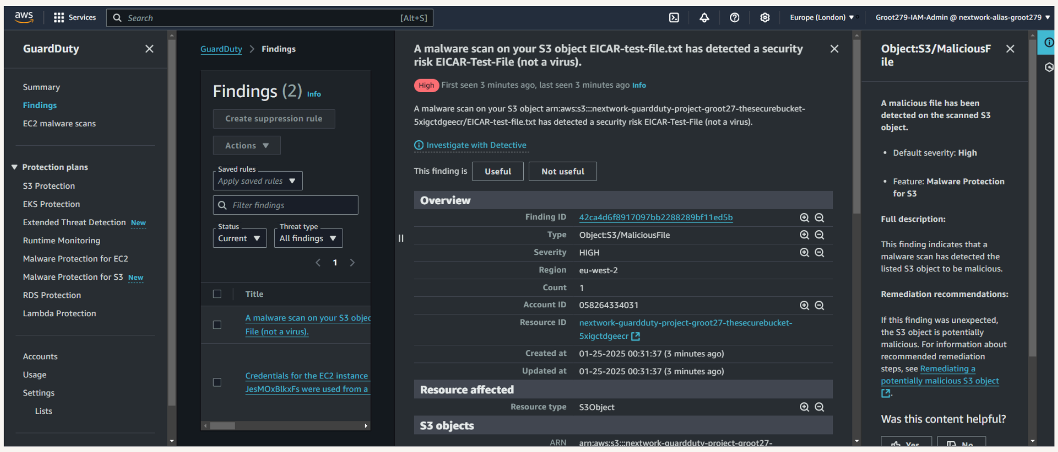
TA
tahirgroot@gmail.com
NextWork Student

Extra: Malware Protection

For my project extension, I enabled Malware protection for s3. Malware is file that contains threats e.g opening the file will cause a data breach or a deleteion of resoruces.

To test Malware Protection, I uploaded an EICAR test file into a protected bucket. The uploaded file won't actually cause damage because the test file is only designed to alert antivirus software.

Once I uploaded the file, GuardDuty instantly triggered a finding called Object:S3/MaliciousFile.This verified that GuardDuty could successfully detect malware. it also mentioned that the threat type is EICAR-Test-File (which means not a virus)



A malware scan on your S3 objec
File (not a virus).

Credentials for the EC2 instance
JesMOxBlkxFs were used from a

A malware scan on your S3 object EICAR-test-file.txt has detected a security risk EICAR-Test-File (not a virus).

High First seen 3 minutes ago, last seen 3 minutes ago Info

A malware scan on your S3 object arn:aws:s3:::nextwork-guardduty-project-groot27-thesecurebucket-5xigctdgeecr/EICAR-test-file.txt has detected a security risk EICAR-Test-File (not a virus).

Investigate with Detective

This finding is Useful Not useful

Overview

Finding ID	42ca4d6f8917097bb2288289bf11ed5b
Type	Object:S3/MaliciousFile
Severity	HIGH
Region	eu-west-2
Count	1
Account ID	058264334031
Resource ID	nextwork-guardduty-project-groot27-thesecurebucket-5xigctdgeecr
Created at	01-25-2025 00:31:37 (3 minutes ago)
Updated at	01-25-2025 00:31:37 (3 minutes ago)

Resource affected

Resource type	S3Object

S3 objects

ARN	arn:aws:s3:::nextwork-guardduty-project-groot27-

Object:S3/MaliciousFile

A malicious file has been detected on the scanned S3 object.

- Default severity: **High**
- Feature: **Malware Protection for S3**

Full description:

This finding indicates that a malware scan has detected the listed S3 object to be malicious.

Remediation recommendations:

If this finding was unexpected, the S3 object is potentially malicious. For information about recommended remediation steps, see Remediating a potentially malicious S3 object.

Was this content helpful?