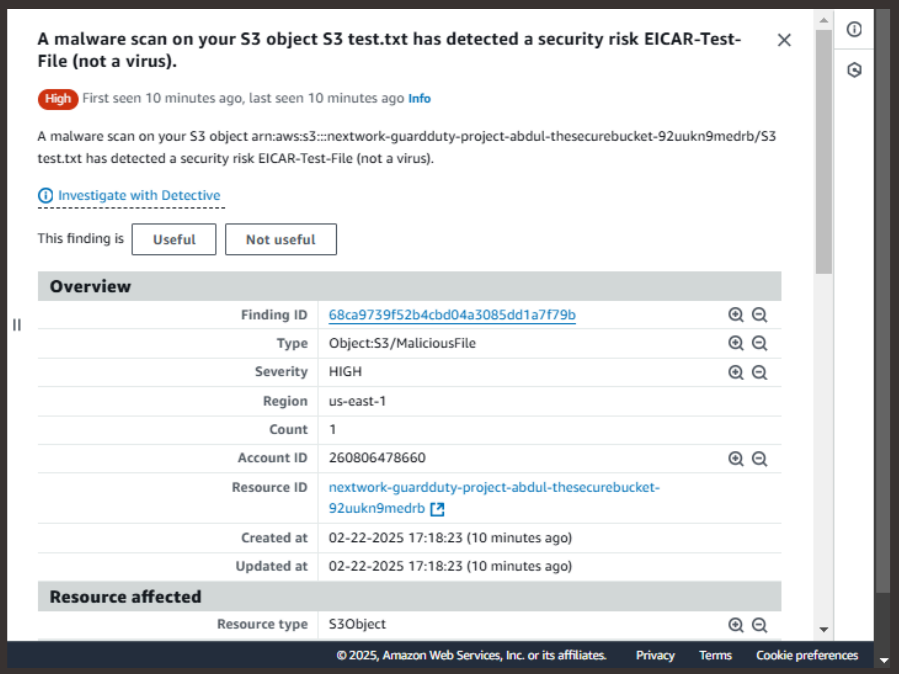
nextwork.org

Threat Detection with GuardDuty

Abdulrahman Sulaiman



Abdulrahman Sulaiman
NextWork Student

Introducing Today's Project!

Tools and concepts

The services I used were, CloudFormation, S3, EC2 instance, cloudshell. GuardDuty Key concepts I learnt include, detecting suspicious activities such as unauthorized access, credential compromise, Analyzes network traffic for potential threats.

Project reflection

This project took me approximately 3 hours including documentation. The most challenging part was analyzing the threats detected. It was most rewarding to know that GuardDutu alerts generated detects potential threats.

I worked on Amazon GuardDuty today to improve my understanding of AWS security monitoring. My goal was to detect potential threats, analyze security findings, and explore how GuardDuty integrates with other AWS services like CloudTrail and flow logs.

Abdulrahman Sulaiman
NextWork Student

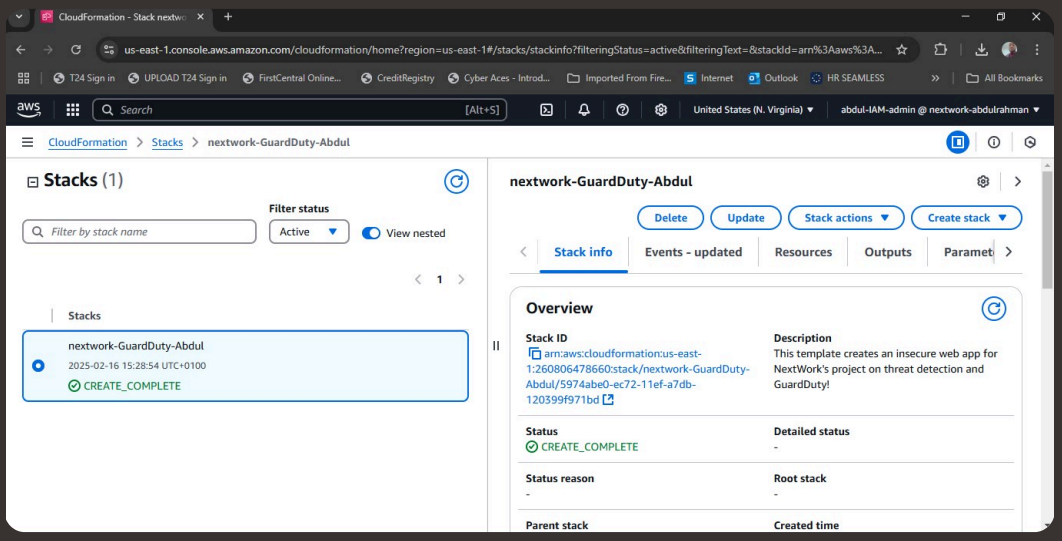
NextWork.org

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 web app infrastructure, S3 bucket and GuardDuty.

The web app deployed is called OWASP juice shop. To practice my GuardDuty skills, I will attack the juice shop and visit GuardDuty cosole to detect and analyze my findings.

GuardDuty is an AI threat detection service that is designed to analyze activities and alerts when there is something suspicious found. It detects and allows investigation.

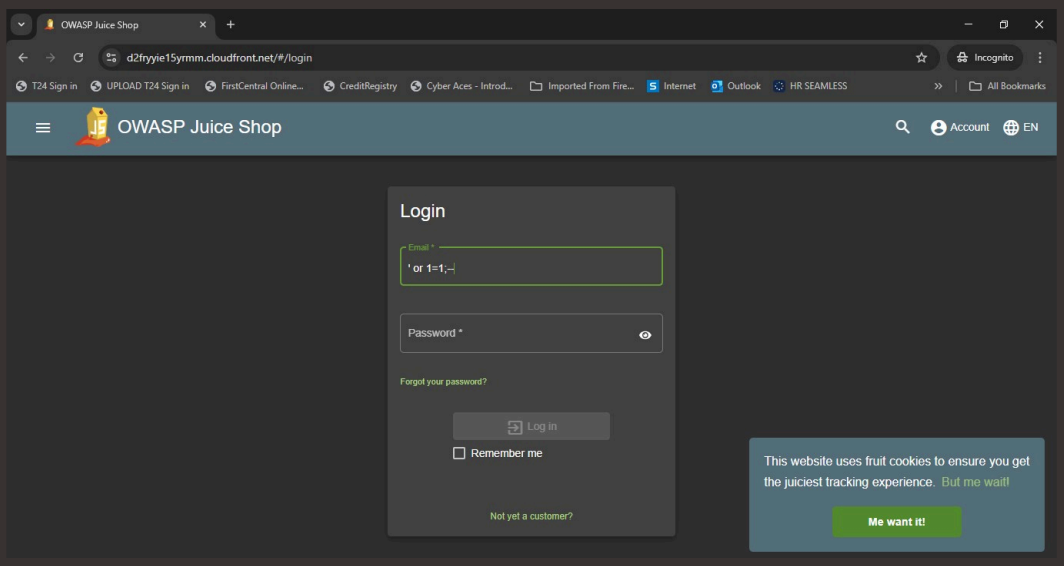


Abdulrahman Sulaiman
NextWork Student

SQL Injection

The first attack I performed on the web app is SQL injection, which means I can bye pass authentication and exploit data.SQL injection is a security risk because is a web vulnerability that an attacker can use to insert malicious codes in a database.

My SQL injection attack involved entering the code "' or 1=1;--" into the email field. This means the login query will always evaluate to true. (That is, the data base is manipulated to always allow login to exist)

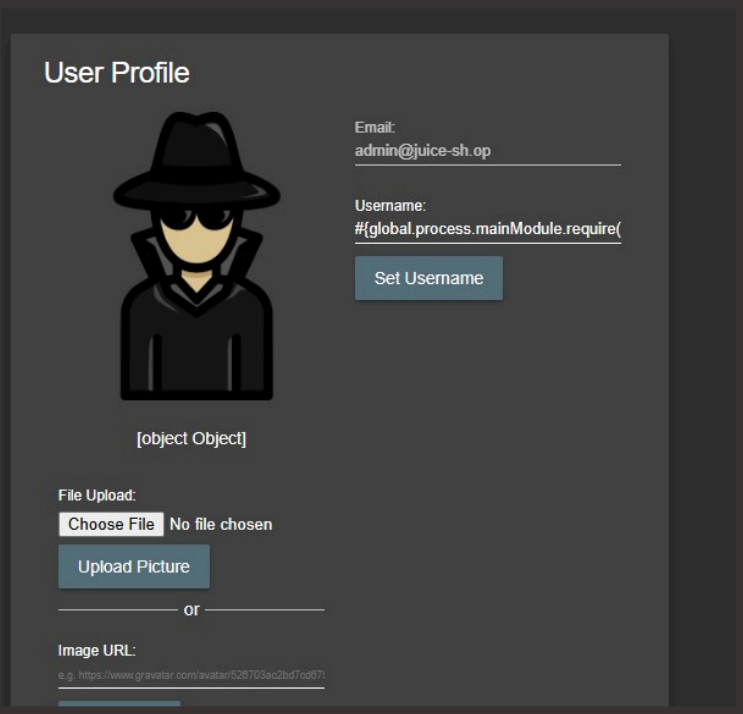


Abdulrahman Sulaiman
NextWork Student

Command Injection

Next, I use Command Injection, which manipulates the web app's web server to run code that has been entered. 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.

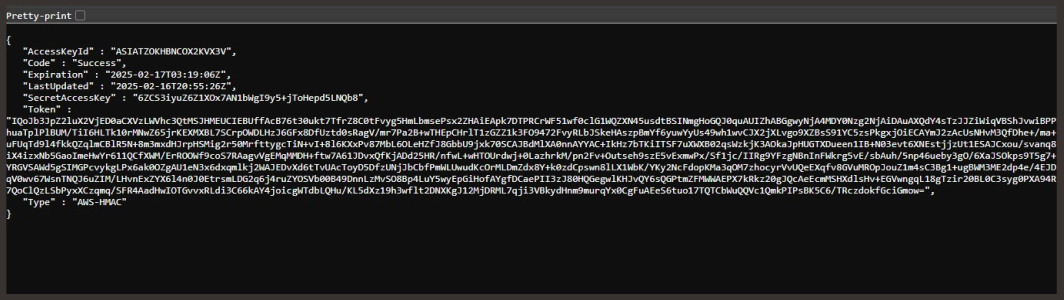
To run command injection, I (as the attacker) inputted code into the username field. The script will access sensitive information, manipulate services, or even take over the server.



Abdulrahman Sulaiman
NextWork Student

Attack Verification

To verify the attack's success, I visited the publicy exposed credenrtials file (i.e. credentials.json). This page showed me access keys that represents my EC2 instance's access to my AWS environement. I can use those keys to get same level of access



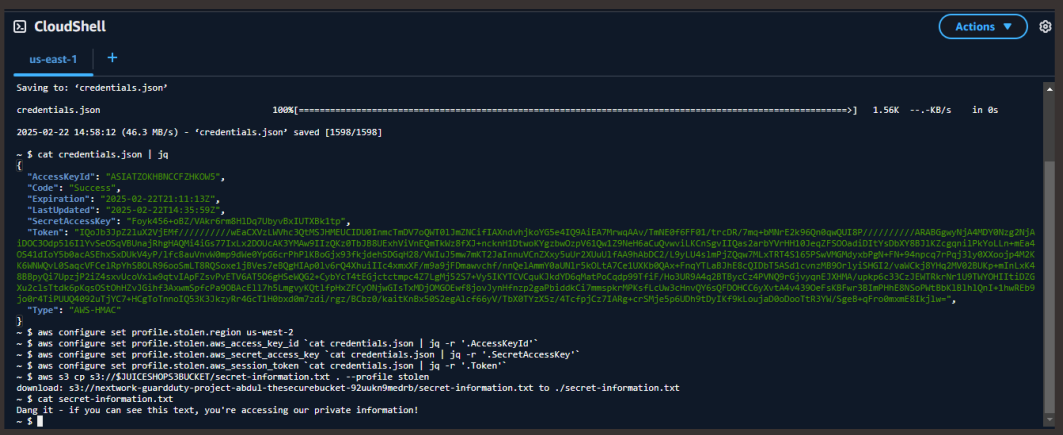
Abdulrahman Sulaiman
NextWork Student

Using CloudShell for Advanced Attacks

The attack continues in CloudShell, because, in this step, I'm trying to get access to private data using stolen credentials. I'll use CloudShell to run commands that a hacker would run to steal data.

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

I then set up a profile, called stolen to use and store all the stolen credentials I have extracted from my attack on the web application.



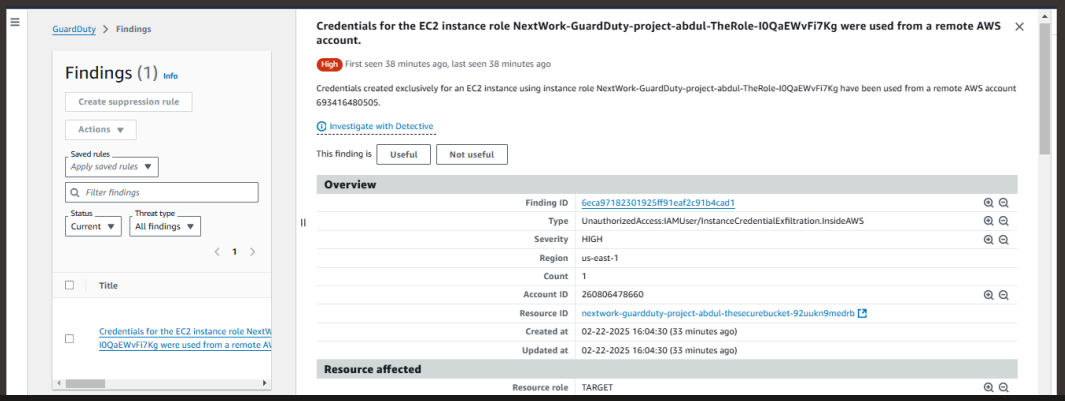
Abdulrahman Sulaiman
NextWork Student

GuardDuty's Findings

After performing the attack, GuardDuty reported a finding within 15 minutes. Findings are notifications from GuarDuty that something susicious has been detected, and they give you details about who, when, where of the attack.

GuardDuty's finding was called "Credentials created exclusively for an EC2 instance using instance role NextWork-GuardDuty-project-abdul-TheRole-I0QaE..." which means credentials belonging to my EC2 has been used in another account. Anomaly detected!

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



Abdulrahman Sulaiman
NextWork Student

Extra: Malware Protection

For my project extension, I enabled malware protection for S3. Malware is Malware is a type of software designed to harm computers. It can damage computers, steal data, or disrupt operations.

To test Malware Protection, I uploaded a .txt malware file. The uploaded file won't actually cause damage because it was designed for this project and it is not an acutual malware.

Once I uploaded the file, GuardDuty instantly triggered the file and detected a security risk. This verified that the action triggered to protect files in S3 buckets has made GuarDuty to actively detect malicious file when uploaded.