Zion Spearman

4/30/2025

CBR 585

Thomas Dodds

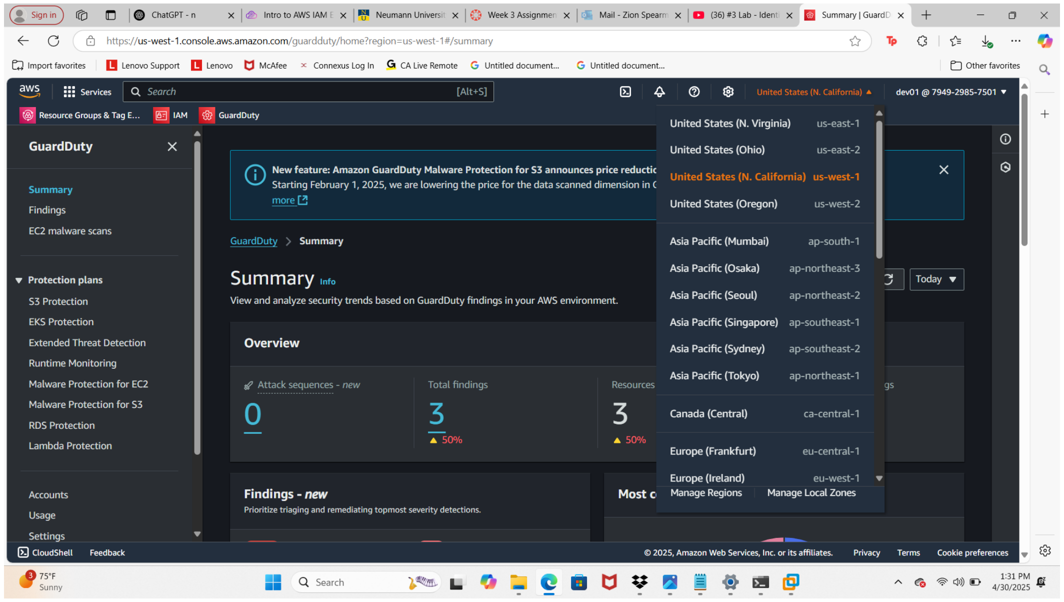
AWS IAM Enumeration Lab Report

Scenario

You are a security consultant hired by the global logistics company, Huge Logistics. Following suspicious activity, you are tasked with enumerating the IAM user 'dev01' and mapping out any potentially compromised resources. Your mission is to enumerate and evaluate IAM roles, policies, and permissions to understand the potential blast radius of a compromised account.

Lab Setup and AWS CLI Configuration

AWS CLI was configured using the provided access key and secret key. The default region used was 'us-west-1' and the output format was set to 'json'. The following screenshots demonstrate the configuration steps:

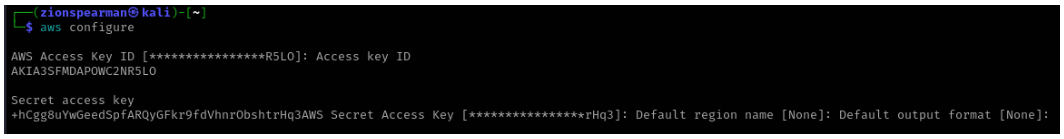

https://us-west-1.console.aws.amazon.com/guardduty/home?region=us-west-1#/summary

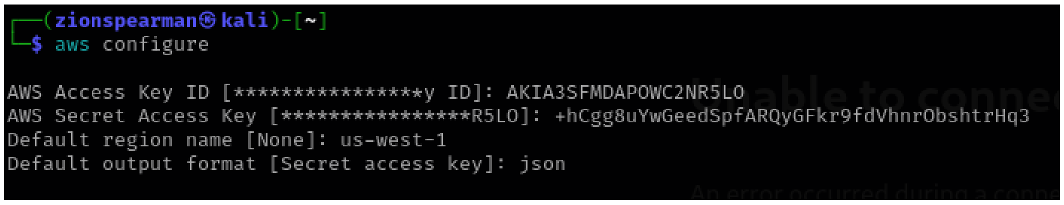
New feature: Amazon GuardDuty Malware Protection for S3 announces price reductic
Starting February 1, 2025, we are lowering the price for the data scanned dimension in (
more

Summary
View and analyze security trends based on GuardDuty findings in your AWS environment.

Findings - new
Prioritize triaging and remediating topmost severity detections.

Services
Search
[Alt+S]
United States (N. California)
dev01 @ 7949-2985-7501
Resource Groups & Tag E...
IAM
GuardDuty
GuardDuty
Summary
Findings
EC2 malware scans
Protection plans
S3 Protection
EKS Protection
Extended Threat Detection
Runtime Monitoring
Malware Protection for EC2
Malware Protection for S3
RDS Protection
Lambda Protection
Accounts
Usage
Settings
GuardDuty > Summary
Today
Overview
Attack sequences - new
0
Total findings
3
50%
Resources
3
50%
Most c
United States (N. Virginia) us-east-1
United States (Ohio) us-east-2
United States (N. California) us-west-1
United States (Oregon) us-west-2
Asia Pacific (Mumbai) ap-south-1
Asia Pacific (Osaka) ap-northeast-3
Asia Pacific (Seoul) ap-northeast-2
Asia Pacific (Singapore) ap-southeast-1
Asia Pacific (Sydney) ap-southeast-2
Asia Pacific (Tokyo) ap-northeast-1
Canada (Central) ca-central-1
Europe (Frankfurt) eu-central-1
Europe (Ireland) eu-west-1
Manage Regions
Manage Local Zones
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
75°F
Sunny
1:31 PM
4/30/2025

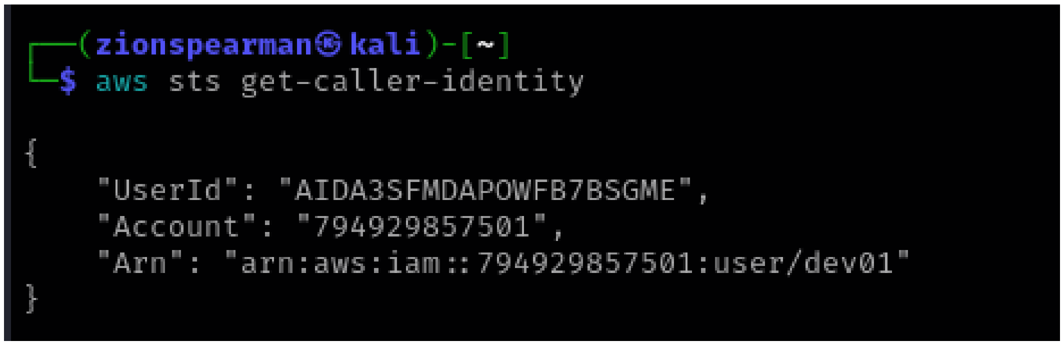

```
┌──(zionspearman㉿kali)-[~]
└─$ aws configure

AWS Access Key ID [****************R5LO]: Access key ID
AKIA3SFMDAPOWC2NR5LO

Secret access key
+hCgg8uYwGeedSpfARQyGFkr9fdVhnrObshtrHq3AWS Secret Access Key [****************rHq3]: Default region name [None]: Default output format [None]:
```



```
┌──(zionspearman㉿kali)-[~]
└─$ aws configure

AWS Access Key ID [****************y ID]: AKIA3SFMDAPOWC2NR5LO
AWS Secret Access Key [****************R5LO]: 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
Default region name [None]: us-west-1
Default output format [Secret access key]: json
```

Identity Verification

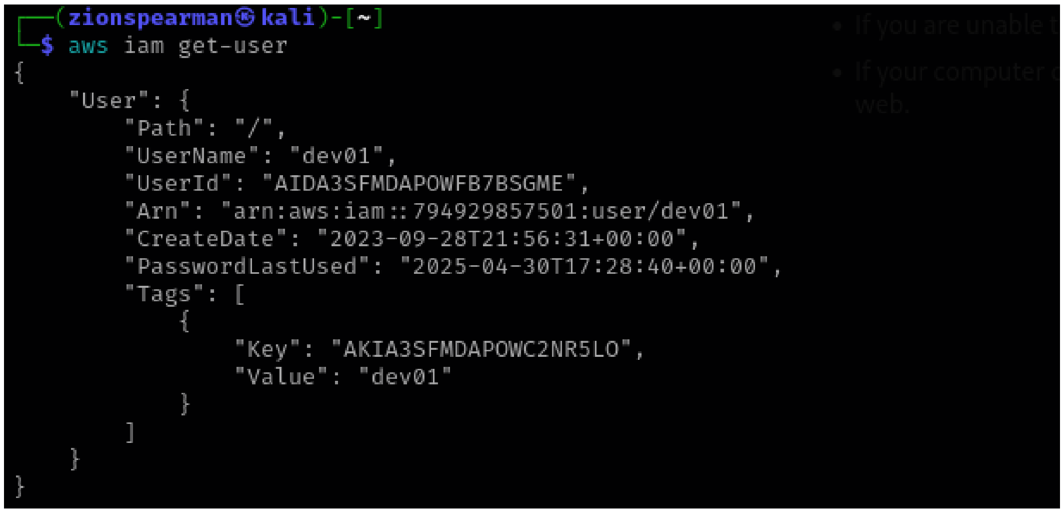
The identity of the user was verified using the AWS CLI command aws sts get-caller-identity, which confirmed the IAM user in context was 'dev01' under the AWS account 794929857501. This step is critical to ensure that all following actions are accurately mapped to the correct user context.



```
┌──(zionspearman㉿kali)-[~]
└─$ aws sts get-caller-identity

{
    "UserId": "AIDA3SFMDAPOWFB7BSGME",
    "Account": "794929857501",
    "Arn": "arn:aws:iam::794929857501:user/dev01"
}
```

IAM User Details

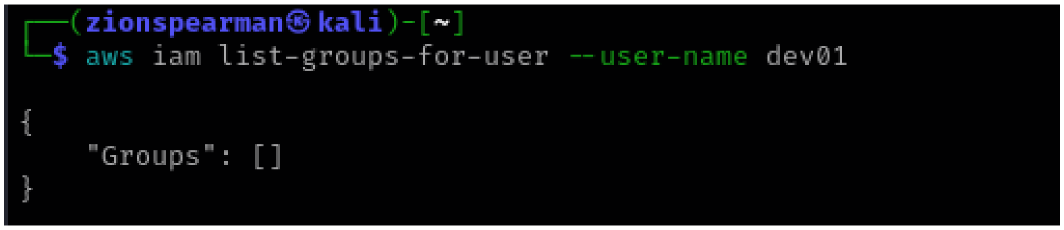
We used the command aws iam get-user to get details about the IAM user 'dev01'. It showed when the user was created and gave basic info like the user's name and ID.



```
┌──(zionspearman㉿kali)-[~]
└─$ aws iam get-user
{
    "User": {
        "Path": "/",
        "UserName": "dev01",
        "UserId": "AIDA3SFMDAPOWFB7BSGME",
        "Arn": "arn:aws:iam::794929857501:user/dev01",
        "CreateDate": "2023-09-28T21:56:31+00:00",
        "PasswordLastUsed": "2025-04-30T17:28:40+00:00",
        "Tags": [
            {
                "Key": "AKIA3SFMDAPOWC2NR5LO",
                "Value": "dev01"
            }
        ]
    }
}
```

Group Membership Check

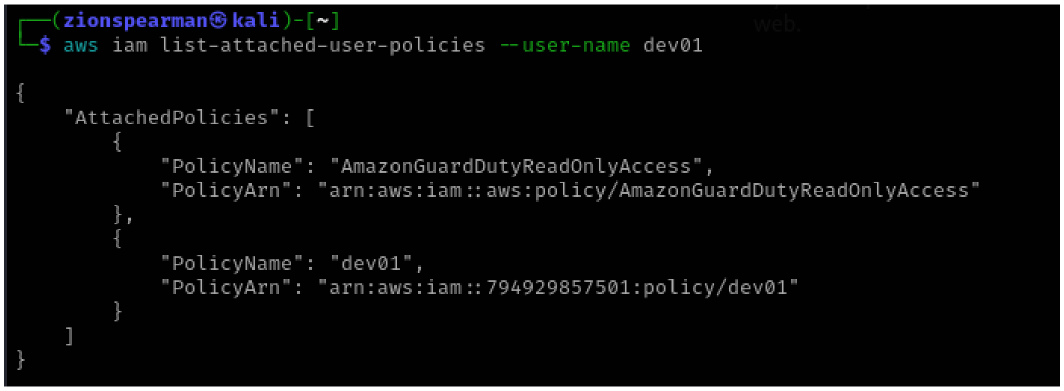
We ran the command aws iam list-groups-for-user --user-name dev01 to check if the user was part of any groups. The result showed that dev01 isn't a member of any groups.



```
┌──(zionspearman㉿kali)-[~]
└─$ aws iam list-groups-for-user --user-name dev01

{
    "Groups": []
}
```

Attached User Policies

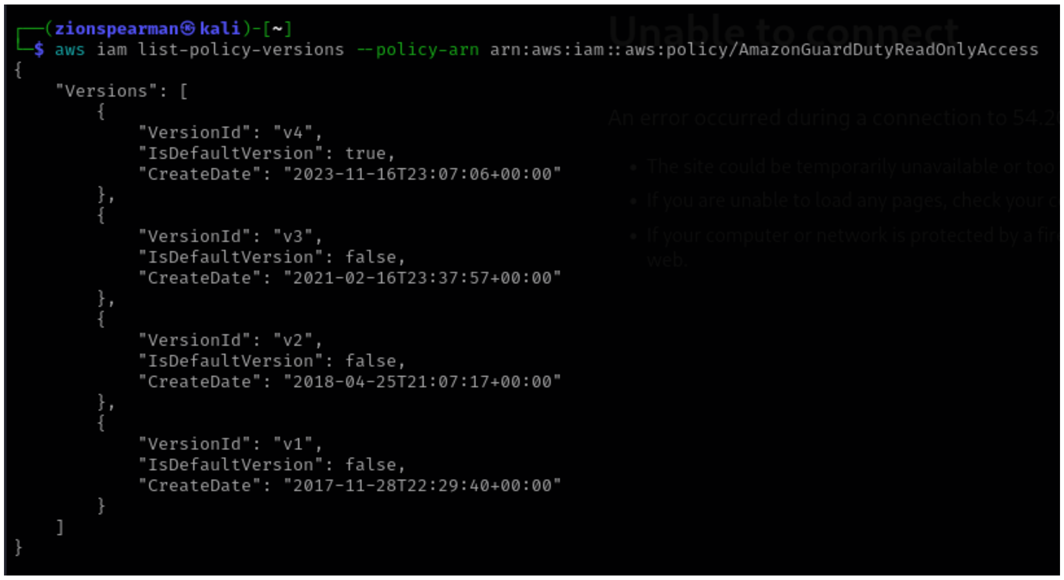
We used the command aws iam list-attached-user-policies --user-name dev01 to see which policies were attached to the user. It showed that dev01 had two policies: one Amazon-managed policy called AmazonGuardDutyReadOnlyAccess and a custom policy named dev01.



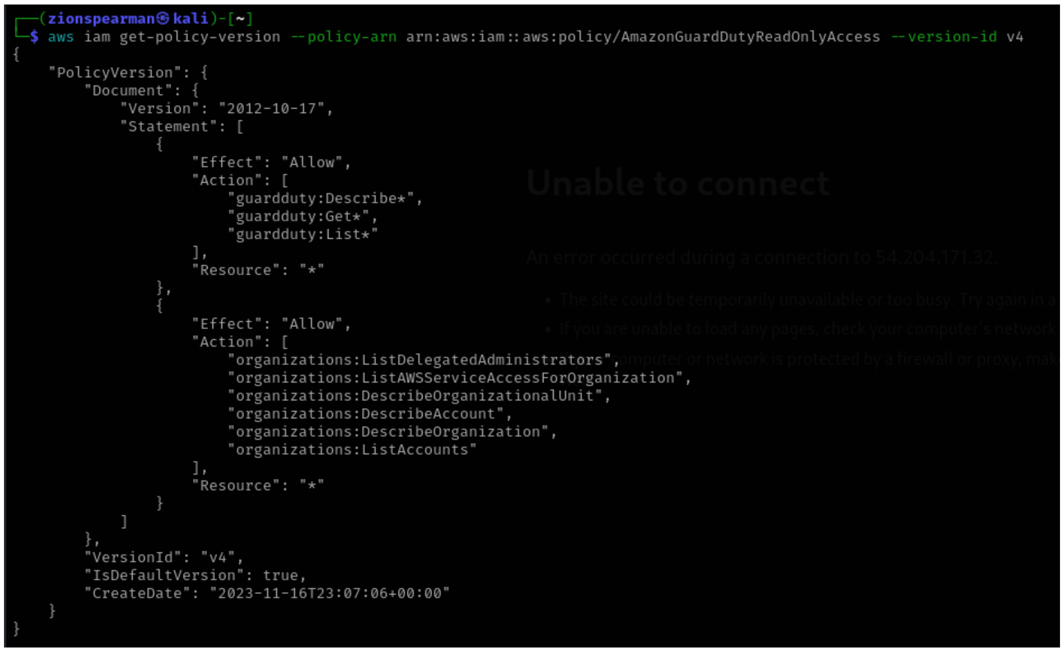
```
┌──(zionspearman㉿kali)-[~]
└─$ aws iam list-attached-user-policies --user-name dev01

{
    "AttachedPolicies": [
        {
            "PolicyName": "AmazonGuardDutyReadOnlyAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonGuardDutyReadOnlyAccess"
        },
        {
            "PolicyName": "dev01",
            "PolicyArn": "arn:aws:iam::794929857501:policy/dev01"
        }
    ]
}
```

GuardDuty Policy Permissions

We explored the managed policy AmazonGuardDutyReadOnlyAccess to see what permissions it included. The policy allows the user to view (but not change) GuardDuty findings, settings, and related AWS Organization details.



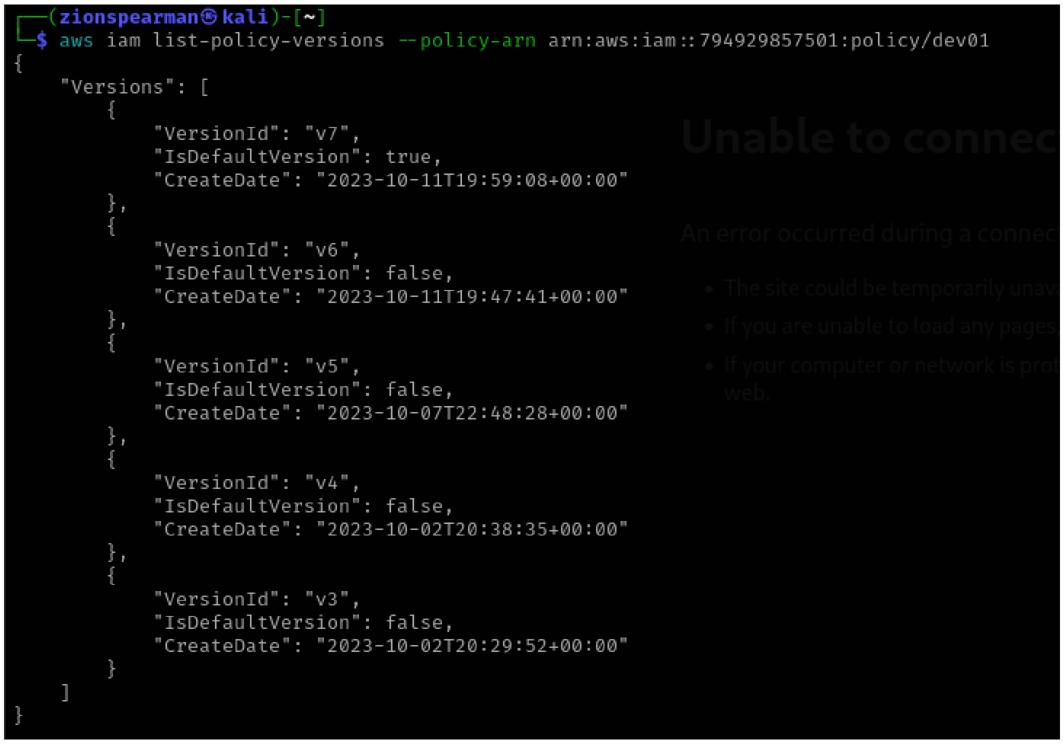
```
┌──(zionspearman㉿kali)-[~]
└─$ aws iam list-policy-versions --policy-arn arn:aws:iam::aws:policy/AmazonGuardDutyReadOnlyAccess
{
    "Versions": [
        {
            "VersionId": "v4",
            "IsDefaultVersion": true,
            "CreateDate": "2023-11-16T23:07:06+00:00"
        },
        {
            "VersionId": "v3",
            "IsDefaultVersion": false,
            "CreateDate": "2021-02-16T23:37:57+00:00"
        },
        {
            "VersionId": "v2",
            "IsDefaultVersion": false,
            "CreateDate": "2018-04-25T21:07:17+00:00"
        },
        {
            "VersionId": "v1",
            "IsDefaultVersion": false,
            "CreateDate": "2017-11-28T22:29:40+00:00"
        }
    ]
}
```



```
┌──(zionspearman㉿kali)-[~]
└─$ aws iam get-policy-version --policy-arn arn:aws:iam::aws:policy/AmazonGuardDutyReadOnlyAccess --version-id v4
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": [
                        "guardduty:Describe*",
                        "guardduty:Get*",
                        "guardduty:List*"
                    ],
                    "Resource": "*"
                },
                {
                    "Effect": "Allow",
                    "Action": [
                        "organizations:ListDelegatedAdministrators",
                        "organizations:ListAWSServiceAccessForOrganization",
                        "organizations:DescribeOrganizationalUnit",
                        "organizations:DescribeAccount",
                        "organizations:DescribeOrganization",
                        "organizations:ListAccounts"
                    ],
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v4",
        "IsDefaultVersion": true,
        "CreateDate": "2023-11-16T23:07:06+00:00"
    }
}
```

Custom Policy Version Listing

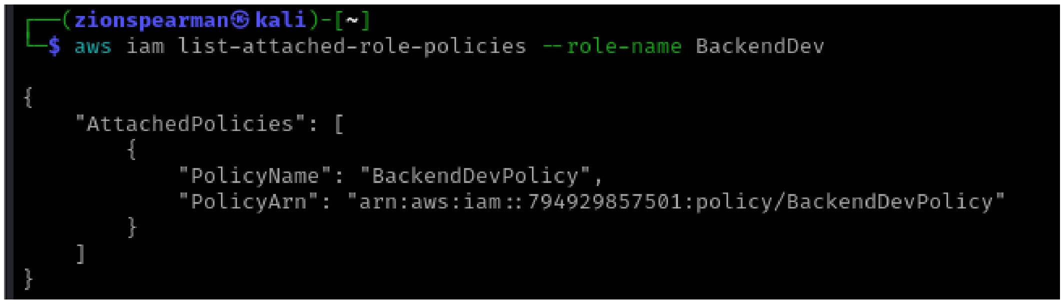
We listed all available versions of the customer-managed policy named dev01. This helped us see how the policy had changed over time and confirmed that version 7 was the most recent and currently in use.



```
┌──(zionspearman㉿kali)-[~]
└─$ aws iam list-policy-versions --policy-arn arn:aws:iam::794929857501:policy/dev01
{
    "Versions": [
        {
            "VersionId": "v7",
            "IsDefaultVersion": true,
            "CreateDate": "2023-10-11T19:59:08+00:00"
        },
        {
            "VersionId": "v6",
            "IsDefaultVersion": false,
            "CreateDate": "2023-10-11T19:47:41+00:00"
        },
        {
            "VersionId": "v5",
            "IsDefaultVersion": false,
            "CreateDate": "2023-10-07T22:48:28+00:00"
        },
        {
            "VersionId": "v4",
            "IsDefaultVersion": false,
            "CreateDate": "2023-10-02T20:38:35+00:00"
        },
        {
            "VersionId": "v3",
            "IsDefaultVersion": false,
            "CreateDate": "2023-10-02T20:29:52+00:00"
        }
    ]
}
```

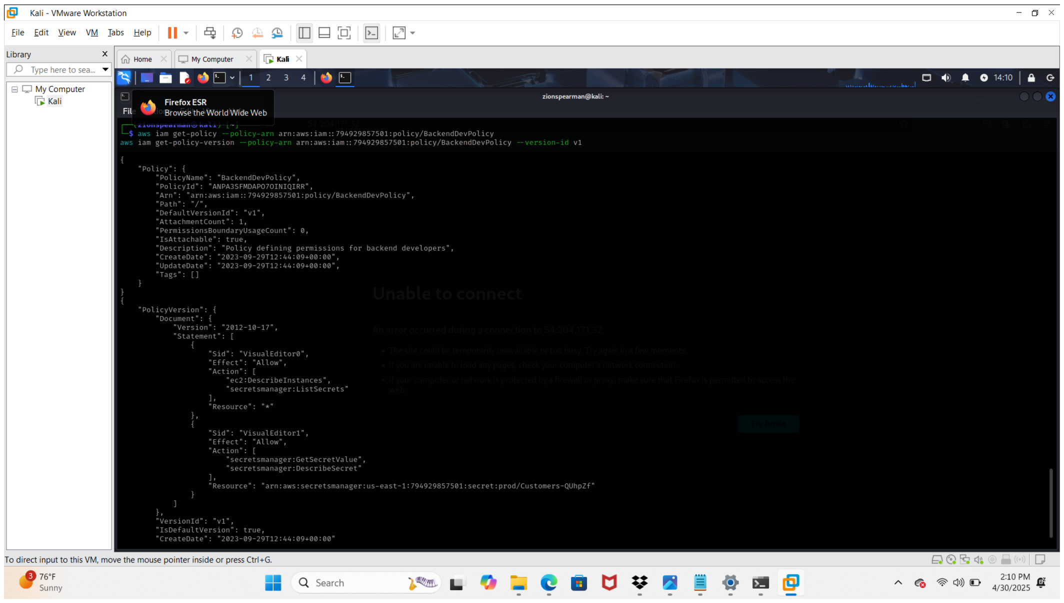
Role Enumeration: BackendDev

We listed the policy attached to the BackendDev role. It showed that the role had one policy attached, named BackendDevPolicy, which defines the permissions granted when the role is assumed.



```
┌──(zionspearman㉿kali)-[~]
└─$ aws iam list-attached-role-policies --role-name BackendDev

{
    "AttachedPolicies": [
        {
            "PolicyName": "BackendDevPolicy",
            "PolicyArn": "arn:aws:iam::794929857501:policy/BackendDevPolicy"
        }
    ]
}
```

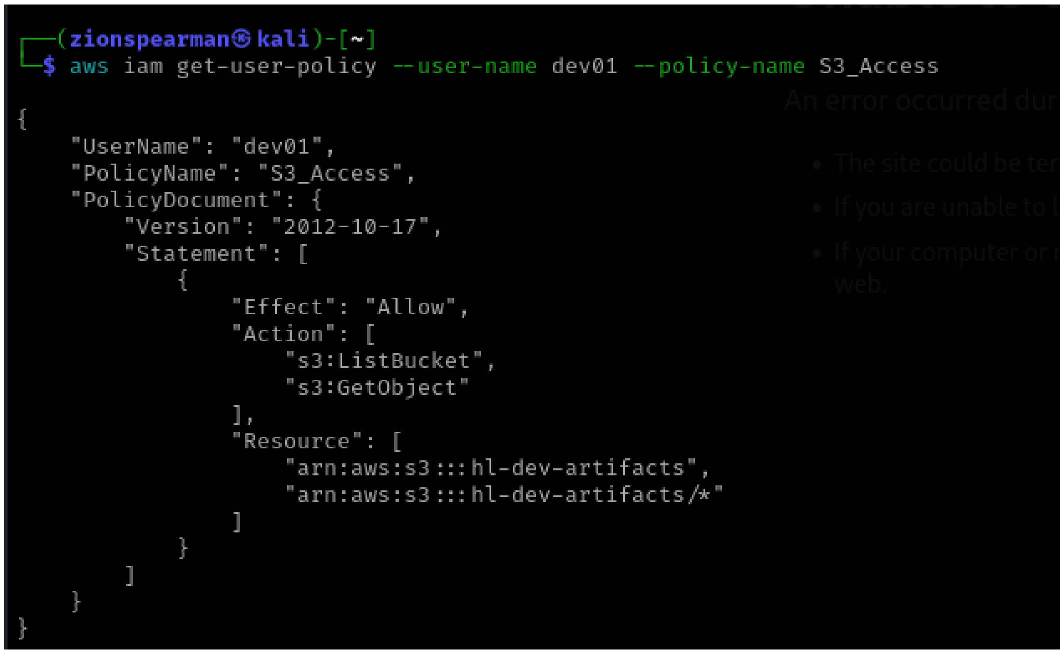
Retrieved the details and permissions granted by BackendDevPolicy:



```
└─$ aws iam get-policy --policy-arn arn:aws:iam::794929857501:policy/BackendDevPolicy
aws iam get-policy-version --policy-arn arn:aws:iam::794929857501:policy/BackendDevPolicy --version-id v1

{
    "Policy": {
        "PolicyName": "BackendDevPolicy",
        "PolicyId": "ANPA3SFMDAPO7OINIQIRR",
        "Arn": "arn:aws:iam::794929857501:policy/BackendDevPolicy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 1,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "Description": "Policy defining permissions for backend developers",
        "CreateDate": "2023-09-29T12:44:09+00:00",
        "UpdateDate": "2023-09-29T12:44:09+00:00",
        "Tags": []
    }
}
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "VisualEditor0",
                    "Effect": "Allow",
                    "Action": [
                        "ec2:DescribeInstances",
                        "secretsmanager:ListSecrets"
                    ],
                    "Resource": "*"
                },
                {
                    "Sid": "VisualEditor1",
                    "Effect": "Allow",
                    "Action": [
                        "secretsmanager:GetSecretValue",
                        "secretsmanager:DescribeSecret"
                    ],
                    "Resource": "arn:aws:secretsmanager:us-east-1:794929857501:secret:prod/Customers-QUhpZf"
                }
            ]
        },
        "VersionId": "v1",
        "IsDefaultVersion": true,
        "CreateDate": "2023-09-29T12:44:09+00:00"
```

Inline User Policy: S3_Access

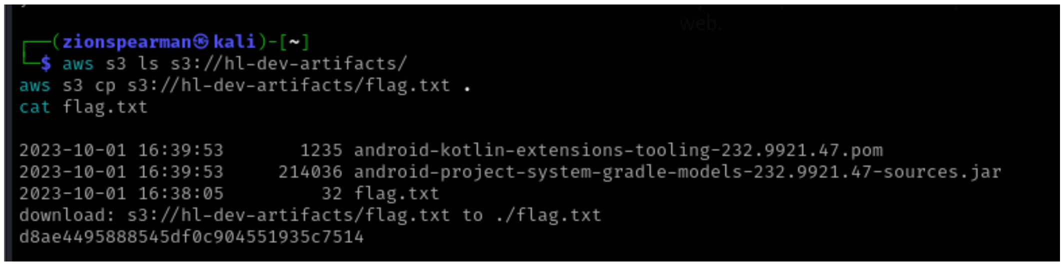
We retrieved the inline policy attached to the user dev01 to see what S3 permissions were granted. The policy allows the user to list and read objects from the S3 bucket named hl-dev-artifacts.



```
┌──(zionspearman㉿kali)-[~]
└─$ aws iam get-user-policy --user-name dev01 --policy-name S3_Access

{
    "UserName": "dev01",
    "PolicyName": "S3_Access",
    "PolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:ListBucket",
                    "s3:GetObject"
                ],
                "Resource": [
                    "arn:aws:s3:::hl-dev-artifacts",
                    "arn:aws:s3:::hl-dev-artifacts/*"
                ]
            }
        ]
    }
}
```

S3 Bucket Enumeration and Flag Retrieval

Enumerated the S3 bucket, found 'flag.txt', downloaded it, and revealed the flag:



```
┌──(zionspearman㉿kali)-[~]
└─$ aws s3 ls s3://hl-dev-artifacts/
aws s3 cp s3://hl-dev-artifacts/flag.txt .
cat flag.txt

2023-10-01 16:39:53       1235 android-kotlin-extensions-tooling-232.9921.47.pom
2023-10-01 16:39:53     214036 android-project-system-gradle-models-232.9921.47-sources.jar
2023-10-01 16:38:05         32 flag.txt
download: s3://hl-dev-artifacts/flag.txt to ./flag.txt
d8ae4495888545df0c904551935c7514
```

Key Security Issues:

- **Overly permissive access** to sensitive services like S3, IAM, and Secrets Manager
- **Hardcoded access keys** that could be leaked or misused
- **Inline policies** that are harder to track and manage securely
- **Single user trust for role assumption**, creating a privilege escalation risk

These findings highlight how important it is to follow the principle of least privilege and regularly audit IAM permissions to reduce the blast radius if a user account is compromised.

Summary

In this lab, we looked into the IAM user dev01 to see what they could access. We used the AWS CLI to check their identity and find out what permissions they had. We found that dev01 could read GuardDuty data, view IAM roles and policies, and access an S3 bucket with a file called flag.txt. The user could also take on a role that lets them view EC2 information and access a secret in Secrets Manager. This shows how important it is to know what a user can do if their account is ever compromised.