AWS re:Inforce

JUNE 10 - 12, 2024 | PHILADELPHIA, PA

TDR432

New tactics and techniques for proactive threat detection

Ben Fletcher
(he/him)
AWS EMEA CIRT LEAD
Amazon Web Services

Steve de Vera
(he/him)
AWS CIRT Manager
Amazon Web Services

Agenda

- About AWS CIRT
- Statistics
- Current threat actor tactics
- New threat actor tactics
- Security best practices

THIS SESSION IS INTERACTIVE!

Feel free to ask questions, make comments, participate

About AWS CIRT

AWS Customer Incident Response Team (CIRT)

A specialized team that assists and advises customers during suspected active security events, on the **customer's side** of the **AWS Shared Responsibility Model**

Respond

Global team 24/7, follow-the-sun model

Recover

Assist and advise customers with active triage and recovery from their security event on AWS

Learn

Assist in root cause analysis of a customer's AWS service logs for their active security event

Educate

Provide advice to customers for long-term recovery from their active security event

Statistics

Threat actors use which initial access method most often?

Cross-account permissions	Vulnerable web apps	Brute force
#1	#2	#3
Lost/leaked access keys/credentials	Open S3 buckets	DDoS
#4	#5	#6

Threat actors use which initial access method most often?

Lost/leaked access keys/credentials

#4

Threat actors use which initial access method most often?

66%

valid IAM credentials

Lost/leaked access keys/credentials

#4

Threat actors use which initial access method most often?

Lost/leaked access keys/credentials

#4

66%

valid IAM credentials

↑

1/3

of those are **root credentials**

[20% of all initial access method use]

Threat actors use which initial access method most often?

Lost/leaked access keys/credentials

#4

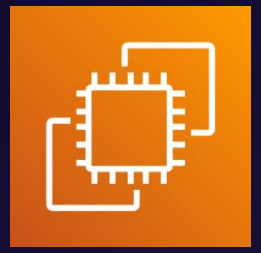
66%

valid IAM credentials

↑

1/3

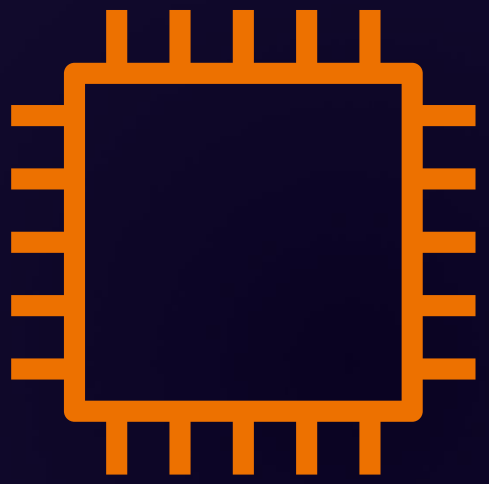
of those are **root credentials**
[20% of all initial access method use]



13%

Public-facing EC2 instance

Threat primary tactics



Resource hijack

Ransom events

Opportunistic destruction

A zero trust strategy

Get the keys

MITRE ATT&CK
Tactic: Initial access
Technique: Valid cloud credentials

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

- (●) **Command Line Interface (CLI)**
  You plan to use this access key to enable the AWS CLI to access your AWS account.
- ( ) **Local code**
  You plan to use this access key to enable application code in a local development environment to access your AWS account.
- ( ) **Application running on an AWS compute service**
  You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
- ( ) **Third-party service**
  You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
- ( ) **Application running outside AWS**
  You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.
- ( ) **Other**
  Your use case is not listed here.

⚠ **Alternatives recommended**

- Use AWS CloudShell, a browser-based CLI, to run commands. Learn more
- Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center. Learn more

Confirmation

☐ I understand the above recommendation and want to proceed to create an access key.

Cancel | Next

If keys are posted on GitHub, how long until they are used?

~Two weeks

#1

~One week

#2

24 hours

#3

4 hours

#4

Minutes, if not seconds

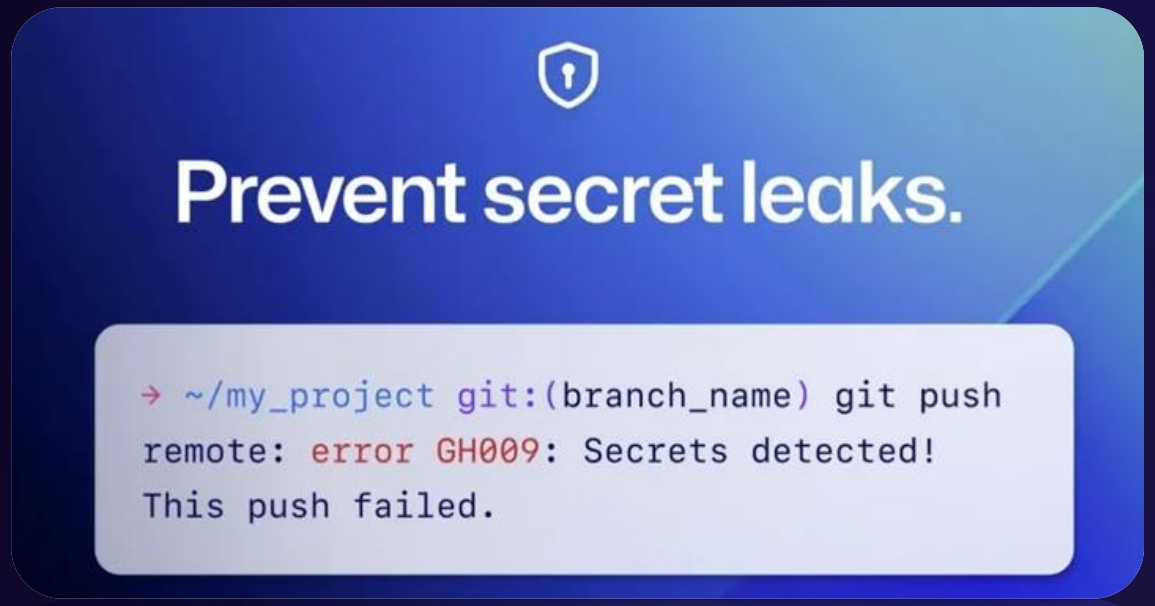
#5

GitHub

MITRE ATT&CK
Tactic: Initial access
Technique: Valid cloud credentials

Minutes, if not seconds

#5



https://thehackernews.com/2024/03/github-rolls-out-default-secret.html

Current threat actor tactics

DISCLAIMER:

Tactics and techniques presented do not constitute vulnerabilities within AWS

Resource hijacking: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

Resource hijacking: Premise

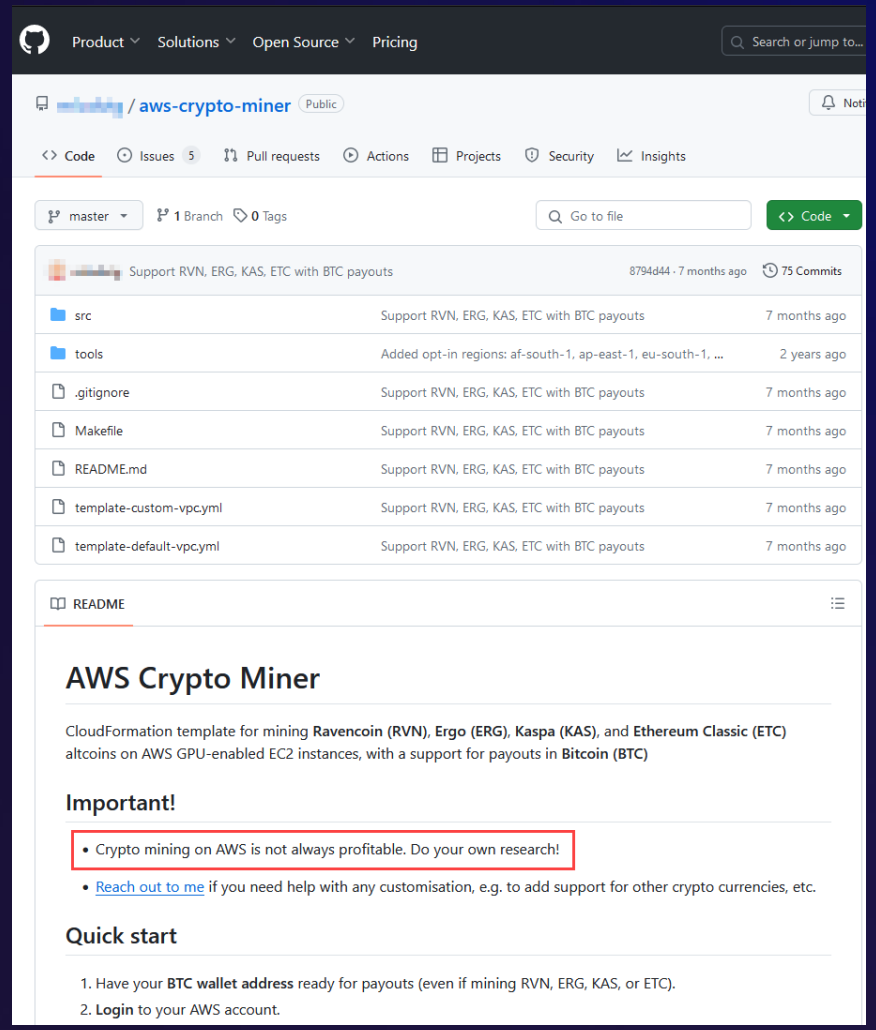
MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

1) Threat actor obtains access to AWS account or hosted resource

Resource hijacking: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

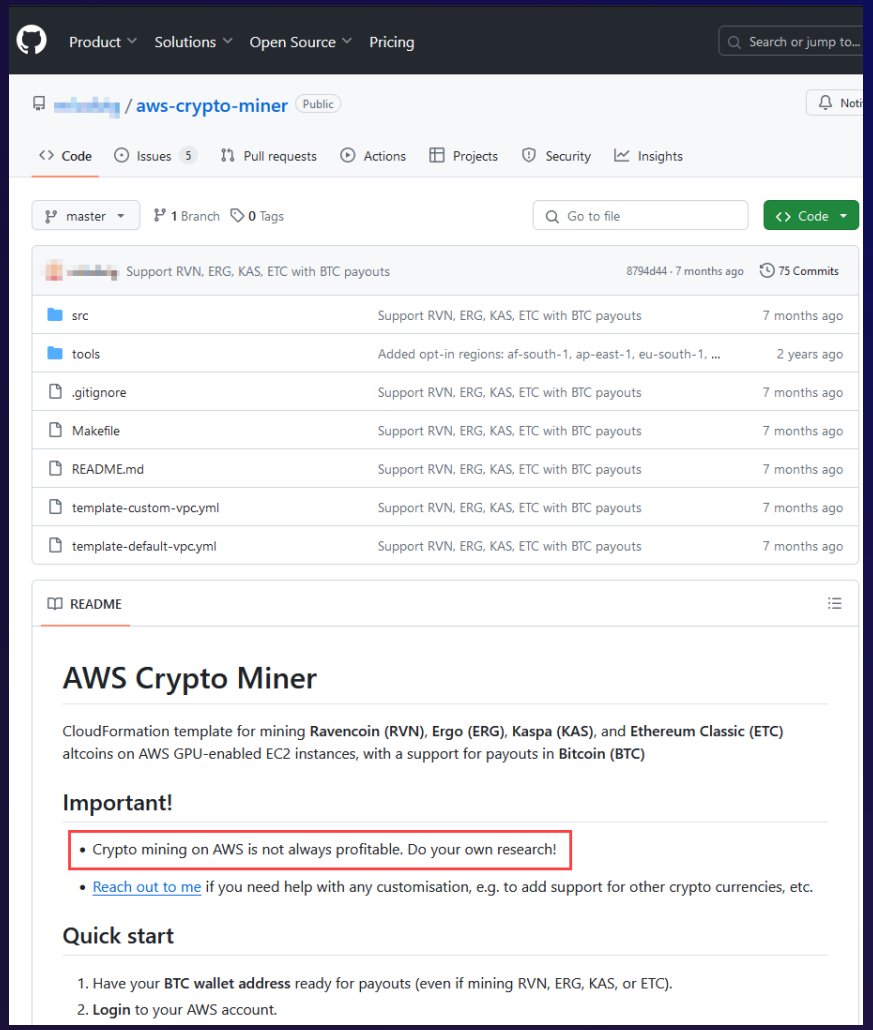
1) Threat actor obtains access to AWS account or hosted resource
2) Threat actor will mine cryptocurrency from the resource



Resource hijacking: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

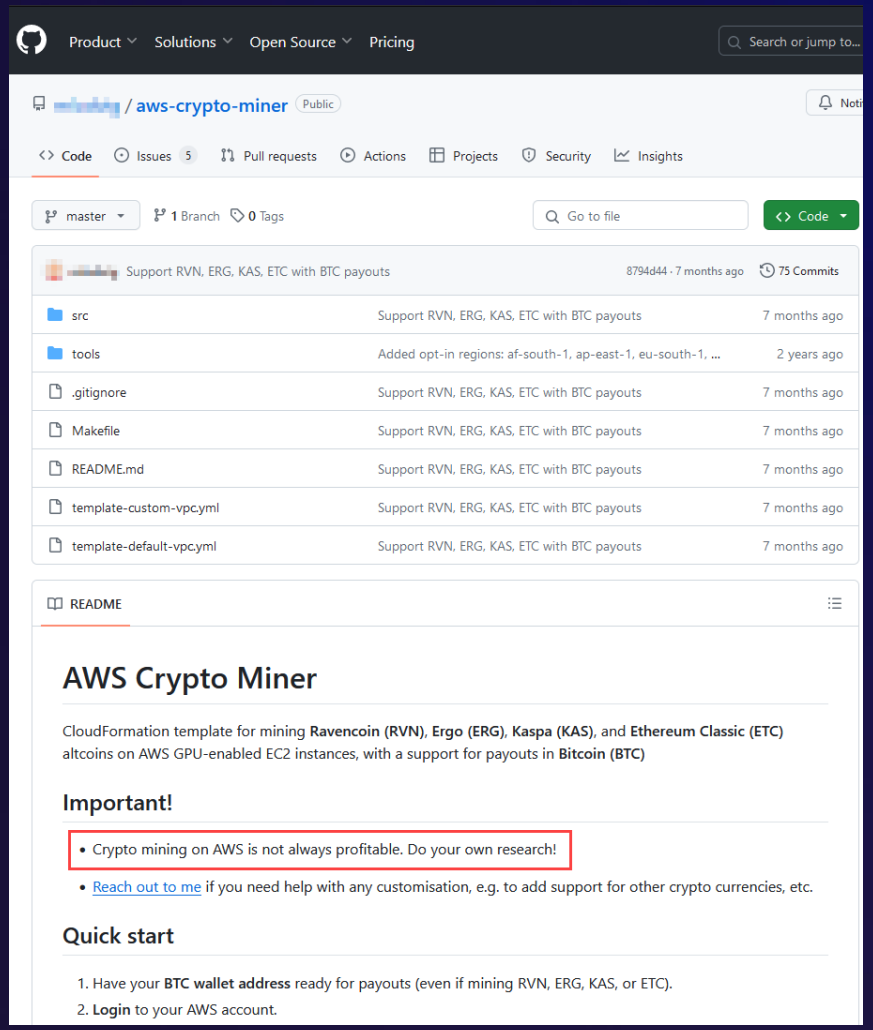
1) Threat actor obtains access to AWS account or hosted resource
2) Threat actor will mine cryptocurrency from the resource
3) Resources created in AWS account:
   - `RunInstances`
   - `CreateStack`
   - `CreateCluster`



Resource hijacking: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

1) Threat actor obtains access to AWS account or hosted resource
2) Threat actor will mine cryptocurrency from the resource
3) Resources created in AWS account:
   - `RunInstances`
   - `CreateStack`
   - `CreateCluster`
4) Resources created in unused AWS Regions



Resource hijacking: Mitigations

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

- Use SCPs to prevent resource creation – especially in unused Regions
- Apply principle of least privilege to assigned permissions

SubDomain takeover: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Defacement

SubDomain takeover: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Defacement

1) Customer has CNAME pointing to a resource (S3 bucket, EC2 instance, Elastic IP)

SubDomain takeover: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Defacement

1) Customer has CNAME pointing to a resource (S3 bucket, EC2 instance, Elastic IP)

2) The resource is deleted, but the CNAME still exists

SubDomain takeover: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Defacement

1) Customer has CNAME pointing to a resource (S3 bucket, EC2 instance, Elastic IP)

2) The resource is deleted, but the CNAME still exists

3) Threat actor creates a resource that the CNAME still points to

SubDomain takeover: Premise

SubDomain takeover: Premise

Customer

SubDomain takeover: Premise

S3 bucket

Customer

s3-newco-random

SubDomain takeover: Premise

S3 bucket

S3 bucket configured as static website

Customer

s3-newco-random

http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

SubDomain takeover: Premise

Customer

S3 bucket

s3-newco-random

S3 bucket configured as static website

http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

CNAME pointing to S3 bucket configured as static website

CNAME: app1.newco.com

points to:
http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

SubDomain takeover: Premise

Customer

S3 bucket

s3-newco-random

S3 bucket configured as static website

http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

CNAME pointing to S3 bucket configured as static website

CNAME: app1.newco.com

points to:
http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

SubDomain takeover: Premise

S3 bucket

S3 bucket configured as static website

CNAME pointing to S3 bucket configured as static website

Customer

s3-newco-random

http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

CNAME: app1.newco.com

points to:
http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

Threat actor

SubDomain takeover: Premise

Customer

S3 bucket

s3-newco-random

S3 bucket configured as static website

http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

CNAME pointing to S3 bucket configured as static website

CNAME: app1.newco.com

points to:
http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

Threat actor

S3 bucket recreated

s3-newco-random

SubDomain takeover: Premise

Customer

S3 bucket

s3-newco-random

S3 bucket configured as static website

http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

CNAME pointing to S3 bucket configured as static website

CNAME: app1.newco.com

points to:
http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

Threat actor

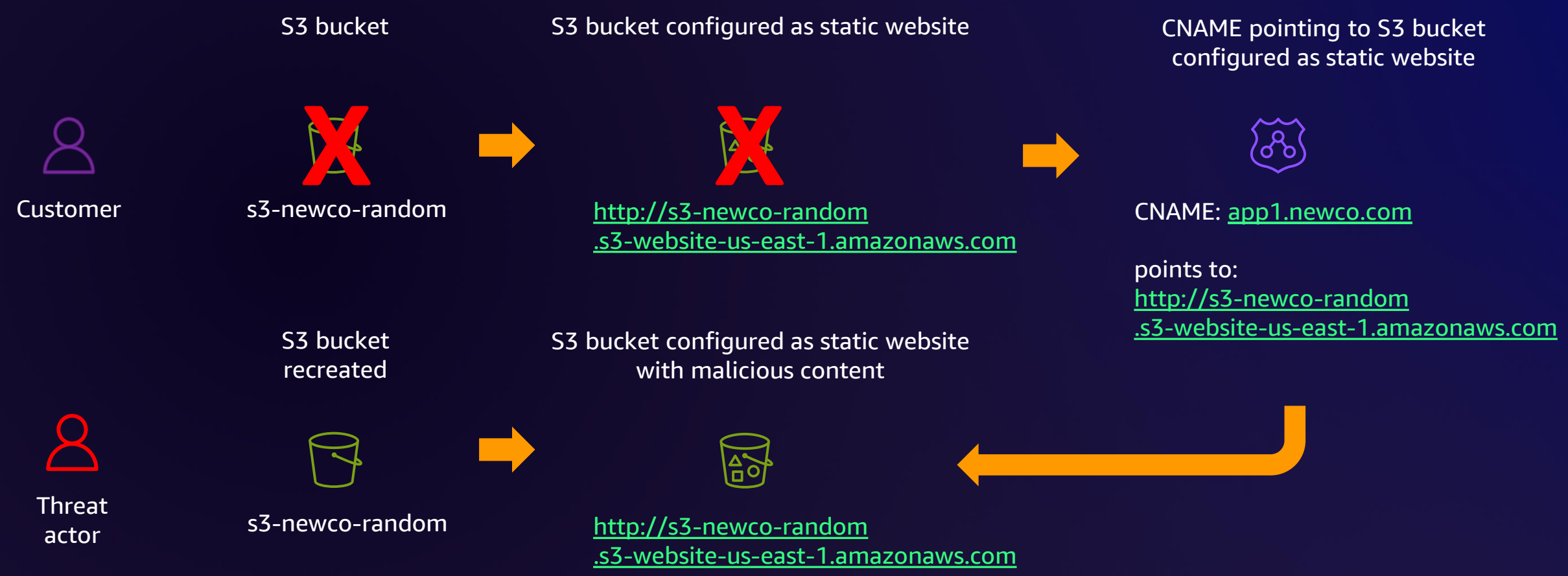
S3 bucket recreated

s3-newco-random

S3 bucket configured as static website with malicious content

http://s3-newco-random
.s3-website-us-east-1.amazonaws.com

SubDomain takeover: Premise



SubDomain takeover: Mitigations

- Review hosted zones and delete unused CNAMEs
- When de-provisioning, remove CNAMEs first

Data destruction: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

Data destruction: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

1) Threat actor obtains access to AWS account or resource (Amazon S3 or Amazon RDS)

Data destruction: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

1) Threat actor obtains access to AWS account or resource (Amazon S3 or Amazon RDS)

2) Threat actor will attempt to delete resources or data

Data destruction: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

1) Threat actor obtains access to AWS account or resource (Amazon S3 or Amazon RDS)

2) Threat actor will attempt to delete resources or data

3) Resources deleted in AWS account:

- `DeleteBucket`
- `DeleteObject`
- `DeleteDBInstance`
- `DeleteDBCluser`
- `DeleteDBSnapshot`
- `AuthorizeSecurityGroupIngress`

Data destruction: Mitigations

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

- Apply and review policies (resource policies and lifecycle policies), S3 Object Lock
- Principle of least privilege
- Use and test backup methodologies

IMDSv1 credential access: Premise

MITRE ATT&CK
Tactic: Credential access
Technique: Unsecured credentials

IMDSv1 credential access: Premise

MITRE ATT&CK
Tactic: Credential access
Technique: Unsecured credentials

1) Threat actor obtains ability to obtain IMDSv1 credentials from resource

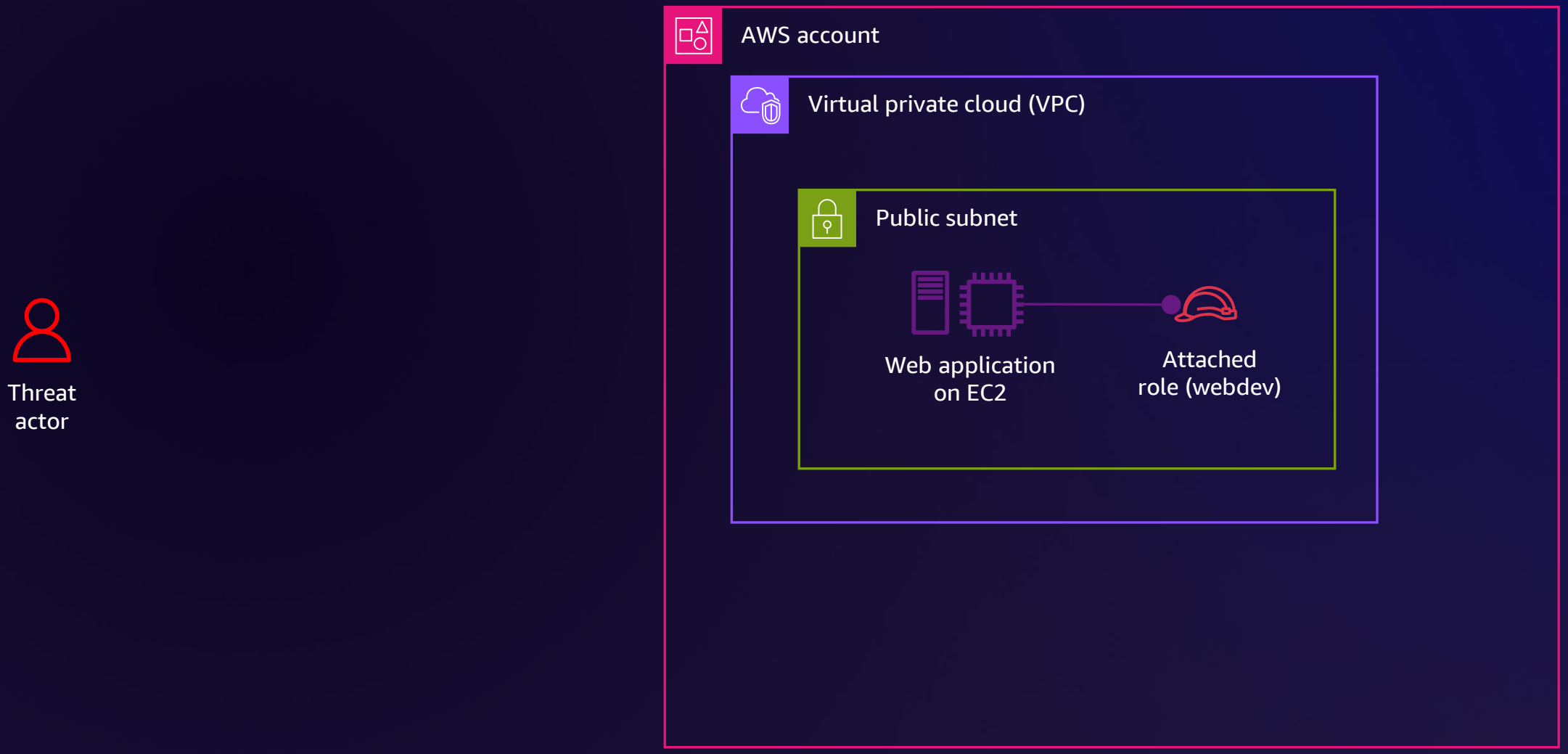
IMDSv1 credential access: Premise

MITRE ATT&CK
Tactic: Credential access
Technique: Unsecured credentials

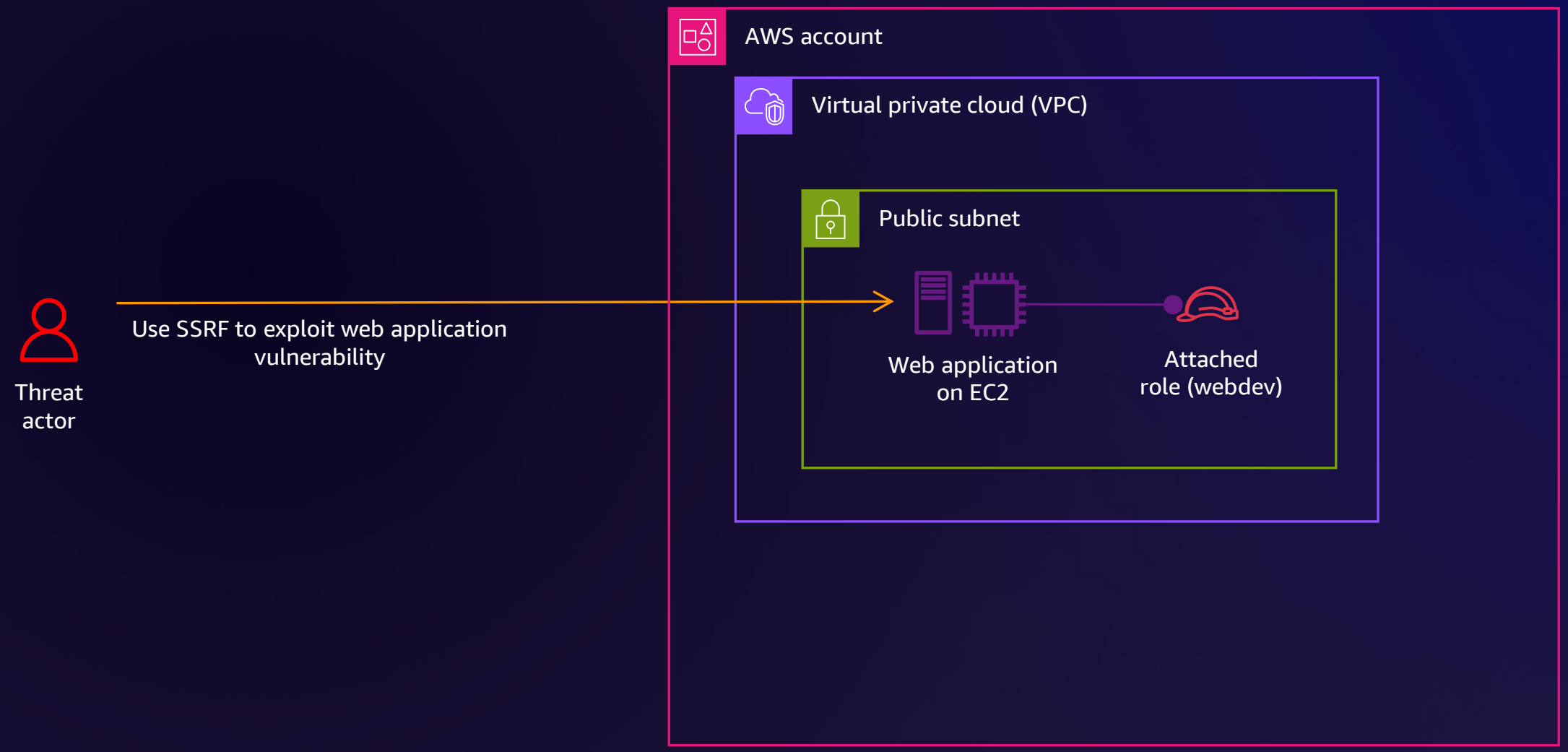
1) Threat actor obtains ability to obtain IMDSv1 credentials from resource

2) Threat actor exports and uses credentials

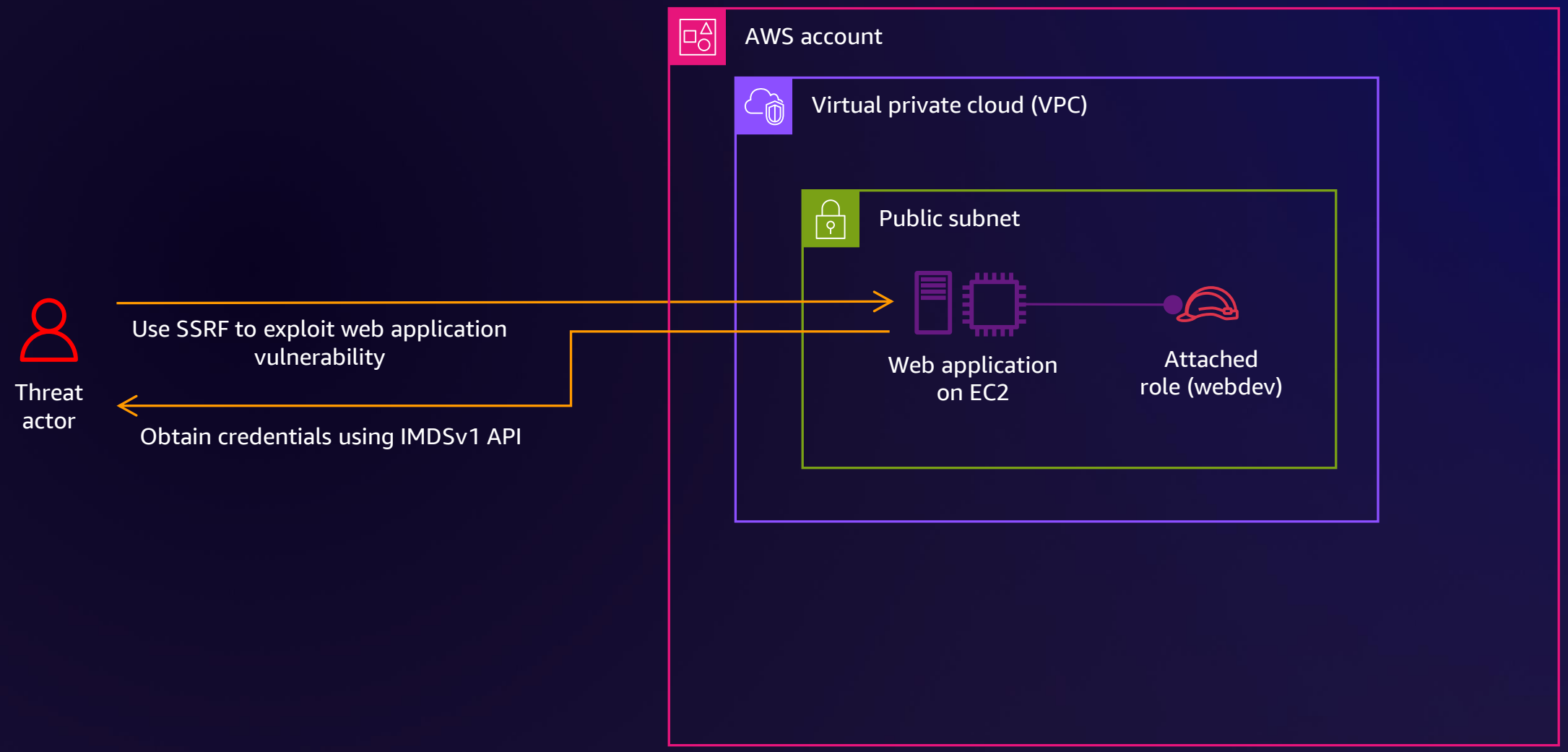
IMDSv1 credential access: Premise



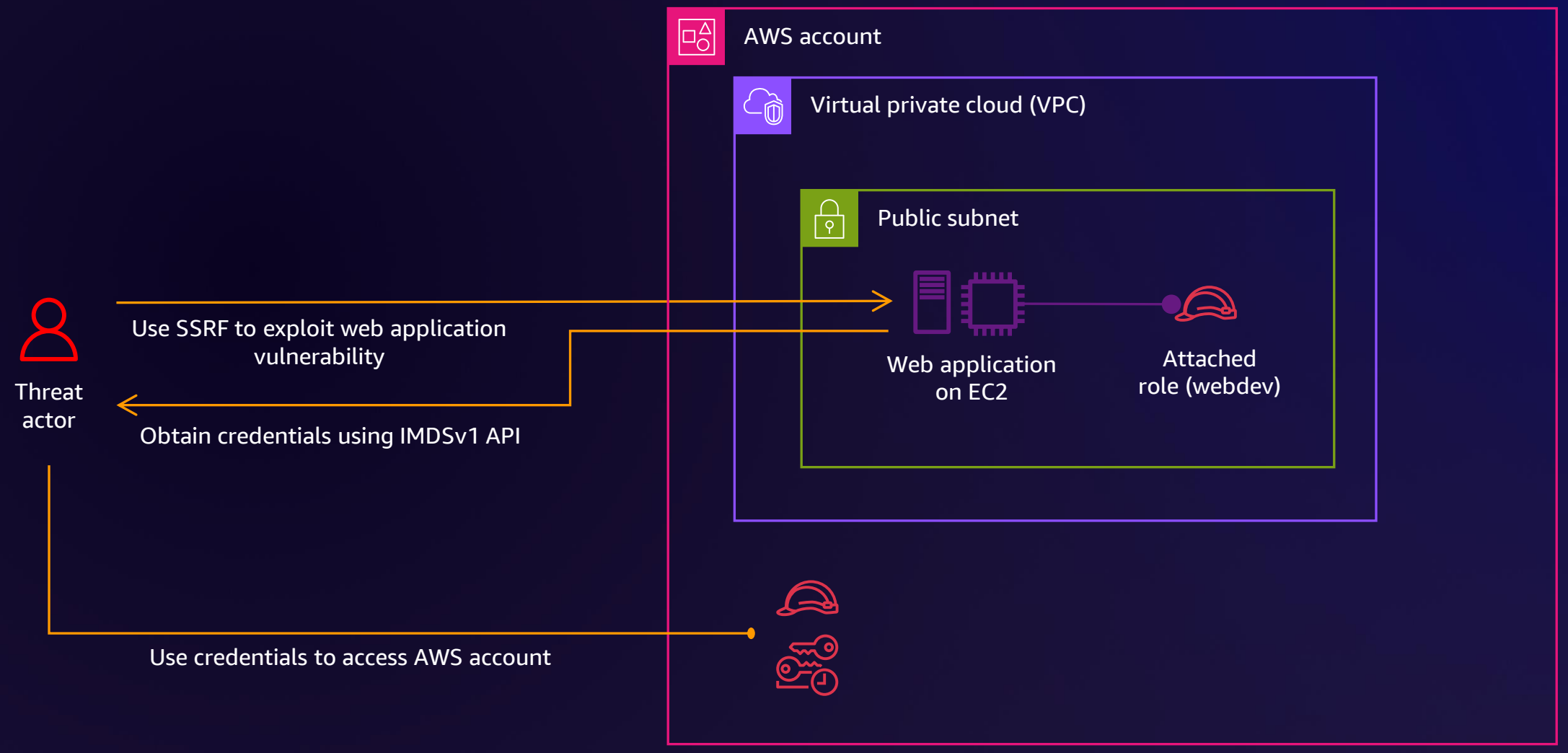
IMDSv1 credential access: Premise



IMDSv1 credential access: Premise

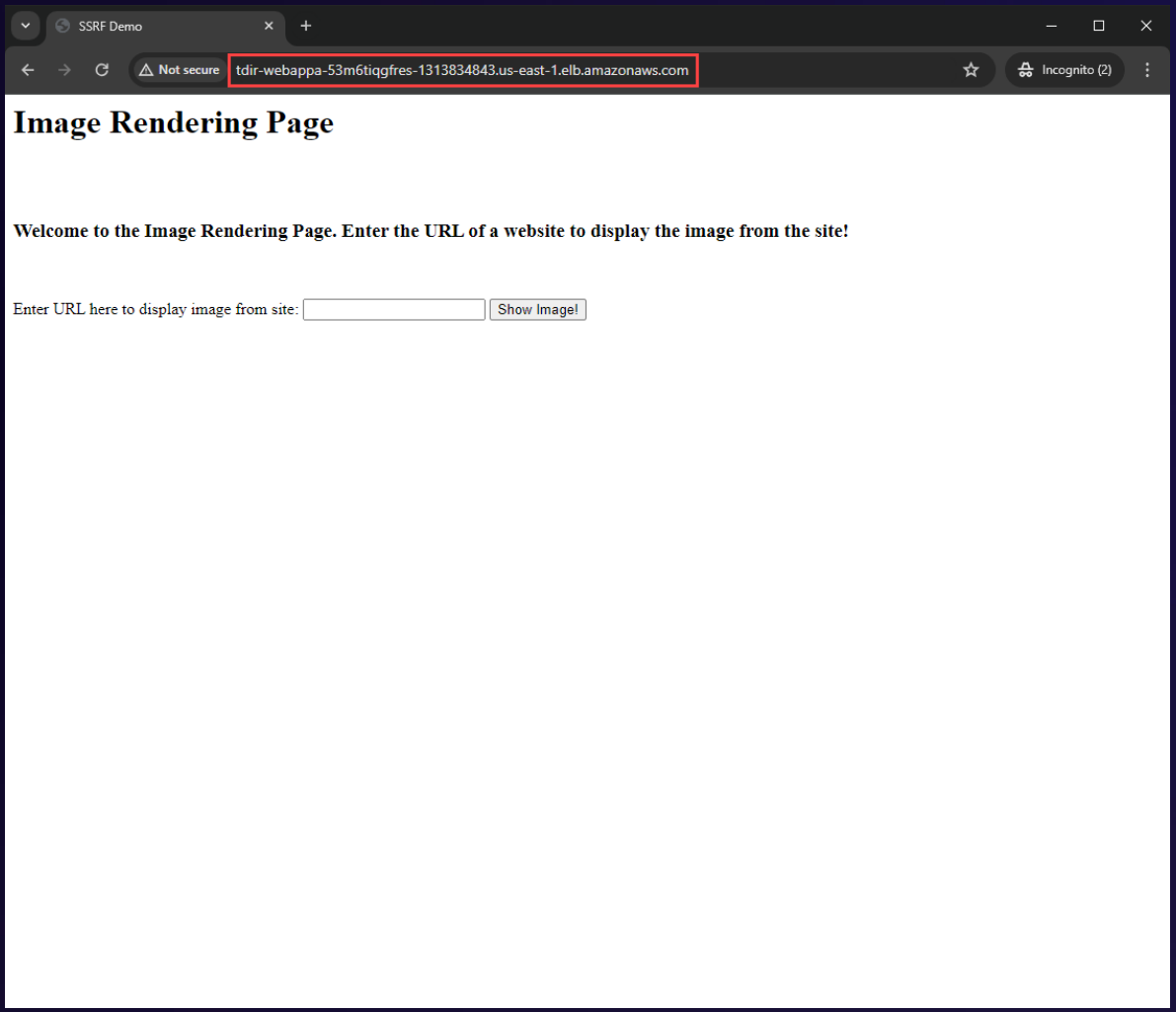


IMDSv1 credential access: Premise

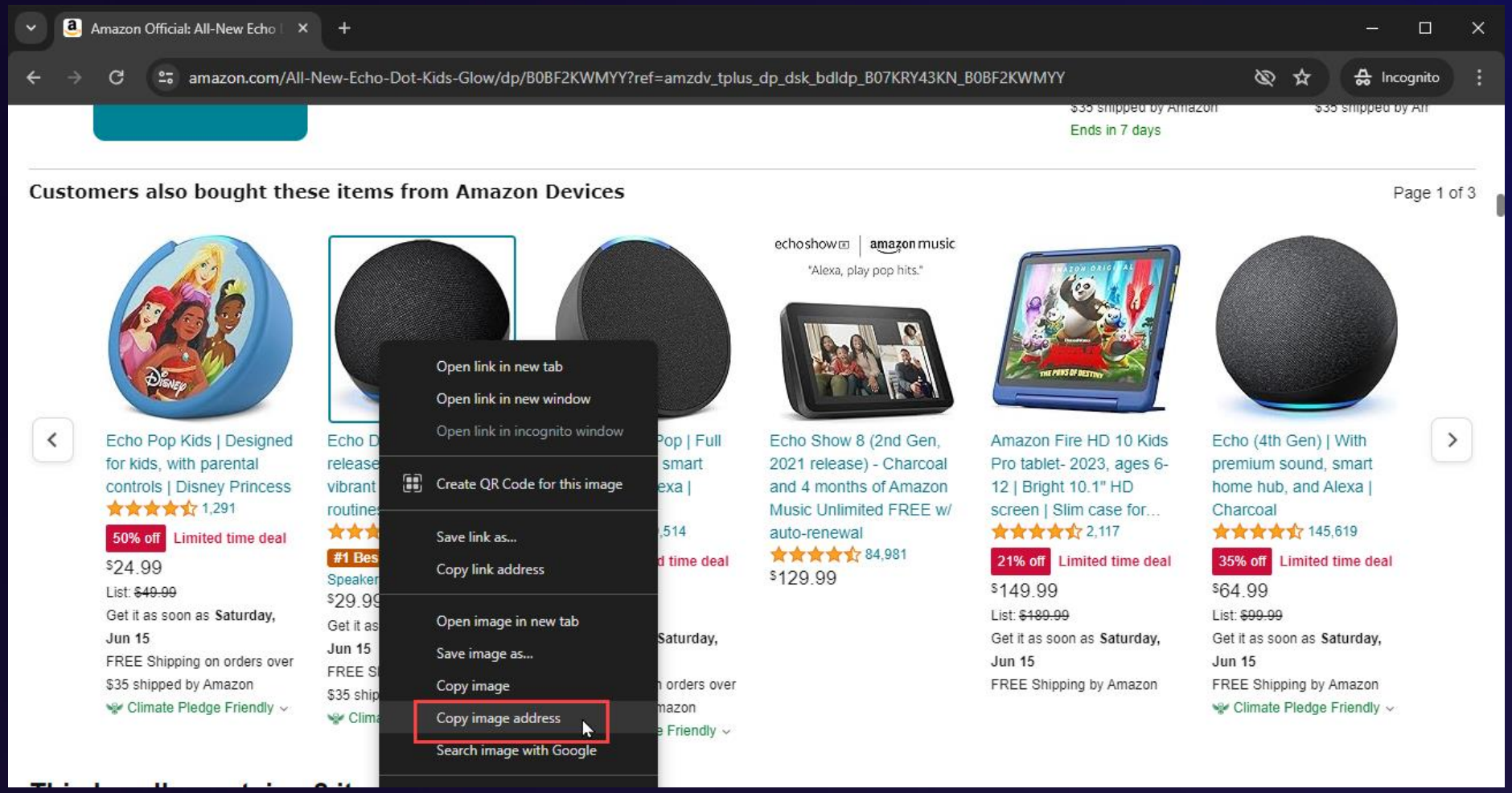


IMDSv1 credential access: Premise

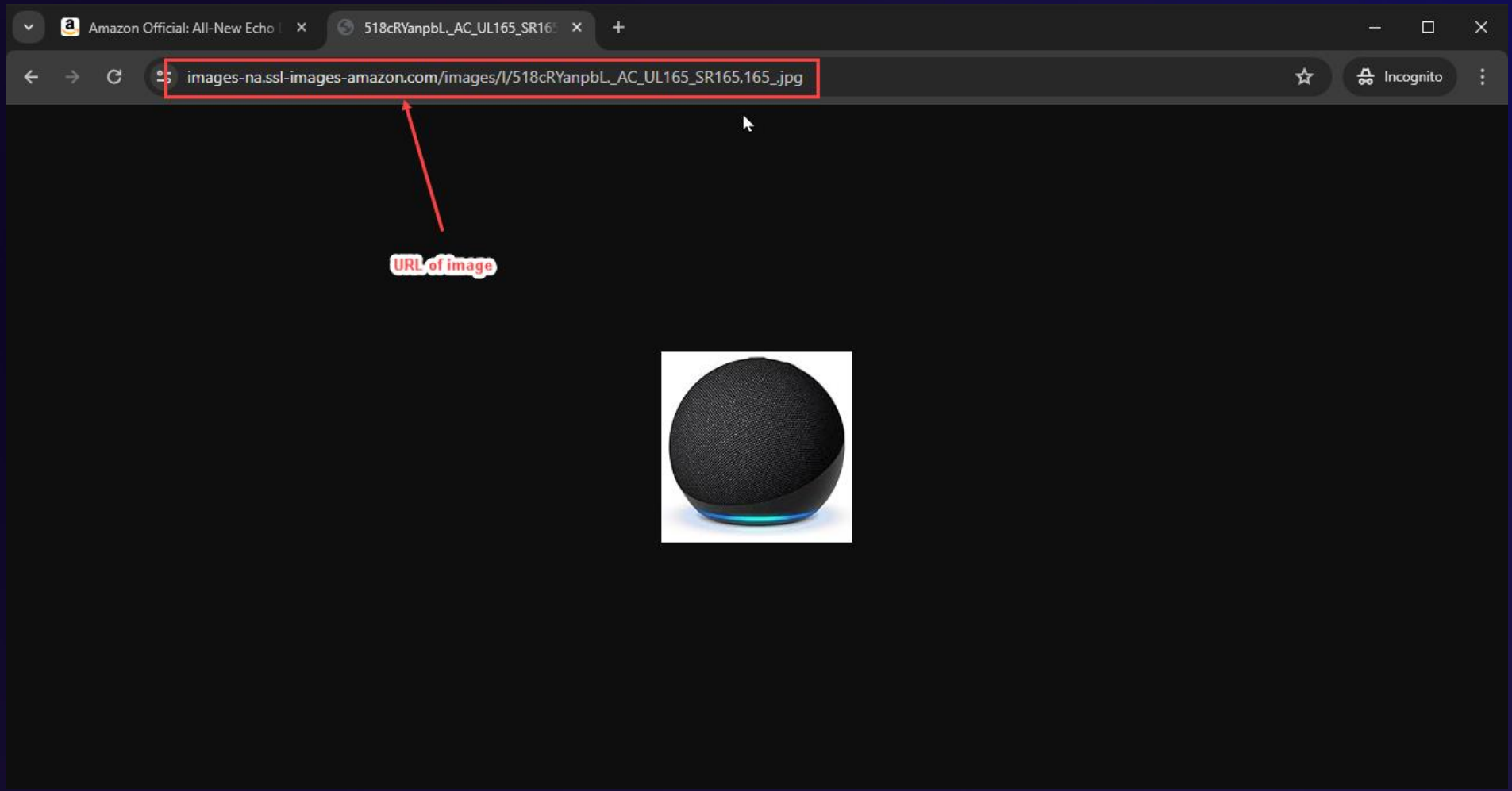
IMDSv1 credential access: Premise



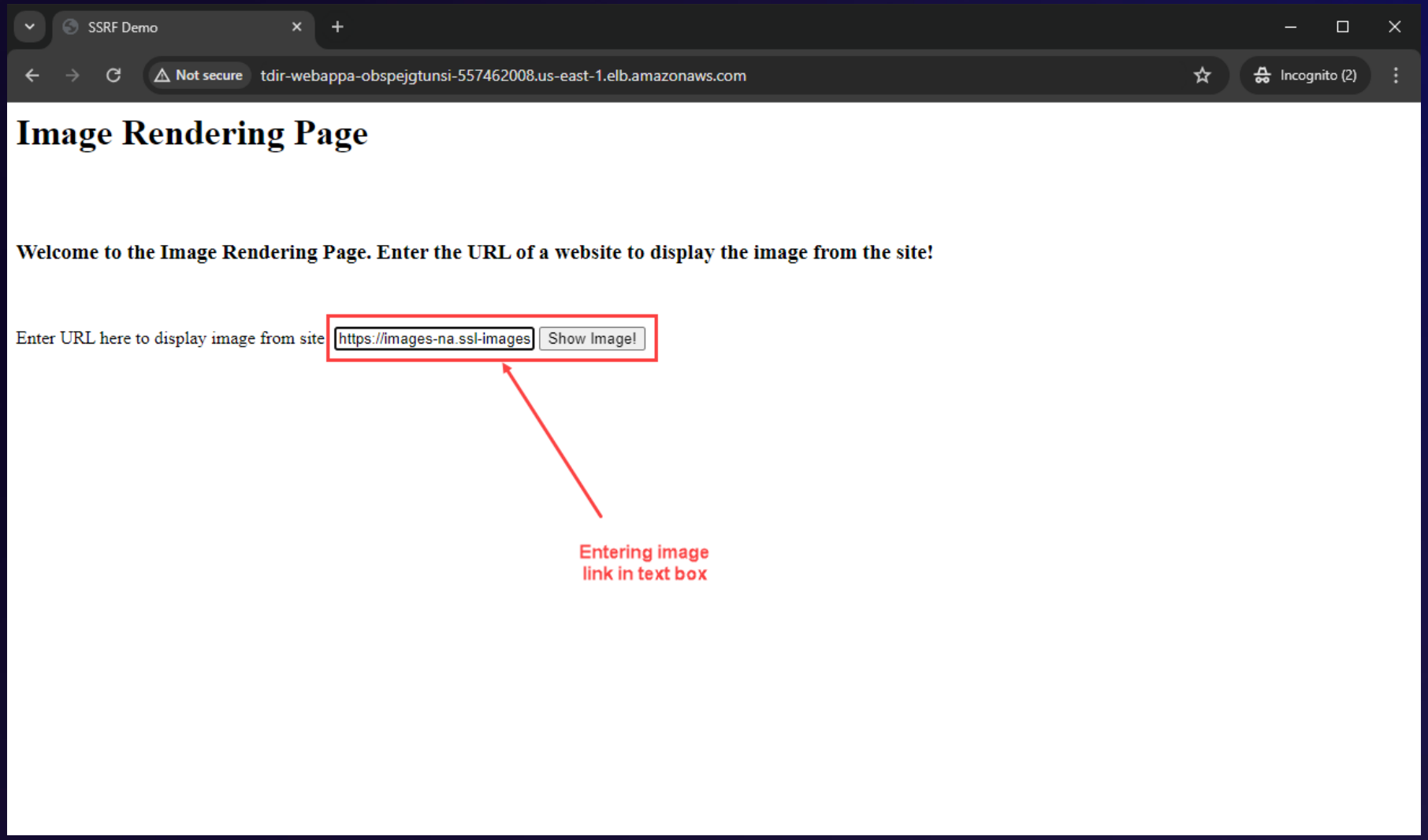
IMDSv1 credential access: Premise



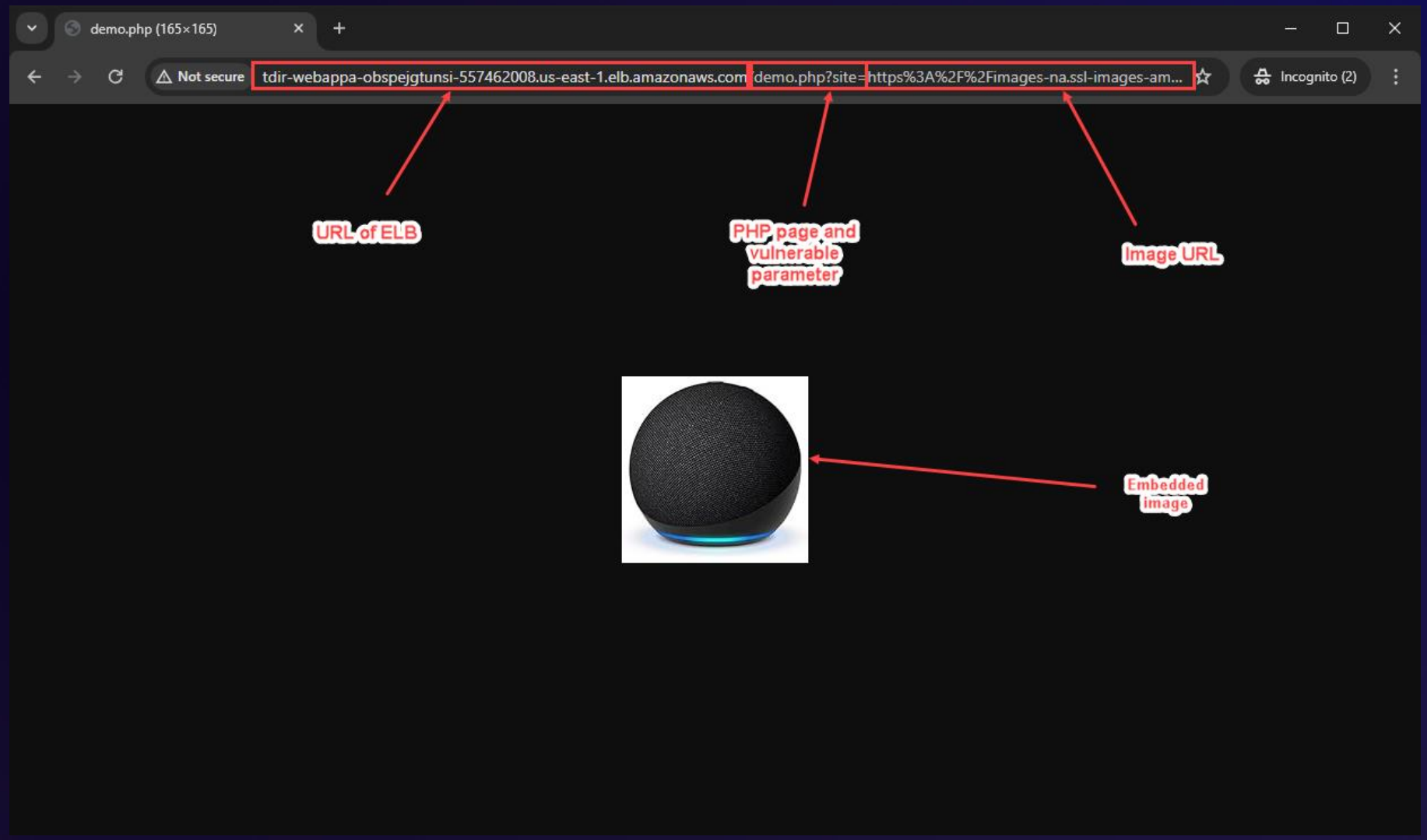
IMDSv1 credential access: Premise



IMDSv1 credential access: Premise



IMDSv1 credential access: Premise



IMDSv1 credential access: Premise

```
sdevera@SEA-1801401059: ~
:~$
:~$ curl 'http://tdir-webappa-53m6tiqgfres-1313834843.us-east-1.elb.amazonaws.com/demo.php?site=http://169.254.169.254/latest/meta-data"
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
iam/
identity-credentials/
instance-action
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
reservation-id
security-groups
services/
system
:~$
```

URL of ELB

PHP page and vulnerable parameter

URL of IMDS

IMDSv1 credential access: Premise

```
:~$
:~$
:~$ curl "http://tdir-webappa-53m6tiqgfres-1313834843.us-east-1.elb.amazonaws.com/demo.php?site=
http://169.254.169.254/latest/meta-data/iam/security-credentials/webdev"
{
  "Code" : "Success",
  "LastUpdated" : "2024-05-25T22:46:55Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "ASIA[redacted]",
  "SecretAccessKey" : "[redacted]",
  "Token" : "[redacted]",
  "Expiration" : "2024-05-26T05:10:18Z"
}
:~$
:~$
```

Requesting credentials from IMDS

Credentials

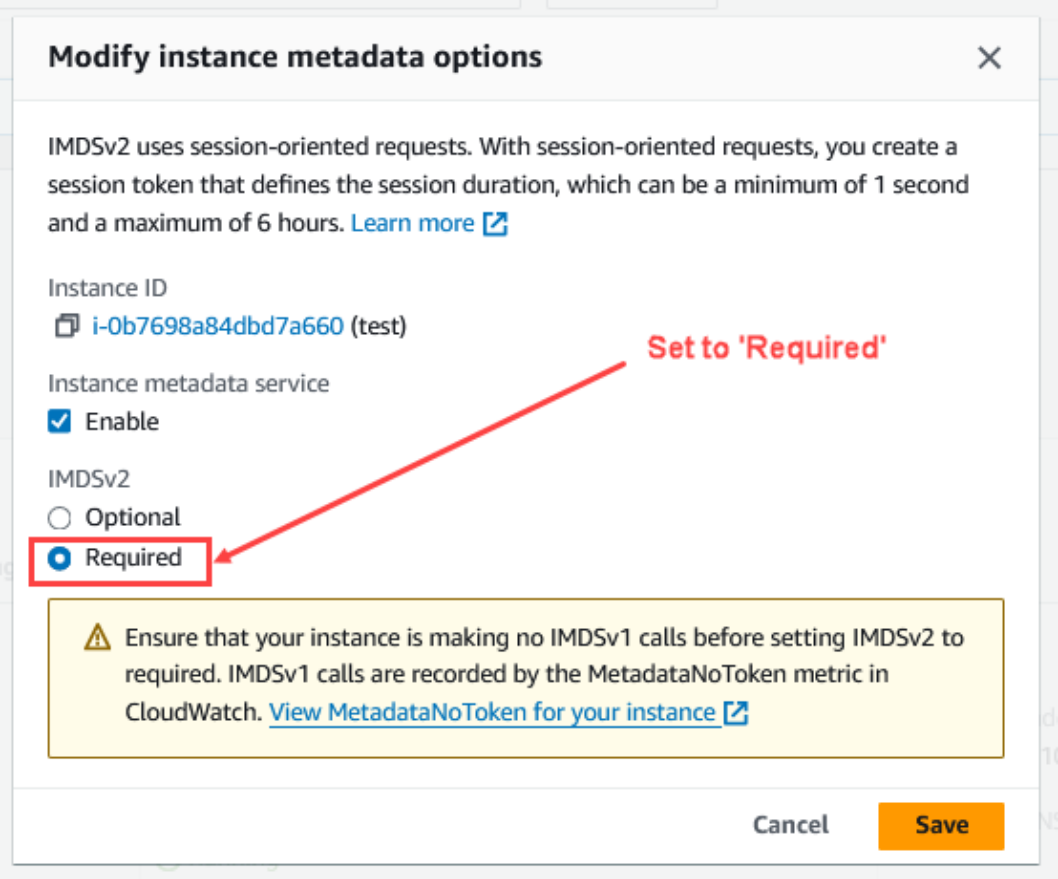
IMDSv1 credential access: Mitigations

MITRE ATT&CK
Tactic: Credential access
Technique: Unsecured credentials

IMDSv1 credential access: Mitigations

MITRE ATT&CK
Tactic: Credential access
Technique: Unsecured credentials

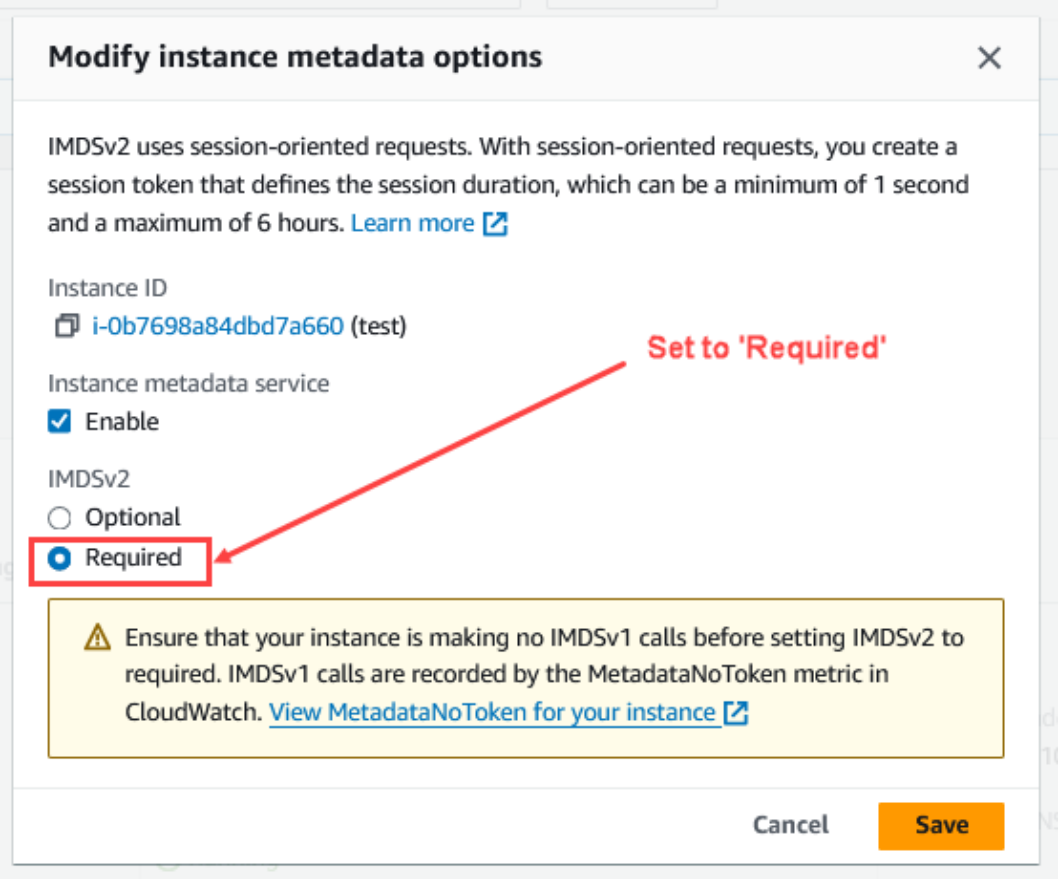
- ~~Use~~ *require* IMDSv2



IMDSv1 credential access: Mitigations

MITRE ATT&CK
Tactic: Credential access
Technique: Unsecured credentials

- ~~Use~~ *require* IMDSv2
- Use principle of least privilege on EC2 instance profile



IMDSv1 credential access: Mitigations

MITRE ATT&CK
Tactic: Credential access
Technique: Unsecured credentials

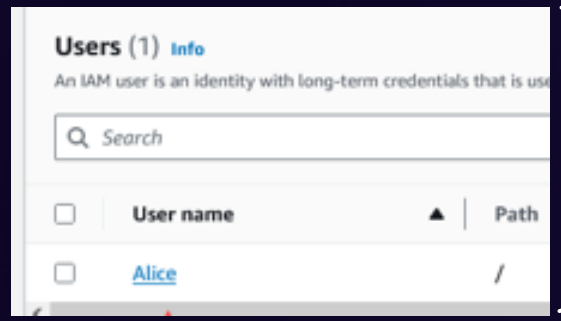
- ~~Use~~ *require* IMDSv2
- Use principle of least privilege on EC2 instance profile
- Use the `aws:EC2InstanceSourceVPC` or `aws:EC2InstanceSourcePrivateIPv4` global condition keys in Service Control Policies

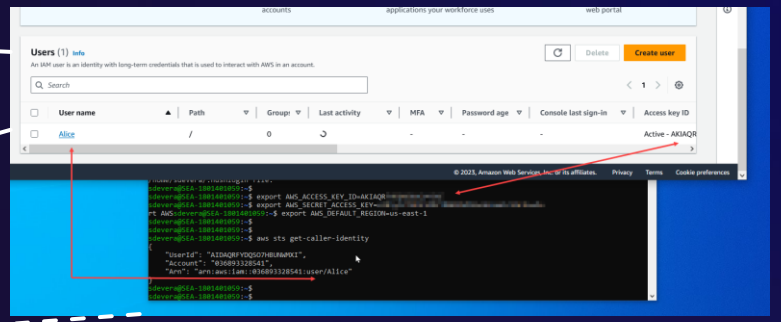
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ec2InstanceSourceVPC": "${aws:SourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::*:role/aws:ec2-infrastructure"
          ]
        }
      }
    },
```

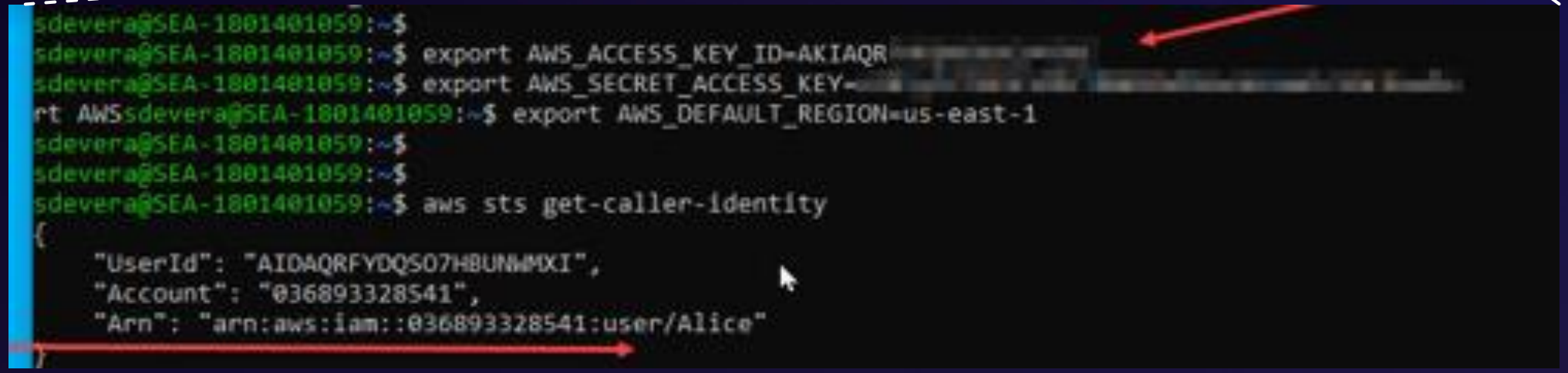
GetFederationToken: Premise

MITRE ATT&CK
Tactic: Persistence
Technique: Additional cloud credentials

1) Credentials exported







GetFederationToken: Premise

2) Federation token generated

```
sdevera@SEA-1801401059:~$
sdevera@SEA-1801401059:~$ aws sts get-federation-token --name HIDDEN_DUE_TO_SECURITY_REASONS --duration-seconds 129600
-policy-arns arn=arn:aws:iam::aws:policy/AdministratorAccess
{
    "Credentials": {
        "AccessKeyId": "[redacted]",
        "SecretAccessKey": "[redacted]",
        "SessionToken": "[redacted]",
        "Expiration": "2023-10-06T09:45:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "036893328541:HIDDEN_DUE_TO_SECURITY_REASONS",
        "Arn": "arn:aws:sts::036893328541:federated-user/HIDDEN_DUE_TO_SECURITY_REASONS"
    },
    "PackedPolicySize": 7
}
```

GetFederationToken: Premise

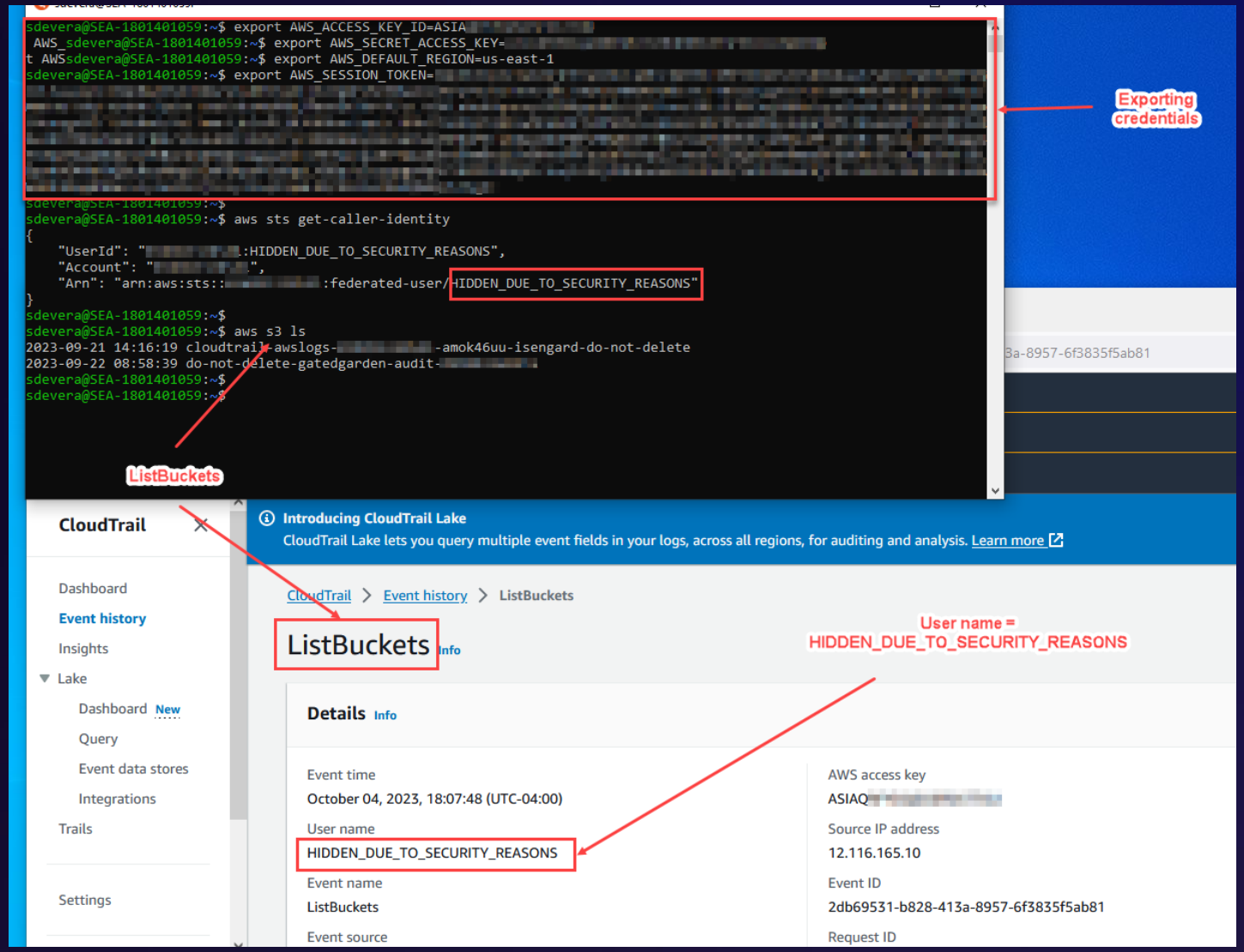
3) Threat actor exports and assumes federation token credentials

```
sdevera@SEA-1801401059:~$ export AWS_ACCESS_KEY_ID=ASIA[redacted]
 AWS_sdevera@SEA-1801401059:~$ export AWS_SECRET_ACCESS_KEY=[redacted]
t AWSsdevera@SEA-1801401059:~$ export AWS_DEFAULT_REGION=us-east-1
sdevera@SEA-1801401059:~$ export AWS_SESSION_TOKEN=[redacted]
sdevera@SEA-1801401059:~$
sdevera@SEA-1801401059:~$ aws sts get-caller-identity
{
    "UserId": "036893328541:HIDDEN_DUE_TO_SECURITY_REASONS",
    "Account": "036893328541",
    "Arn": "arn:aws:sts::036893328541:federated-user/HIDDEN_DUE_TO_SECURITY_REASONS"
}
sdevera@SEA-1801401059:~$
```

Exporting credentials

GetFederationToken: Premise

4) Use exported credentials from federation token



GetFederationToken: Premise

- The session name or ‘user name’ can be changed
- Still need to review actions by ‘masked’ user


```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "FederatedUser",
        "principalId": "036[redacted]:HIDDEN_DUE_TO_SECURITY_REASONS",
        "arn": "arn:aws:sts::036[redacted]:federated-user/HIDDEN_DUE_TO_SECURITY_REASONS",
        "accountId": "036[redacted]",
        "accessKeyId": "ASIA[redacted]",
        "sessionContext": {
            "sessionIssuer": {
                "type": "IAMUser",
                "principalId": "AIDAQRFYDQSO7HBUNWMXI",
                "arn": "arn:aws:iam::036[redacted]:user/Alice",
                "accountId": "036[redacted]",
                "userName": "Alice"
            },
            "attributes": {
                "creationDate": "2023-10-04T21:45:07Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-10-04T22:07:48Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "ListBuckets",
    "awsRegion": "us-east-1",
```


Named session

Originating principal

GetFederationToken: Mitigations

- `GetSessionToken` also used
- Generally considered unauthorized if observed
- With both `GetFederationToken` and `GetSessionToken`, you can delete the originating access key and the session will persist
- Can delete/recreate the user

GetFederationToken: Mitigations

- Apply inline policy to IAM user (deny based on `aws:TokenIssueTime`)

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

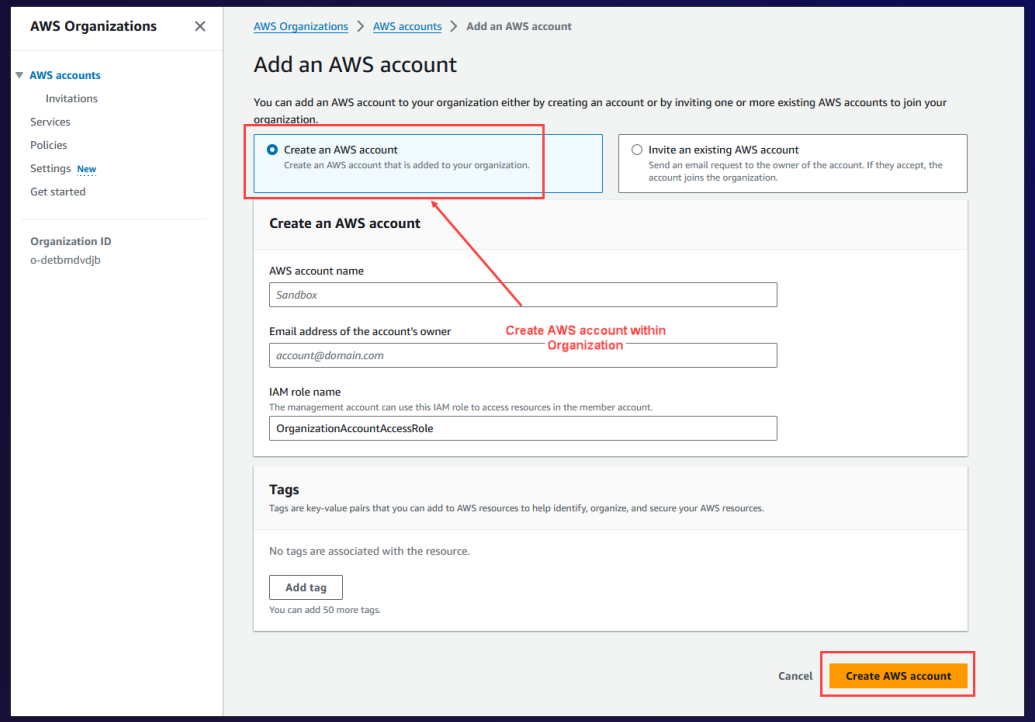
Time in ISO 8601 format based on time of GetFederationToken event

Novel threat actor tactics

Create account: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

1) Threat actor creates an account in an AWS organization

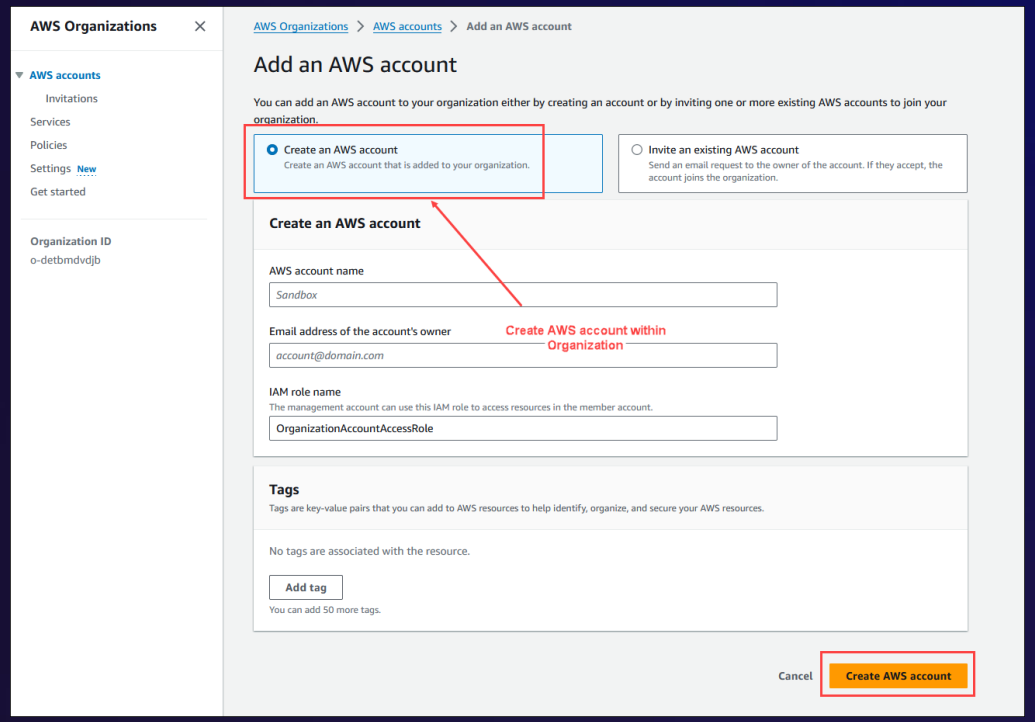


Create account: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

1) Threat actor creates an account in an AWS organization

2) Created account is used for defense evasion, resource hijacking



Create account: Alternative

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

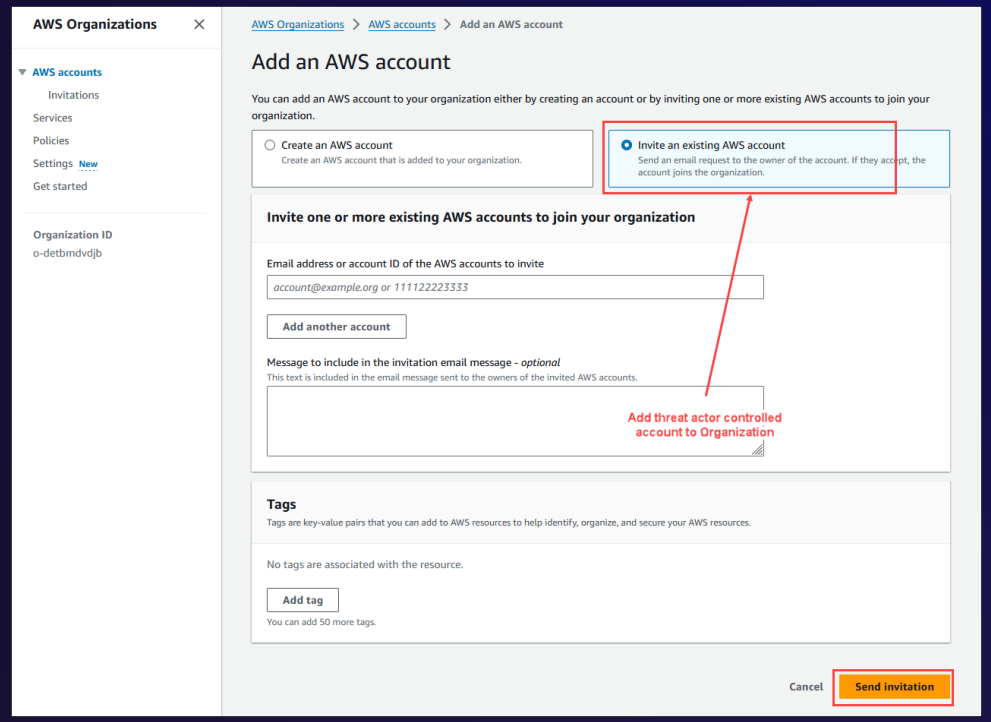
1) Threat actor creates a standalone account with a stolen credit card

Create account: Alternative

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

1) Threat actor creates a standalone account with a stolen credit card

2) Invites account to compromised AWS organization



Create account: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

1) Threat actor can remove *OrganizationAccountAccessRole*

Create account: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

1) Threat actor can remove *OrganizationAccountAccessRole*

2) Victim can apply SCPs, but this prevents new actions (existing threat actor resources not affected)

Create account: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

1) Threat actor can remove *OrganizationAccountAccessRole*

2) Victim can apply SCPs, but this prevents new actions (existing threat actor resources not affected)

3) May need support case to remove account

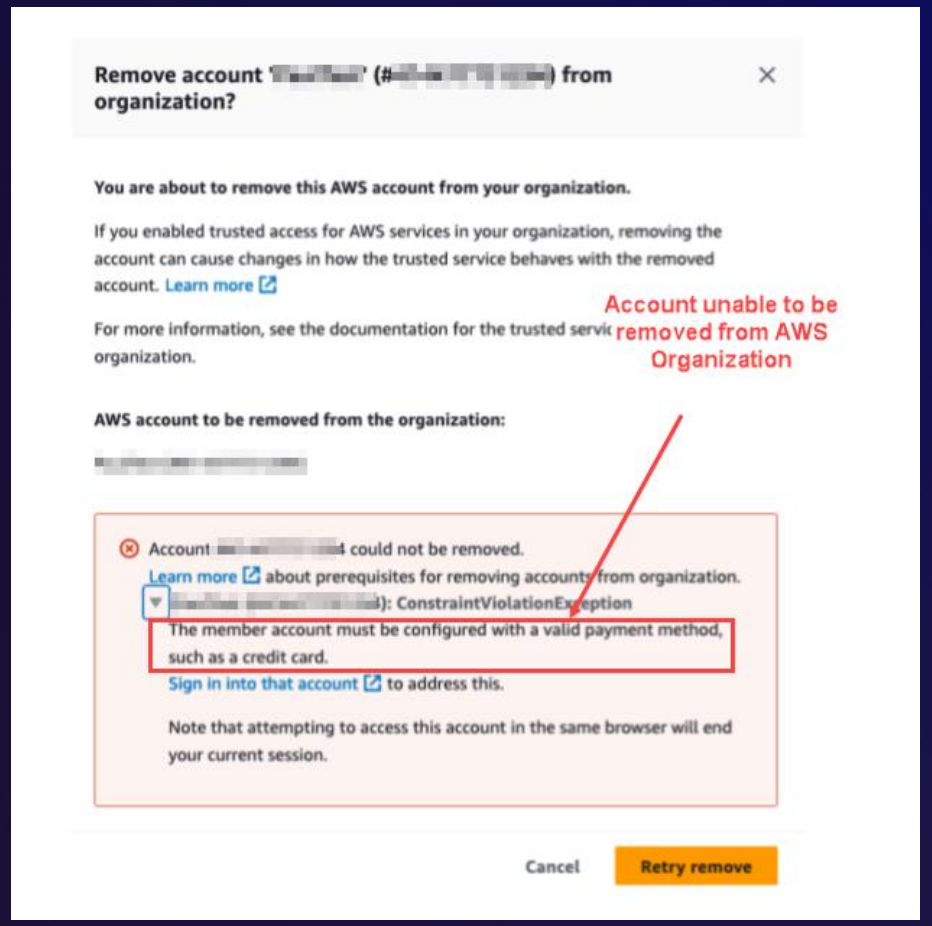
Create account: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

1) Threat actor can remove *OrganizationAccountAccessRole*

2) Victim can apply SCPs, but this prevents new actions (existing threat actor resources not affected)

3) May need support case to remove account



Create account: Mitigations

MITRE ATT&CK
Tactic: Defense evasion
Technique: Unused/unsupported cloud regions

- Create custom groups or roles
- Use principle of least privilege to restrict account creation
- Amazon CloudWatch alarm/SCP for `InviteAccountToOrganization` API call

Lifecycle deletion: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

1) Threat actor uses S3 lifecycle policies to set parameters to delete objects within 1 day

Lifecycle deletion: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

1) Threat actor uses S3 lifecycle policies to set parameters to delete objects within 1 day


```
"eventTime": "2023-10-18T10:29:40Z",
"eventSource": "s3.amazonaws.com",
"eventName": "PutBucketLifecycle",
"awsRegion": "eu-west-1",
"sourceIPAddress": "[redacted]",
"userAgent": "[aws-cli/2.13.2 Python/3.11.4 Darwin/22.6.0 source/x86_64 prompt/off command/s3api.put-bucket-lifecycle-configuration]",
"requestParameters": {
  "lifecycle": "",
  "bucketName": "[redacted]",
  "LifecycleConfiguration": {
    "xmlns": "http://s3.amazonaws.com/doc/2006-03-01/",
    "Rule": {
      "Status": "Enabled",
      "Filter": "",
      "NoncurrentVersionExpiration": {
        "NoncurrentDays": 1
      },
      "Expiration": {
        "Days": 1
      },
      "ID": "Delete_all_Oct",
      "AbortIncompleteMultipartUpload": {
        "DaysAfterInitiation": 1
      }
    }
  },
  "Host": "[redacted].s3.eu-west-1.amazonaws.com"
},
"responseElements": null,
"additionalEventData": {
  "SignatureVersion": "SigV4",
  "CipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "bytesTransferredIn": 408,
  "AuthenticationMethod": "AuthHeader",
```


Policy sets parameters to minimum values

Lifecycle deletion: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

1) Threat actor uses S3 lifecycle policies to set parameters to delete objects within 1 day

2) Form of data destruction


```
"eventTime": "2023-10-18T10:29:40Z",
"eventSource": "s3.amazonaws.com",
"eventName": "PutBucketLifecycle",
"awsRegion": "eu-west-1",
"sourceIPAddress": "[redacted]",
"userAgent": "[aws-cli/2.13.2 Python/3.11.4 Darwin/22.6.0 source/x86_64 prompt/off command/s3api.put-bucket-lifecycle-configuration]",
"requestParameters": {
  "lifecycle": "",
  "bucketName": "[redacted]",
  "LifecycleConfiguration": {
    "xmlns": "http://s3.amazonaws.com/doc/2006-03-01/",
    "Rule": {
      "Status": "Enabled",
      "Filter": "",
      "NoncurrentVersionExpiration": {
        "NoncurrentDays": 1
      },
      "Expiration": {
        "Days": 1
      },
      "ID": "Delete_all_Oct",
      "AbortIncompleteMultipartUpload": {
        "DaysAfterInitiation": 1
      }
    }
  },
  "Host": "[redacted].s3.eu-west-1.amazonaws.com"
},
"responseElements": null,
"additionalEventData": {
  "SignatureVersion": "SigV4",
  "CipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "bytesTransferredIn": 408,
  "AuthenticationMethod": "AuthHeader",
```


Policy sets parameters to minimum values

Lifecycle deletion: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

1) Threat actor uses S3 lifecycle policies to set parameters to delete objects within 1 day

2) Form of data destruction

3) Bypasses permissions and detections against `DeleteObject`


```
"eventTime": "2023-10-18T10:29:40Z",
"eventSource": "s3.amazonaws.com",
"eventName": "PutBucketLifecycle",
"awsRegion": "eu-west-1",
"sourceIPAddress": "[redacted]",
"userAgent": "[aws-cli/2.13.2 Python/3.11.4 Darwin/22.6.0 source/x86_64 prompt/off command/s3api.put-bucket-lifecycle-configuration]",
"requestParameters": {
  "lifecycle": "",
  "bucketName": "[redacted]",
  "LifecycleConfiguration": {
    "xmlns": "http://s3.amazonaws.com/doc/2006-03-01/",
    "Rule": {
      "Status": "Enabled",
      "Filter": "",
      "NoncurrentVersionExpiration": {
        "NoncurrentDays": 1
      },
      "Expiration": {
        "Days": 1
      },
      "ID": "Delete_all_Oct",
      "AbortIncompleteMultipartUpload": {
        "DaysAfterInitiation": 1
      }
    }
  },
  "Host": "[redacted].s3.eu-west-1.amazonaws.com"
},
"responseElements": null,
"additionalEventData": {
  "SignatureVersion": "SigV4",
  "CipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "bytesTransferredIn": 408,
  "AuthenticationMethod": "AuthHeader",
```


Policy sets parameters to minimum values

Lifecycle deletion: Mitigations

MITRE ATT&CK
Tactic: Impact
Technique: Data destruction

- Apply SCPs to prevent use of `PutBucketLifecycle`
- Use principle of least privilege
- AWS Config rule for `s3-lifecycle-policy-check`

SMS pumping: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

1) Threat actor obtains block of high rate SMS phone numbers from telecom provider

2) Threat actor identifies service that sends SMS text messages

3) Service used to send numerous text messages

SMS pumping: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

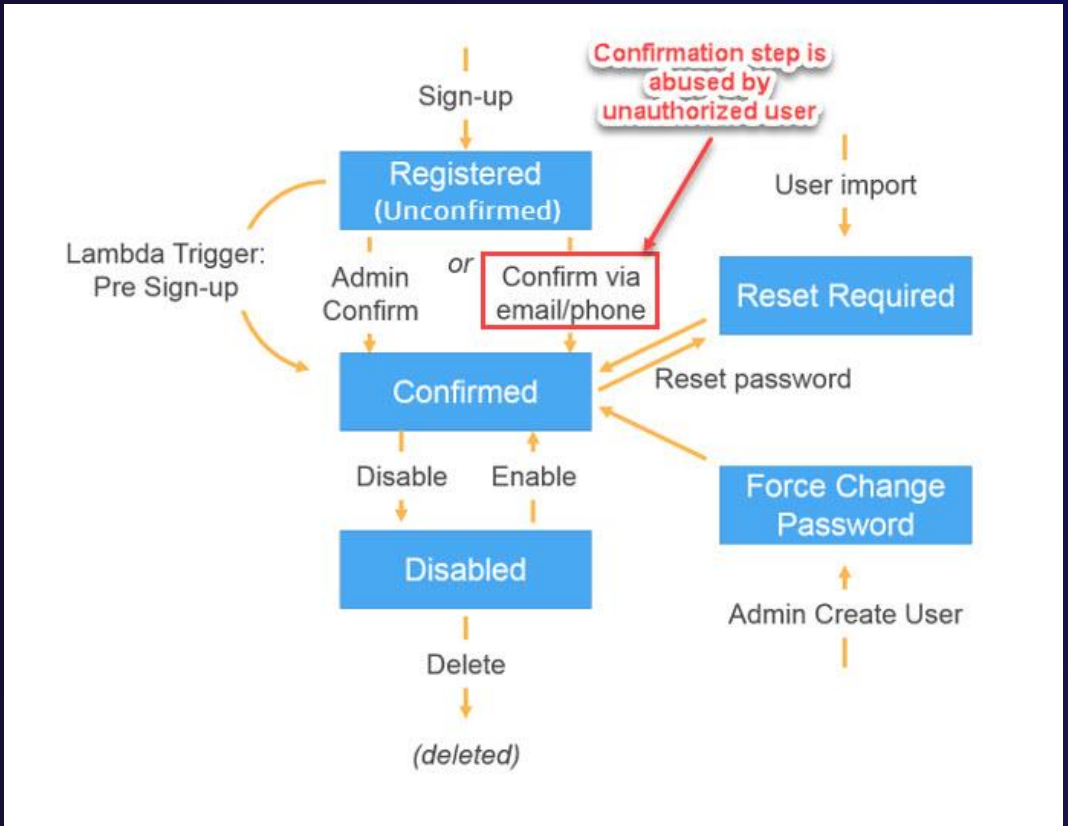
4) Amazon Cognito used

SMS pumping: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

4) Amazon Cognito used

5) APIs observed are `SignUp` or `ResendConfirmationCode`



SMS pumping: Mitigations

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

- Change attribute verification and user account confirmation
- Apply AWS WAF to present CAPTCHA
- Apply web ACL rule to inspect request body and match the SMS area code
- Amazon Fraud Detector (may require rearchitected solution)

Leave organization: Premise

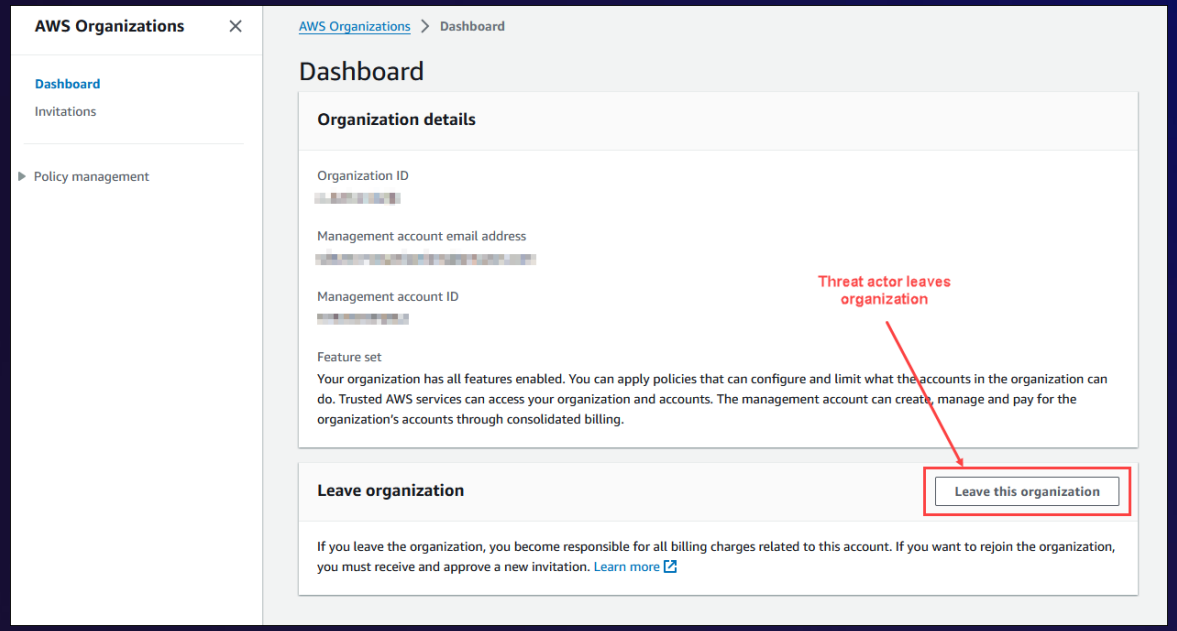
MITRE ATT&CK
Tactic: Defense evasion
Technique: Indicator removal

1) Threat actor attempts to leave an AWS organization

Leave organization: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Indicator removal

1) Threat actor attempts to leave an AWS organization

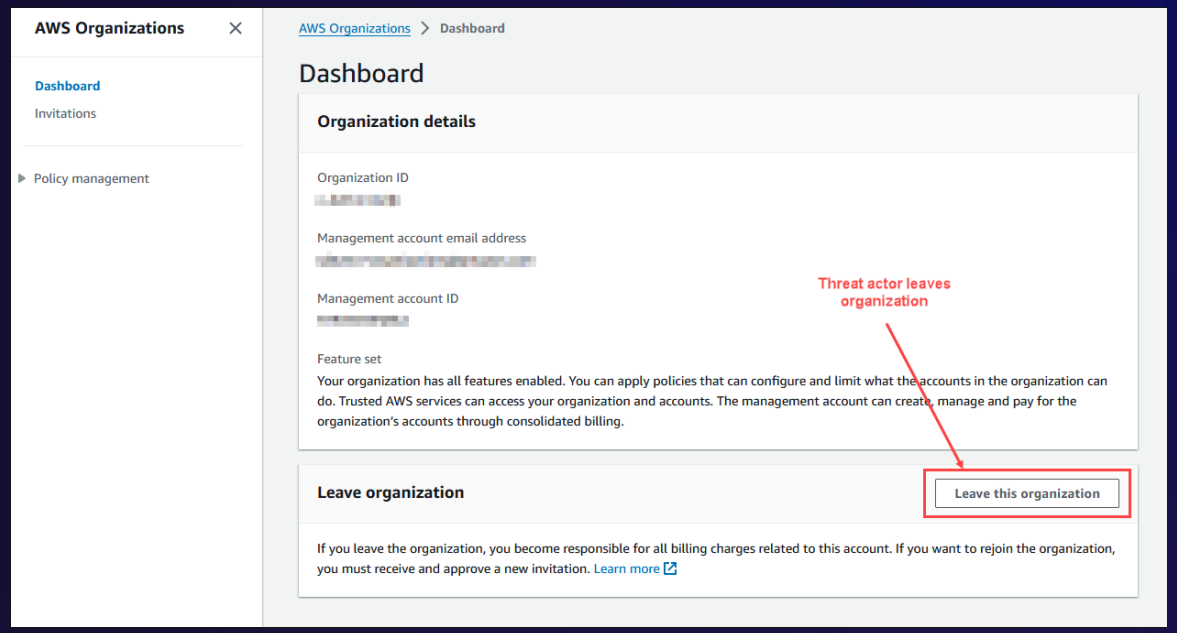


Leave organization: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Indicator removal

1) Threat actor attempts to leave an AWS organization

2) Prevents SCPs from being applied, used for resource hijacking



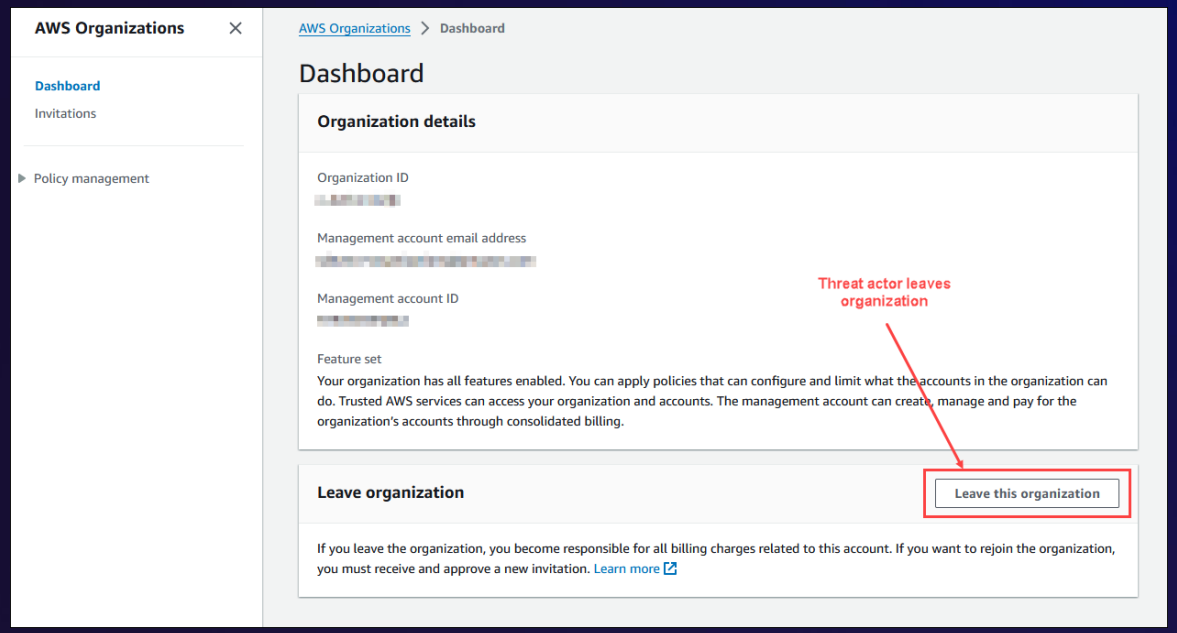
Leave organization: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Indicator removal

1) Threat actor attempts to leave an AWS organization

2) Prevents SCPs from being applied, used for resource hijacking

3) Form of defense evasion, AWS billing reports migrate



Leave organization: Mitigations

MITRE ATT&CK
Tactic: Defense evasion
Technique: Indicator removal

- Apply SCPs to prevent `LeaveOrganization` API call in member account
- Use principle of least privilege to limit use of `RemoveAccountFromOrganization` in management account

Create identity provider: Premise

1) Threat actor gains access to an AWS organization

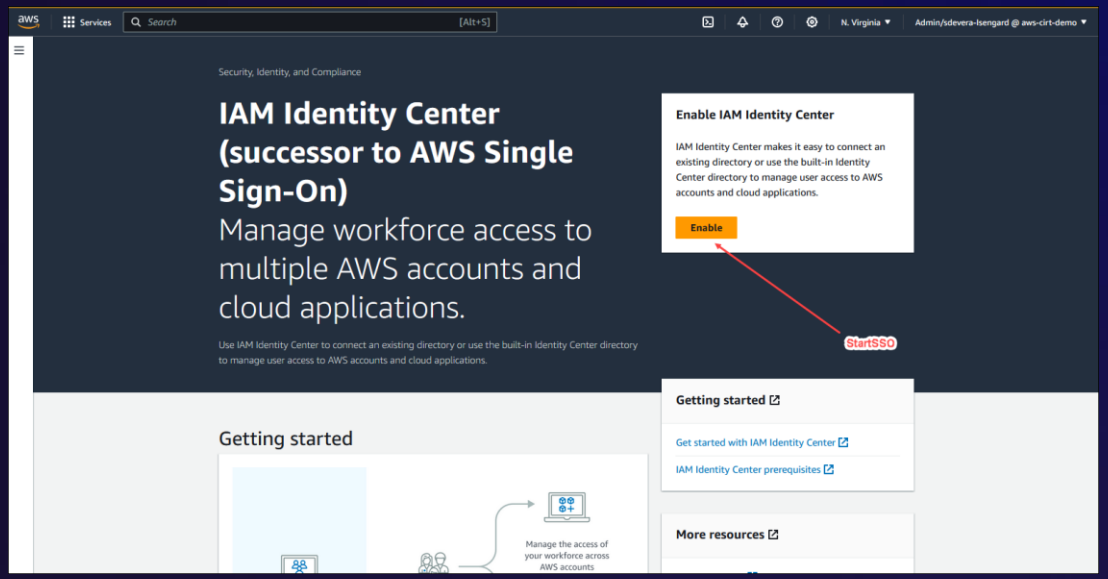
MITRE ATT&CK
Tactic: Persistence
Technique: Create account

Create identity provider: Premise

MITRE ATT&CK
Tactic: Persistence
Technique: Create account

1) Threat actor gains access to an AWS organization

2) AWS IAM Identity Center enabled to provision access to accounts



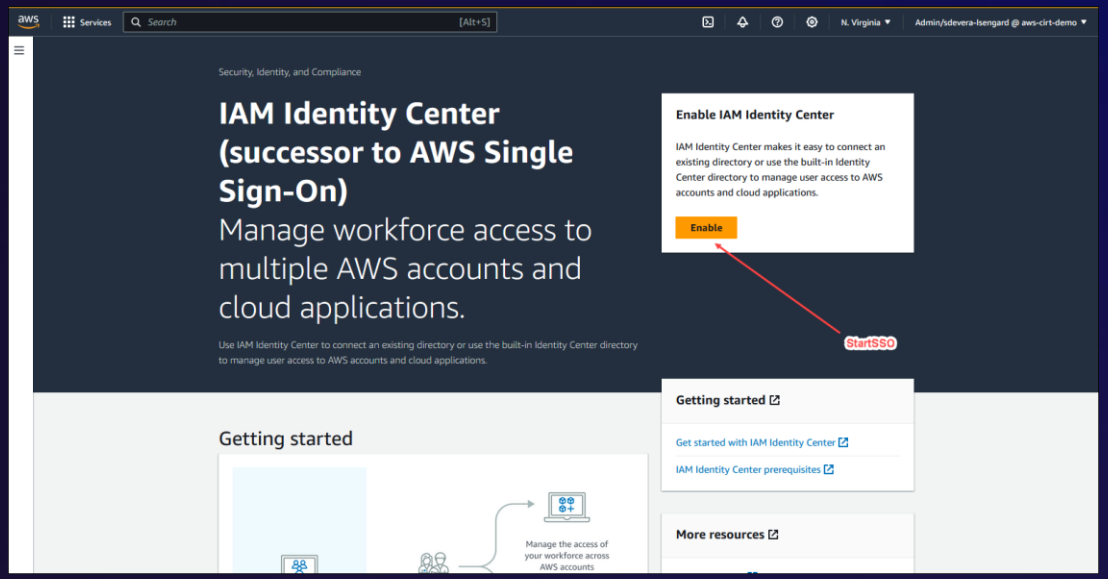
Create identity provider: Premise

MITRE ATT&CK
Tactic: Persistence
Technique: Create account

1) Threat actor gains access to an AWS organization

2) AWS IAM Identity Center enabled to provision access to accounts

3) Adds extra steps to containment



Create identity provider: Alternative

MITRE ATT&CK
Tactic: Persistence
Technique: Create account

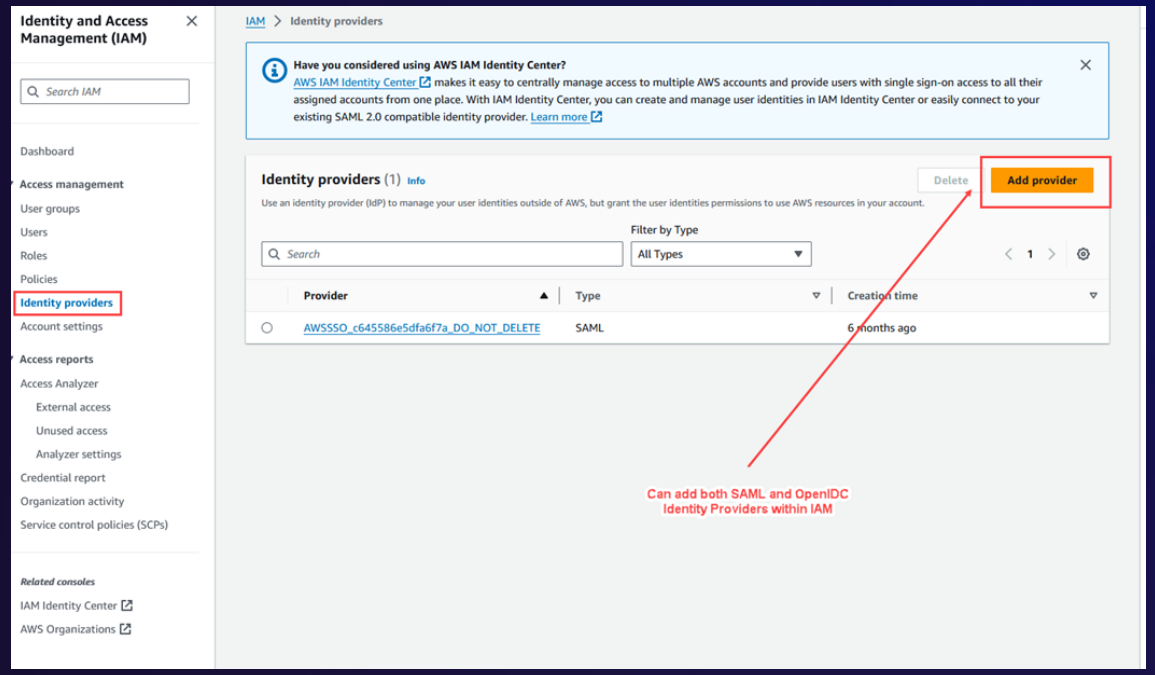
3) Access to a specific account/s within an AWS organization

Create identity provider: Alternative

MITRE ATT&CK
Tactic: Persistence
Technique: Create account

3) Access to a specific account/s within an AWS organization

4) IAM used to add a SAML or OpenIDC provider



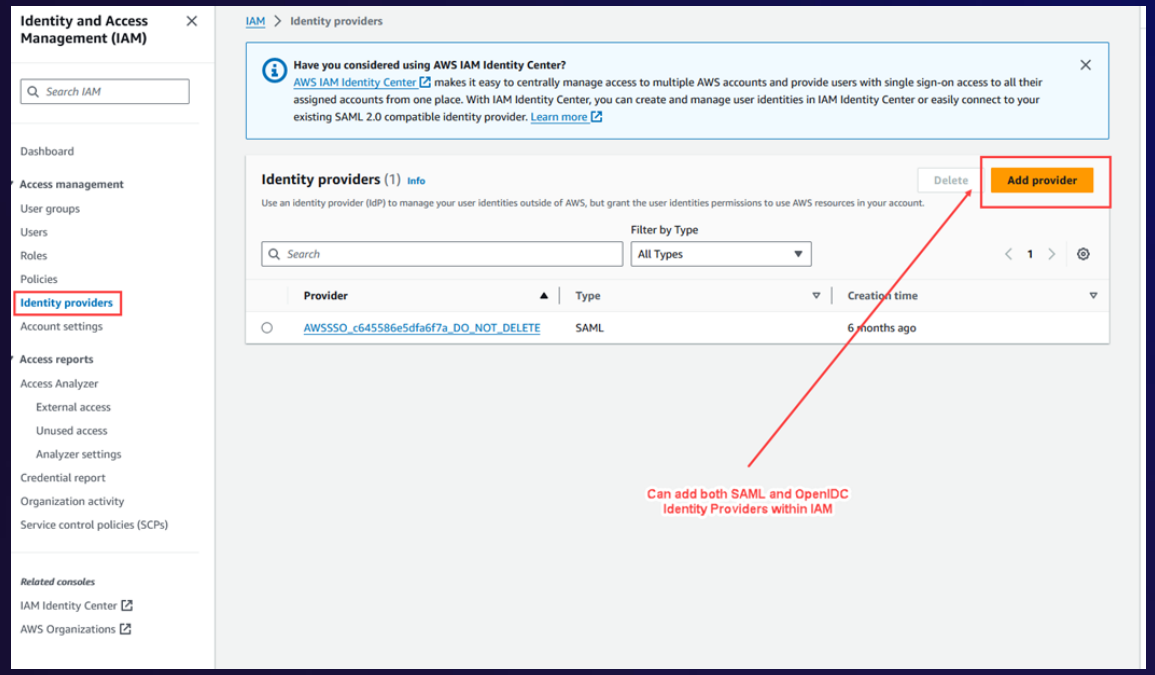
Create identity provider: Alternative

MITRE ATT&CK
Tactic: Persistence
Technique: Create account

3) Access to a specific account/s within an AWS organization

4) IAM used to add a SAML or OpenIDC provider

5) Look for `CreateSAMLProvider` or `CreateOIDCProvider` events in AWS CloudTrail



Create identity provider: Mitigations

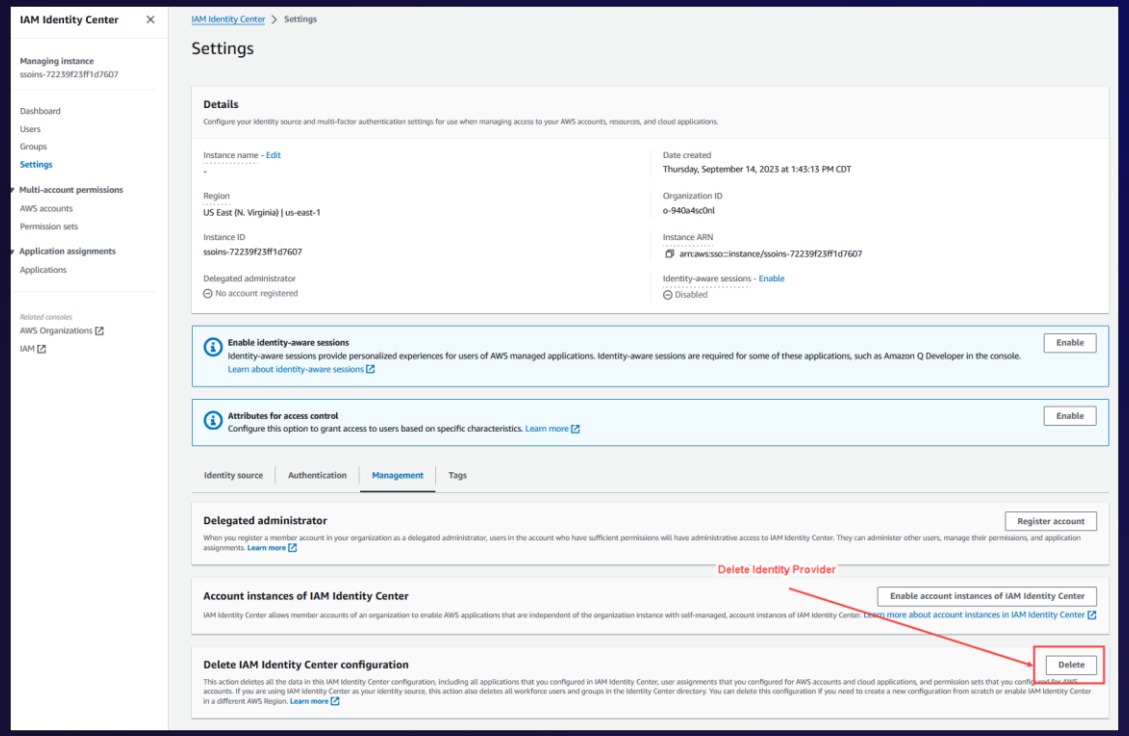
MITRE ATT&CK
Tactic: Persistence
Technique: Create account

- Remove identity provider from IAM Identity Center or IAM

Create identity provider: Mitigations

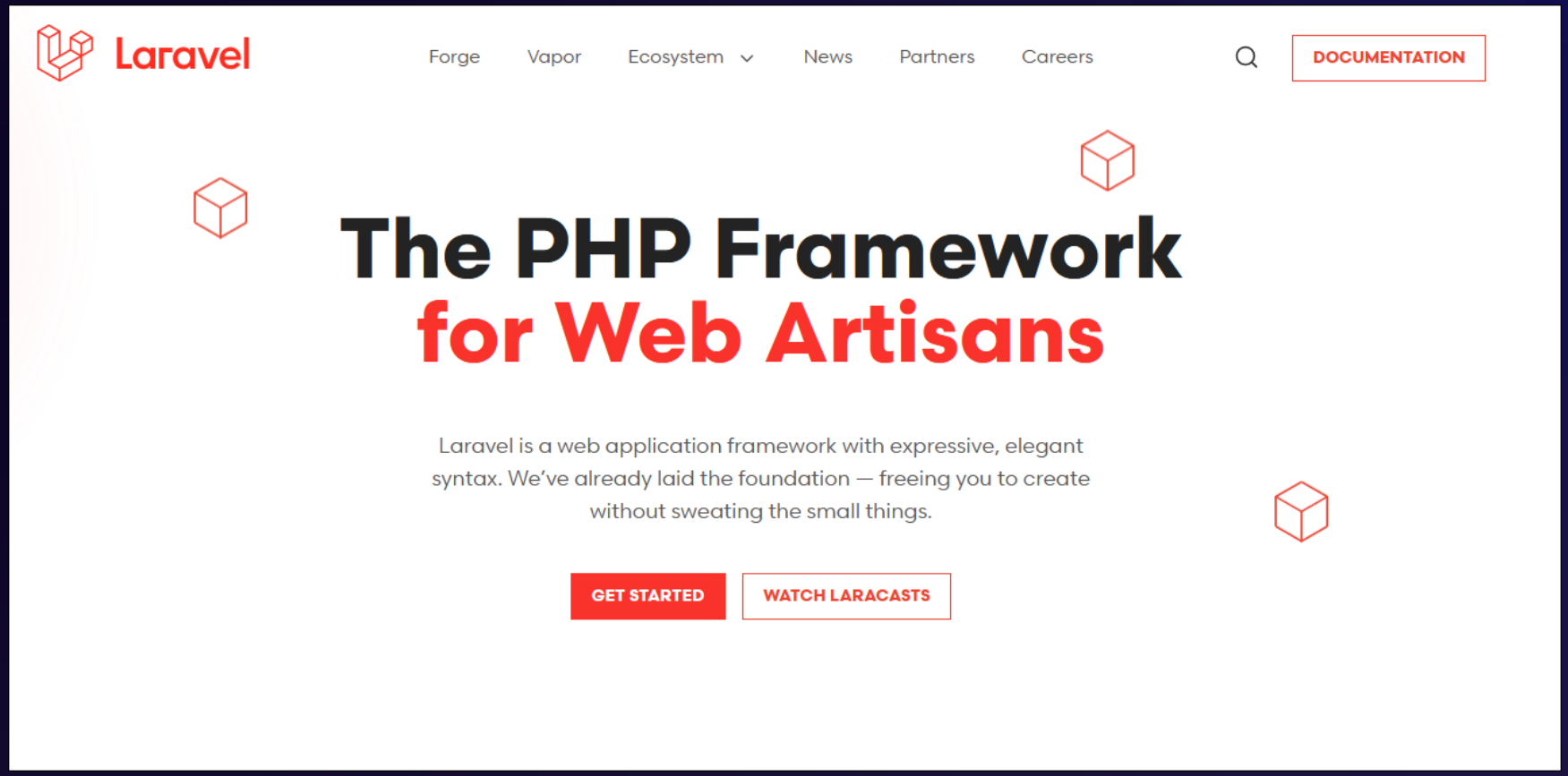
MITRE ATT&CK
Tactic: Persistence
Technique: Create account

- Remove identity provider from IAM Identity Center or IAM
- Use Amazon EventBridge to watch for `StartSSO`, `CreateSAMLProvider` or `CreateOIDCProvider` events in CloudTrail



Laravel framework access: Premise

MITRE ATT&CK
Tactic: Initial access
Technique: Exploit public-facing application



Laravel framework access: Premise

MITRE ATT&CK
Tactic: Initial access
Technique: Exploit public-facing application

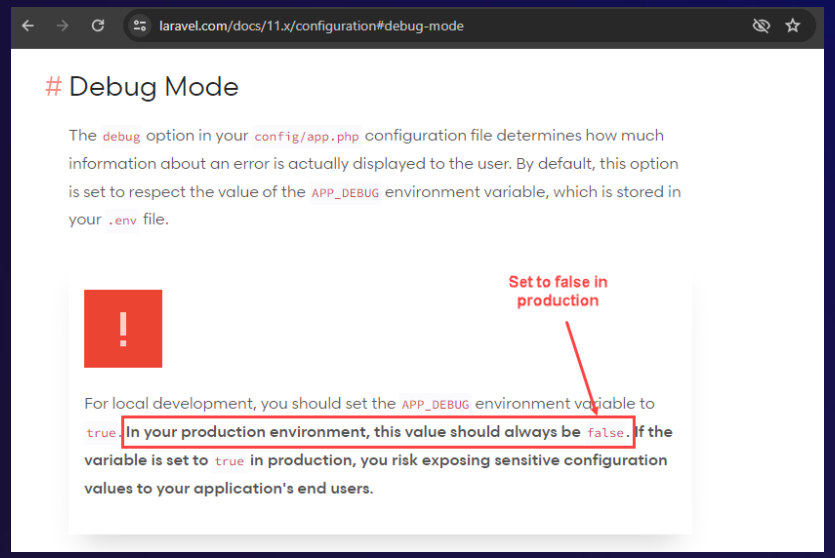
1) Threat actor identifies vulnerable version of Laravel
   - CVE-2021-3129
   - Debug mode

Laravel framework access: Premise

MITRE ATT&CK
Tactic: Initial access
Technique: Exploit public-facing application

1) Threat actor identifies vulnerable version of Laravel
   - CVE-2021-3129
   - Debug mode

2) Debug mode allows access to .env file



Laravel framework access: Premise

MITRE ATT&CK
Tactic: Initial access
Technique: Exploit public-facing application

1) Threat actor identifies vulnerable version of Laravel
   - CVE-2021-3129
   - Debug mode

2) Debug mode allows access to .env file

3) .env configured with AWS credentials

```
APP_NAME=My App
APP_ENV=local
APP_KEY=base64:[redacted]
APP_DEBUG=true
APP_URL=http://localhost

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=my_database
DB_USERNAME=root
DB_PASSWORD=password

BROADCAST_DRIVER=log
CACHE_DRIVER=file
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_MAILER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_ENCRYPTION=null
MAIL_FROM_ADDRESS=null
MAIL_FROM_NAME="${APP_NAME}"

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
PUSHER_APP_CLUSTER=mt1

MIX_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
MIX_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```

Laravel framework access: Premise

MITRE ATT&CK
Tactic: Initial access
Technique: Exploit public-facing application

3) For server in debug mode, specific data sent generates a debug file

Laravel framework access: Premise

MITRE ATT&CK
Tactic: Initial access
Technique: Exploit public-facing application

3) For server in debug mode, specific data sent generates a debug file

4) File contains .env variables including AWS credentials

```
:/tmp$ cat laravel.dump |grep -A 1 AWS_ACCESS_KEY
                <td>AWS_ACCESS_KEY_ID</td>
                <td><pre class=sf-dump id=sf-dump-639057879 data-indent-pad="  ">"<span class=sf-dump-str title="11 characters"
>AKIAEXAMPLE</span>"
--
                <td>AWS_ACCESS_KEY_ID</td>
                <td><pre class=sf-dump id=sf-dump-1669199535 data-indent-pad="  ">"<span class=sf-dump-str title="11 characters
">AKIAEXAMPLE</span>"
:/tmp$
:/tmp$ cat laravel.dump |grep -A 1 SECRET_ACCESS_KEY
                <td>AWS_SECRET_ACCESS_KEY</td>
                <td><pre class=sf-dump id=sf-dump-1009507608 data-indent-pad="  ">"<span class=sf-dump-str title="11 characters
">AKIAEXAMPLE</span>"
--
                <td>AWS_SECRET_ACCESS_KEY</td>
                <td><pre class=sf-dump id=sf-dump-874671302 data-indent-pad="  ">"<span class=sf-dump-str title="11 characters"
>AKIAEXAMPLE</span>"
/tmp$
/tmp$
/tmp$
/tmp$
/tmp$
```

Keys

Laravel framework access: Mitigations

MITRE ATT&CK
Tactic: Initial access
Technique: Exploit public-facing application

- Confirm Laravel is up-to-date and fully patched
- Disable debug mode in production – set `APP_DEBUG = FALSE`
- Use principle of least privilege for credentials in Laravel .env
- AWS Secrets Manager for hardcoded secrets

CloudTrail modification: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Impair defenses

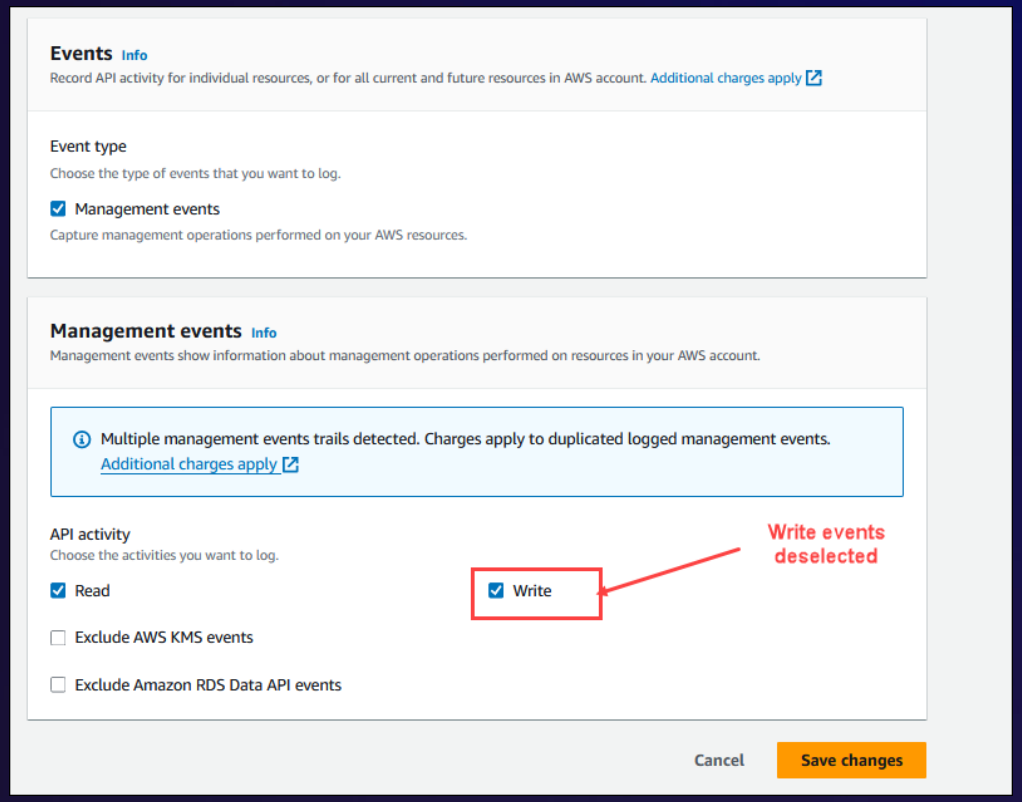
1) Threat actor gains access to AWS account

CloudTrail modification: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Impair defenses

1) Threat actor gains access to AWS account

2) Modifies CloudTrail using `PutEventSelectors`



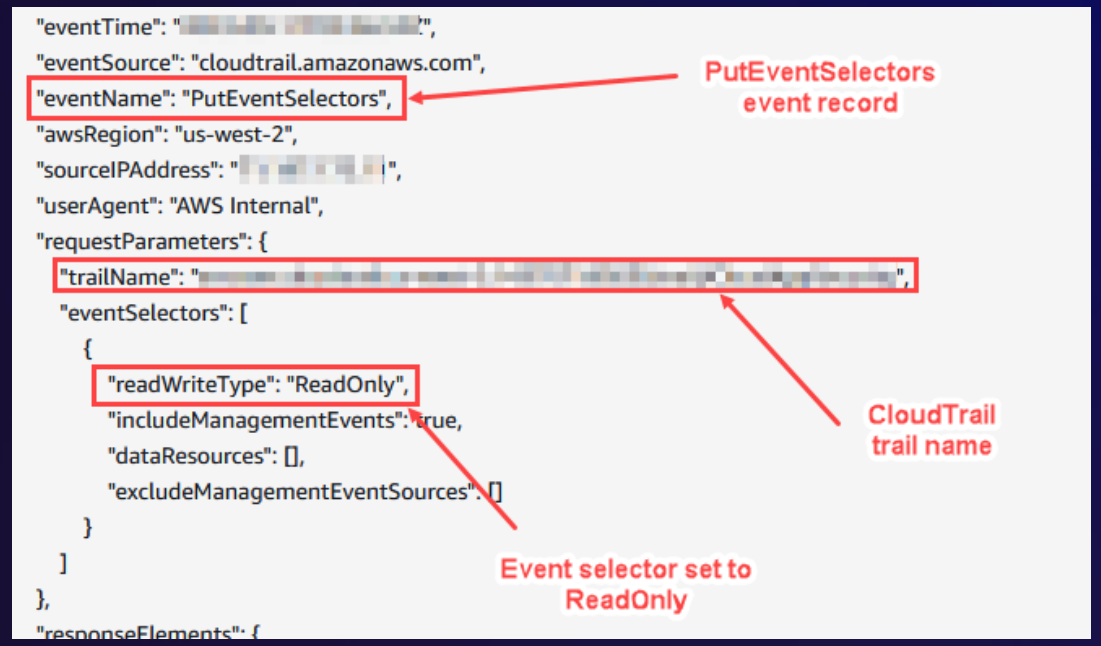
CloudTrail modification: Premise

MITRE ATT&CK
Tactic: Defense evasion
Technique: Impair defenses

1) Threat actor gains access to AWS account

2) Modifies CloudTrail using `PutEventSelectors`

3) Prevents logging of mutating events



CloudTrail modification: Alternative

MITRE ATT&CK
Tactic: Defense evasion
Technique: Impair defenses

1) Threat actor gains access to AWS account

2) Modifies CloudTrail using `PutEventSelectors`

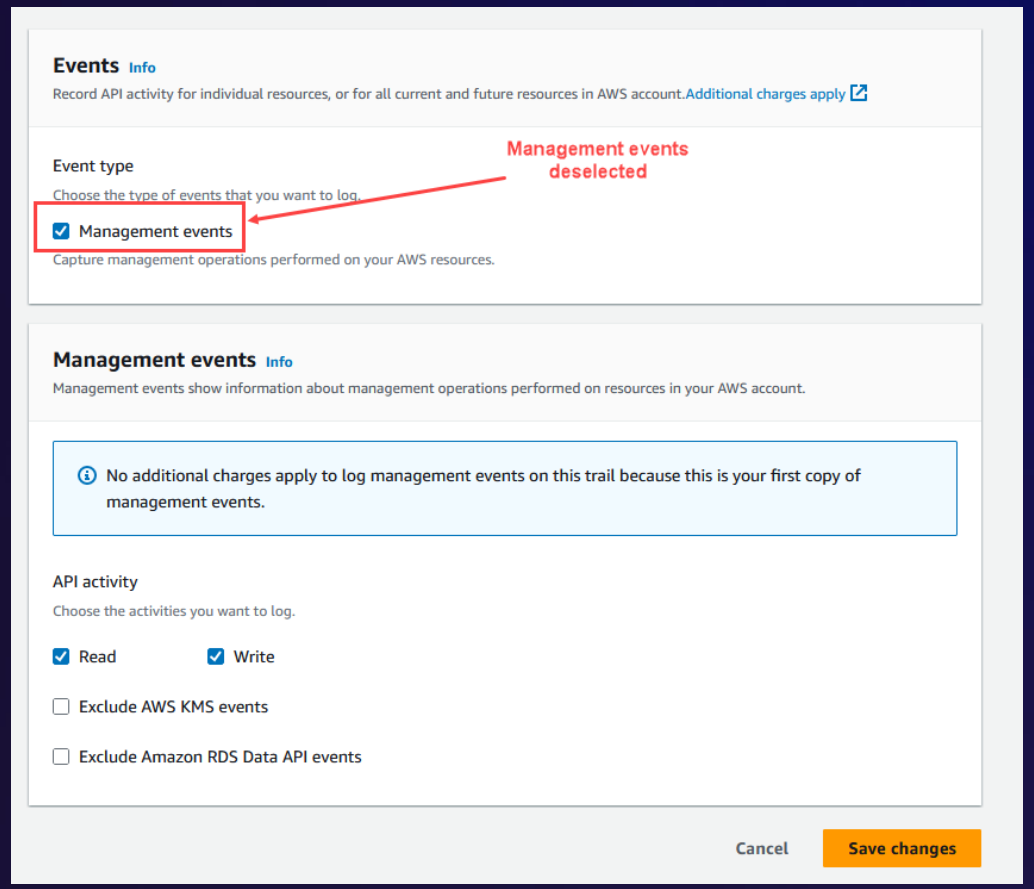
CloudTrail modification: Alternative

MITRE ATT&CK
Tactic: Defense evasion
Technique: Impair defenses

1) Threat actor gains access to AWS account

2) Modifies CloudTrail using `PutEventSelectors`

3) Prevents logging of management events



CloudTrail modification: Alternative

MITRE ATT&CK
Tactic: Defense evasion
Technique: Impair defenses

1) Threat actor gains access to AWS account

2) Modifies CloudTrail using `PutEventSelectors`

3) Prevents logging of management events


```
"eventTime": "[redacted]",
"eventSource": "cloudtrail.amazonaws.com",
"eventName": "PutEventSelectors",
"awsRegion": "us-west-2",
"sourceIPAddress": "[redacted]",
"userAgent": "AWS Internal",
"requestParameters": {
  "trailName": "[redacted]",
  "eventSelectors": [
    {
      "readWriteType": "All",
      "includeManagementEvents": false,
      "dataResources": [
        {
          "type": "AWS::S3::Object",
          "values": [
            "[redacted]",
            "[redacted]",
            "[redacted]",
            "[redacted]"
          ]
        }
```


PutEventSelectors event record

CloudTrail trail name

Event selector with includeManagementEvents to false

CloudTrail modification: Mitigations

MITRE ATT&CK
Tactic: Defense evasion
Technique: Impair defenses

- Use SCPs to restrict CloudTrail modification including use of `PutEventSelectors` API
- Consider AWS Config remediation rules for CloudTrail

LLM resource hijacking: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

1) Threat actor obtains access to AWS account

LLM resource hijacking: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

1) Threat actor obtains access to AWS account

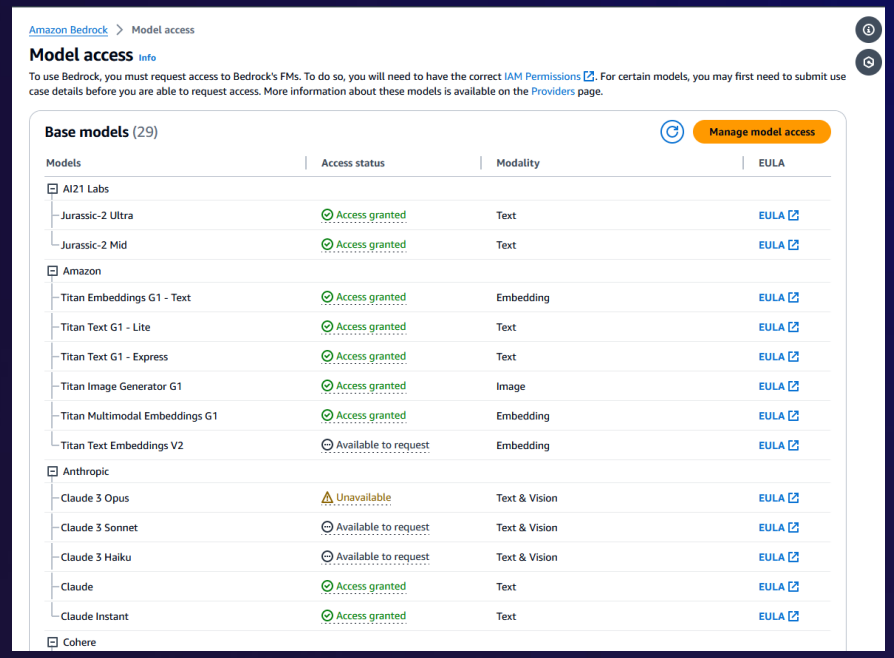
2) Threat actor enables access to LLMs through Amazon Bedrock

LLM resource hijacking: Premise

MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

1) Threat actor obtains access to AWS account

2) Threat actor enables access to LLMs through Amazon Bedrock



LLM resource hijacking: Premise

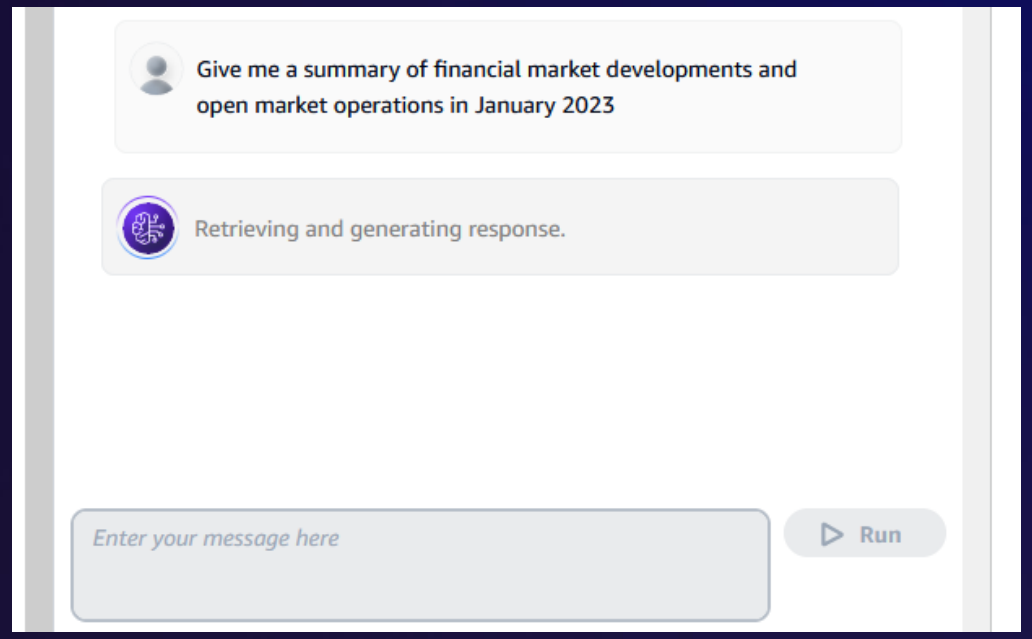
MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

1) Threat actor obtains access to AWS account

2) Threat actor enables access to LLMs through Amazon Bedrock

3) Models used and prompts sent:

- `InvokeModel`
- `InvokeModelWithResponseStream`



LLM resource hijacking: Premise

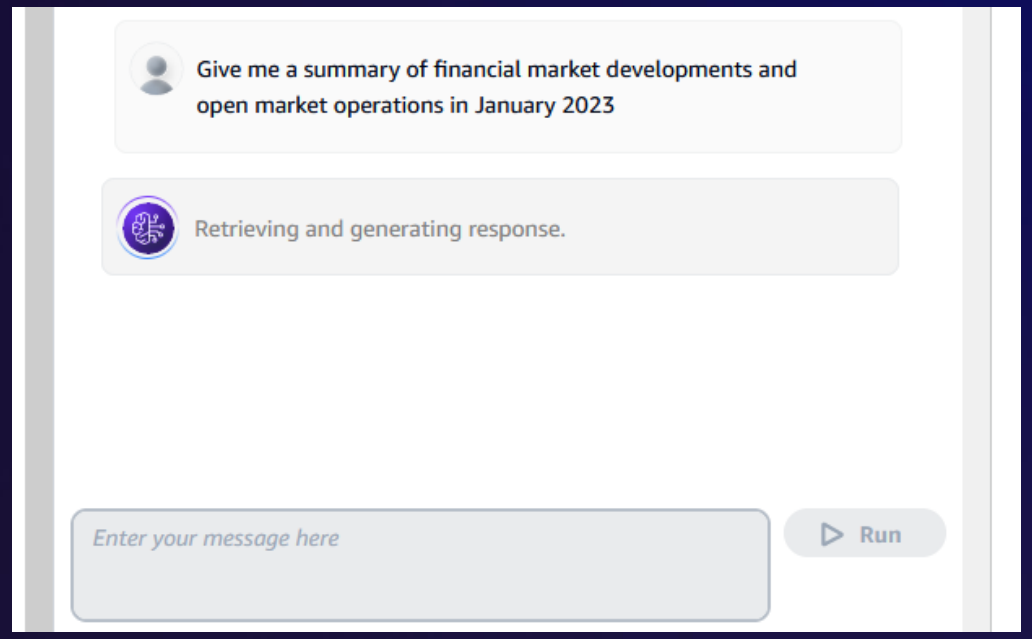
MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

1) Threat actor obtains access to AWS account

2) Threat actor enables access to LLMs through Amazon Bedrock

3) Models used and prompts sent:
   - `InvokeModel`
   - `InvokeModelWithResponseStream`

4) Can be performed in unused AWS Regions



LLM resource hijacking: Mitigations

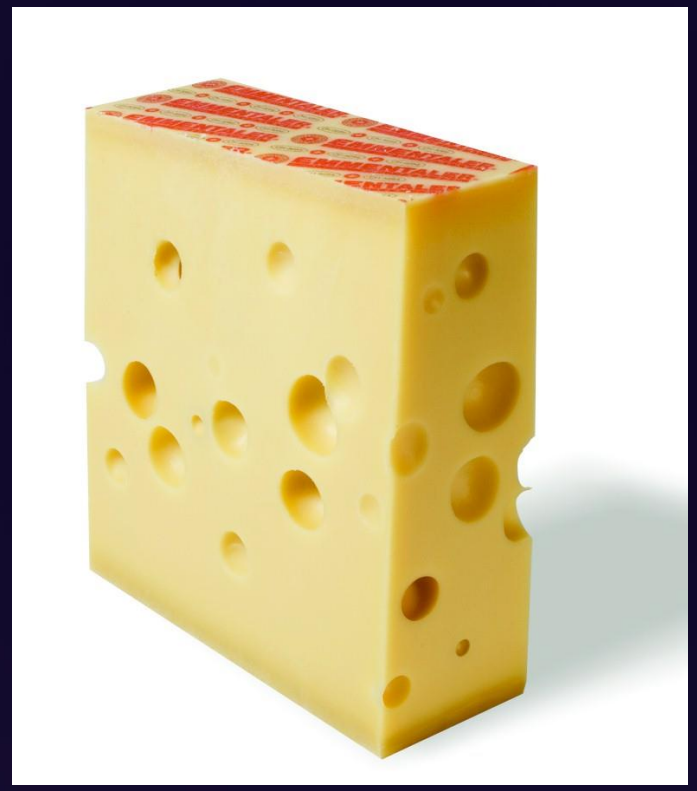
MITRE ATT&CK
Tactic: Impact
Technique: Resource hijacking

Use SCPs to limit access to Amazon Bedrock using

- Specific principals
- Specific Regions

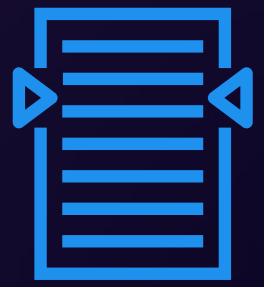
Security best practices

Swiss cheese model (industrial accidents)



A failure **cannot** be traced back to a **single root cause**;
accidents are often the **result of a combination of factors**

Get the basics right



Inaccurate AWS account contact information

Unintended disclosure of credentials and secrets

Ineffective response to detective controls

Lack of continuous vulnerability management

Insecure AWS resource configuration