



## **Attack technique report**

**Demo**

# AWS EC2 (AttachVolume, DetachVolume) Take Over

## Technique description

An attacker who compromises an AWS identity that has the right permissions can detach a volume from an EC2 instance, attach it to the attacker's instance, add new SSH keys to the instance, then reattach it to the first instance, and in this way take control of the EC2 instance.

### MITRE technique alignment

[T1078](#) , [T1078.004](#)

## AWS EC2 (AttachVolume, DetachVolume) Take Over

An attacker with a stolen AWS Identity that possesses the required permissions 'ec2:RunInstances', 'ec2:DetachVolume', 'ec2:AttachVolume', 'ec2:StopInstances', and 'ec2:StartInstances' can take control of an EC2 instance.

Steps taken by an attacker are:

- Stop execution of the target instance
- Detach the volume and attach it to a new running instance under the attacker's control, and mount it
- Add new SSH keys to the instance
- Stop the instance, detach the volume, and re-attach it to the original instance
- Log into the instance using the new SSH keys

...

## Remediations



Limit principal permissions for reading and writing EC2 instance volumes.



Protect the users or roles that can manage EC2 instances.



Use boundary-set to reduce the permissions users/roles have.

# Limit principal permissions for reading and writing EC2 instance volumes.

Remediation (1 of 3)

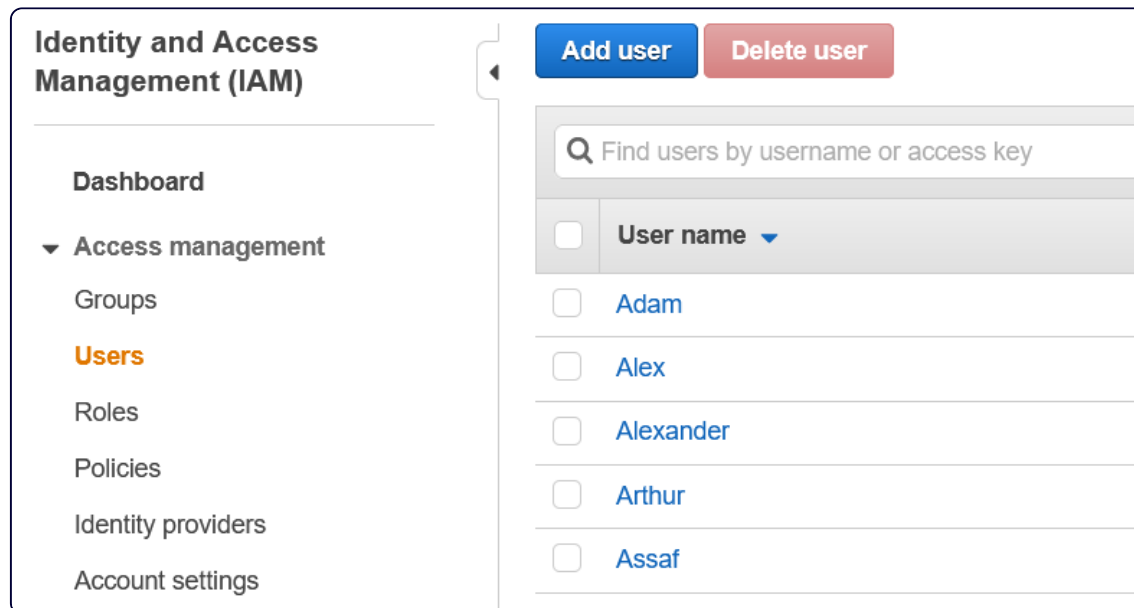
## Access Controls on AWS

On AWS, access is mainly controlled by policies, either at the IAM level or at the resource level.

### Identity-Based Policies

IAM is a service that provides granular access controls to AWS resources.

You can view all of the users in an account from the IAM panel.



User profiles can be inspected to view attached policies, and the permission associated with each policy.

▼ Permissions policies (2 policies applied)

Add permissions

Policy name ▼

Attached directly

▼ ListBucketsPolicy

Policy summary

{ } JSON

Edit policy

Q Filter

Service ▼

Access level

Resource

Allow (1 of 238 services) Show remaining 237

S3

Limited: List

All resources

► listMyOwnBucketPolicy

## Groups

Permissions may be granted to many users at the same time using groups. Groups are a convenient way to grant permissions based on job functions or team membership.

## Identity and Access Management (IAM)

Dashboard

▼ Access management

Groups

Users

Roles

Policies

Create New Group

Group Actions ▼

XM Cyber

☐

Group Name ↕

☐

XM Cyber\_Group

IAM > Groups > XM Cyber\_Group

▼ Summary

Group ARN:

arn:aws:iam: [redacted] :group/XM Cyber\_Group

Users (in this group):

4

Path:

/

Creation Time:

2020-10-18 10:00 UTC+0300

Users

Permissions

Access Advisor

This view shows all users in this group: 4 Users

User	Actions
<div><div></div>Noah</div>	<a href="#">Remove User from Group</a>
<div><div></div>Isabella</div>	<a href="#">Remove User from Group</a>
<div><div></div>Adam</div>	<a href="#">Remove User from Group</a>
<div><div></div>Inbar</div>	<a href="#">Remove User from Group</a>

IAM > Groups > XM Cyber\_Group

▼ Summary

Group ARN:

arn:aws:iam: [redacted] :group/XM Cyber\_Group

Users (in this group):

4

Path:

/

Creation Time:

2020-10-18 10:00 UTC+0300

Users

Permissions

Access Advisor

Managed Policies

The following managed policies are attached to this group. You can attach up to 10 managed policies.


Attach Policy

Policy Name	Actions
<div><div></div>SecurityAudit</div>	<a href="#">Show Policy</a>   <a href="#">Detach Policy</a>   <a href="#">Simulate Policy</a>

Group membership is also listed in every user's IAM page.

Users > Adam

## Summary

**User ARN**    arn:aws:iam::[redacted]:user/Adam 

**Path**    /

**Creation time**    2020-01-21 14:46 UTC+0300

Permissions    **Groups (1)**    Tags    Security credentials    Access Advisor

[Add user to groups](#)

Group name ▾	Attached permissions
<a href="#">XMCyber_Group</a>	<a href="#">SecurityAudit</a> and <a href="#">CloudWatchLogsFullAccess</a>

Additionally, IAM provides roles. A role is a way to grant permissions to many different AWS entities:

- An IAM user from another account or from the same account
- Application code running in another AWS service
- An AWS service that needs to perform actions on your behalf
- Security Assertion Markup Language (SAML) Users

Roles include associated policies, like users. Moreover, roles also feature an additional (innate) Trust Policy that specifies which users or services can assume which roles.

Permissions    **Trust relationships**    Tags    Access Advisor    Revoke sessions

You can view the trusted entities that can assume the role and the access conditions for the role. [Show policy document](#)

[Edit trust relationship](#)

**Trusted entities**

The following trusted entities can assume this role.

Trusted entities
arn:aws:iam::[redacted]:user/Leo

## Resource-Based Policies

Resource based policies are policies that are attached to a specific resource, like an S3 bucket or an ECR repository. Resource-based policies can be used, coupled with an appropriate identity-based policy on the other account, to allow a principal from another account to perform actions on resources in the hosting account.

### Example - ECR

Navigate to the service page and open a specific repository, then click "Permissions", and create a new statement.

Statement name

new statement

Effect

Specifies whether the statement results in an allow or an explicit deny.

☒ Allow
☐ Deny

Principal ( ☐ use NotPrincipal )

The entities (AWS service, IAM user, role, group, AWS account ID, or Everyone) you want the statement to apply to. For more information, see [Principal](#).

☐ Everyone (\*)

Service principal - *optional*

The service principal to apply the statement to.

Comma delimited list

AWS account IDs - *optional*

The AWS account(s) to apply the statement to. All users under the AWS account will be affected.

Comma delimited list

IAM entities (109)

Find entities

< 1 2 3 4 5 6 7 ... 11 >

<input type="checkbox"/>	Name	▼	Path
<input type="checkbox"/>	AdminRole		/


From here, you can deny or allow AWS entities access to the repository. Note that according to the AWS policy evaluation logic, if you deny access to a certain entity here, other relevant "Allow" policies will not take effect.

Additional information about resource-based policies for each service is available on the associated documentation page.

## Simulating Policies

The [IAM Policy Simulator](#) can be used to simulate policies and quickly determine which policies grant which permissions.

First, a user is selected and their policies, whether they were attached directly or via a group, are presented.



# IAM Policy Simulator

Policies


Back


Create New Policy


Selected user: Adam

IAM Policies

Filter

☒  AmazonS3ReadOnlyAccess

☒  listMyOwnBucketPolicy

☒  AmazonECS\_FullAccess

Then, an action can be simulated. In this example,

Adam

can list an S3 bucket because he has the

AmazonS3ReadOnlyAccess

policy, and the selected bucket has an appropriate resource policy.

Amazon S3

1 Action(s) sele...

Select All

Deselect All


Reset Contexts


Clear Results

Run Simulation

Global Settings ⓘ ⚠

Action Settings and Results [1 actions selected. 0 actions not simulated. 1 actions allowed. 0 actions denied. ]


Service	Action	Resource Type	Simulation Resource	Permission
Amazon S3	ListBucket	bucket	bucket	 <b>allowed</b> 2 matching statements.

[Show statement](#) in AmazonS3ReadOnlyAccess (IAM Policy)
 [Show statement](#) in  (Resource Policy)

Resource You can specify the resource and context keys used to simulate this action. By default the simulation resource is "".

bucket

arn:aws:s3:::



s3:prefix

/home/Adam

☒ Include Resource Policy

## References

- [IAM Documentation - Access Management](#)
- [IAM Documentation - IAM groups](#)
- [IAM Documentation - Testing IAM policies with the IAM policy simulator](#)



# Protect the users or roles that can manage EC2 instances.

Remediation (2 of 3)

## Protecting Users on AWS

There are many ways to protect users on AWS:

- Restrict access to AWS root account.
- Create individual IAM user accounts.
- Use groups to assign permissions to IAM users.
- Grant least privilege.
- Use permissions with AWS managed policies.
- Use customer managed policies instead of inline policies.
- Use access levels to review IAM permissions.
- Configure and enforce a strong user password policy.
- Enable MFA (Multi-Factor Authentication).
- Use roles for applications that run on Amazon EC2 instances.
- Use roles to delegate permissions.
- Do not share access keys.
- Rotate credentials regularly.
- Remove unnecessary credentials.
- Use policy conditions for extra security.
- Monitor activity in your AWS account.

## References

- [AWS - Security Best Practices in IAM](#)

...

# Use boundary-set to reduce the permissions users/roles have.

Remediation (3 of 3)

## AWS Boundary Set

AWS supports permissions boundaries for IAM entities (users or roles). A permissions boundary is an advanced feature for using a managed policy to set the maximum permissions that an identity-based policy can grant to an IAM entity. An entity's permissions boundary allows it to perform only the actions that are allowed by both its identity-based policies and its permissions boundaries.

You can use an AWS managed policy or a custom managed policy to set the boundary for an IAM entity (user or role). That policy limits the maximum permissions for the user or role.

For example, assume that the IAM user named ShirleyRodriguez should be allowed to manage only Amazon S3, Amazon CloudWatch, and Amazon EC2. To enforce this rule, you can use the following policy to set the permissions boundary for the ShirleyRodriguez user:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:*",
        "cloudwatch:*",
        "ec2:*"
      ],
      "Resource": "*"
    }
  ]
}
```

When you use a policy to set the permissions boundary for a user, it limits the user's permissions but does not provide permissions on its own. In this example, the policy sets the maximum permissions of ShirleyRodriguez as all operations in Amazon S3, CloudWatch, and Amazon EC2. Shirley can never perform operations in any other service, including IAM, even if she has a permissions policy that allows it. For example, you can add the following policy to the ShirleyRodriguez user:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

This policy allows creating a user in IAM. If you attach this permissions policy to the ShirleyRodriguez user, and Shirley tries to create a user, the operation fails because the permissions boundary does not allow the iam:CreateUser operation.

## References

[Permissions boundaries for IAM entities](#)