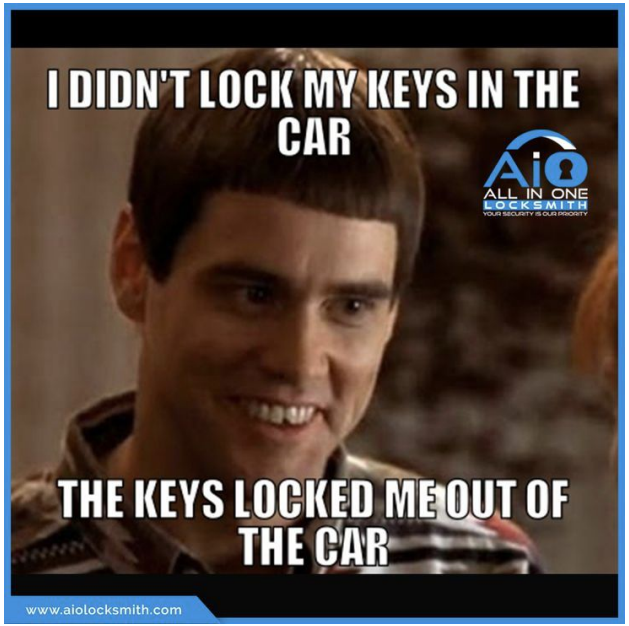
Women Techmakers

When it comes to SECRETS, how secure is your application?

- Sakshi Nasha

How much security is too much security for you ?



Common Ways

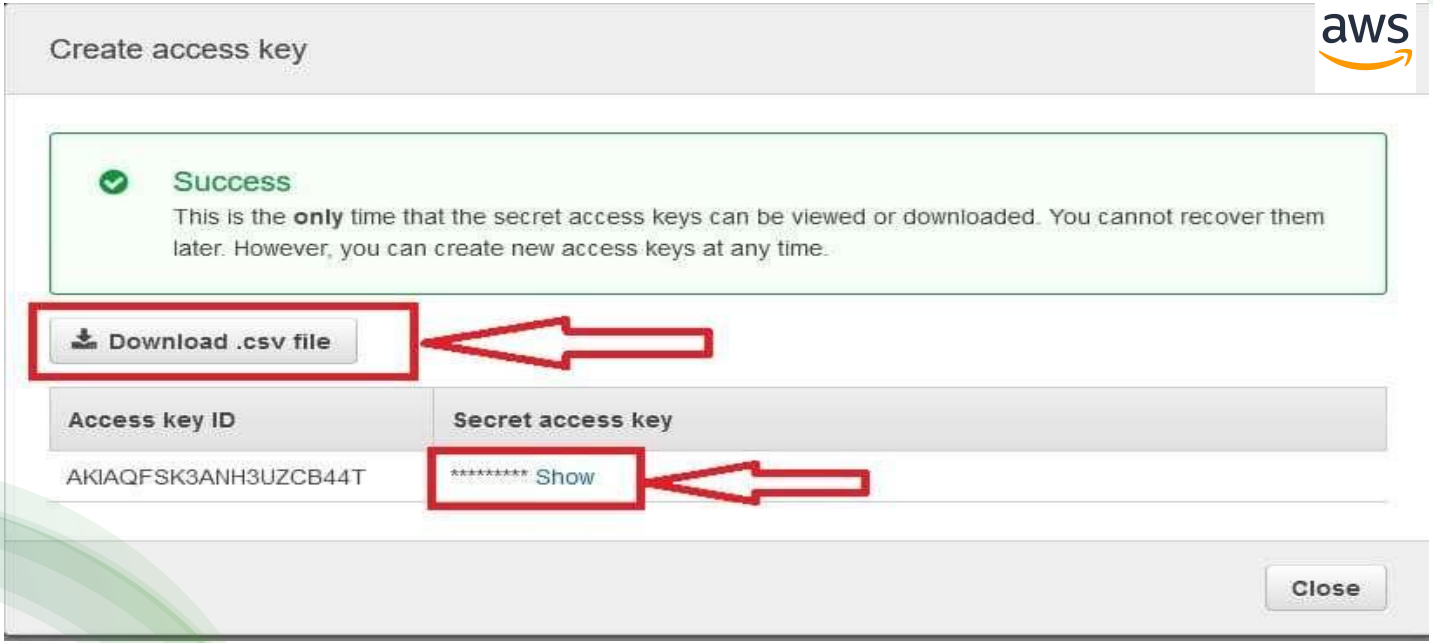
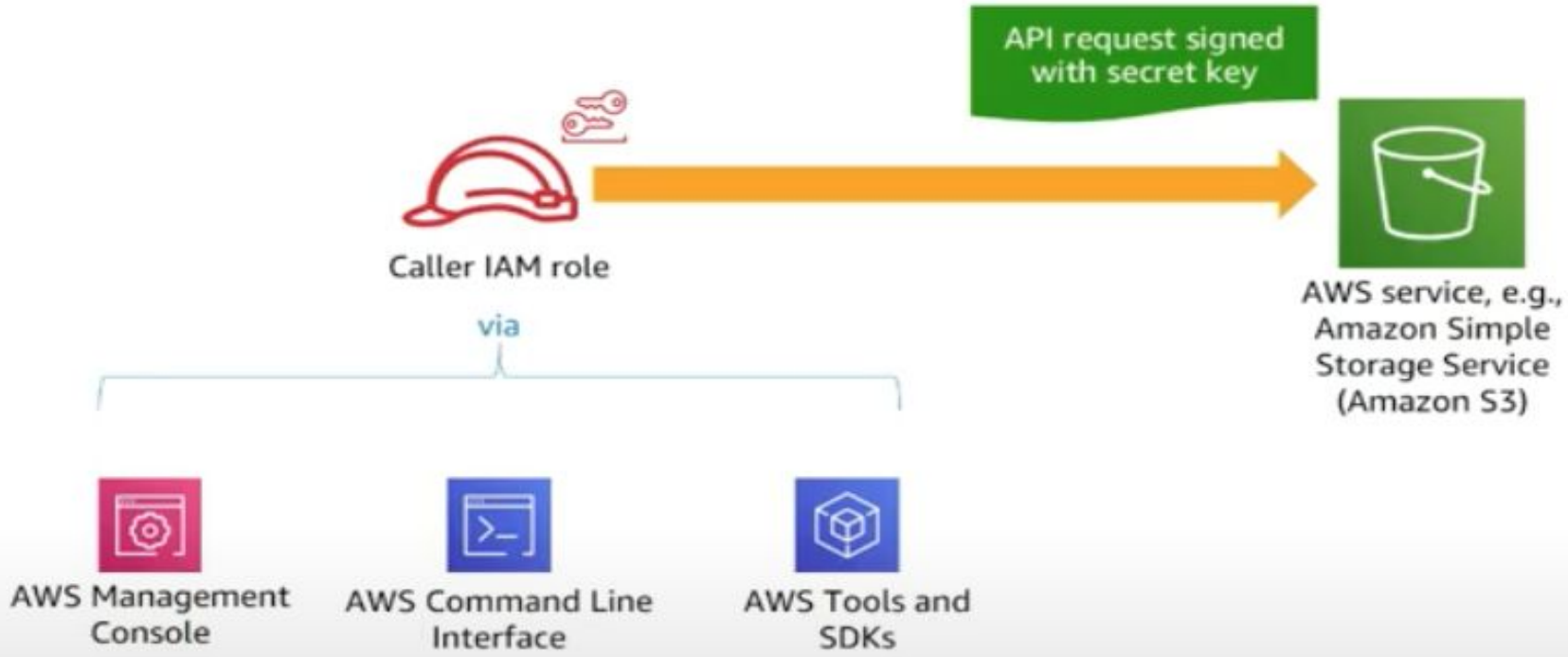


Diagram1: Creating a Credential (Access key and secret) with the AWS S3 bucket (IAM ROLE)

How an authentication works in AWS



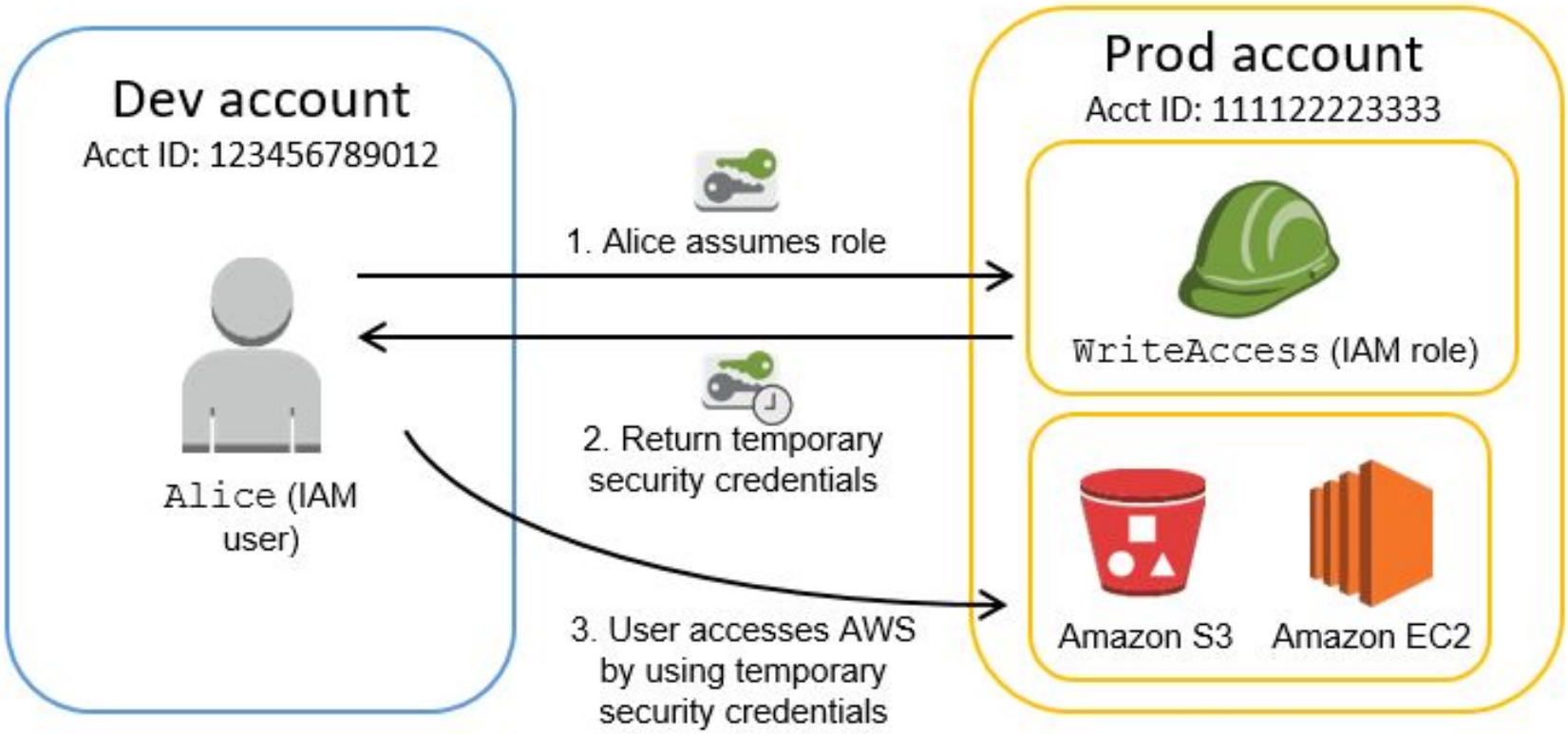
Disadvantages of using Standard Credentials in your application

- These standard credentials are **long lived** in nature.
- If compromised, they give attackers **ample time** to exploit the application.
- If they are **stolen** it would be a nightmare to discern which operations are legitimate.
- Thus, the only **fail-safe choice** is to cumbersomely **rotate the keys** and **redistribute to customers**. This is often overlooked action and adds extra pain for the DevOps.

Introducing short lived token-based authentication.

- Short lived tokens are **JWT** (JSON Web Token) with **session time**
- As the name implies “**dynamic means changing**” ie: they have short term validity.
- In technical sense, they are generated on-demand and automatically revoked after a specified time period called expiration time.
- After these tokens expire, AWS no longer recognizes them and hence revokes any kind of access from API requests made with them.
- You can create these tokens using AWS Security Token Service (STS)
- While creating the tokens, you can specify the duration : How long the STS is valid.
- No pain of manual key rotation.
- After they expire, they cannot be reused.
- You can specify how long the credentials are valid, up to a maximum limit.
- You entirely own the token from controlling the expiration to revocation. Thats the real beauty!

USE CASE: Granting Temporary Access to AWS Production Resources Using Cross-Account Roles



```
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAR52LBBISYNJOBJ7O",
            "sessionToken":
"IQoJb3JpZ2luX2VjEHsaCXVzLWVhc3QtMSJIMEYCIQD3qSmqdpaSQAdJWJH/Yg0WuSkkvVxqlFZuzuG3j8uQ3QIhAKusdUmLkAQbOLnRJmN6DMJEmiT4OK2aKZ8AqH3+g5fNK
uwCCKT//////////wEQARoMMTMyNzY0NDY5Nzk3Igx0D61kLHtcI7YLvDQqwALQxxRBHheYyehMRlZ4kPnk2YaZIHRDsF/O1snPqq4ExmGs7+ABKaduT3aF/jjSWUyV4iQTk9h
Tbn4m3iB4oScSohSSUkmuk0qXm/4lUUQ/6rEkY3gou/h2RnCtP/q27npTYzpgrl/QE+RYqd+fOn+/T4e/EY5drPHrw3VsLVn2MSGU3vXh01uAgm1XoKzH3bkQqeAWU26A4B33z
i8PEI1CIrOUHNHE5XpCBAhAcGAJ4t0hOKdB3rssZ0diN59/V/En2bhoxkndQgf4Vg2JXyA9MapG8CZf8VgONqfgWyjTOuukGMooyjwZTUZrnGsXP5Z2IY3lRx6VqCTPSAniDhs
cnSINZ04kh8uvMCbzfONYaeoGZAKnRhnv3lvcNdMgzsCGe3vXKYNr2bAaAKpoqmK99NHVuKmWtoi+8aMj4rLWiDD9htycBjq+AWihUWMw5D4JsNih7AdPlaW61/GkP1LxmSieq
8MdECThUsOzt4M/PXEaMK8Fp7penrGBUXdriIInojBEEu/6+PQVrpqCw7T+PXYE/PpVKL0ZZOKHsVHC3behBbxU0NsjYZ25GbuF5HeJ5fV060lOBhhns18twqFePKgxKypRiEF
tnEno3aI1bRCmyHlJ6REHdmdskt[redacted]dr9JiToaQUefOawhA6p/ODZ4k=",
            "expiration": "Dec 12, 2022, 10:48:33 AM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "[redacted]CompositionSession",
            "arn": "arn:aws:sts:[redacted]:assumed-role/[redacted]CompositionSession"
        }
    },
```

Perks of using Short lived Tokens

1 

Enhancing Security

- Short-lived tokens have a **limited lifespan**, reducing the exposure window for potential attacks.
- If a token is compromised, its validity period is short, minimizing the risk of unauthorized access.
- Regular token expiration forces users to **re-authenticate**, ensuring better security.

2 

Mitigating Token Abuse

- Tokens are often used to authorize access to resources.
- By making tokens short lived, we limit the time an attacker can use to abuse a stolen token.
- Thus, **minimizing** the **risk window** significantly

3 

Least Privilege Principle

- A user should only have access to what they absolutely **need** and not what they want.
- When permissions **change** (e.g., user roles or access levels), short-lived tokens automatically reflect the updates upon renewal.
- Long-lived tokens may retain outdated permissions, leading to security risks.

Future References :

My Blog

Let’s connect over LinkedIn :

Women Techmakers