Identity Round Robin Workshop

Serverless Round – Static Hosted Website

What to expect from this round

- AWS IAM core concepts
- Using Cognito for Application User Management
- WildRydes Serverless Application
- Group Exercises
- Review and Discussion

https://awssecworkshops.com

Considerations for layers of principals

Applications

- Identities: Application Users, Application Administrators

Operating Systems

- Identities: Developers, and/or Systems Engineers

Amazon Web Services

- Identities: Developers, Solutions Architects, Testers, Software/Platform
- Interaction of AWS Identities:
  - Provisioning/deprovisioning EC2 instances and EBS storage.
  - Configuring Elastic Load Balancers.
  - Accessing S3 Objects or data in DynamoDB.
  - Accessing data in DynamoDB.
  - Interacting with SQS queues.
  - Sending SNS notifications.

aws

Considerations for layers of principals

Applications

- Identities: Application Users, Application Administrators

Operating Systems

- Identities: Developers, and/or Systems Engineers

Amazon Web Services

- Identities: Developers, Solutions Architects, Testers, Software/Platform
- Interaction of AWS Identities:
  - Provisioning/deprovisioning EC2 instances and EBS storage.
  - Configuring Elastic Load Balancers.
  - Accessing S3 Objects or data in DynamoDB.
  - Accessing data in DynamoDB.
  - Interacting with SQS queues.
  - Sending SNS notifications.

aws

AWS Principals

Account Owner ID (Root Account)

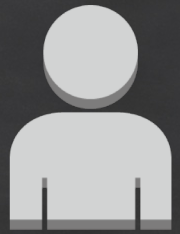
- Access to all subscribed services.
- Access to billing.
- Access to console and APIs.
- Access to Customer Support.

AWS Identity and Access Management (IAM)

- Access to specific services.
- Access to console and/or APIs.
- Access to Customer Support (Business and Enterprise).

AWS Identity and Access Management (IAM)

*Enables **you** to **control** who can do what in your AWS account*



IAM Users

IAM Groups

IAM Roles

Policies

aws

AWS IAM Policy Types

Identity-based policies

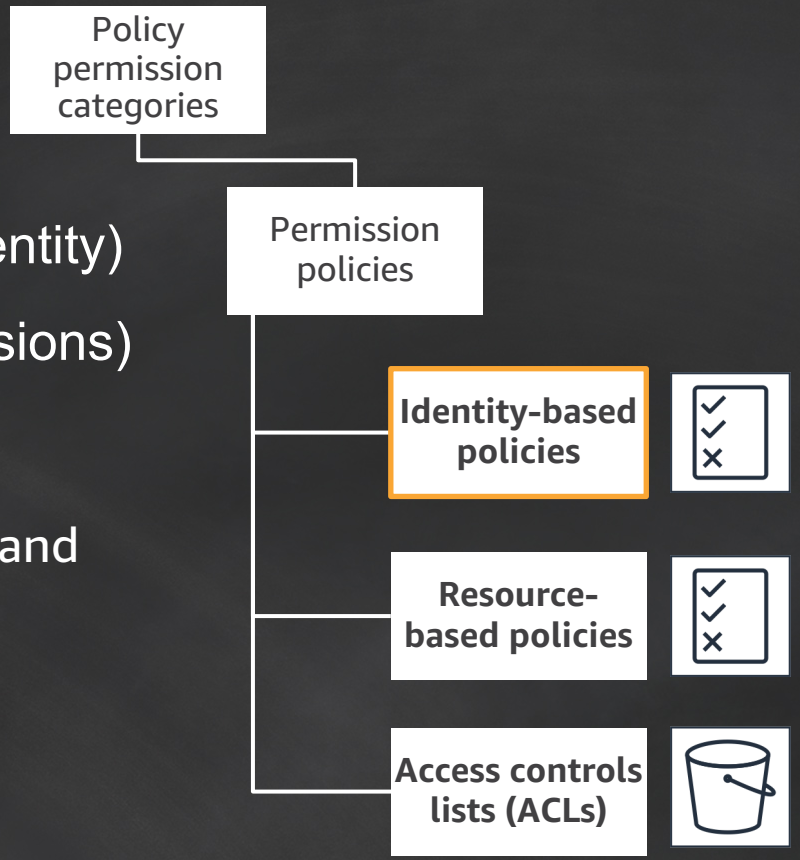
Resource-based policies

Access Control Lists

AWS IAM Policy Types

- JSON-formatted documents
- Attached to a **principal** (or identity)
- Contains a statement (permissions) that specifies:
  - what actions that identity can perform, on which resources, and under what conditions.

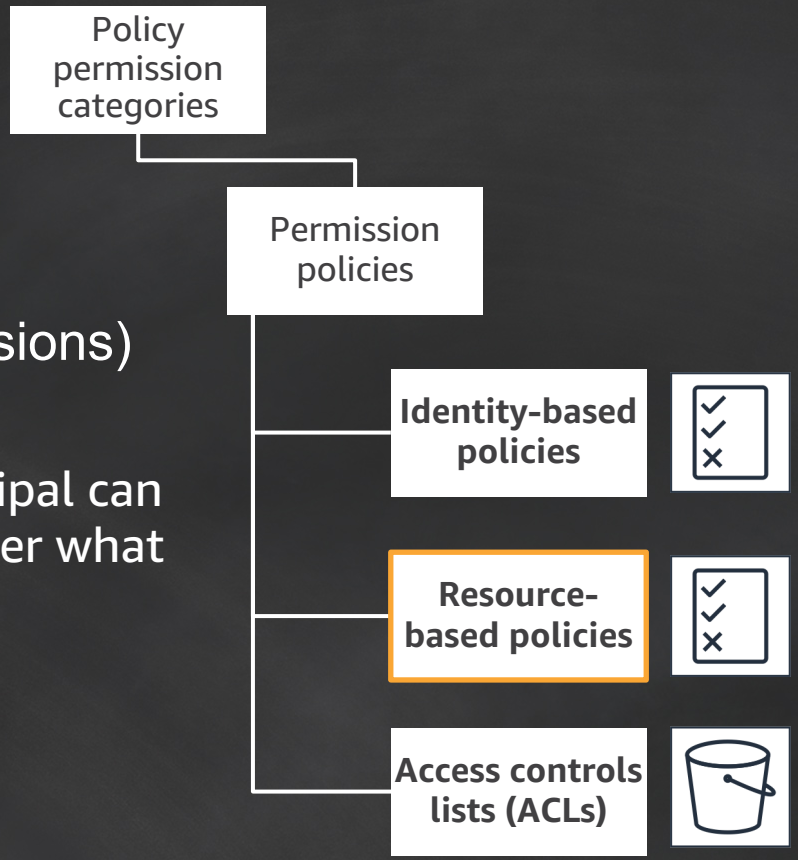
Principal (implicit)
Action
Resource
Condition



AWS IAM Policy Types

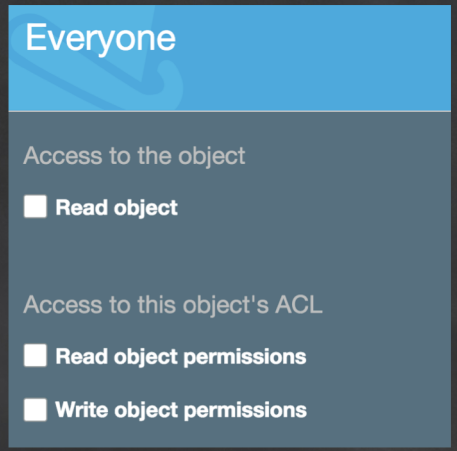
- JSON-formatted documents
- Attached to a **resource**
- Contains a statement (permissions) that specifies:
  - what actions a specified principal can perform on that resource under what conditions.

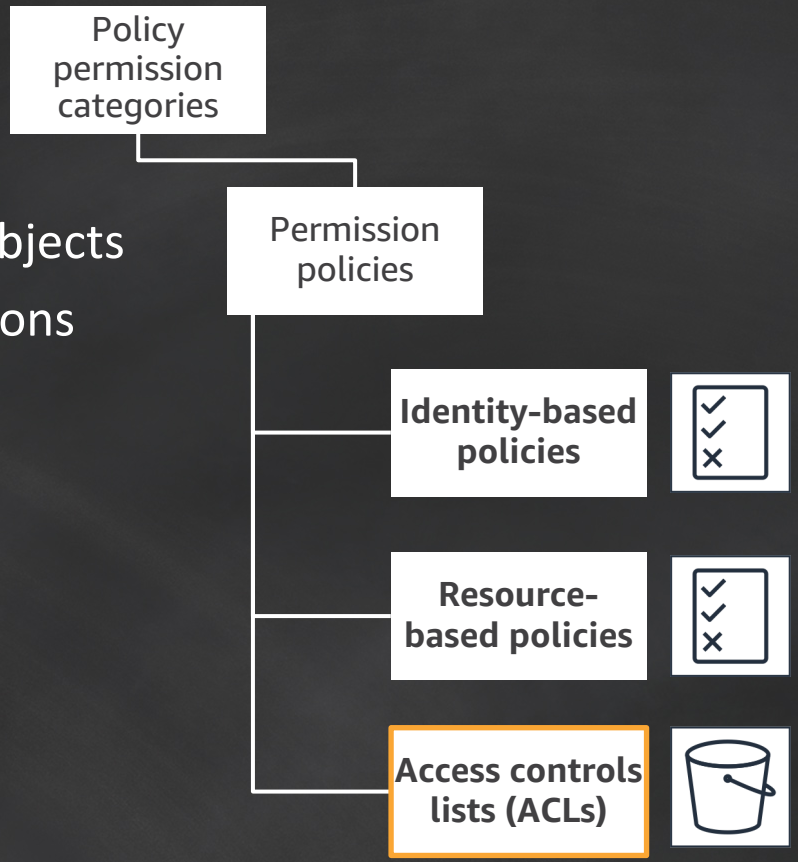
Principal
Action
Resource
Condition



AWS IAM Policy Types

- Manage access to buckets and objects
- Contains a Grantee and Permissions





AWS IAM Identity-Based Policy Example

AWS IAM User
(Alice)

AWS IAM
Identity-Based
Policy

```
"Version": "2012-10-17"
"Statement": [
   {
     "Effect": "Allow",
     "Action": "s3:GetObject",
     "Resource": "arn:aws:s3:::examplebucket/*"
    }
 ]
```

AWS IAM Identity-Based Policy Example

AWS IAM User
(Alice)

AWS IAM
Identity-Based
Policy

```
"Version": "2012-10-17"
"Statement": [
   {
     "Effect": "Allow",
     "Action": "s3:GetObject",
     "Resource": "arn:aws:s3:::examplebucket/*”,
     "Condition": {
       "StringEquals": {
         "s3:ExistingObjectTag/classification": ”sensitive"
       }
     }
   ]
```

AWS IAM Resource-Based Policy Example

Amazon S3 Bucket

AWS IAM
Resource-Based
Policy

```
"Version": "2012-10-17"
"Statement": [
   {
     "Effect": "Allow",
     ”Principal": {
       "AWS" : "*"
     },
     "Action": "s3:GetObject",
     "Resource": "arn:aws:s3:::examplebucket/*”
   }
 ]
```

AWS IAM Resource-Based Policy Example

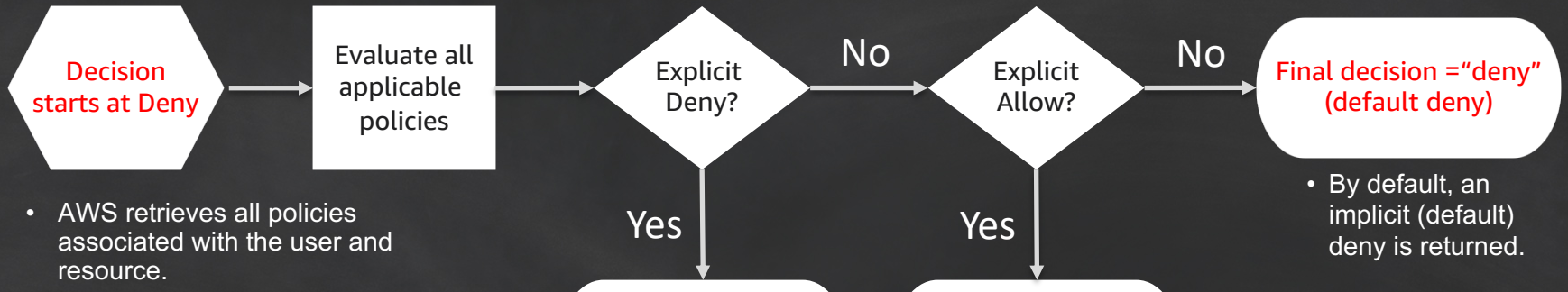
Amazon S3 Bucket

AWS IAM Resource-Based Policy

```
"Version": "2012-10-17"
"Statement": [
   {
     "Effect": "Allow",
     ”Principal": {
       "AWS" : "*"
     },
     "Action": "s3:GetObject",
     "Resource": "arn:aws:s3:::examplebucket/*”,
     "Condition" : {
       "IpAddress" : {
         "aws:SourceIp": "192.168.143.0/24"
       },
     }
    }
  ]
```


aws

AWS IAM Policy Evaluation Logic



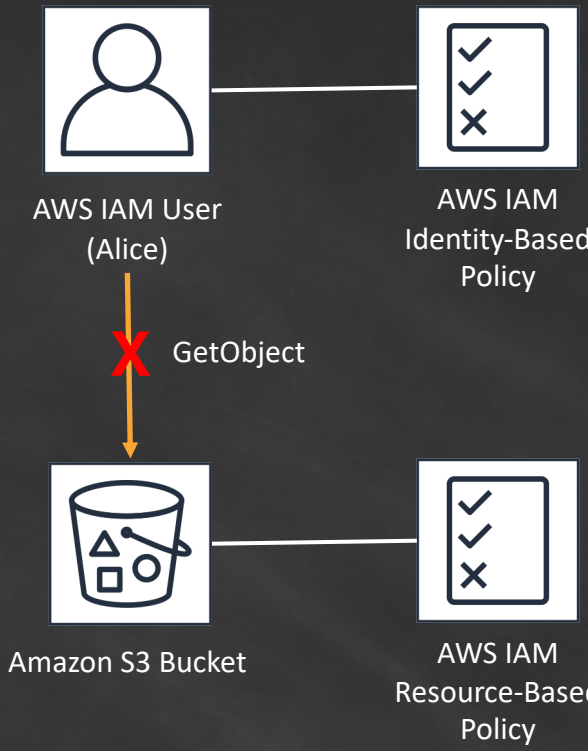
- AWS retrieves all policies associated with the user and resource.
- Only policies that match the action and conditions are evaluated.

- If a policy statement has a deny, it trumps all other policy statements.

- Access is granted if there is an explicit allow and no deny.

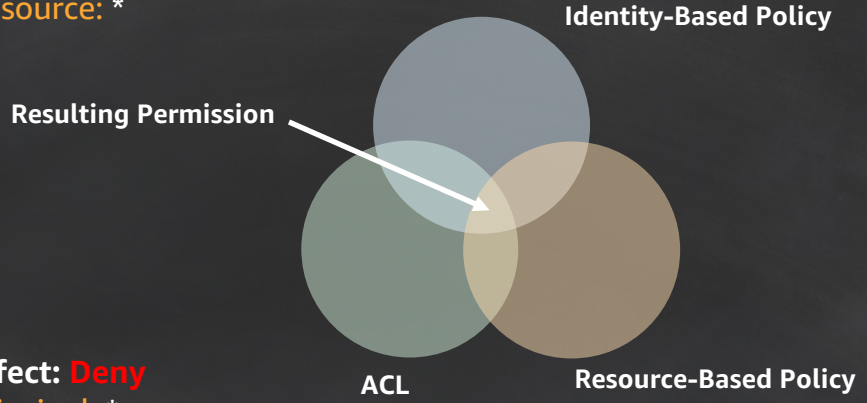
- By default, an implicit (default) deny is returned.

AWS IAM Policy Evaluation Logic – Example 1

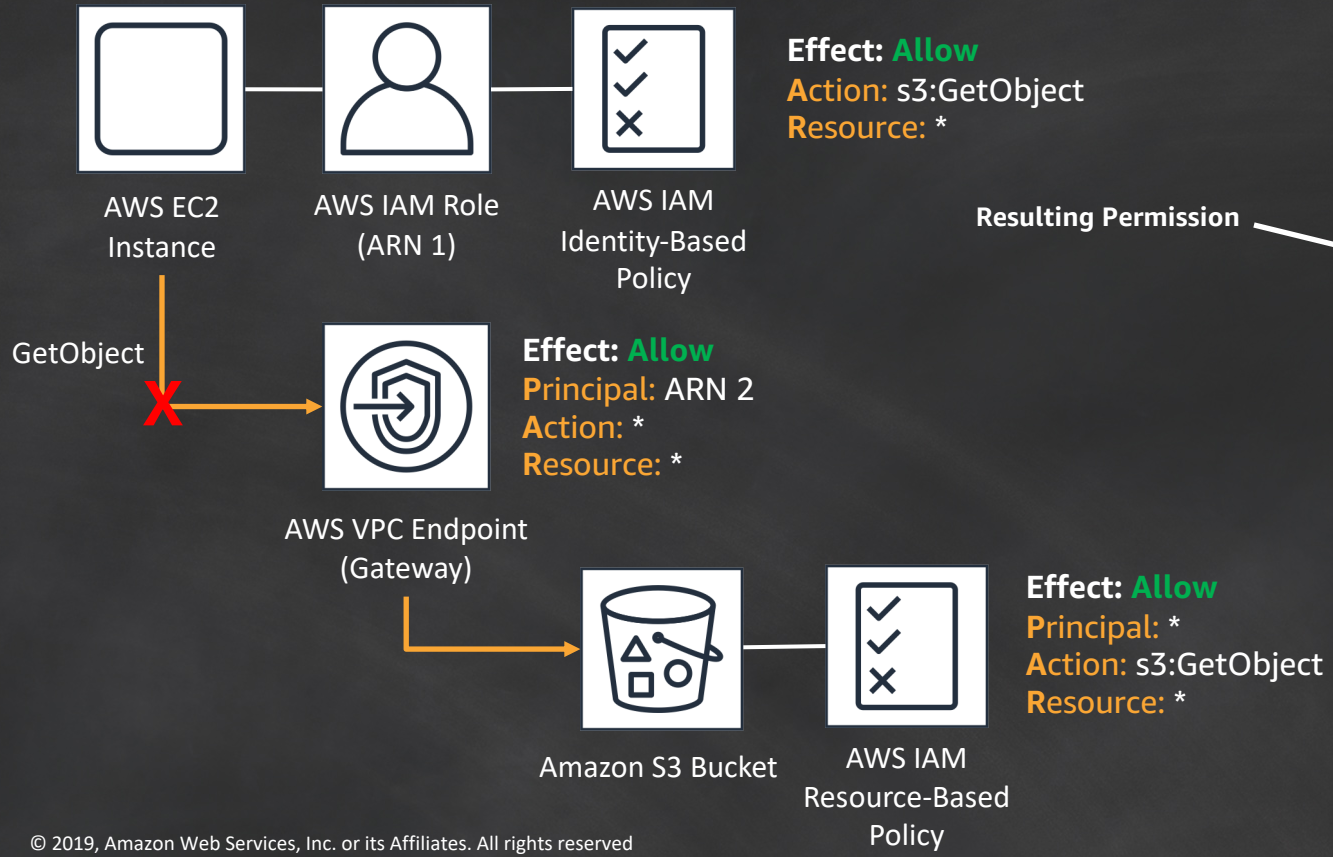


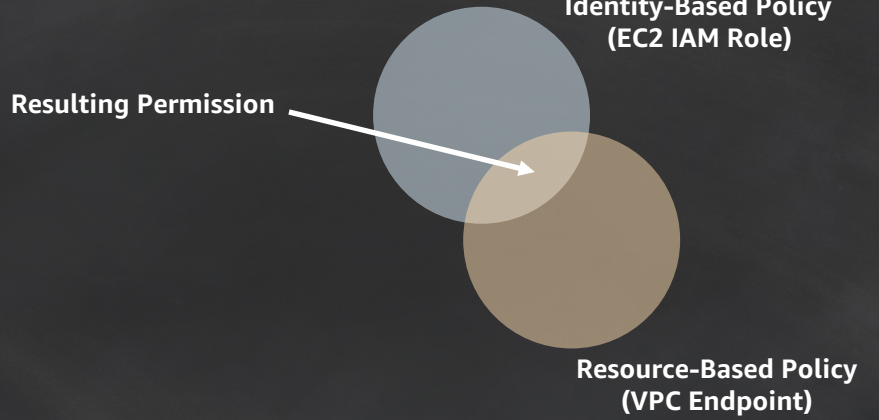
Effect: Allow
Action: s3:GetObject
Resource: *

Effect: Deny
Principal: *
Action: s3:GetObject
Resource: *



AWS IAM Policy Evaluation Logic – Example 2





Considerations for layers of principals

Applications

- Identities: Application Users, Application Administrators

Operating Systems

- Identities: Developers, and/or Systems Engineers

Amazon Web Services

- Identities: Developers, Solutions Architects, Testers, Software/Platform
- Interaction of AWS Identities:
  - Provisioning/deprovisioning EC2 instances and EBS storage.
  - Configuring Elastic Load Balancers.
  - Accessing S3 Objects or data in DynamoDB.
  - Accessing data in DynamoDB.
  - Interacting with SQS queues.
  - Sending SNS notifications.

aws

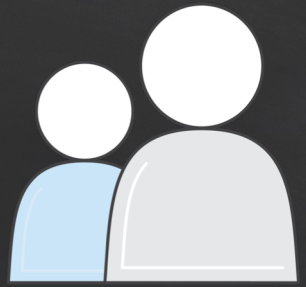
Amazon Cognito

Identity for your web and mobile apps

Security & Access

User Ownership

Experience

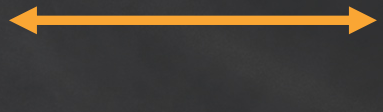


Customer Relationships

aws

Amazon Cognito

Web and Mobile Apps



Amazon Cognito

Cognito User Pools →

Cognito Identity Pools →

- Managed User Directory
- AWS Credentials
- Federation (SAML)
- Hosted-UI
- Standard Tokens (OAuth 2)

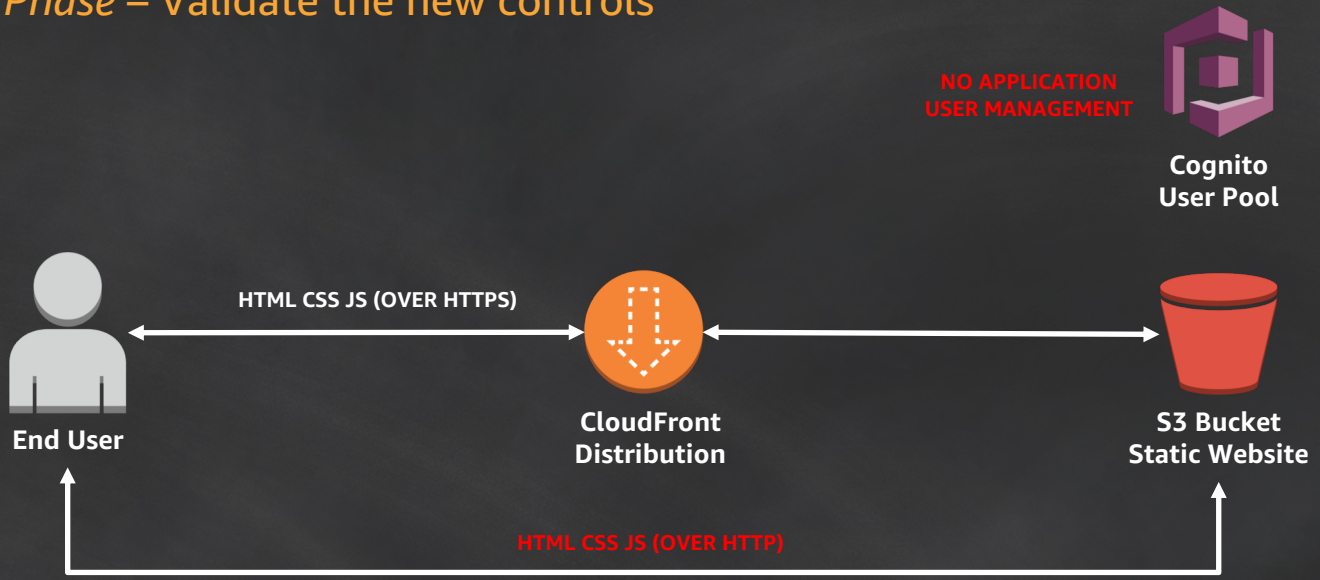
Developers focus on what is special about their app

Cognito handles auth and identity

aws

WildRydes Serverless Application

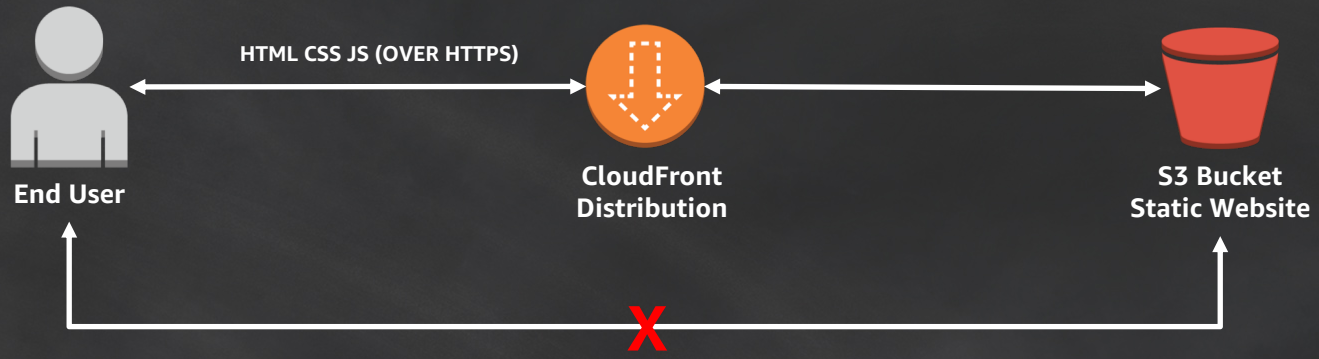
- *Build Phase* – Complete the access controls
- *Verify Phase* – Validate the new controls



WildRydes Serverless Application

Build Phase - Task 1

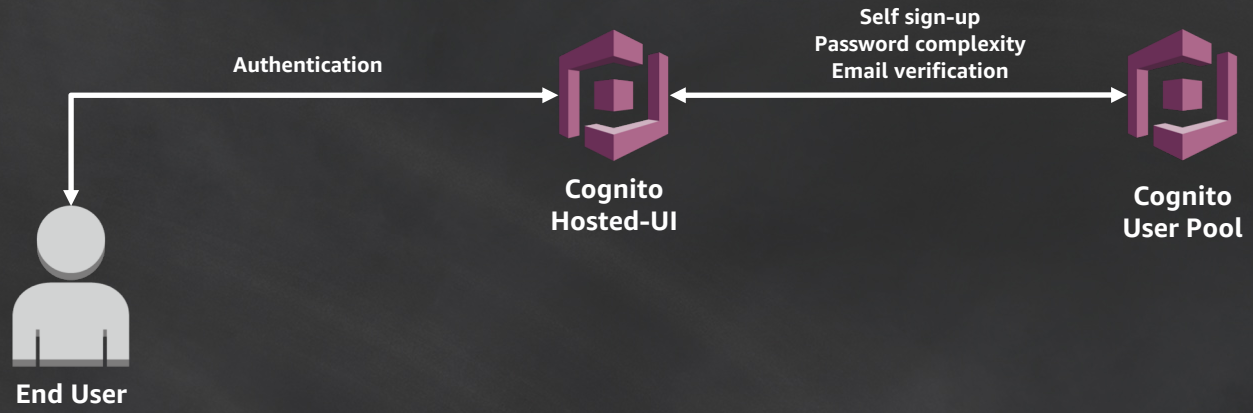
Reduce the attack surface of your origin



WildRydes Serverless Application

Build Phase - Task 2

Setup application user management



WildRydes Serverless Application

Build Phase (1 hour)

https://awssecworkshops.com/

Directions:

- Workshops (top navigation)
  - Identity Round Robin
  - **Serverless Round – Scenario (bottom right)**
    - ***AWS Sponsored Event***
  - **Build Phase (bottom right)**
    - **Task 1 – Reduce the attack surface of the origin**
    - **Task 2 – Set up application user management**

WildRydes Serverless Application

Verify Phase (15 Minutes)

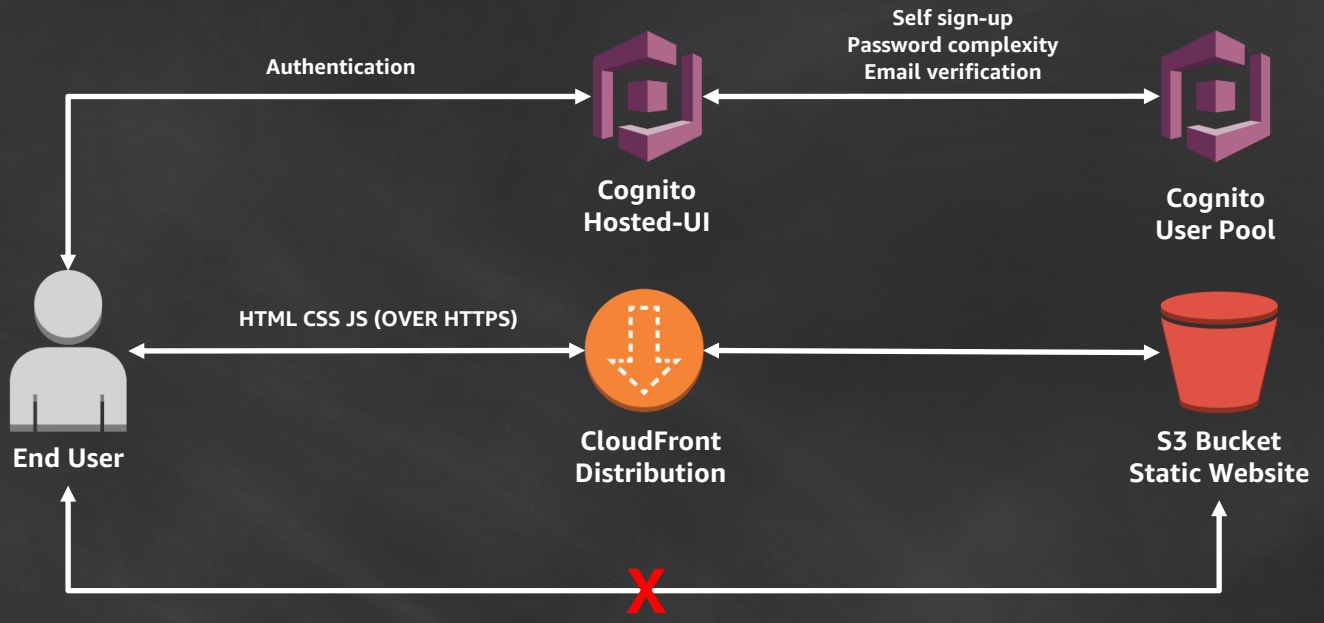
https://awssecworkshops.com/

Directions:

- Workshops
  - Identity Round Robin
  - Serverless Round – Scenario
  - Build Phase
  - **Verify Phase (bottom right)**
    - **Login to another teams AWS Account**
    - **Verify Task 1 & 2**

WildRydes Serverless Application

Architecture



Review and Discussion

How did you restrict access to the S3 Bucket?

Review and Discussion

What response type did you put in your Hosted-UI URL?

Thank You!