
AWS
re:Invent

SVS309

Best practices for security governance in serverless applications

Lia Vader

Serverless Solutions Architect

AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

Proactively enforce your organization’s standards for serverless developers and resources

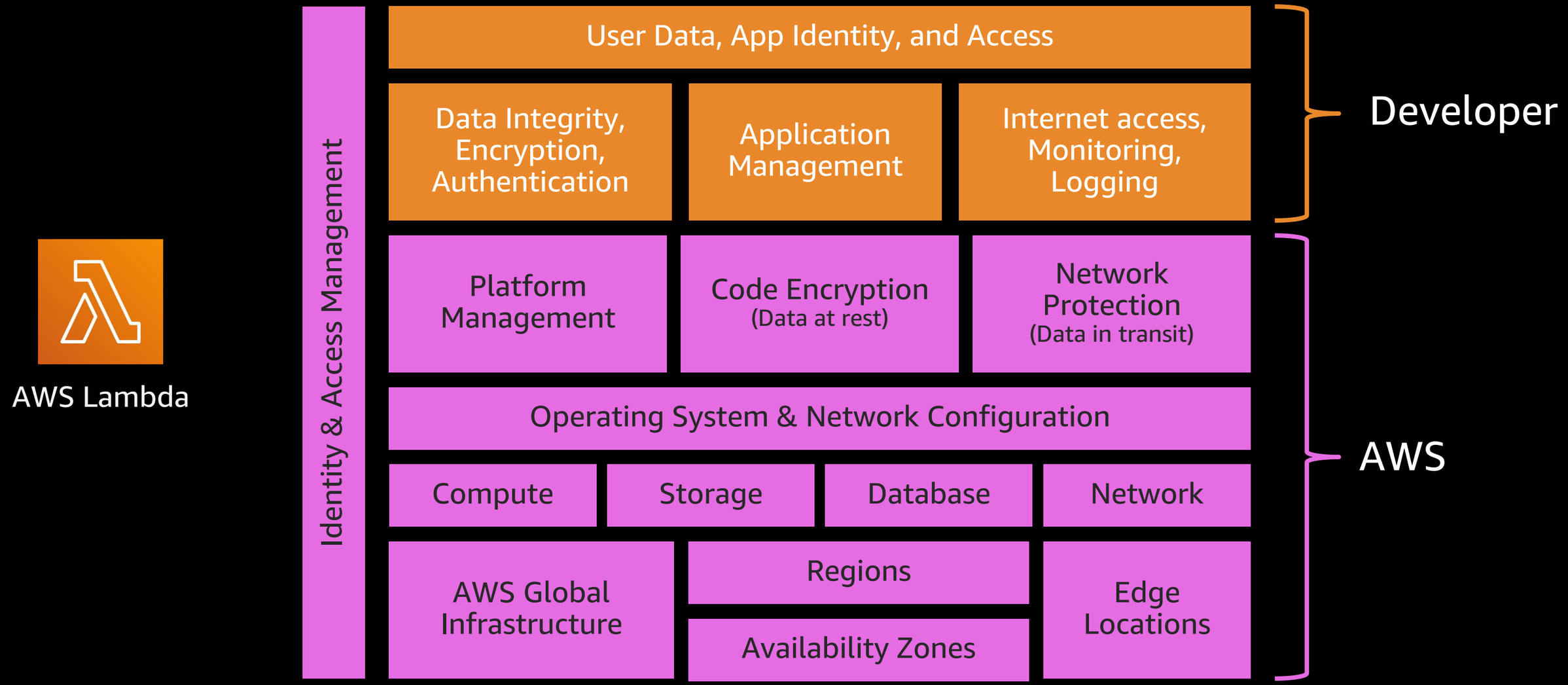
Continuously monitor and ensure your resources are in compliance with your standards

Ensure the integrity and validity of your code



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS shared responsibility model applied to AWS Lambda



Example standards to build

- AWS Identity and Access Management (IAM)
- Tagging
- Logging
- Networking boundaries
- TLS requirements
- Encryption
- Monitoring and observability

Example standards to build

- AWS Identity and Access Management (IAM)
- Tagging
- Logging
- Networking boundaries
- TLS requirements
- Encryption
- Monitoring and observability

But are we actually doing those things?

Mechanisms for your developers



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

IAM policy specification basics

JSON-formatted documents

Contain a statement (permissions) that specifies:

- Which actions a principal can perform
- Which resources can be accessed

```
{
 "Statement":[{
   "Effect":"effect",
   "Principal":"principal",
   "Action":"action",
   "Resource":"arn",
   "Condition":{
     "condition":{
       "key":"value" }
     }
   }
  ]
 }
```

Principal
Action
Resource
Condition

You can have multiple statements, and each statement is composed of PARC

Generic policy for a single user on AWS Lambda



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "LambdaCreateFunctionOnly",
            "Effect": "Allow",
            "Action": [
                "lambda:CreateFunction”,
                "lambda:UpdateFunctionConfiguration"
            ],
            "Resource": "*"
        },
        …
}
```

Note: To create a Lambda function, you will also need IAM permissions like iam:PassRole

AWS Lambda specific policy conditions

Supported Condition Keys for Lambda – Examples	
lambda:FunctionArn	Restrict access for a specific function
lambda:Layer	Enforce a specific layer
lambda:Principal	Restrict AWS service or account that can invoke a function
lambda:VpcIds	Enforce functions to use VPC configuration for specified VPCs
lambda:SecurityGroupIds	Enforce functions to use VPC configuration with the specified security groups
lambda:SubnetIds	Enforce functions to use VPC configuration with the specified subnets

Example AWS Lambda policy using conditions



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "LambdaCreateFunctionOnly",
            "Effect": "Allow",
            "Action": [
                "lambda:CreateFunction",
                "lambda:UpdateFunctionConfiguration"
          ],
            "Resource": "*”,
             "Condition": {
                "ForAllValues:StringLike": {
                    "lambda:Layer": [
                        "arn:aws:lambda:*:*:layer:reinvent-*:*"
                     ]
                }
            }
        }
}
```

Note: To create a Lambda function, you will also need IAM permissions like iam:PassRole

Managing across the organization



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Example policy as a service control policy (SCP)

```
{
        "Version": "2012-10-17",
        "Statement": [{
                "Sid": "Stmt159186333251",
                "Action": ["lambda:CreateFunction", "lambda:UpdateFunctionConfiguration"],
                "Effect": "Deny",
                "Resource": "*",
                "Condition": {
                        "StringNotEquals": {
                                "lambda:VpcIds":
                                        ["vpc-0abcd123efg456hij", "vpc-vwxyz987stu654pqr"]
                        }
                }
        }]
}
```

Mechanisms for your serverless applications



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

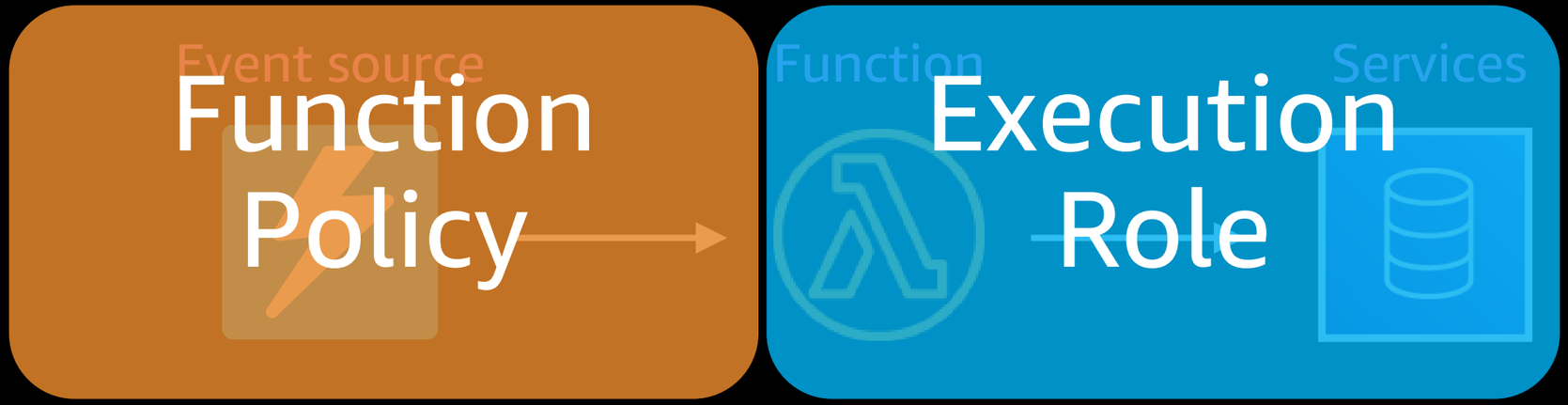
Lambda permissions model

Function policies:

- “Actions on bucket X can invoke Lambda function Z"
- Resource policies allow for cross-account access

Execution role:

- “Lambda function A can read from DynamoDB table users”
- Define what AWS resources / API calls this function can access via IAM



Execution role

Lambda Function



+

Execution Role



=

Allowed Actions

IAM role with:

- Trust policy allowing Lambda to AssumeRole
- IAM policy permissions

Example: Allow Lambda to write to DynamoDB table

IAM policy

```
{
        "Effect": "Allow",
        "Action": [
                "dynamodb:PutItem"
        ],
        "Resource": "arn:aws:dynamodb:us-west-2:123456789012:table/reinvent”
}
```

Trust policy

```
{
   "Effect":"Allow",
   ”Principal": {
      “Service”: ”lambda.amazonaws.com"
   },
   ”Action": ”sts:AssumeRole”
}
```

Sometimes our developers do this

IAM policy

```
{
    "Effect": "Allow",
    "Action": [
        "dynamodb:*"
    ],
    "Resource": "*"
}
```



Trust policy

```
{
  "Effect":"Allow",
  "Principal": {
    "Service": "lambda.amazonaws.com"
  },
  "Action": "sts:AssumeRole"
}
```

How do I enforce least privilege policies for my developers?

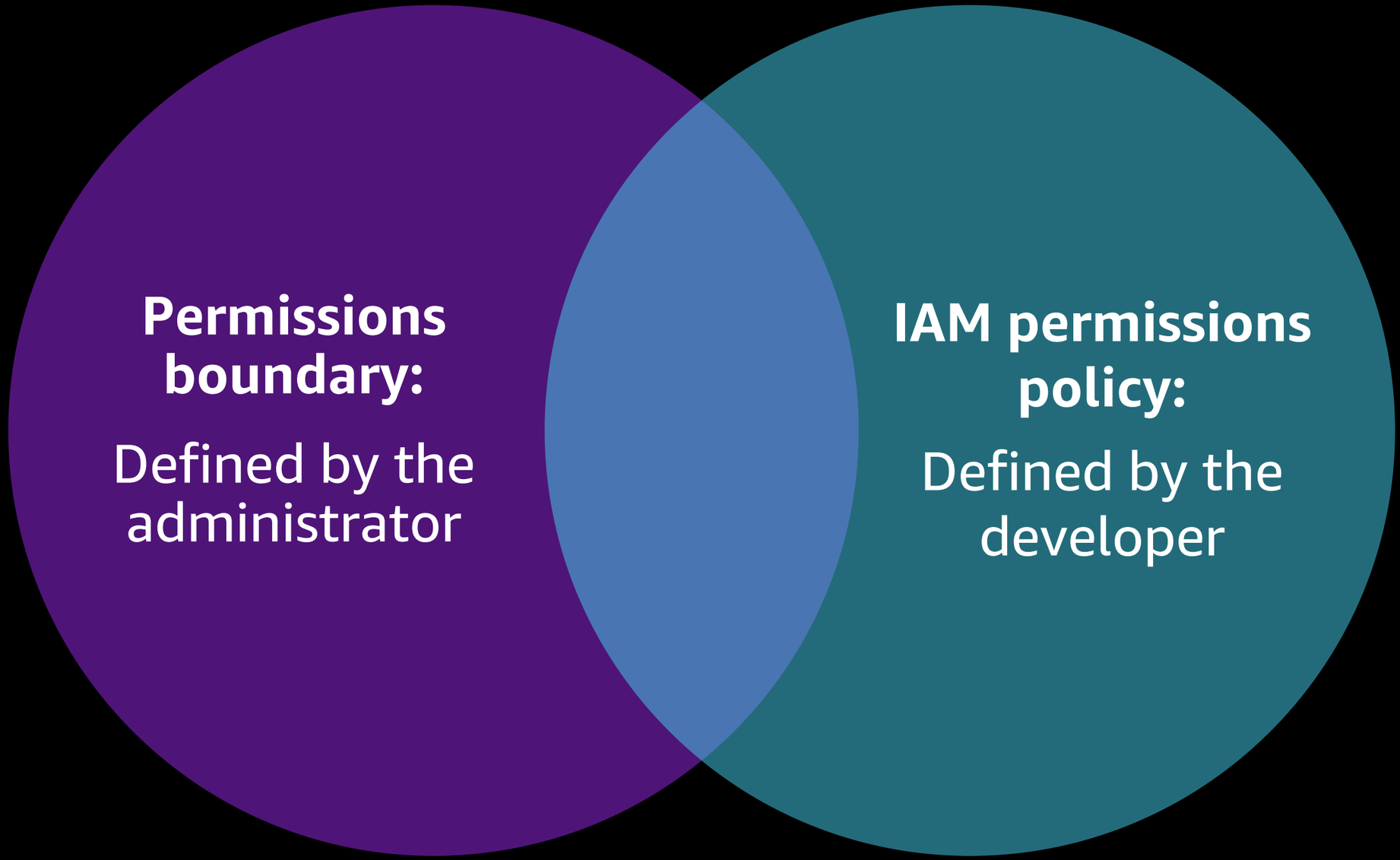


© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

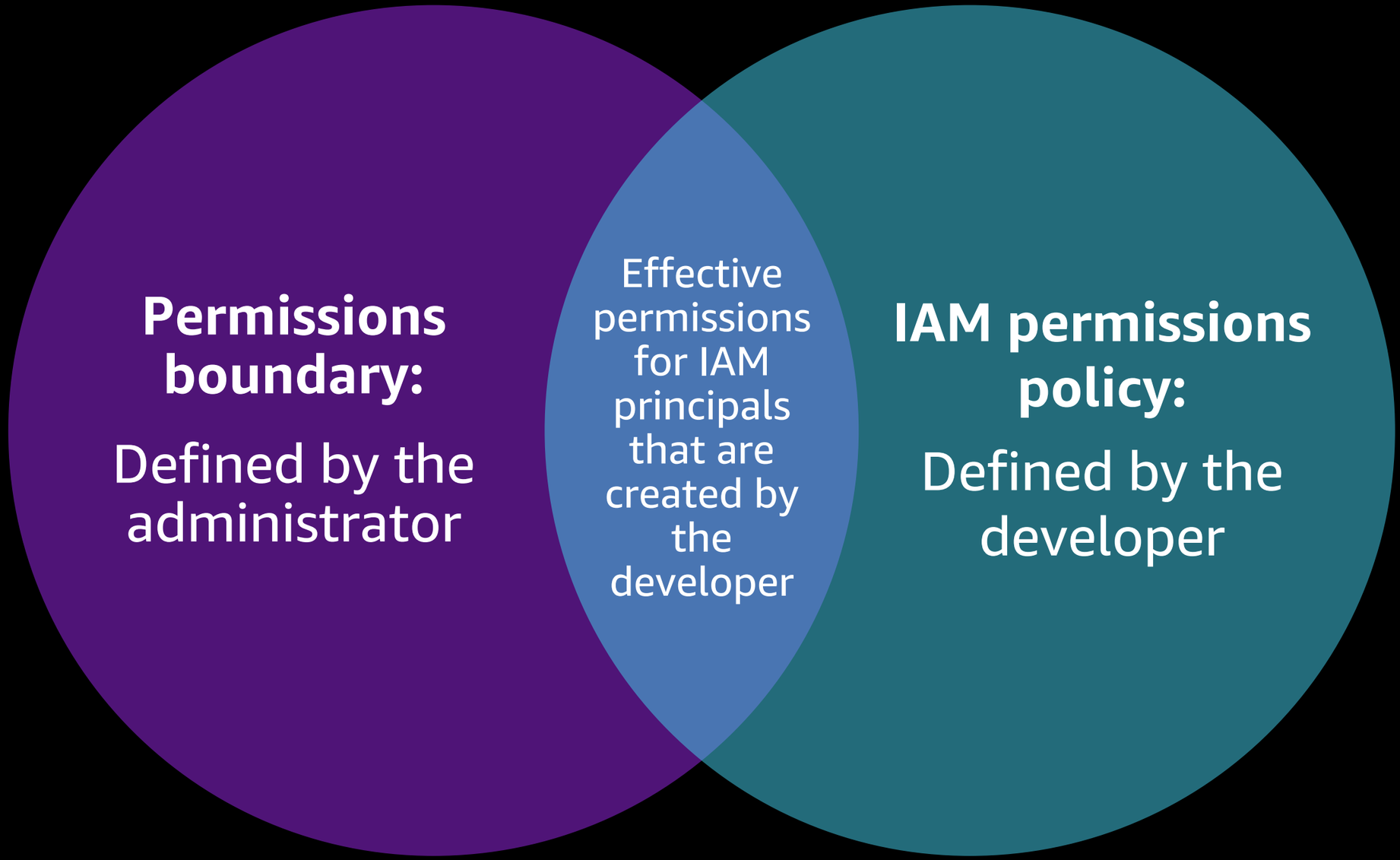
IAM permissions boundaries



IAM permissions boundaries



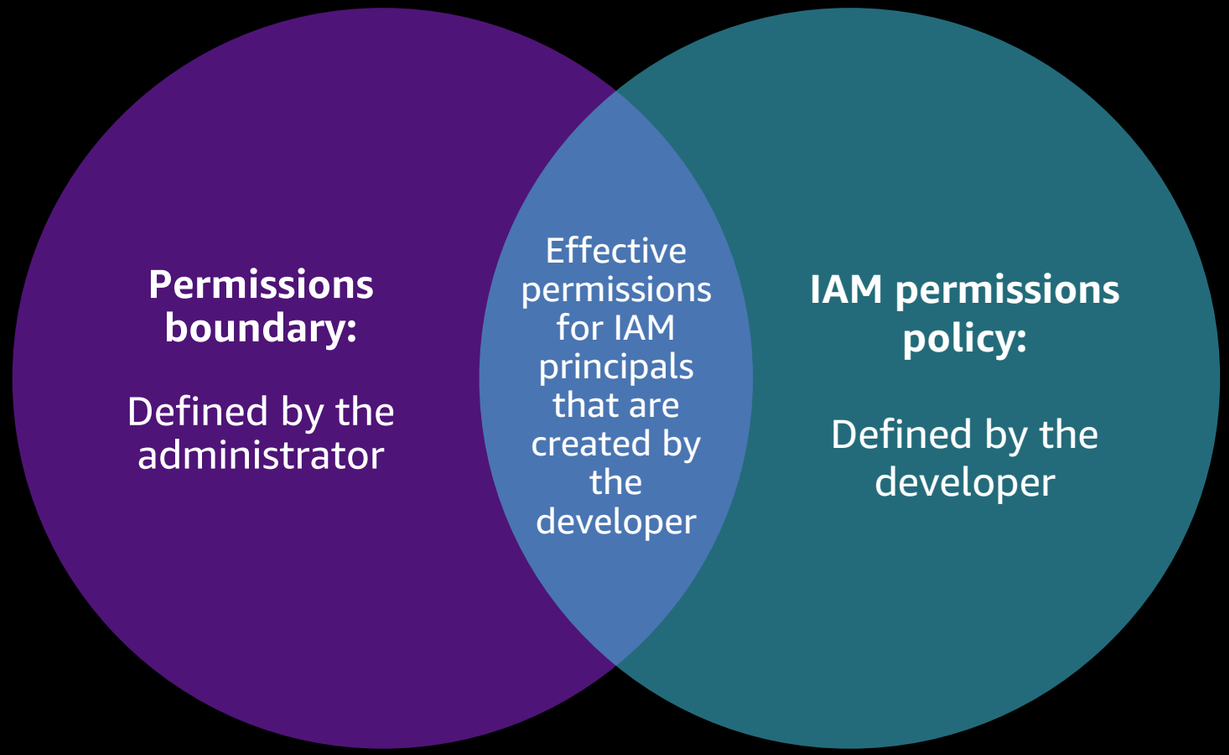
IAM permissions boundaries



IAM permissions boundaries

Scale and delegate permissions management to developers safely

Control the maximum permissions that employees can grant



Permissions boundary

- ✓ Attach to users, roles, group
- ✓ Defines max permissions that identity-based policies can grant
- ✓ Does not limit permissions granted in resource-based policies
- ✓ Does not grant permissions

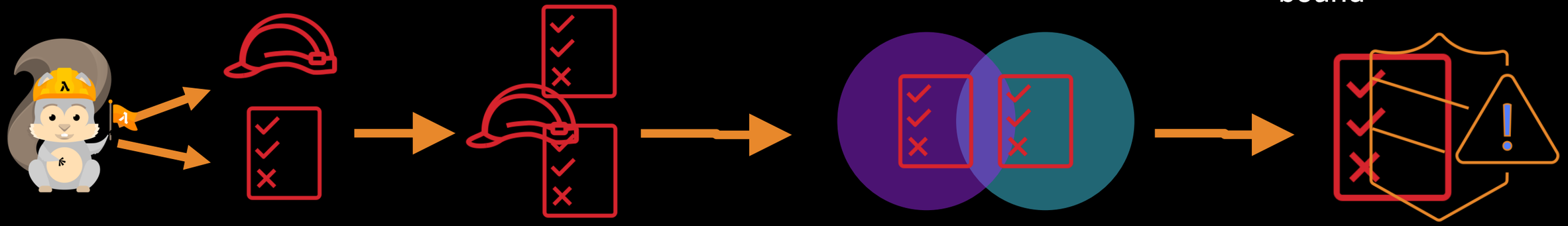
The solution = bound roles

Grant Sam the permission to create policies and roles

Require that permissions boundary is also attached to the role

The effective permission of the role is the intersection of the two policies

In this way, you can set the maximum permission of the roles that Sam creates – the roles are bound



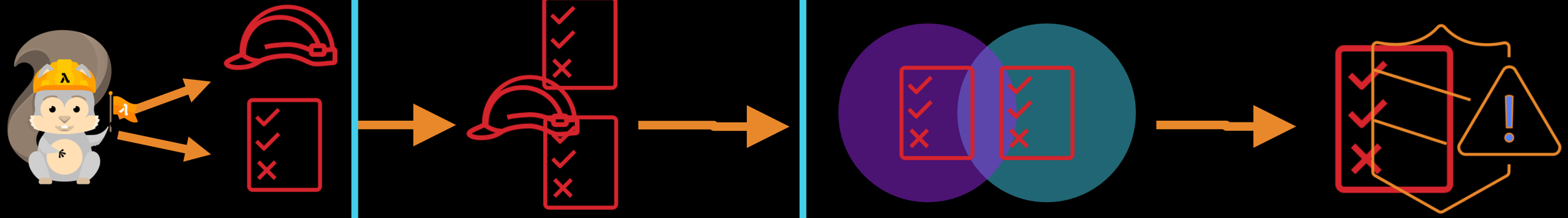
Taking a closer look at permissions boundaries

Grant Sam the permission to create policies and roles

Require that permissions boundary is also attached to the role

The effective permission of the role is the intersection of the two policies

In this way, you can set the maximum permission of the roles that Sam creates – the roles are bound



Example: IAM permissions boundary

Sam has permission to create policies and roles; Sam then attaches this permissions boundary



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SetPermissionsBoundary",
            "Effect": "Allow",
            "Action": [
                "*"
            ],
            "Resource": [
                "arn:aws:lambda:us-east-1:123456789012:function:reinvent*",
                "arn:aws:dynamodb:us-east-1:123456789012:table/reinvent*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

How do I require permissions boundaries?

```
"Condition": {"StringEquals":
        {"iam:PermissionsBoundary":
        "arn:aws:iam::123456789012:policy/permissionboundary"
        }
}
```

How do I require permissions boundaries?

```
"Condition": {"StringEquals":
        {"iam:PermissionsBoundary":
        "arn:aws:iam::123456789012:policy/permissionboundary"
        }
}
```

. . . applied to principal actions

```
"Effect": "Allow",
"Action": ["iam:CreateRole"],
"Resource": ["arn:aws:iam::123456789012:role/path/*"],
"Condition": {"StringEquals":
        {"iam:PermissionsBoundary":
        "arn:aws:iam::ACCOUNT_ID:policy/permissionboundary"
        }
}
```

End-user experience changes little

Create role for a Lambda function

 # Step 1: Create role and attach permissions boundary

 # Step 2: Create identity-based policy

No change

 # Step 3: Attach identity-based policy

No change

AWS SAM template example

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: 'AWS::Serverless-2016-10-31'
Resources:
  MyFunction:
    Type: 'AWS::Serverless::Function'
    Properties:
      Handler: index.handler
      Runtime: nodejs12.x
      CodeUri: 's3://my-bucket/function.zip'
      Policies:
        - DynamoDBCrudPolicy:
            TableName: !Ref reinventTable
      PermissionsBoundary: arn:aws:iam::123456789012:policy/reinventPermissionsBoundary
```

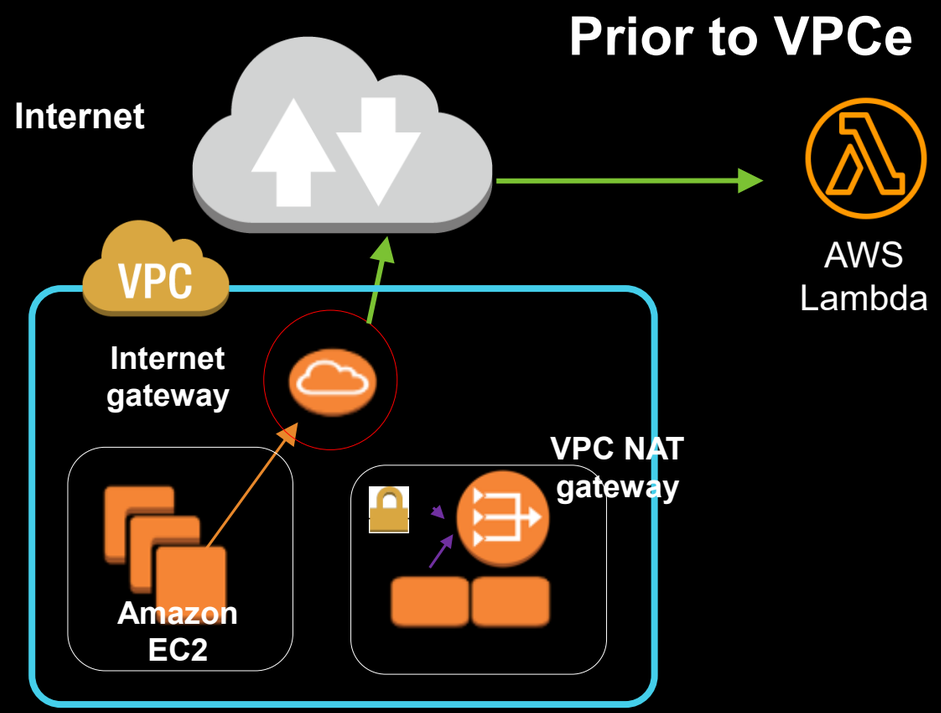
We just covered developers; what about my resources?



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

NEW

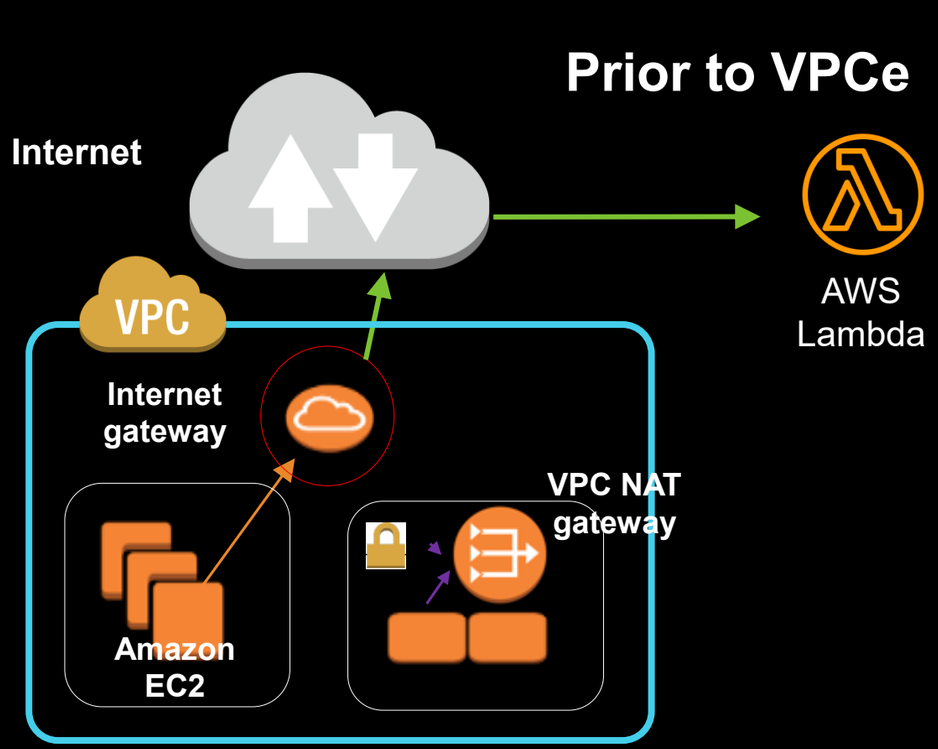
Access to AWS Lambda API from VPC



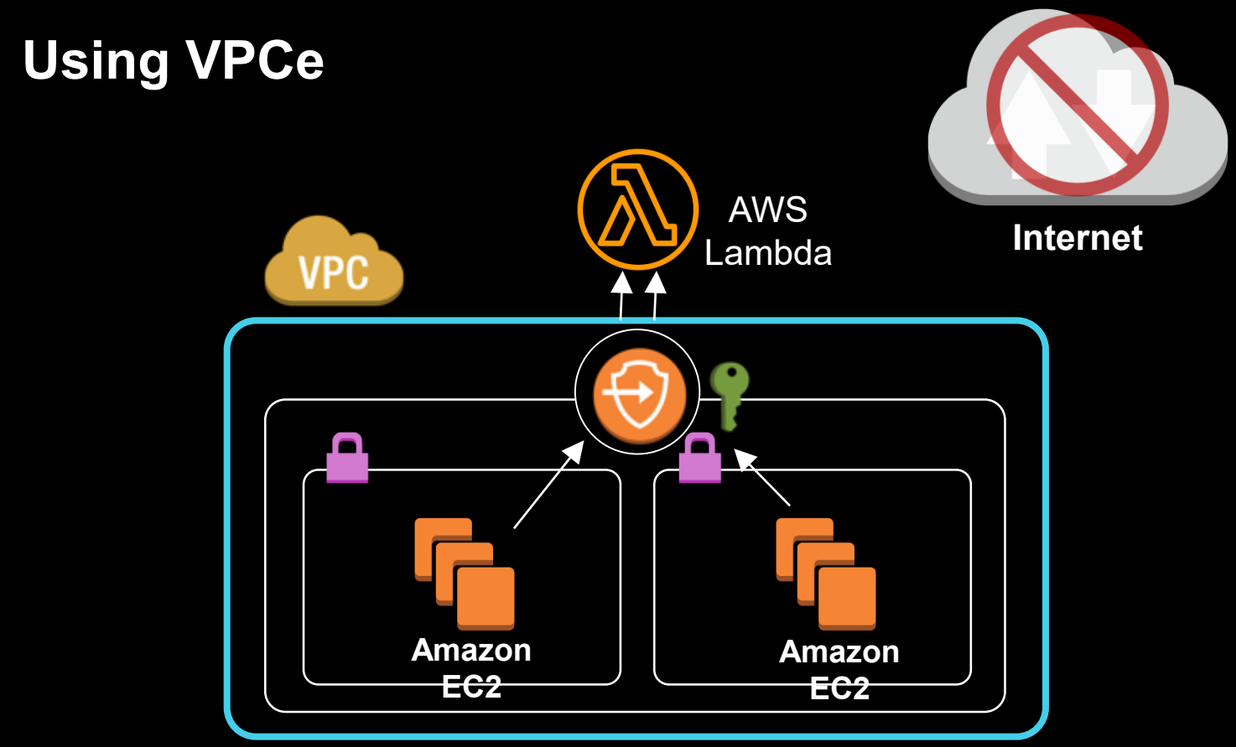
- Public IP on EC2 instances and internet gateway
- Private IP on EC2 instances and NAT

VPC endpoints (VPCe)

NEW



- Public IP on EC2 instances and internet gateway
- Private IP on EC2 instances and NAT



- Access Lambda using interface endpoint without using NAT instances or gateways

VPC endpoint policies for AWS Lambda

NEW

```
{
   "Statement": [{
      "Principal": {
         "AWS": [
            "arn:aws:iam::123412341234:role/MyRoleName"
         ]
      },
      "Effect": "Allow",
      "Action": [
         "lambda:InvokeFunction"
      ],
      "Resource": [
         "arn:aws:lambda:us-west-2:123412341234:function:reinvent*"
      ]
   }]
}
```

Can I control access to my private Amazon API Gateway APIs?



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Enforce source VPC for Amazon API Gateway via resource policies

```
…
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "execute-api:Invoke",
            "Resource": [
                "execute-api:/*"
            ]
        },
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "execute-api:Invoke",
            "Resource": [
                "execute-api:/*"
            ],
            "Condition" : {
                "StringNotEquals": {
                   "aws:SourceVpc": "vpc-1a2b3c4d"
                }
            }
        }
…
```

How do I know my code is what is running in AWS Lambda?

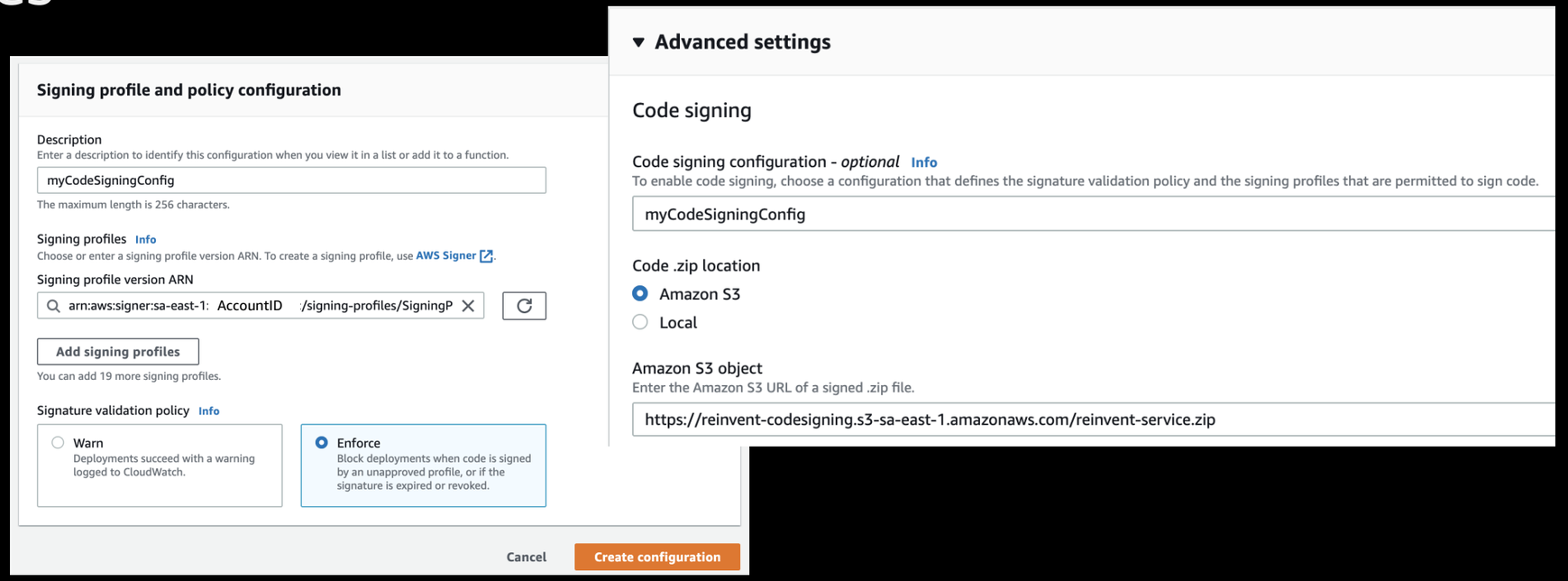


© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Code signing for AWS Lambda

NEW

- Digitally sign code artifacts deployed to Lambda functions
- Ensure integrity of code artifacts and validate they come from a trusted publisher prior to deployment
- Configure signature expiry and revoke unauthorized signatures
- Configure whether to reject or alarm on signature validation failures



Code signing is supported by AWS Signer

NEW

- Fully managed code signing service allows customers to generate signed code packages
- Provide permissions for cross-account use of Signing Profiles
- Revoke signatures by individual jobs, for all jobs created by a Signing Profile version, or jobs created within a certain time period

How does code signing work?

NEW

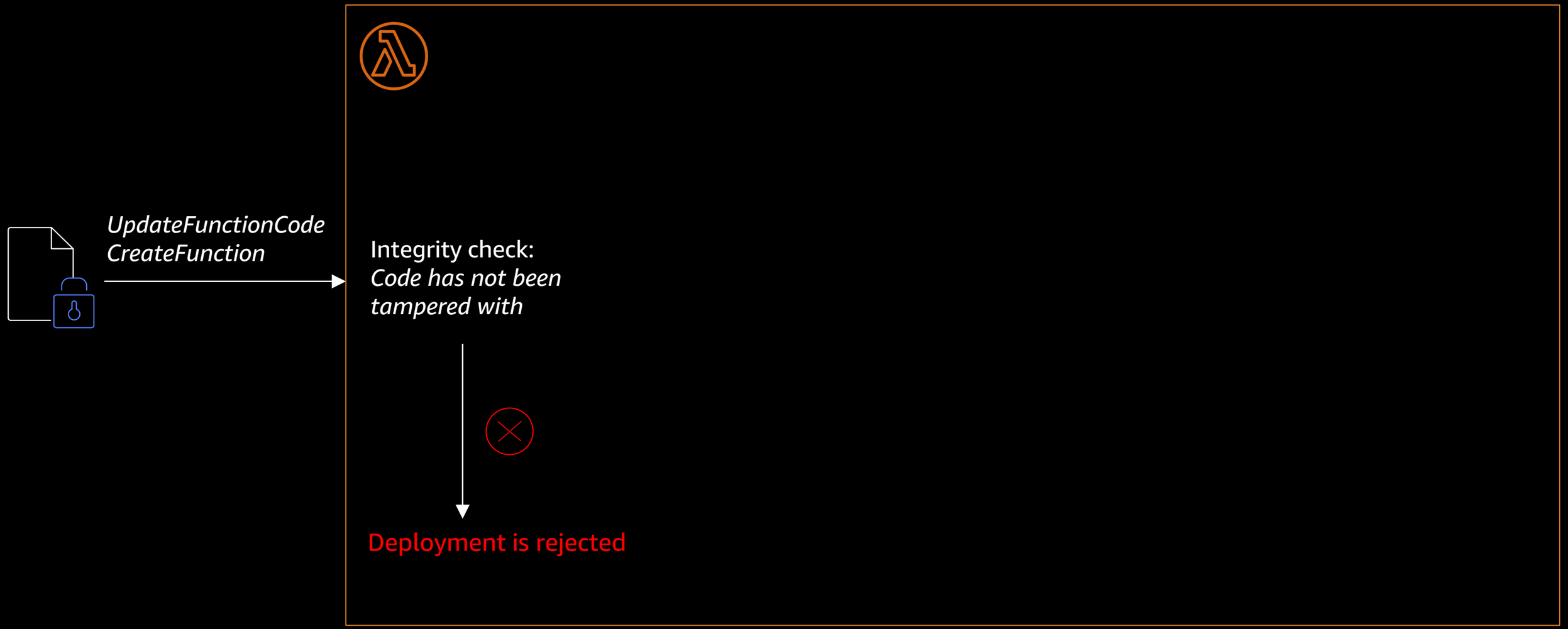


UpdateFunctionCode
CreateFunction



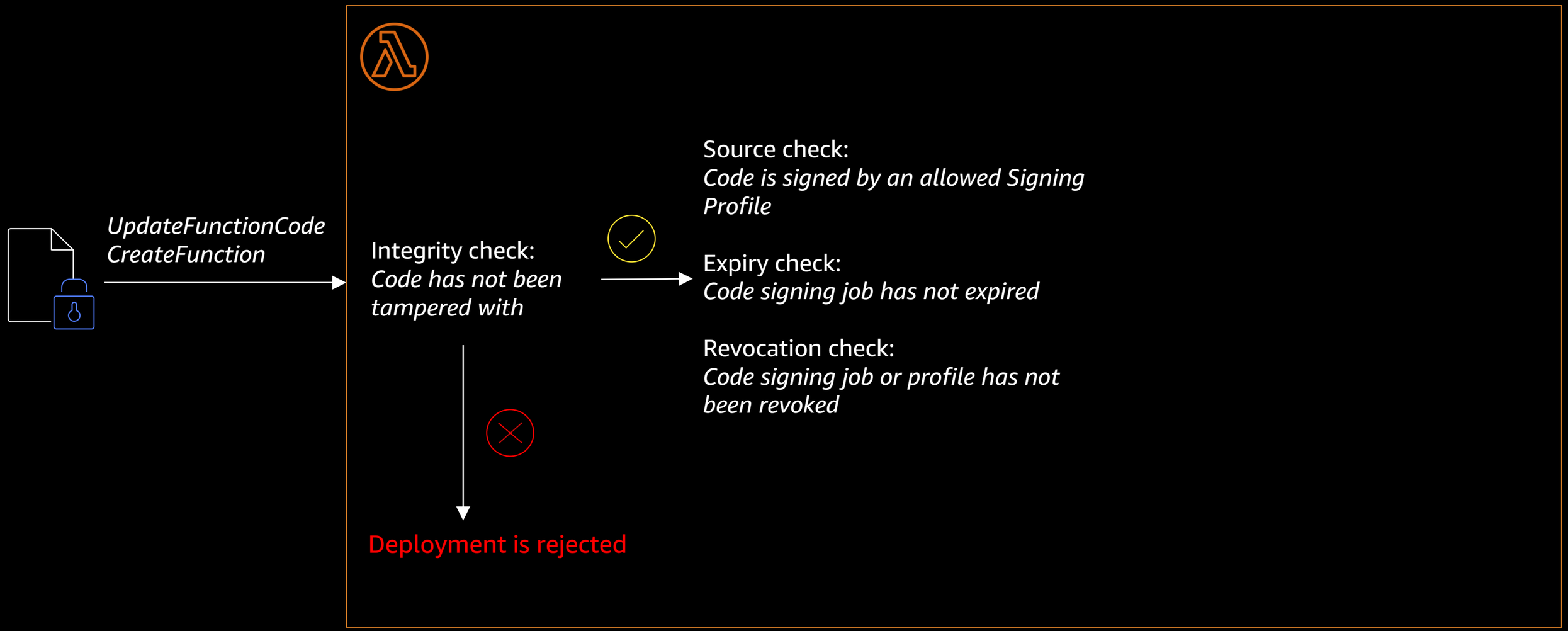
How does code signing work?

NEW



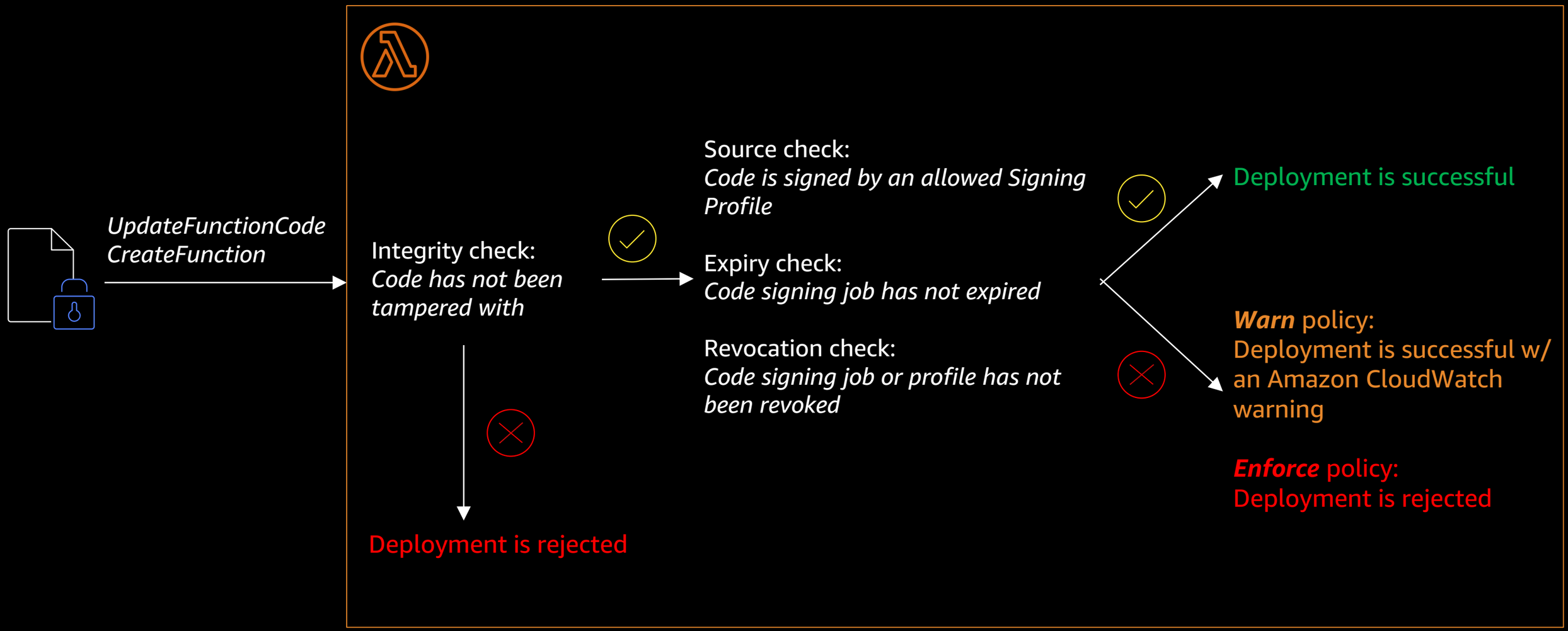
How does code signing work?

NEW



How does code signing work?

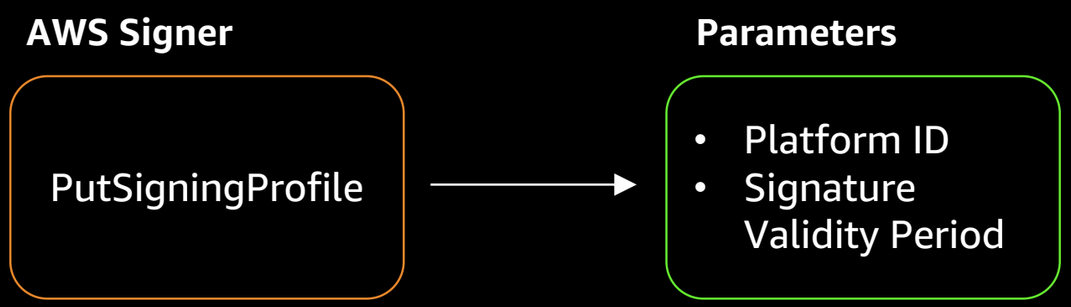
NEW



How does code signing work?

NEW

1. Create Signing Profile in AWS Signer

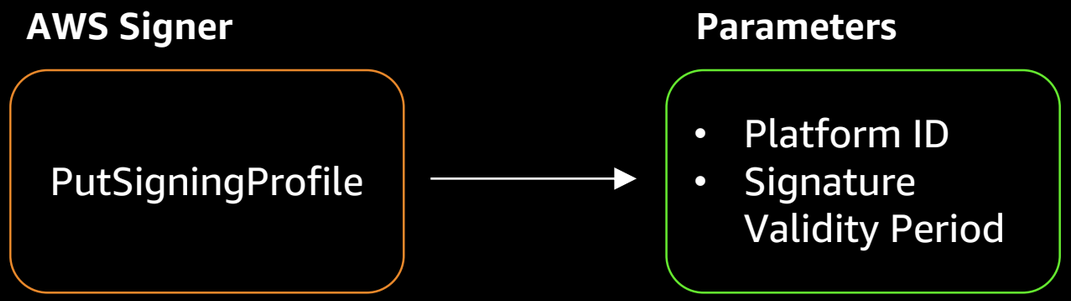


Platform Id: AWSLambda-SHA384-ECDSA
Signature Validity Period: Default is 135 months

How does code signing work?

NEW

1. Create Signing Profile in AWS Signer



Platform ID: AWSLambda-SHA384-ECDSA
Signature Validity Period: Default is 135 months

```
AWSTemplateFormatVersion: "2010-09-09"
Description: >
  Provisions Signing Profile
Resources:
  SigningProfileForReinvent:
    Type: AWS::Signer::SigningProfile
    Properties:
      PlatformId: "AWSLambda-SHA384-ECDSA"
      SignatureValidityPeriod:
        Type: "DAYS"
        Value: 365
      Tags:
        Key: "Name"
        Value: "reinvent2020"
```

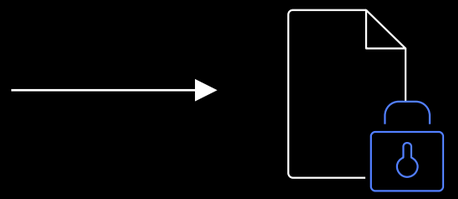
How does code signing work?

NEW

2. Create signed code packages

AWS Signer

Create SigningJob with a SigningProfile

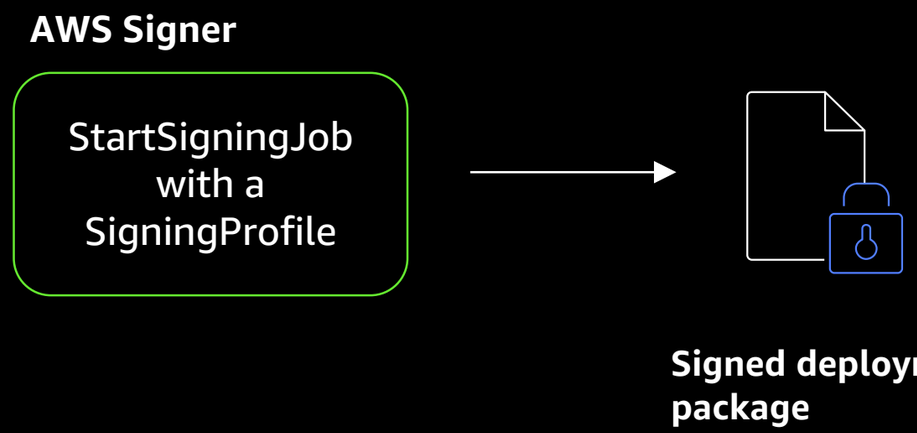


Signed deployment package

How does code signing work?

NEW

2. Create signed code packages

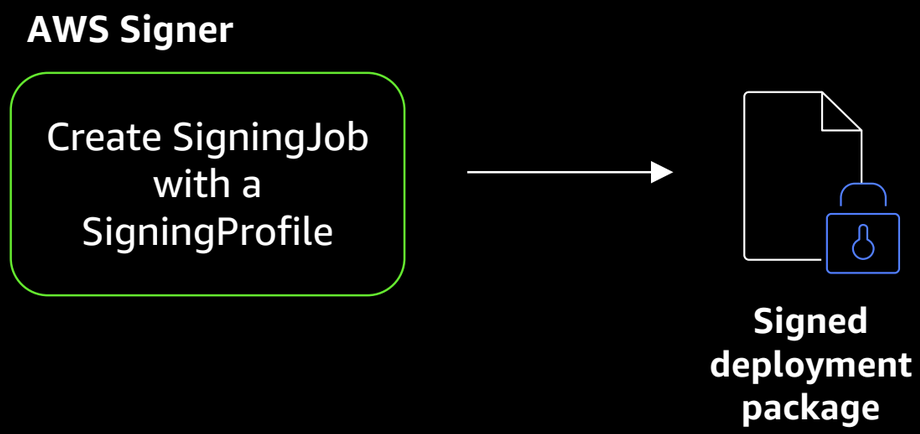



```
aws signer start-signing-job \
--source 's3={bucketName=signer-source,key=lambdafunction.zip,version=PNyFaUTgsQh5ZdMCcoCe6pT1gOpgB_M4}' \
--destination 's3={bucketName=signer-destination,prefix=signed-}' \
--profile-name SigningProfileForReinvent
```

How does code signing work?

NEW

2. Create signed code packages



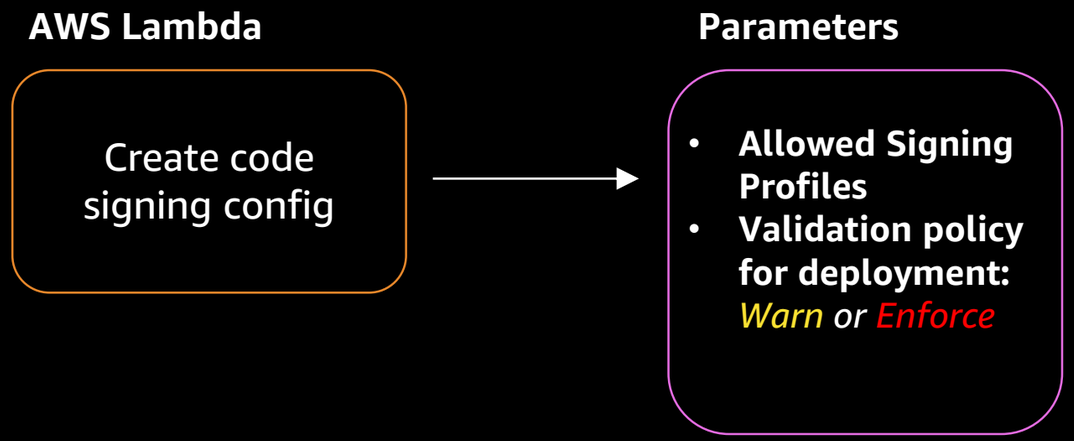
Response:

```
{
    "jobId": "9371a7c2-ab7b-4cbf-ac7b-b39e5788f23f",
    "jobOwner": "123456789012"
}
```

How does code signing work?

NEW

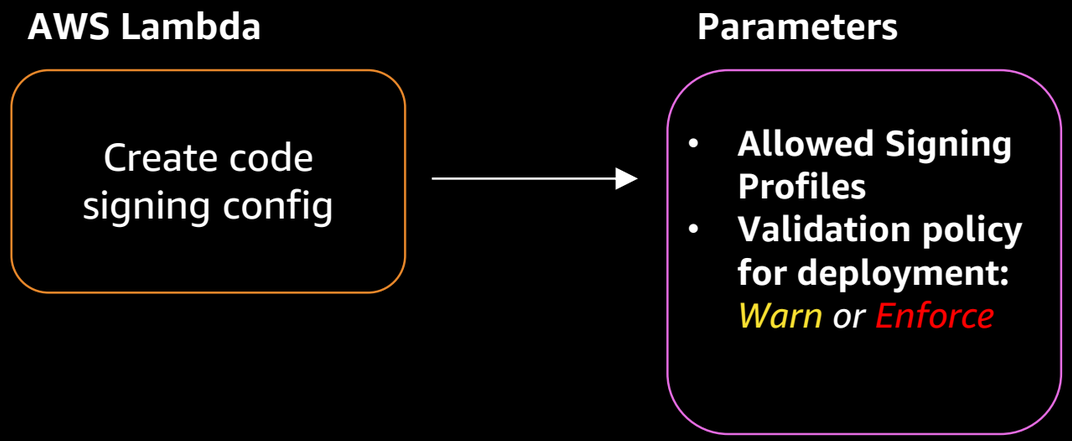
3. Create code signing configuration resource



How does code signing work?

NEW

3. Create code signing configuration resource




```
Resources:
  MyCodeSigningConfig:
    Type: AWS::Lambda::CodeSigningConfig
    Properties:
      Description: "New Code Signing for re:Invent"
      AllowedPublishers:
        SigningProfileVersionArns:
          - MySigningProfile-profileVersionArn
      CodeSigningPolicies:
        UntrustedArtifactOnDeployment: "Enforce"
```

How does code signing work?

NEW

4. Create function with code signing enabled

AWS Lambda

CreateFunction with CodeSigningConfig ARN



Function with code signing enabled

```
Resources:
  MyLambdaFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: .
      Handler: src/lambda.handler
      Runtime: nodejs12.x
      CodeSigningConfigArn: !Ref MyCodeSigningConfig
```

NEW

Code signing with SAM

```
Resources:
  MyLambdaFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: .
      Handler: src/lambda.handler
      Runtime: nodejs12.x
      CodeSigningConfigArn: !Ref MyCodeSigningConfig

  MyCodeSigningConfig:
    Type: AWS::Lambda::CodeSigningConfig
    Properties:
      Description: "New Code Signing for re:Invent"
      AllowedPublishers:
        SigningProfileVersionArns:
          - MySigningProfile-profileVersionArn
      CodeSigningPolicies:
        UntrustedArtifactOnDeployment: "Enforce"
```

New AWS Lambda condition key

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": ”EnforceCodeSigningConfig",
      "Effect": "Allow",
      "Action": [
          "lambda:CreateFunction",
        ],
      "Resource": "*",
      "Condition": {
          "StringEquals": {
              "lambda:codeSigningConfig: {
                  ”arn:aws:lambda:us-west-2:123456789012:code-signing-config:csc-0d79387a58dc63303”
               }
          }
      }
    }
  ]
}
```

Code signing operator controls

NEW

IAM condition keys

Admins can ensure all functions created in their account or organization use code signing

AWS Config

Admins can detect functions with code signing disabled or with undesired validation policies

Amazon CloudWatch

Admins can use new CloudWatch metric to track signature validation errors per function

AWS CloudTrail

Admins can audit signature details, signature validation errors, and signature status

AWS Lambda code signing API operations

NEW

Control Plane:

CreateCodeSigningConfig

UpdateCodeSigningConfig

DeleteCodeSigningConfig

PutFunctionCodeSigningConfig

DeleteFunctionCodeSigningConfig

Read Only:

ListCodeSigningConfigs

GetCodeSigningConfig

GetFunctionCodeSigningConfig

ListFunctionsByCodeSigningConfig

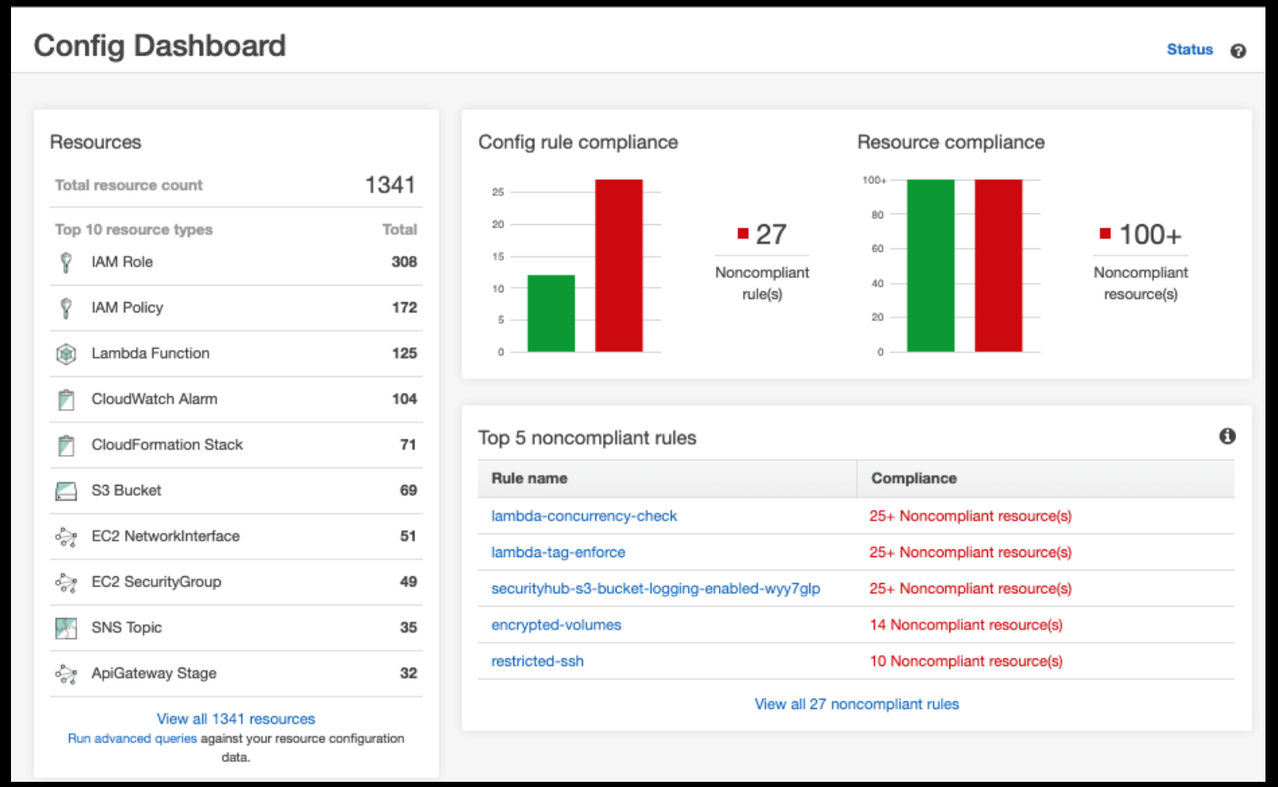
Continuously monitor your configuration



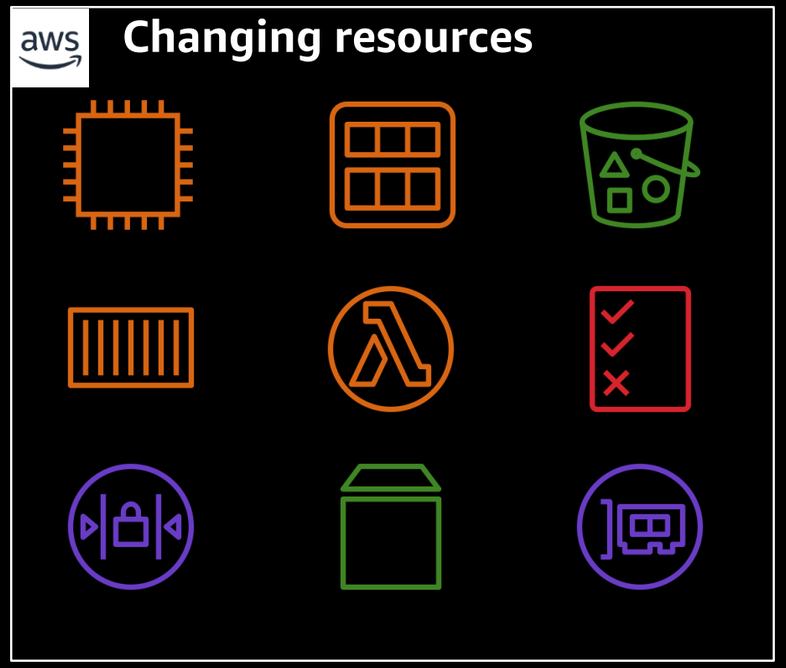
© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Config

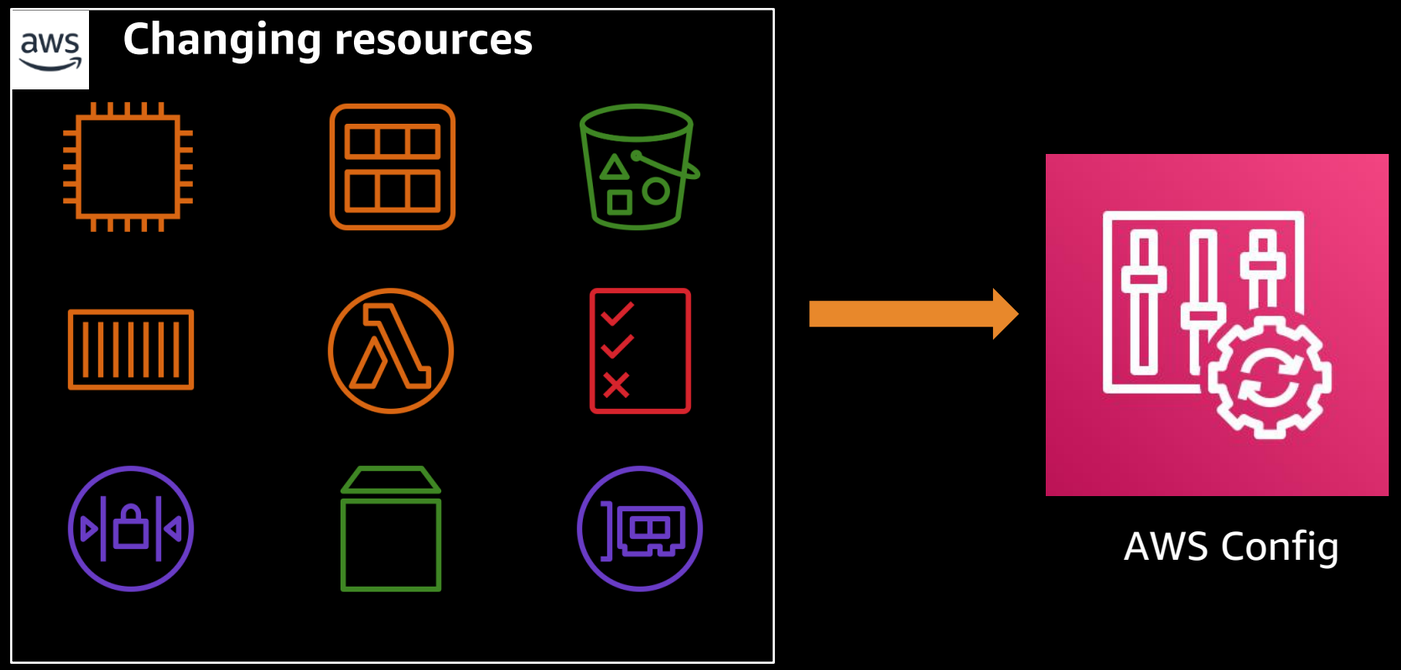
- Configuration auditor
- Monitors configuration changes over time
- Evaluates the configuration against policies defined using AWS Config rules
- Alerts you if the configuration is noncompliant with your policies



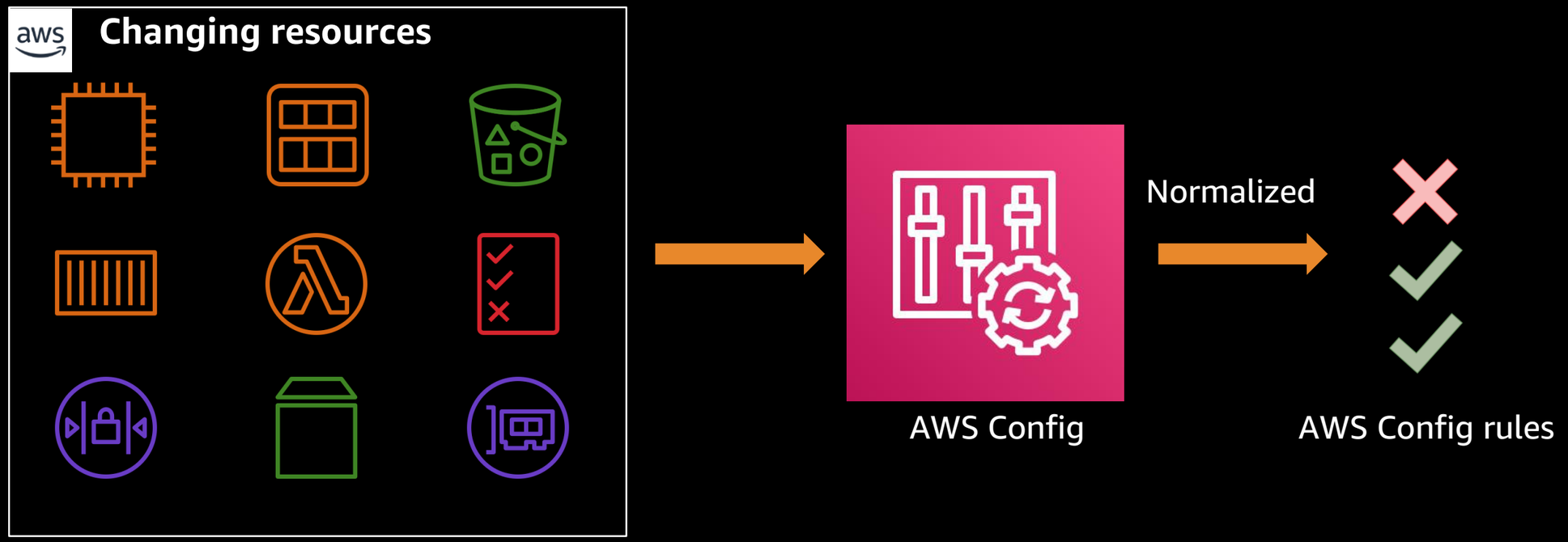
AWS Config rules



AWS Config rules



AWS Config rules



AWS Config rules



AWS Config

Normalized

AWS Config rules

Amazon SNS topic

AWS API endpoint

Amazon EventBridge

AWS Systems Manager automation

Example managed AWS Config rules

lambda-concurrency-check

Checks whether the AWS Lambda function is configured with function-level concurrent execution limit. The rule is NON_COMPLIANT if the Lambda function is not configured with

Lambda

lambda-dlq-check

Checks whether an AWS Lambda function is configured with a dead-letter queue. The rule is NON_COMPLIANT if the Lambda function is not configured with a dead-letter queue.

SNS . Lambda . SQS . DLQ

lambda-function-public-access-prohi…

Checks whether the Lambda function policy prohibits public access. The rule is NON_COMPLIANT if the Lambda function policy allows public access.

Lambda . Zelkova

lambda-function-settings-check

Checks that the AWS Lambda function settings for runtime, role, timeout, and memory size match the expected values.

Lambda

lambda-inside-vpc

Checks whether an AWS Lambda function is in an Amazon Virtual Private Cloud. The rule is NON_COMPLIANT if the Lambda function is not in a VPC.

VPC . Lambda

api-gw-cache-enabled-and-encrypted

Checks that all methods in Amazon API Gateway stages have cache enabled and cache encrypted. The rule is NON_COMPLIANT if any method in Amazon

API Gateway . REST API

api-gw-endpoint-type-check

Checks that Amazon API Gateway APIs are of type as specified in the rule parameter 'endpointConfigurationTypes'. The rule returns COMPLIANT if any of the RestApi

API Gateway . REST API

api-gw-execution-logging-enabled

Checks that all methods in Amazon API Gateway stage has logging enabled. The rule is NON_COMPLIANT if logging is not enabled. The rule is NON_COMPLIANT if

API Gateway . Logging

fms-webacl-resource-policy-check

Checks whether the web ACL is associated with Application Load Balancers, API Gateway stage or CloudFront distributions. When AWS Firewall Manager creates this

FM . FMS . WebACL

Open source custom AWS Config rules

API Gateway:

- Authorizer is in place
- API Gateway Cache is enabled and encrypted
- API Gateway endpoint type check
- API Gateway execution logging enabled
- Private APIs uses resource policy restricting to VPC endpoints or VPC in the same AWS account

Lambda:

- Code is versioned
- Concurrency check
- DLQ Check
- VPC enabled
- Logging to CloudWatch allowed

https://github.com/awslabs/aws-config-rules

Key takeaways

- Use IAM policies and conditions
- Configure permissions boundaries
- Use VPC endpoints and policies where appropriate
- Enable AWS Config and Config rules for reactive governance
  - Enforce authentication, encryption, resource policies, logging, alarms, etc.
- NEW code signing for AWS Lambda

Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.