

Validate IAM policies with Access Analyzer using AWS Config rules

by Anurag Jain, Swara Gandhi, and Matt Luttrell | on 04 OCT 2023 | in [Best Practices](#), [Intermediate \(200\)](#), [Security, Identity, & Compliance](#), [Technical How-to](#) | [Permalink](#) | [Comments](#) | [Share](#)

You can use [AWS Identity and Access Management \(IAM\) Access Analyzer policy validation](#) to validate IAM policies against IAM [policy grammar](#) and [best practices](#). The [findings generated by Access Analyzer policy validation](#) include errors, security warnings, general warnings, and suggestions for your policy. These findings provide actionable recommendations that help you author policies that are functional and conform to security best practices.

You can use the [IAM Policy Validator for AWS CloudFormation](#) and the [IAM Policy Validator for Terraform](#) solutions to integrate Access Analyzer policy validation in a proactive manner within your continuous integration and continuous delivery CI/CD pipeline before deploying IAM policies to your [Amazon Web Service \(AWS\)](#) environment. Customers requested a similar capability to validate policies already deployed within their environments as part of the defense-in-depth strategy.

In this post, you learn how to set up and continuously validate and report on compliance of the IAM policies in your environment using [AWS Config](#). [AWS Config](#) evaluates the configuration settings of your AWS resources with the help of [AWS Config rules](#), which represent your ideal configuration settings. AWS Config continuously tracks the configuration changes that occur among your resources and checks whether these changes conform to the conditions in your rules. If a resource doesn't conform to a rule, AWS Config flags the resource and the rule as noncompliant.

You can use this solution to validate identity-based and resource-based IAM policies attached to resources in your AWS environment that might have grammatical or syntactical errors or might not follow AWS best practices. The code used in this post is hosted in a [GitHub repository](#).

Prerequisites

Before you get started, you need:

- An [AWS account](#)
- Basic knowledge of [AWS Config](#), [AWS CloudFormation](#) and [Amazon Simple Storage Service](#)
- [AWS Command Line Interface \(AWS CLI\) installed](#)
- [An Amazon S3 bucket](#)

Step 1: Enable AWS Config to monitor global resources

To get started, enable AWS Config in your AWS account by following the instructions in the [AWS Config Developer Guide](#).

Next, enable the recording of global resources:

1. Open the AWS Management Console and go to the [AWS Config console](#).
2. Go to **Settings** and choose **Edit** to see the AWS Config recorder settings.
3. Under **General settings**, select the **Include globally recorded resource types** to enable [AWS Config](#) to monitor IAM configuration items.
4. Leave the other settings at their defaults.
5. Choose **Save**.

Figure 1: AWS Config settings page showing inclusion of globally recorded resource types

6. After choosing **Save**, you should see **Recording is on** at the top of the window.

Figure 2: AWS Config settings page showing recorder settings

Note: You only need to enable globally recorded resource types in the AWS Region where you've configured AWS Config because they aren't tied to a specific Region and can be used in other Regions. The globally recorded resource types that AWS Config supports are IAM users, groups, roles, and customer managed policies.

Step 2: Deploy the CloudFormation template

In this section, you deploy and test a sample [AWS CloudFormation](#) template that creates the following:

- An AWS Config rule that reports the compliance of IAM policies.
- An [AWS Lambda](#) function that implements and then makes the requests to IAM Access Analyzer and returns the policy validation findings.
- An IAM role that's used by the Lambda function with permissions to validate IAM policies using the Access Analyzer ValidatePolicy API.
- An optional [Amazon CloudWatch](#) alarm and [Amazon Simple Notification Service \(Amazon SNS\)](#) topic to provide notification of Lambda function errors.

Follow the steps below to deploy the AWS CloudFormation template:

1. To deploy the CloudFormation template using the following command, you must have the [AWS Command Line Interface \(AWS CLI\) installed](#).
2. Make sure you have configured your [AWS CLI credentials](#).
3. Clone the solution repository.

```
git clone https://github.com/aws-labs/aws-iam-access-analyzer-policy-validation-confi
```

4. Navigate to the iam-access-analyzer-config-rule folder of the cloned repository.

```
cd aws-iam-access-analyzer-policy-validation-config-rule
```

5. Deploy the CloudFormation template using the AWS CLI.

Note: Change the Region for the parameter — `RegionToValidateGlobalResources` — to the Region you enabled for global resources in Step 1. Optionally, you can add an email address if you want to receive notifications if the AWS Config rule stops working. Use the code that follows, replacing `<us-east-1>` with the Region you enabled and `<EMAIL_ADDRESS>` with your chosen address.

```
aws cloudformation deploy \  
  --stack-name iam-policy-validation-config-rule \  
  --template-file templates/template.yaml \  
  --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM \  
  --region us-east-1
```

```
--parameter-overrides RegionToValidateGlobalResources='<us-east-1>' \
ErrorNotificationsEmailAddress='<EMAIL_ADDRESS>'
```

6. After successful deployment, you will see the message Successfully created/updated stack – iam-policy-validation-config-rule.

```
Waiting for changeset to be created..
Waiting for stack create/update to complete
Successfully created/updated stack – iam-policy-validation-config-rule
```

Figure 3: Successful CloudFormation stack creation reported on the terminal

Note: If the CloudFormation stack creation fails, go to the [CloudFormation console](#) and select the **iam-policy-validation-config-rule** stack. Choose **Events** to review the failure reason.

7. After deployment, open the [CloudFormation console](#) and select the **iam-policy-validation-config-rule** stack.
8. Choose **Resources** to see the resources created by the template.

Step 3: Check noncompliant resources discovered by AWS Config

The AWS Config rule is designed to mark resources that have IAM policies as noncompliant if the resources have validation findings found using the IAM Access Analyzer [ValidatePolicy API](#).

1. Open the [AWS Config console](#)
2. Choose **Rules** from the navigation pane on the left and select **policy-validation-config-rule**.

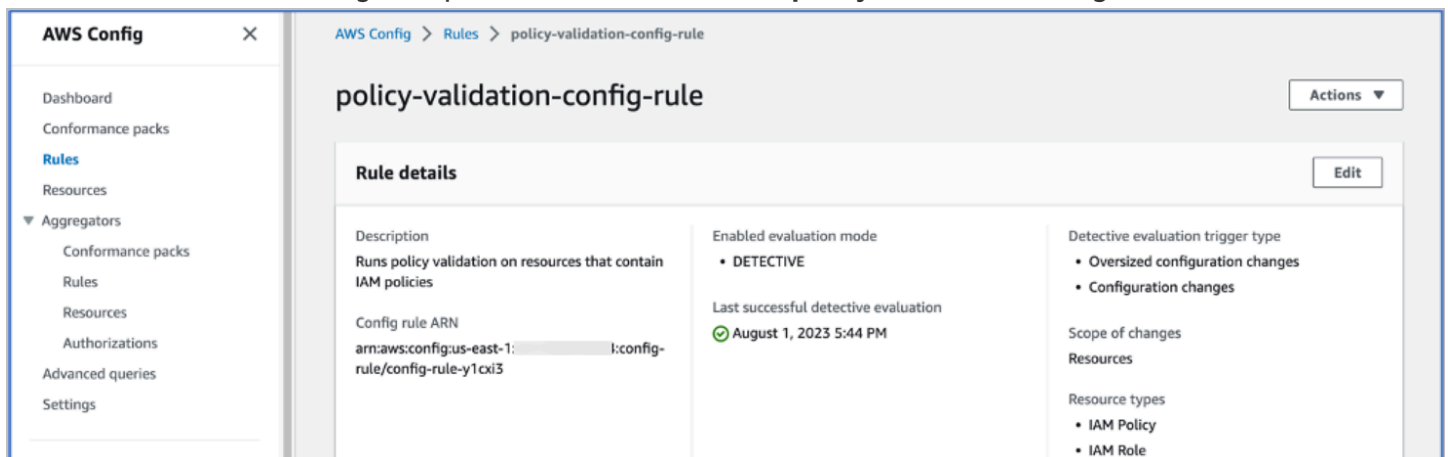


Figure 4: AWS Config rules page showing the rule details

3. Scroll down on the page and filter **Resources in Scope** to see the noncompliant resources.

Resources in scope					
Noncompliant					
ID	Type	Status	Annotation	Compliance	
ConfigAccessPolicy	IAM Policy	-	2 noncompliant finding(s) with issue codes: INVALID_ACTION	Noncompliant	
IAuditorRole-DO-NOT-DELE...	IAM Role	-	6 noncompliant finding(s) with issue codes: INVALID_ACTION	Noncompliant	
InternalAudit	IAM Role	-	2 noncompliant finding(s) with issue codes: MISSING_QUALIFIER	Noncompliant	
Admin1	IAM Role	-	1 noncompliant finding(s) with issue codes: PASS_ROLE_WITH_STAR_IN_ACTION_AND_RESOURCE	Noncompliant	
dRole-DO-NOT-DELETE	IAM Role	-	1 noncompliant finding(s) with issue codes: PASS_ROLE_WITH_STAR_IN_ACTION_AND_RESOURCE	Noncompliant	

Figure 5: AWS Config rules page showing noncompliant resources

Note: If the AWS Config rule isn't invoked yet, you can choose **Actions** and select **Re-evaluate** to invoke it.

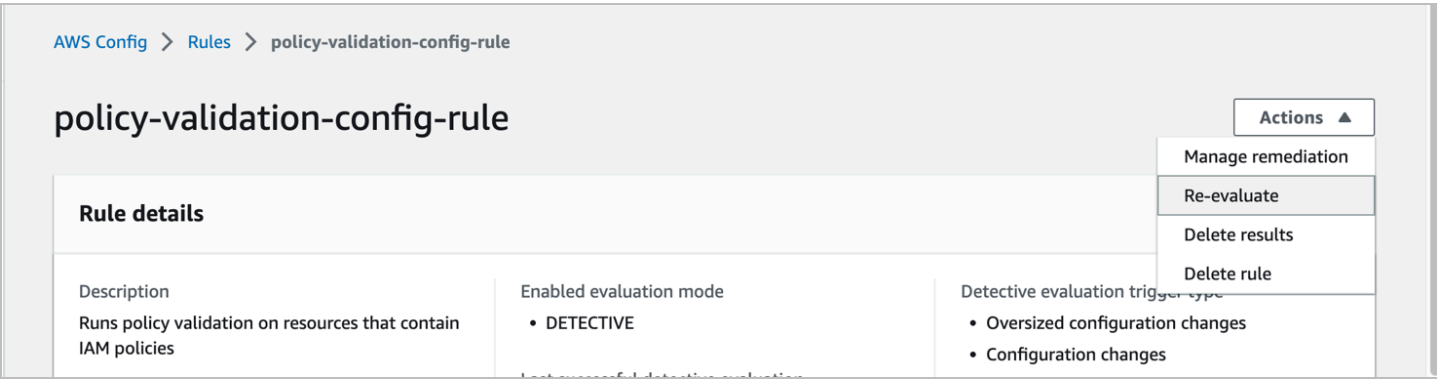


Figure 6: AWS Config rules page showing evaluation invocation

Step 4: Modify the AWS Config rule for exceptions

You might want to exempt certain resources from specific policy validation checks. For example, you might need to deploy a more privileged role—such as an administrator role—to your environment and you don't want that role's policies to have policy validation findings.

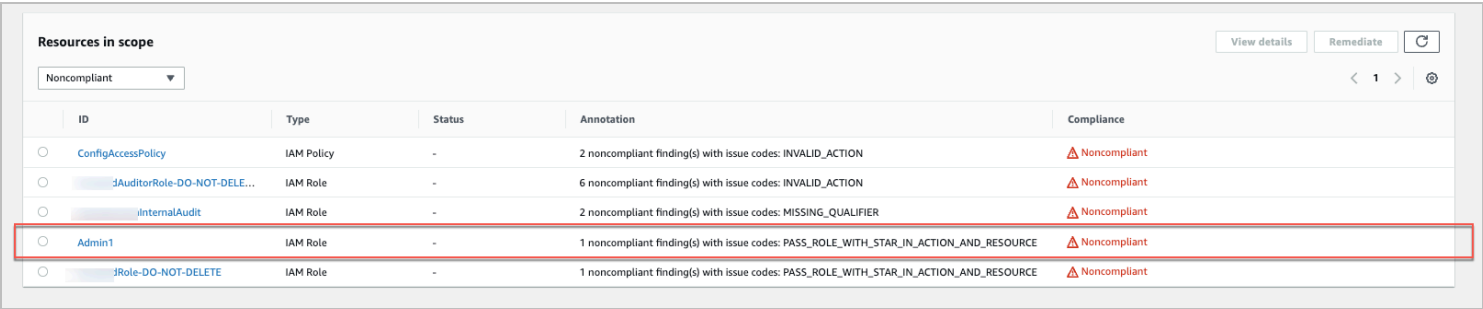


Figure 7: AWS Config rules page showing a noncompliant administrator role

This section shows you how to configure an exceptions file to exempt specific resources.

1. Start by configuring an exceptions file similar to the one that follows to log general warning findings across the accounts in your organization to make sure your policies conform to best practices by setting **ignoreWarningFindings** to **False**.
2. Additionally, you might want to create an exception that allows administrator roles to use the `iam:PassRole` action on another role. This combination of action and resource is usually reserved for privileged users. The example file below shows an exception for all the roles created with `Administrator` in the role path from account `12345678912`.

Example exceptions file:

```
JSON
{
  "global": {
```

```

    "ignoreWarningFindings": false
  },
  "12345678912": {
    "ignoreFindingsWith": [
      {
        "issueCode": "PASS_ROLE_WITH_STAR_IN_ACTION_AND_RESOURCE",
        "resourceType": "AWS::IAM::Role",
        "resourceName": "Administrator/*"
      }
    ]
  }
}

```

3. After the exceptions file is ready, upload the JSON file to the S3 bucket you created as a part of the prerequisites.

You can manage this exceptions file by hosting it in a central Git repository. When teams need to exempt a particular resource from these policy validation checks, they can submit a pull request to the central repository. An approver can then approve or reject this request and, if approved, deploy the updated exceptions file.

4. Modify the bucket policy so that the bucket is accessible to your AWS Config rule if the rule is operating in a different account than the bucket was created in. Below is an example of a bucket policy that allows the accounts in your organization to read the exceptions file.

```

{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": "*"},
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::EXAMPLE-BUCKET/my-exceptions-file.json",
    "Condition": {
      "StringEquals": {
        "aws:PrincipalOrgId": "<your organization id here>"
      }
    }
  }]
}

```

Note: For more examples visit [example policy validation exceptions file contents](#).

5. Deploy the CloudFormation template again using the `ExceptionsS3BucketName` and `ExceptionsS3FilePrefix` parameters. The file prefix should be the full prefix of the S3 object exceptions file.

```
aws cloudformation deploy \
  --stack-name iam-policy-validation-config-rule \
  --template-file templates/template.yaml \
  --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM \
  --parameter-overrides RegionToValidateGlobalResources='<us-east-1>' \
    ExceptionsS3BucketName='EXAMPLE-BUCKET' \
    ExceptionsS3FilePrefix='my-exceptions-file.json'
```

6. After you see the Successfully created/updated stack – iam-policy-validation-config-rule message on the terminal or command line and the AWS Config rule has been re-evaluated, the resources mentioned in the exception file should show as **Compliant**.

<input type="radio"/>	MetricStreams-FirehoseToS3-IPR45877	IAM Role	-	Policy has no validation findings.	Compliant
<input type="radio"/>	Admin1	IAM Role	-	Policy has no validation findings.	Compliant
<input type="radio"/>	AVMContainersUserRole	IAM Role	-	Policy has no validation findings.	Compliant
<input type="radio"/>	policy-validation-config-rule	IAM Role	-	Policy has no validation findings.	Compliant
<input type="radio"/>	AwsSecurityAudit	IAM Role	-	Policy has no validation findings.	Compliant

Figure 8: Resource exception result

You can find additional customization options in the [exceptions file schema](#).

Cleanup

To avoid recurring charges and to remove the resources used in testing the solution outlined in this post, use the CloudFormation console to delete the **iam-policy-validation-config-rule** CloudFormation stack.

The screenshot shows the AWS CloudFormation console. On the left, the 'Stacks' section is active, displaying a list of stacks. The stack 'iam-policy-validation-config-rule' is highlighted, showing its status as 'CREATE_COMPLETE'. In the main panel, the 'Stack info' tab is selected for the 'iam-policy-validation-config-rule' stack. The 'Delete' button is highlighted with a red box, indicating the action to be taken to remove the stack. The 'Overview' section on the right provides details about the stack, including its ID, status, and creation time.

Figure 9: AWS CloudFormation stack deletion

Conclusion

In this post, we demonstrated how you can set up a centralized compliance and monitoring workflow using AWS IAM Access Analyzer policy validation with AWS Config rules to validate identity-based and resource-based policies attached to resources in your account. Using this solution, you can create a single pane of glass to monitor resources and govern centralized compliance for AWS Config-supported resources across accounts. You can also build and maintain exceptions customized to your environment as shown in the [example policy validation exceptions file](#). You can visit the [Access Analyzer policy checks reference page](#) for a complete list of policy check validation errors and resolutions.

If you have feedback about this post, submit comments in the **Comments** section below. If you have questions about this post, [contact AWS Support](#).

Want more AWS Security news? Follow us on [Twitter](#).

TAGS: [AWS Config](#), [AWS IAM Access Analyzer](#), [AWS Identity and Access Management](#), [Security Blog](#)

Comments

ALSO ON AWS SECURITY BLOG

Hardening the RAG chatbot ...

17 days ago • 1 comment

Mitigate risks like data exposure, model exploits, and ethical lapses when ...

Access AWS services programmatically ...

2 months ago • 1 comment

With the introduction of trusted identity propagation, applications can now ...

How the unique culture of security ...

4 months ago • 1 comment

Our customers depend on Amazon Web Services (AWS) for their ...

How to 2.0 in A


5 months a

Implemer authenticat authorizat

o Comments

4

Prashanth ▼



Start the discussion...

 Share

Best

NewestOldest

Be the first to comment.

Subscribe

Privacy

Do Not Sell My Data