AWS News Blog

AWS Config Rules – Dynamic Compliance Checking for Cloud Resources

by Jeff Barr | on 07 OCT 2015 | in AWS Config, AWS Re:Invent | Permalink | Share

The flexible, dynamic nature of the AWS cloud gives developers and admins the flexibility to launch, configure, use, and terminate processing, storage, networking, and other resources as needed. In any fast-paced agile environment, security guidelines and policies can be overlooked in the race to get a new product to market before the competition.

Imagine that you had the ability to verify that existing and newly launched AWS resources conformed to your organization's security guidelines and best practices without creating a bureaucracy or spending your time manually inspecting cloud resources.

Last year I announced that you could Track AWS Resource Configurations with AWS Config. In that post I showed you how AWS Config captured the state of your AWS resources and the relationships between them. I also discussed Config's auditing features, including the ability to select a resource and then view a timeline of configuration changes on a timeline.

New AWS Config Rules

Today we are extending Config with a powerful new rule system. You can use existing rules from AWS and from partners, and you can also define your own custom rules. Rules can be targeted at specific resources (by id), specific types of resources, or at resources tagged in a particular way. Rules are run when those resources are created or changed, and can also be evaluated on a periodic basis (hourly, daily, and so forth).

Rules can look for any desirable or undesirable condition. For example, you could:

- Ensure that EC2 instances launched in a particular VPC are properly tagged.
- Make sure that every instance is associated with at least one security group.
- Check to make sure that port 22 is not open in any production security group.

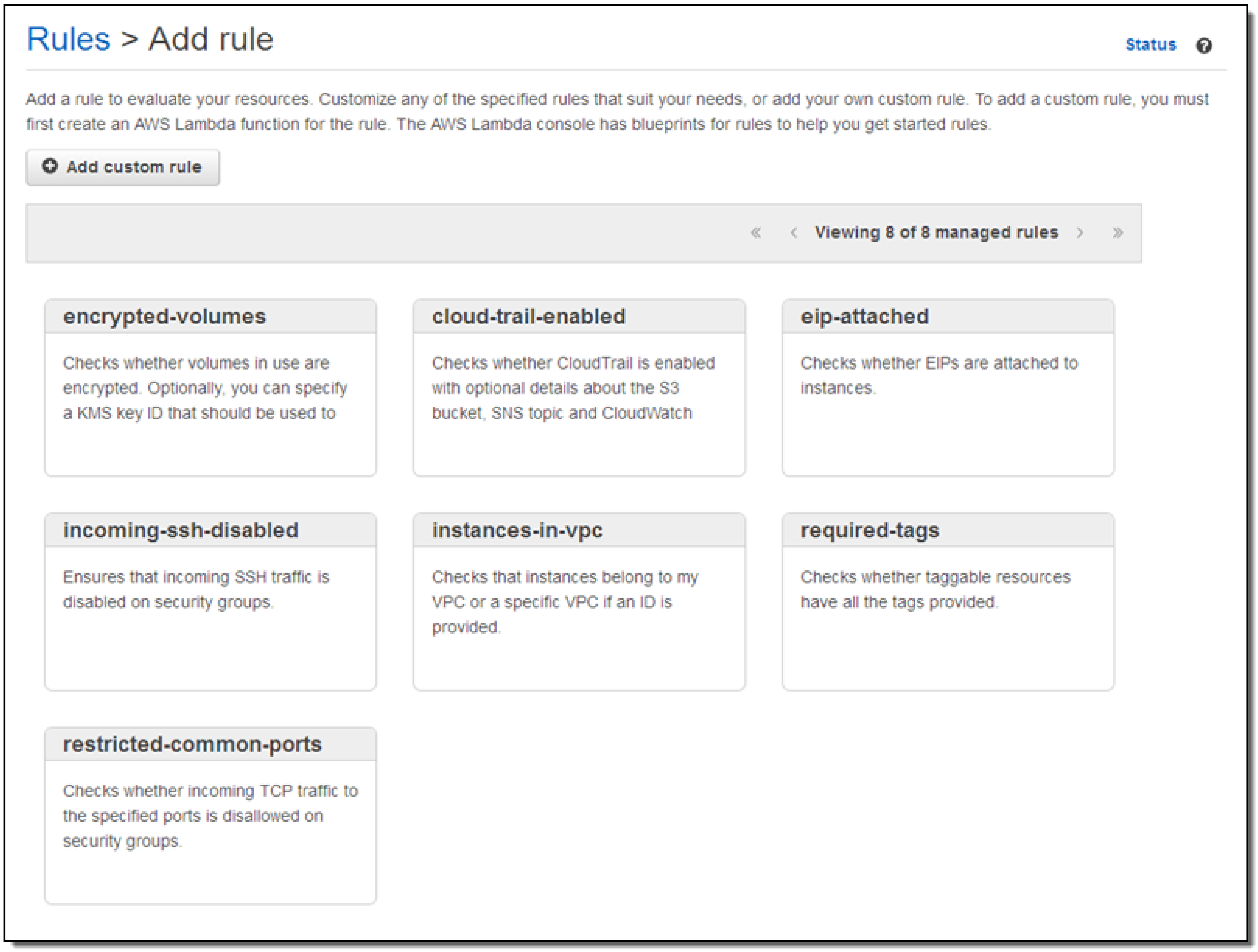
Each custom rule is simply an AWS Lambda function. When the function is invoked in order to evaluate a resource, it is provided with the resource's Configuration Item. The function can inspect the item and can also make calls to other AWS API functions as desired (based on permissions granted via an IAM role, as usual). After the Lambda function makes its decision (compliant or not) it calls the `PutEvaluations` function to record the decision and returns.

The results of all of these rule invocations (which you can think of as compliance checks) are recorded and tracked on a per-resource basis and then made available to you in the AWS Management Console. You can also access the results in a report-oriented form, or via the Config API.

Let's take a quick tour of AWS Config Rules, with the proviso that some of what I share with you will undoubtedly change as we progress toward general availability. As usual, we will look forward to your feedback and will use it to shape and prioritize our roadmap.

Using an Existing Rule

Let's start by using one of the rules that's included with Config. I open the Config Console and click on **Add Rule**:



Rules > Add rule

Add a rule to evaluate your resources. Customize any of the specified rules that suit your needs, or add your own custom rule. To add a custom rule, you must first create an AWS Lambda function for the rule. The AWS Lambda console has blueprints for rules to help you get started rules.

encrypted-volumes
Checks whether volumes in use are encrypted. Optionally, you can specify a KMS key ID that should be used to

cloud-trail-enabled
Checks whether CloudTrail is enabled with optional details about the S3 bucket, SNS topic and CloudWatch

eip-attached
Checks whether EIPs are attached to instances.

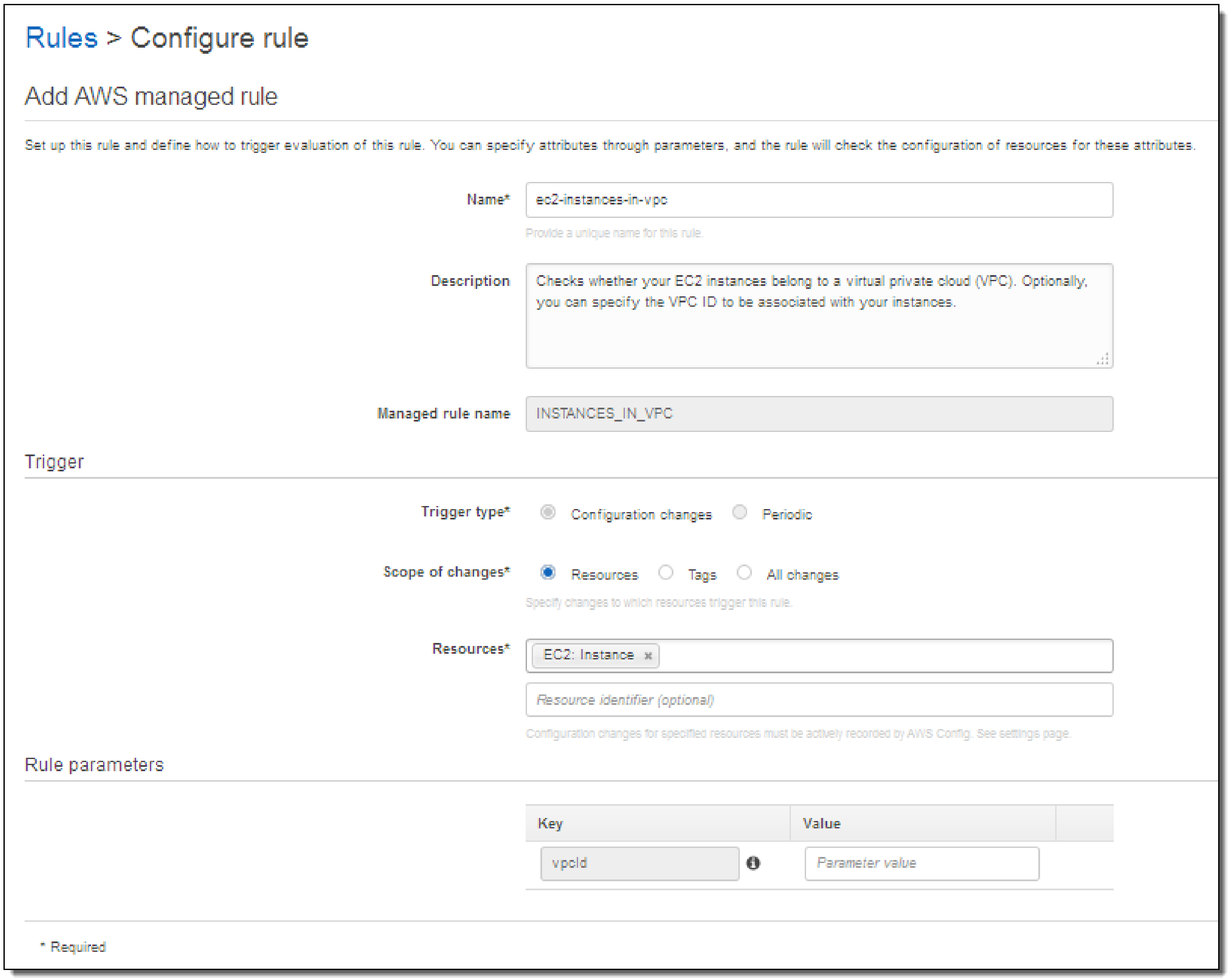
incoming-ssh-disabled
Ensures that incoming SSH traffic is disabled on security groups.

instances-in-vpc
Checks that instances belong to my VPC or a specific VPC if an ID is provided.

required-tags
Checks whether taggable resources have all the tags provided.

restricted-common-ports
Checks whether incoming TCP traffic to the specified ports is disallowed on security groups.

I browse through the rules and decide to start with **instances-in-vpc**. This rule verifies that an EC2 instance belong to a VPC, with the option to check that it belongs to a specific VPC. I click on the rule and customize it as needed:


Rules > Configure rule

Add AWS managed rule

Set up this rule and define how to trigger evaluation of this rule. You can specify attributes through parameters, and the rule will check the configuration of resources for these attributes.

Name*: ec2-instances-in-vpc
Provide a unique name for this rule.

Description: Checks whether your EC2 instances belong to a virtual private cloud (VPC). Optionally, you can specify the VPC ID to be associated with your instances.

Managed rule name: INSTANCES_IN_VPC

Trigger

Trigger type*: [✓ Configuration changes] Periodic

Scope of changes*: [✓ Resources] Tags All changes
Specify changes to which resources trigger this rule.

Resources*: EC2: Instance
Resource identifier (optional)
Configuration changes for specified resources must be actively recorded by AWS Config. See settings page.

Rule parameters

Key	Value
vpcid	Parameter value

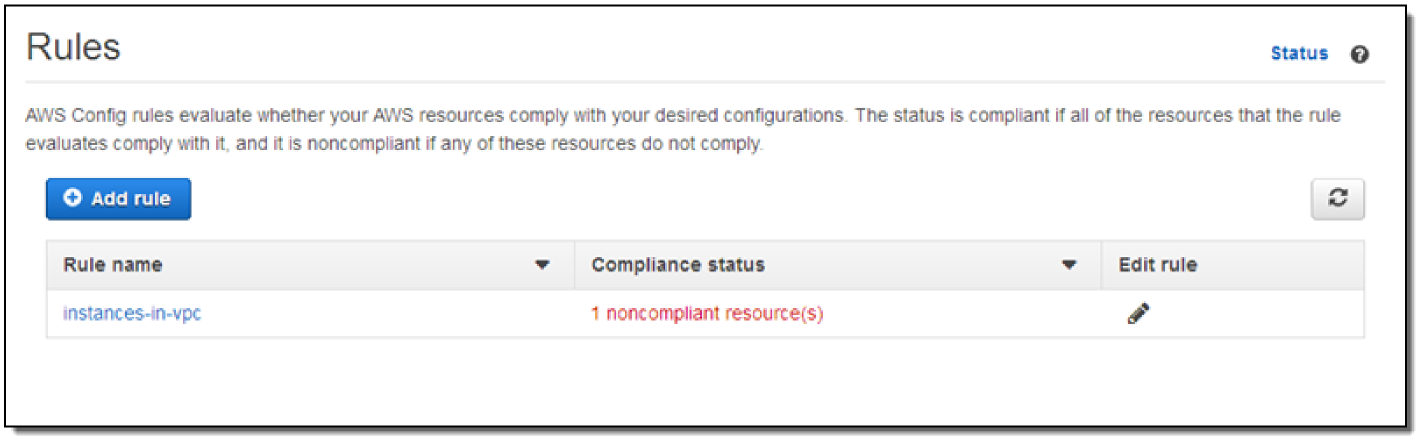
* Required

I have a lot of choices here. The **Trigger type** tells Config to run the rule when the resource is changed, or periodically. The **Scope of changes** tells Config which resources are of interest. The scope can be specified by resource type (with an optional identifier) by tag name, or by a combination of tag name and value. If I am checking EC2 instances, I can trigger on any of the following:

- All EC2 instances.
- Specific EC2 instances, identified by a resource identifier.
- All resources tagged with the key “Department.”
- All resources tagged with the key “Stage” and the value “Prod.”

The **Rule parameters** allows me to pass additional key/value pairs to the Lambda function. The parameter names, and their meaning, will be specific to the function. In this case, supplying a value for the **vpcid** parameter tells the function to verify that the EC2 instance is running within the specified VPC.

The rule goes in to effect after I click on **Save**. When I return to the Rules page I can see that my AWS configuration is now noncompliant:

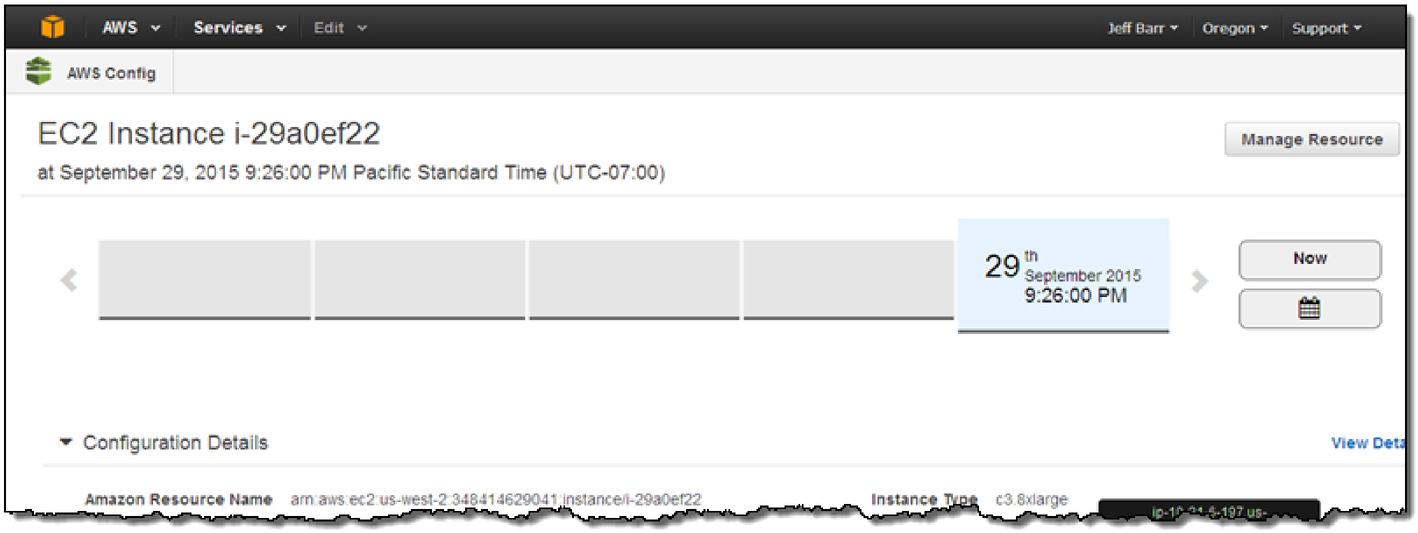


Rules

AWS Config rules evaluate whether your AWS resources comply with your desired configurations. The status is compliant if all of the resources that the rule evaluates comply with it, and it is noncompliant if any of these resources do not comply.

Rule name	Compliance status	Edit rule
instances-in-vpc	1 noncompliant resource(s)	✎

I can investigate the issue by examining the Config timeline for the instance in question:



EC2 Instance i-29a0ef22

at September 29, 2015 9:26:00 PM Pacific Standard Time (UTC-07:00)

29th September 2015 9:26:00 PM

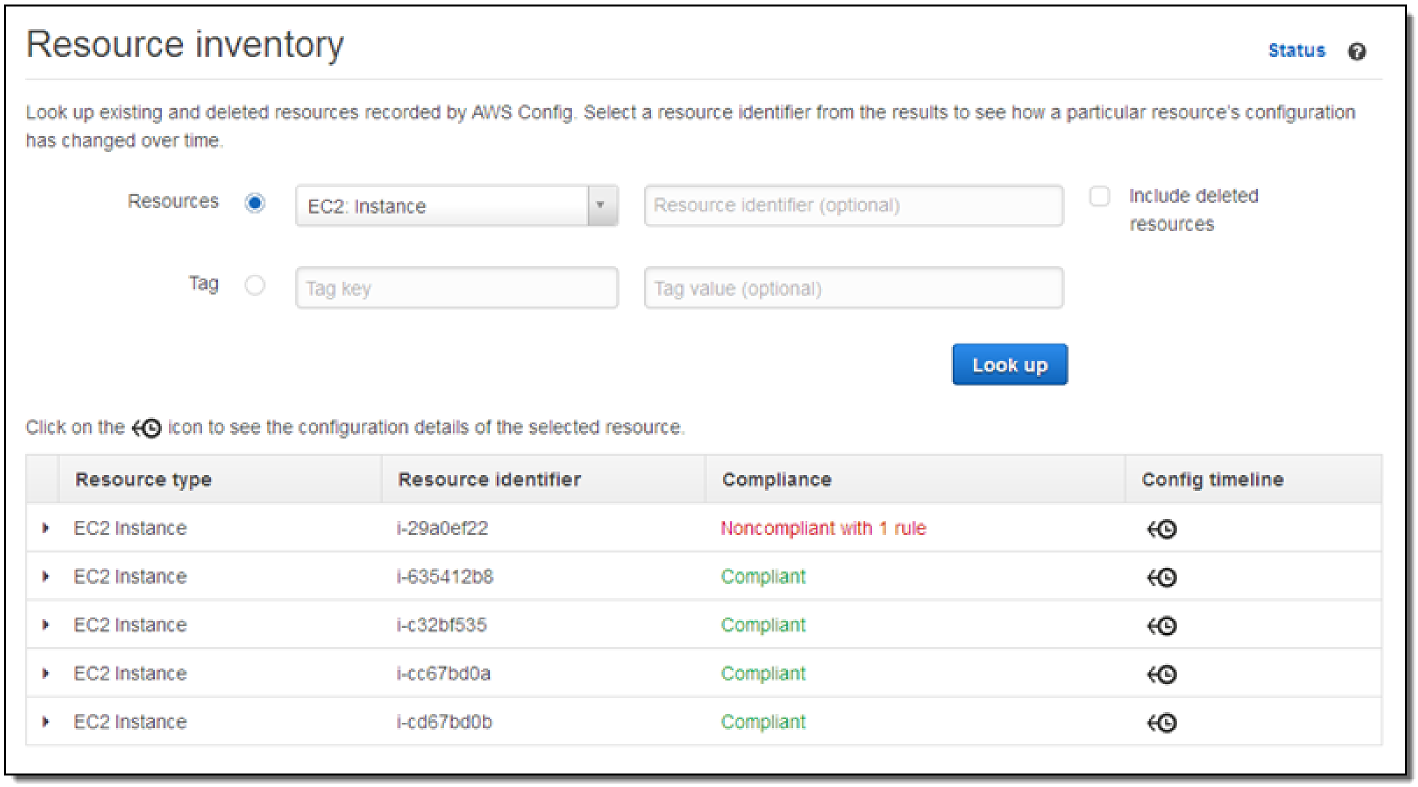
Configuration Details

Amazon Resource Name arn:aws:ec2:us-west-2:348414629041:instance/i-29a0ef22

Instance Type c3.8xlarge

It turns out that this instance has been sitting around for a while (truth be told I forgot about it). This is a perfect example of how useful the new Config Rules can be!

I can also use the Config Console to look at the compliance status of all instances of a particular type:


Resource inventory

Look up existing and deleted resources recorded by AWS Config. Select a resource identifier from the results to see how a particular resource's configuration has changed over time.

Resources [✓] EC2: Instance | Resource identifier (optional) | Include deleted resources

Tag | Tag key | Tag value (optional)

Click on the icon to see the configuration details of the selected resource.

Resource type	Resource identifier	Compliance	Config timeline
EC2 Instance	i-29a0ef22	Noncompliant with 1 rule	⟲
EC2 Instance	i-635412b8	Compliant	⟲
EC2 Instance	i-c32bf535	Compliant	⟲
EC2 Instance	i-cc67bd0a	Compliant	⟲
EC2 Instance	i-cd67bd0b	Compliant	⟲

Creating a New Rule

I can create a new rule using any language supported by Lambda. The rule receives the Configuration Item and the rule parameters that I mentioned above, and can implement any desired logic.

Let's look at a couple of excerpts from a sample rule. The rule applies to EC2 instances, so it checks to see if was invoked on one:

JavaScript
```javascript
function evaluateCompliance(configurationItem, ruleParameters) {
    if (configurationItem.resourceType !== 'AWS::EC2::Instance') {
        return 'NOT_APPLICABLE';
    } else {
        var securityGroups = configurationItem.configuration.securityGroups;
        var expectedSecurityGroupId = ruleParameters.securityGroupId;
        if (hasExpectedSecurityGroup(expectedSecurityGroupId, securityGroups)) {
            return 'COMPLIANT';
        } else {
            return 'NON_COMPLIANT';
        }
    }
}
```

If the rule was invoked on an EC2 instance, it checks to see if any one of a list of expected security groups is attached to the instance:

JavaScript

```
function hasExpectedSecurityGroup(expectedSecurityGroupId, securityGroups) {
    for (var i = 0; i < securityGroups.length; i++) {
        var securityGroup = securityGroups[i];
        if (securityGroup.groupId === expectedSecurityGroupId) {
            return true;
        }
    }
    return false;
}
```

Finally, the rule stores the result of the compliance check by calling the Config API's putEvaluations function:

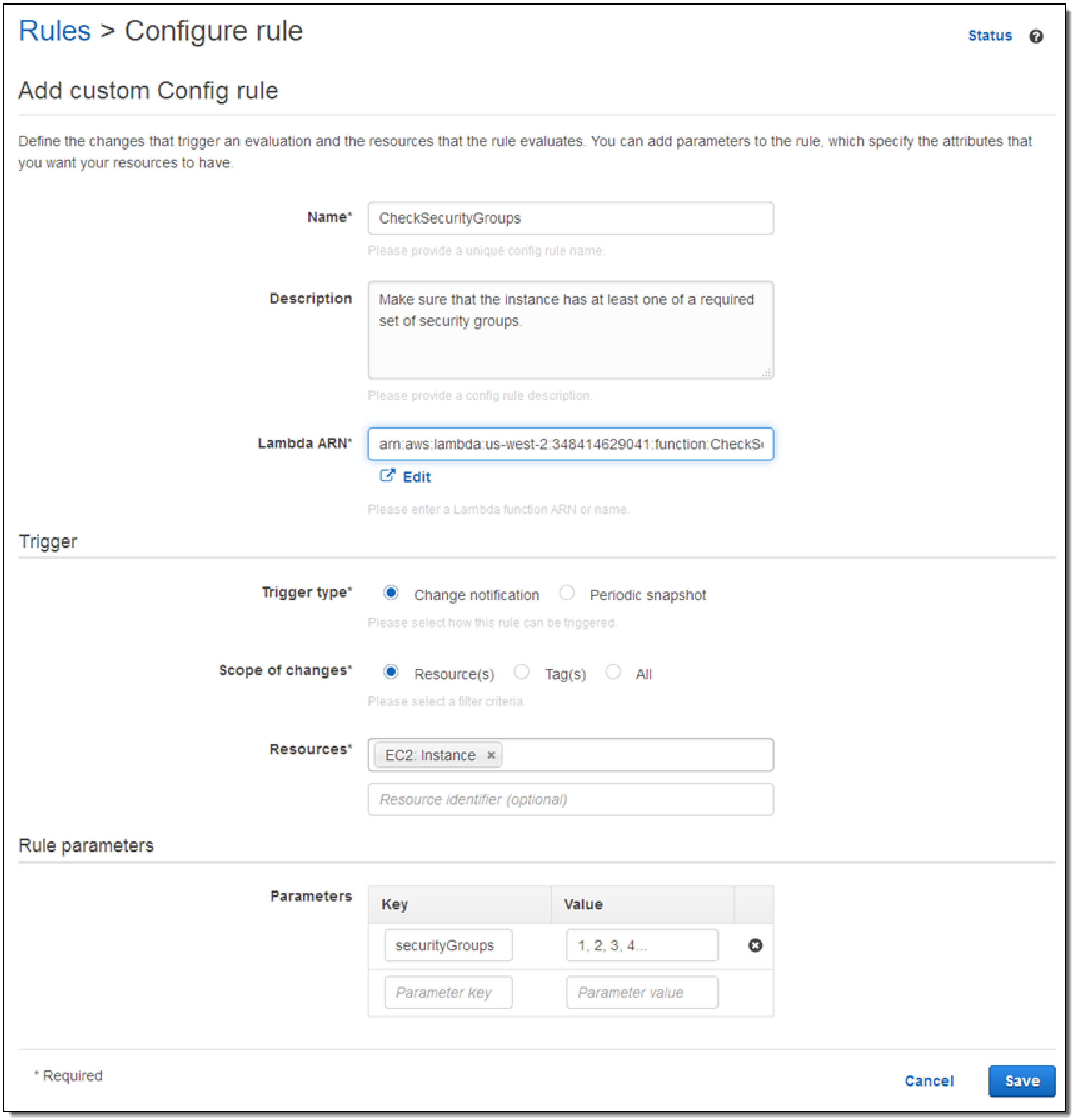
JavaScript

```

config.putEvaluations(putEvaluationsRequest, function (err, data) {
    if (err) {
        context.fail(err);
    } else {
        context.succeed(data);
    }
});
```

The rule can record results for the item being checked or for any related item. Let's say you are checking to make sure that an Elastic Load Balancer is attached only to a specific kind of EC2 instance. You could decide to report compliance (or noncompliance) for the ELB or for the instance, depending on what makes the most sense for your organization and your compliance model. You can do this for any resource type that is supported by Config.

Here's how I create a rule that references my Lambda function:


Rules > Configure rule

Status

Add custom Config rule

Define the changes that trigger an evaluation and the resources that the rule evaluates. You can add parameters to the rule, which specify the attributes that you want your resources to have.

Name*	CheckSecurityGroups
	Please provide a unique config rule name.
Description	Make sure that the instance has at least one of a required set of security groups.
	Please provide a config rule description.
Lambda ARN*	arn:aws:lambda:us-west-2:348414629041:function:CheckS
	Edit
	Please enter a Lambda function ARN or name.

Trigger

Trigger type*	[● Change notification] ○ Periodic snapshot
	Please select how this rule can be triggered.
Scope of changes*	[● Resource(s)] ○ Tag(s) ○ All
	Please select a filter criteria.
Resources*	EC2: Instance ×
	Resource identifier (optional)

Rule parameters

Parameters

Key	Value
securityGroups	1, 2, 3, 4...
Parameter key	Parameter value

* Required

Cancel Save

On the Way

AWS Config Rules are being launched in preview form today and you can sign up now. Stay tuned for additional information!

— Jeff;

PS – re:Invent attendees can attend session SEC 314: Use AWS Config Rules to Improve Governance of Your AWS Resources (5:30 PM on October 8th in Palazzo K).

TAGS: AWS re:Invent

Resources

Getting Started
What's New
Top Posts
Official AWS Podcast
Case Studies

Follow

Twitter
Facebook
LinkedIn
Twitch
RSS Feed
Email Updates

New Launches From re:Invent
Discover the latest services and features from AWS
Visit the News Blog »

Related Posts

Viewing permission issues with service-linked roles

DevSecOps for auto healing PCI DSS 3.2.1 violations in AWS using custom AWS Config conformance packs, AWS Systems Manager and AWS CodePipeline

Automate FedRAMP controls in your AWS environment using AWS Config conformance packs

Automate Amazon S3 versioning using AWS Config rules

How data recipients can implement Open Banking on AWS

Audit your SAP systems with AWS Config – Part II

Audit your SAP systems with AWS Config – Part I

AWS Management and Governance at Re:Invent 2020