



ITIL Change Management for AWS

Table of Contents

01. Executive summary	3
02. Change management	4
CMDB:	5
Time-based changes	5
Configuration update and dependency	5
Types of changes on AWS	6
Standard changes	6
Normal changes	6
03. Infrastructure delta	7
04. Tools for change management	7
AWS CloudTrail	8
AWS Config	9
05. Automation	10
06. nOps.io	11

Executive summary

There is considerable discussion on implementing effective change management in the cloud. Some opinions go so far as declaring ITIL change management incompatible with DevOps. One thing is certain, AWS has changed developers' and infrastructure teams' expectations of what is possible and when.

Of course, infrastructure teams are still responsible for maintaining reliability, but this gets challenging when there are 100-times more changes than with physical infrastructure. With multiple developers launching resources on AWS using different workflows, things can spiral out of control quickly. It's hard to gauge the impact of each change, and there are typically wasted resources and time cleaning up unused resources.

Lean thinking calls for eliminating Muda (the Japanese word for waste). Waste is anything that doesn't add value to the customer. Companies move to the cloud so they can utilize its dynamic nature, launching resources, as needed. But this practice hasn't worked on AWS, as expected. Teams launch resources on AWS like they are free. We see teams launch resources like mytest1 and mytest2, and suddenly, their account is full of test instances. A year later, everyone is afraid to stop these instances because no one knows what might break. Normally, when resources get out of hand, a company puts together a SWAT team to eliminate waste on the cloud. This leads to the company wasting more time trying to eliminate waste. If that's the position your organization is in, you need a change management process.

Security is another major concern on AWS. Cyberattacks are on the rise, and every company is a target. The number of security incidents increased 40% in 2016 (Bloomberg), with the cost of a data breach costing companies an average of \$4 million per incident (IBM). Automation on AWS can help you respond quickly to attacks, but automation could be a pain. When things don't work, people usually log in and make "temporary" changes for debugging, and then forget to revert these changes. You need a system to monitor all of your infrastructure changes to ensure they are in compliance with the configurations specified in your internal guidelines. In addition to going through the code approval process for infrastructure changes, like git pull requests for all your infrastructure, you need to make sure actual resources provisioned through automation are in compliance, and they remain in compliance after provisioning.

This is where change management comes in.

Change management

An essential objective of the change management process is to ensure all changes made by IT cause minimum disruption and meet internal guidelines. As noted, when working with AWS, there are often multiple engineers making changes, so it's hard to track who is doing what. With change management, all AWS changes are reviewed and approved, simultaneously helping reduce waste and bringing visibility to changes for improved security and reliability. It can save hours for auditors, assessors, or anyone else working in your value stream.

But to have effective change management in a dynamic environment like AWS, there are essential components you need, like auto-discovery change management database (CMDB), an infrastructure delta, and standard and manual change filters.

Let's talk about what these components look like on AWS.



CMDB

The change management database (CMDB) needs to auto update on AWS, registering and storing information that other services need to interact with (e.g., IP address, port numbers, URIs). This solves the manual nature of the ITIL CMDB and is absolutely necessary when services are comprised of hundreds (or thousands, or even millions) of nodes, each with dynamically assigned IP addresses.

CMDB provides config history for each configuration item. This is needed since cloud changes are so rapid, and resources come up and go down all the time. On AWS, there are also recurring events, such as resources running out of limits and scheduled retirements from AWS. CMDB should always be monitoring for these events. Let's talk about these items in bit more detail.

Time-based changes

In the cloud, some changes might become noncompliant after a specified duration, regardless of whether they were regular changes originally. For example, your engineers might generate an access key to launch various resources; they need that key for day-to-day operation. However, if they never use the key again and forget to delete it, this suddenly becomes a security issue. In another example, your team may launch resources for a specified duration, but if they forget to shut the resources down, the resources will end up getting wasted. CMDB needs to regularly update config items to make sure they are compliant with company policy, and to increase security and eliminate waste.

Configuration update and dependency

Although your team may have a good practice in place for reviewing and testing your infrastructure code before you apply it, you should review changes and dependencies after resources have been provisioned. CMDB should provide context for all configuration updates and dependencies, so you can ensure the launched resources are compliant and meet the original intent.

Types of changes on AWS

Standard changes

Standard changes are low-risk changes. For example, automated deployment, autoscaling, and ephemeral infrastructure for testing are all changes that can be pre-approved. However, you should log the changes for traceability. To ensure these repeated changes don't have any effect on infrastructure reliability or security, you need to have a config history of similar changes.

Normal changes

In traditional environments, these are higher-risk changes that require review or approval from the agreed upon change authority. In many organizations, this responsibility is inappropriately placed on the change advisory board (CAB) or emergency change advisory board (ECAB), which may lack the required expertise to understand the full impact of the change.

Therefore, context is extremely important. Without the context, the CAB will spend hours trying to figure out the impact of the changes, or might approve the changes without knowing the impact.

Infrastructure delta

Most companies leverage auto scaling heavily, where new resources can be launched based on specified metrics. However, this behavior makes approving changes difficult. If new resources are appearing and disappearing, how do you track what truly changed in the infrastructure? To gain an accurate picture, we need an infrastructure delta, which shows resource updates for the still-running resources for a defined period.

Tools for change management

There are several AWS tools that help you gain better visibility into all changes made to your infrastructure. These tools not only give you the ability to keep track of all your AWS resource configurations, but they also help you gain insights into the business impact of infrastructure changes and reduce the risk of experiencing service disruptions.

03



04


CloudTrail

AWS CloudTrail is an indispensable tool for your AWS account. It allows you to manage governance, compliance, operational auditing, and risk auditing, and also gives you a history of AWS API calls made on your account so you can quickly track all your infrastructure changes. Although CloudTrail is a regional service, you can enable it across your entire cloud environment with one click.

API Activity History

Look up API activity captured for your AWS account in the last 7 days. Filter using one of the attributes to troubleshoot operational issues or security incidents.

Filter: Select attribute ▼ Enter lookup value
Time range: Select time range 

Event time	User name	Event name	Resource type	Resource name
▼ 2015-06-15, 11:06:29 AM	root	UpdateTrail	Topic and 2 more	cloudtrail-notifications and ...
<div style="display: flex; justify-content: space-between;"> <div> AWS access key [redacted] AWS region us-east-1 Error code Event ID 0c8a453e-4044-4bbe-b67f-317593a94973 Event name UpdateTrail </div> <div> Event source cloudtrail.amazonaws.com Event time 2015-06-15, 11:06:29 AM Request ID dd11c8fe-1380-11e5-8f98-ad37d834e999 Source IP address [redacted] User name root </div> </div>				
Resources Referenced (3)				
cloudtrail-notifications Topic	[redacted] cloudtrail Bucket	Default Trail		
View event				
▶ 2015-06-15, 10:55:04 AM	root	StartLogging	Trail	Default
▶ 2015-06-15, 10:55:04 AM	root	CreateTrail	Bucket and 1 more	[redacted] and 1 mo...
▶ 2015-06-15, 10:51:38 AM	root	ConsoleLogin		
No more events				

AWS CloudTrail console in AWS.

AWS Config

AWS Config allows you to assess, audit, and evaluate the configurations of your AWS resources. The service continuously monitors and records your AWS resource configurations and enables you to evaluate your change management process against desired configurations.

Unlike AWS CloudTrail, AWS Config requires you to enable it across multiple regions and accounts using tools like Terraform. You can also write custom Lambda rules based on your needs.

The screenshot displays the AWS Config console interface for the resource **EC2 VPC vpc-f60ce293**. The top navigation bar shows the AWS Config service and the resource name. Below the resource name, there is a timeline of configuration changes. The timeline shows five events, all occurring on November 7th, 2014, at various times between 12:05 PM and 12:40 PM. The events are numbered 1 through 5. A calendar widget is visible, showing the date November 7th, 2014, is selected.

Below the timeline, the **Configuration Details** section is expanded, showing the following information:

- Amazon Resource Name:** arn:aws:ec2:us-west-2:203559510903:vpc/vpc-f60ce293
- Resource type:** AWS::EC2::VPC
- Resource ID:** vpc-f60ce293
- Availability zone:** Multiple Availability Zones
- Created at:** null
- Tags (0):**
- VPC ID:**
- State:**
- VPC CIDR:**
- DHCP Options Set:**
- Default VPC:**
- Instance tenancy:** default

The **Relationships** section is also expanded, showing the following resources:

- EC2 NetworkAcl:** acl-dad937bf
- EC2 NetworkInterface:** eni-00b5a777, eni-041c0573, eni-07bf4e63, eni-09eaf47e, eni-0c7c657b
- EC2 Instance:** i-06da290a, i-081adc04, i-08e82e04, i-0913d505, i-0934f205
- EC2 InternetGateway:** igw-f0608795
- EC2 RouteTable:** rtb-f50ee790

The **Changes (2)** section is expanded, showing the following changes:

- Configuration Changes (0):**
- Relationship Changes (2):**

Field	From	To
INSTANCE	"i-0e28ee02"	null
INSTANCE	null	"i-a431f7a8"

AWS Config showing config history.

Automation

You should record and visualize standard changes in your infrastructure using your change management systems such as Remedy or ServiceNow. You should also use deployment pipeline tools such as Chef, Jenkins and Puppet to perform deployments and record results automatically. Doing so allows everyone in your organization to have visibility into every change that happens in your company.

The ServiceNow logo, featuring the word "servicenow" in a lowercase, sans-serif font.

nOps.io

We built nOps with the objective of automating change management processes for cloud computing environments.

Its change management features and powerful CMDB allow you to review all newly introduced resources quickly. It gives you security and billing context, saving engineers hours looking up this information manually, and saving thousands of dollars by eliminating undocumented, unused resources.

nOps is a proven and trusted tool that provides real-time security monitoring for your AWS infrastructure. nOps can be an essential tool for your organization, helping you implement effective and efficient change management on AWS.

Request a demo at www.nops.io

The screenshot displays the nOps.io web application interface. The top navigation bar includes the nOps logo, 'AWS Resources', 'Change management', and 'Event log'. A search bar and a user profile icon labeled 'admin' are also present. Below the navigation bar, the 'Config Changes' section is active, showing a diff of configuration changes. The diff highlights changes in a JSON configuration object, with red lines indicating deletions and green lines indicating additions. The configuration object includes fields for 'eventVersion', 'userIdentity', 'eventTime', 'eventSource', 'eventName', 'awsRegion', 'sourceIPAddress', 'userAgent', 'requestParameters', 'instancesSet', 'instanceId', 'imageId', 'minCount', 'maxCount', 'keyName', 'instanceType', 'blockDeviceMapping', 'tenancy', and 'monitoring'.

```

{
  "Configuration": {
    "eventVersion": "1.05",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "AIDAIZIT55ZBDULGNKA4C",
      "arn": "arn:aws:iam::202279780353:user/carlos",
      "accountId": "202279780353",
      "accessKeyId": "AKIAI34C6NPFJA73JH2A",
      "userName": "carlos"
    },
    "eventTime": "2017-05-23T15:57:04Z",
    "eventTime": "2017-05-22T20:29:13Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "TerminateInstances",
    "eventName": "RunInstances",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "200.6.205.75",
    "userAgent": "Boto/2.46.1 Python/2.7.12 Linux/4.4.0-78-generic",
    "requestParameters": {
      "instancesSet": {
        "items": [
          {
            "instanceId": "i-0b4ce06f9590e77b9",
            "imageId": "ami-b7a114d7",
            "minCount": 1,
            "maxCount": 1,
            "keyName": "oregon-test"
          }
        ]
      },
      "instanceType": "t2.micro",
      "blockDeviceMapping": {},
      "tenancy": "default",
      "monitoring": {
        "enabled": false
      }
    }
  }
}

```

Showing diffs from latest changes.

Change Management (Active)
Apr 29 2017 2:27:36 AM - May 6 2017 2:27:36 AM

Date ▾ Assignee ▾ Project ▾ Security ▾ Event Name ▾ IAM ▾ Status ▾ Enter keyword to search...

1w x Active x

Open (56) Acknowledged (0) Approved (1) All (57)

Showing 20 Terraform changes CHANGES TYPE: All Terraform (20) Manual (16) CloudFormation (20)

G gagan@nclouds.com invoked ConsoleLogin api in region us-east-1
Created on: May 24 2017 1:18:03 AM | Status: Open | Cost: \$11.91
12 events ▶ Project: UAT-nOps SSH Violation

R roman@nclouds.com invoked ConsoleLogin api in region us-east-1
Created on: May 24 2017 1:18:03 AM | Status: Open | Cost: \$200
3 events ▼

R roman@nclouds.com invoked ConsoleLogin api in region us-east-1
#Resource Name: i-0aefedbee457750e1 #Resource Type: AWS::EC2::Instance #project: UAT-nOps SSH Violation
Created on: May 24 2017 1:18:03 AM | Status: Open | Cost: \$10

R adam@nclouds.com invoked ConsoleLogin api in region us-east-1
#Resource Name: i-0aefedbee457750e1 #Resource Type: AWS::EC2::Instance #project: UAT-nOps

Filter manual changes from Terraform and CloudFormation.

nOps Rules > Inactive key rule details

Inactive keys for last 9 days Acknowledged until May 19 2017 6:21:20 AM Rule Configuration Acknowledged ▾

Description Checks whether access key of the IAM user is used for last 9 days.
Overall rule status Last successful invocation on May 26 2017 3:45:36 AM

List of access keys not used recently

User Name	Access Key	Last Used	Created	Status
abel	AKIAIGVB0MSIYBK0IWA	May 5 2017 2:30:00 AM	Apr 28 2017 9:50:55 PM	Active
abel	AKIAIPGQ5SZMRO6YWZJA	N/A	Apr 28 2017 5:10:44 AM	Active
ali	AKIAJY5AMBXGQBFPDOPQ	N/A	Mar 22 2017 1:13:03 AM	Active
braulio	AKIAJK4OBEQE4PRC22Q	May 11 2017 3:09:00 AM	May 11 2017 3:08:51 AM	Active
braulio	AKIAJZXQPLGXOEHPYTA	May 16 2017 1:31:00 AM	May 10 2017 4:38:01 AM	Active
deepu	AKIAIRC3HJA4G4GHG5Q	Apr 21 2017 1:39:00 AM	Apr 20 2017 3:33:44 AM	Active
dockerAbel	AKIAJ7VSFZLWSK7XX4LA	May 11 2017 11:31:00 PM	May 3 2017 9:17:19 PM	Active
ecr	AKIAJAHKM2TVXCZUC3Q	Dec 15 2016 2:34:00 PM	Oct 31 2016 1:26:19 AM	Active
ecr	AKIAJSGCSIGAHZIDICIQ	Jan 6 2017 10:49:00 PM	Oct 31 2016 9:42:02 AM	Active

Use nOps to monitor inactive keys and receive notifications.



We can help you assess and implement change management for AWS.

Let us share with you what we've learned from hundreds of managed services and cloud infrastructure implementations.

Request a demo at www.nops.io

Contact Info



866-673-9330



sales@nops.io



99 S Almaden Blvd Suite 600,
San Jose, CA 95113