AWS Health

User Guide

AWS Health: User Guide

Table of Contents

What is AWS Health?

AWS Health provides ongoing visibility into your resource performance and the availability of your AWS services and accounts. You can use AWS Health *events* to learn how service and resource changes might affect your applications running on AWS. AWS Health provides relevant and timely information to help you manage events in progress. AWS Health also helps you be aware of and to prepare for planned activities. The service delivers alerts and notifications triggered by changes in the health of AWS resources, so that you get near-instant event visibility and guidance to help accelerate troubleshooting.

All customers can use the AWS Personal Health Dashboard (PHD), powered by the AWS Health API. The dashboard requires no setup, and it's ready to use for authenticated AWS users (p. 23). For more service highlights, see the AWS Personal Health Dashboard detail page.

To understand the basics of AWS Health and how you can use the service, see Are you a first-time AWS Health user? (p. 2).

For a list of terms that you will see when you use AWS Health, see Concepts for AWS Health (p. 3).

Additionally, AWS Support customers who have a Business or Enterprise Support plan can use the AWS Health API to integrate with in-house and third-party systems.

Are you a first-time AWS Health user?

If you are a first-time user of AWS Health, begin by reading the following sections:

- What is AWS Health? (p. 1) – This section describes the underlying data model, the operations it supports, and the AWS SDKs that you can use to interact with the service.
- Concepts for AWS Health (p. 3) – Learn the basics about AWS Health and terms that you will encounter while you use the service.
- Getting started with the AWS Personal Health Dashboard (p. 7) – The AWS Personal Health Dashboard section describes using the AWS Personal Health Dashboard to view events and affected entities and perform advanced filtering.
- Accessing the AWS Health API (p. 12) – The AWS Health API section describes the operations that retrieve information about events and entities.

AWS Health provides a console, called the AWS Personal Health Dashboard, to all customers. You do not need to write code or perform any actions to set up the dashboard. If you have a Business or Enterprise Support plan, you can access the information presented on the dashboard programmatically. You can use the AWS Command Line Interface (AWS CLI) or write code to make requests, by using either the REST API directly or the AWS SDKs.

For more information about using AWS Health with the AWS CLI, see the AWS CLI Reference for AWS Health. For instructions for installing the AWS CLI, see Installing the AWS Command Line Interface.

Concepts for AWS Health

Learn about AWS Health concepts and understand how you can use the service to maintain the health of your applications, services, and resources in your AWS account.

Topics

AWS Health event

AWS Health events, also known as Health events, are notifications that AWS Health sends on behalf of other AWS services. You can use these events to learn about upcoming or scheduled changes that might affect your account. For example, AWS Health can send an event if AWS Identity and Access Management (IAM) plans to deprecate a managed policy or AWS Config plans to deprecate a managed rule. AWS Health also sends events when there are service availability issues in an AWS Region. You can review the event description to understand the issue, identify any affected resources, and take any recommended actions.

There are two types of Health events:

Contents

Account-specific event

Account-specific events are local to either your AWS account or an account in your AWS organization. For example, if there's an issue with an Amazon Elastic Compute Cloud (Amazon EC2) instance type in a Region that you use, AWS Health provides information about the event and the name of the affected resources.

You can find account-specific events from your AWS Personal Health Dashboard, the AWS Health API, or use Amazon CloudWatch Events to receive notifications (p. 58).

Public event

Public events are reported service events that aren't specific to an account. For example, if there's a service issue for Amazon Simple Storage Service (Amazon S3) in the US East (Ohio) Region, AWS Health provides information about the event, even if you don't use that service or have S3 buckets in that Region. We recommend that you review public notifications before you take action on them.

You can find public events from your AWS Personal Health Dashboard or the Service Health Dashboard.

> **Note**
> You can't use CloudWatch Events to return public events because they're not related to a specific account. For more information, see About public events for AWS Health (p. 59).

AWS Personal Health Dashboard

The AWS Personal Health Dashboard provides a complete view for events and communications about AWS services. You can use the dashboard to find account-specific events that might affect the health of your account and public service events that provide general awareness. Sign in to the AWS Management Console to view your AWS Personal Health Dashboard.

The AWS Personal Health Dashboard presents information in two ways:

- A dashboard that shows recent and upcoming events organized by event type category
- A full event log that shows all events from the past 90 days

> **Note**
> If you enabled the organizational view feature, you can view events for all accounts in your AWS organization.

For more information, see Getting started with the AWS Personal Health Dashboard (p. 7).

Service Health Dashboard

The Service Health Dashboard shows public events, which are reported service issues for AWS that provide information about service availability. The Service Health Dashboard only shows public events, which aren’t specific to any account.

Your AWS Personal Health Dashboard shows both *public* events and *account-specific* events. We recommend that you use the AWS Personal Health Dashboard to learn about events that might affect you directly, such as a deprecated resource, and to learn about events that provide general awareness, such as an upcoming maintenance issue for a service in a Region.

Event type code

The event type codes shown in a Health event include the affected service and the type of event. For example, if you receive a Health event that has the `AWS_EC2_SYSTEM_MAINTENANCE_EVENT` event type code, this means that the service is scheduling a maintenance event that might affect you. Use this information to plan ahead or take action for your account.

Event type categories

All Health events have an associated event type category. For some events, the event type category might appear in the event type code, such as the `AWS_RDS_MAINTENANCE_SCHEDULED` code. In this example, the category is *scheduled*. You can use this information to understand event categories at a high level.

We recommend that you monitor all event type categories. Note that each category appears for different types of events. You can also use the DescribeEventTypes API operation to find the event type category.

Account notification

These events provide information about the administration or security of your accounts and services. These events might be informative, or they might require urgent action from you. We recommend that you pay attention for these types of events and review all recommended actions.

The following are example event type codes for account notifications:

- `AWS_S3_OPEN_ACCESS_BUCKET_NOTIFICATION` – You have an Amazon S3 bucket that might allow public access.
- `AWS_BILLING_SUSPENSION_NOTICE` – Your account has outstanding charges and has been suspended, or you deactivated your account.
- `AWS_WORKSPACES_OPERATIONAL_NOTIFICATION` – There's a service issue for Amazon WorkSpaces.

Issue

These events are unexpected events that affect AWS services or resources. Common events in this category include communications about operational issues that are causing service degradation, or localized resource-level issues for your awareness.

The following are example event type codes for issues:

- `AWS_EC2_OPERATIONAL_ISSUE` – An operational issue for a service, such as delays in using a service.
- `AWS_EC2_API_ISSUE` – An operational issue for a service's API, such as increased latency for an API operation.
- `AWS_EBS_VOLUME_ATTACHMENT_ISSUE` – A localized resource-level issue that might affect your Amazon Elastic Block Store (Amazon EBS) resources.
- `AWS_ABUSE_PII_CONTENT_REMOVAL_REPORT` – This event means that your account might be suspended if you don't take action.

Scheduled change

These events provide information about upcoming changes to your services and resources. Some events might recommend that you take action to avoid service disruptions, while others will occur automatically without any action on your part. Your resource might be temporarily unavailable during the scheduled change activity. All events in this category are account-specific events.

The following are example event type codes for scheduled changes:

- `AWS_EC2_SYSTEM_REBOOT_MAINTENANCE_SCHEDULED` – An Amazon EC2 instance requires a reboot.
- `AWS_SAGEMAKER_SCHEDULED_MAINTENANCE` – SageMaker requires a maintenance event, such as fixing a service issue.

Tip
If you use the AWS Health API or the AWS Command Line Interface (AWS CLI) to return event details, the `Event` object contains the `eventScopeCode` field with the `ACCOUNT_SPECIFIC` value. For more information, see the AWS Health API Reference.

Event status

The event status tells you if the Health event is open, closed, or upcoming. You can view Health events in the AWS Personal Health Dashboard or the AWS Health API for up to 90 days.

Affected entities

Affected entities are AWS resources that might be affected by the event. For example, if you receive a scheduled event for Amazon EC2 maintenance for a specific instance type that you're using in your account, you can use the Health event to determine the ID of the affected instances. Use this information to address any potential service issue, such as creating or deprecating resources.

AWS Health API

You can use the AWS Health API to programmatically access the information that appears in the AWS Personal Health Dashboard, such as the following:

- Get information about events that might affect your AWS services and resources
- Enable or disable the organizational view feature for your AWS organization
- Filter your events by specific services, event type categories, and event type codes

For more information, see the AWS Health API Reference.

Note
You must have a Business or Enterprise Support plan from AWS Support to use the AWS Health API. If you call the AWS Health API from an account that doesn't have a Business or Enterprise Support plan, you receive a `SubscriptionRequiredException` error.

Organizational view

You can use this feature to aggregate all health events for AWS accounts in your AWS Organizations into a single view in the AWS Personal Health Dashboard. You can then sign in to the management account of your organization or use the AWS Health API to view all events that might affect the different accounts and resources. You can enable this feature from the AWS Health console or API. For more information, see Aggregating AWS Health events across accounts with organizational view (p. 46).

Getting started with the AWS Personal Health Dashboard

You can use the AWS Personal Health Dashboard to learn about AWS Health events that can affect your AWS services or account. The AWS Personal Health Dashboard presents information in two ways: a dashboard that shows recent and upcoming events organized by category, and a full event log that shows all events from the past 90 days.

To view your AWS Personal Health Dashboard

1. Sign in to the AWS Management Console and open the AWS Personal Health Dashboard at https://phd.aws.amazon.com/phd/home.
2. Choose **Dashboard** to view recent and upcoming events or **Event log** to view all events for the past 90 days.

Topics

Dashboard

The AWS Personal Health Dashboard organizes issues in three groups: open issues, scheduled changes, and other notifications. By default, open issues and other notifications are restricted to issues whose start time is within the last seven days. The scheduled changes group contains items that are ongoing or upcoming.

When you choose an event in the dashboard list, the **Event Details** pane appears with information about the event and resources that are affected by the event. For more information, see Event details pane (p. 9).

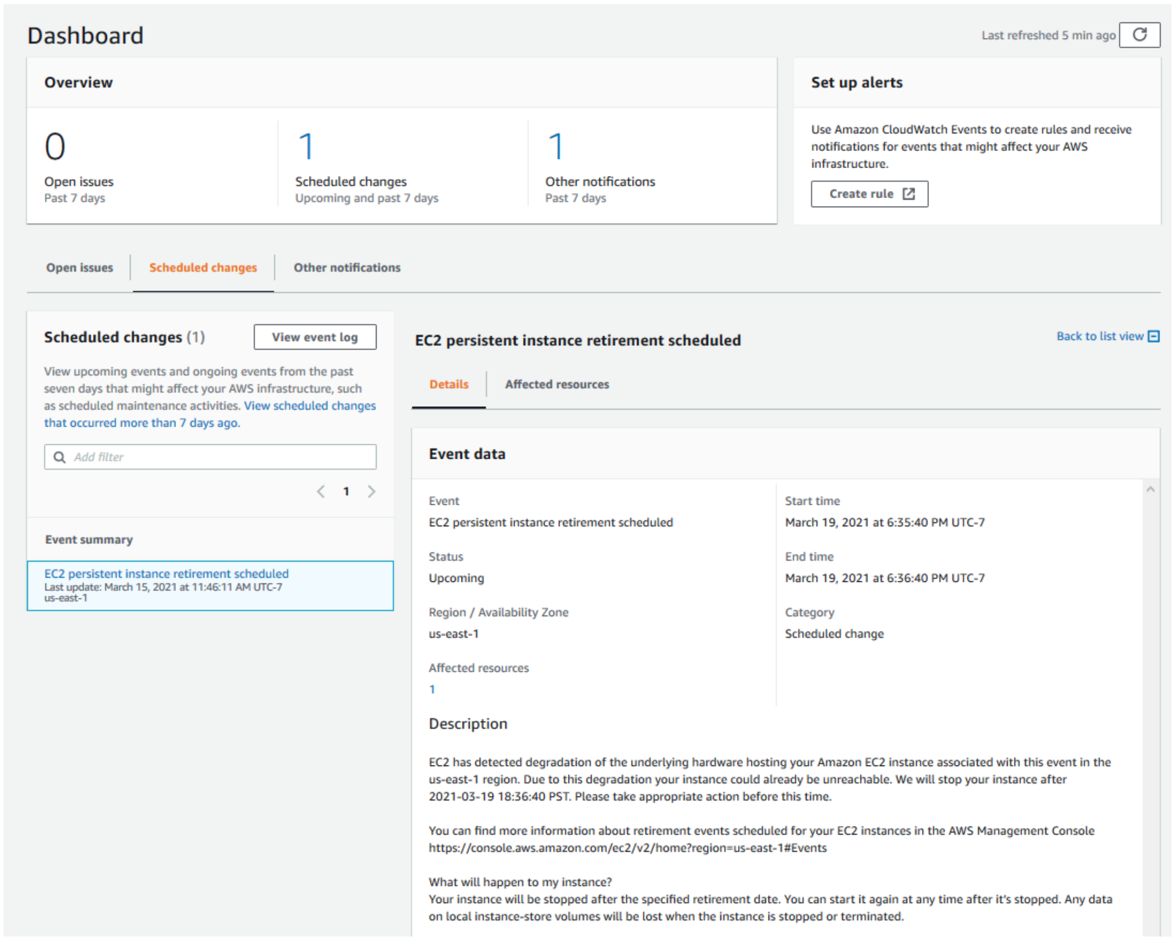
You can filter the items that appear in any group by choosing options from the filter list. For example, you can narrow the results by Availability Zone, Region, event end time or last update time, AWS service, and so on.

To see all the events, rather than the recent ones that appear in the dashboard, choose **See all issues** above the list to open the Event log (p. 8).

> **Note**
> Currently, you can't delete notifications for events that appear in the AWS Personal Health Dashboard. After an AWS service resolves an event, the notification is removed from your dashboard view.

Example : Scheduled event for Amazon Elastic Compute Cloud (Amazon EC2)

The following image shows a scheduled event for an Amazon EC2 instance type that is scheduled for retirement.



Use Amazon CloudWatch Events to create rules and receive notifications for events that might affect your AWS infrastructure.

View upcoming events and ongoing events from the past seven days that might affect your AWS infrastructure, such as scheduled maintenance activities. View scheduled changes that occurred more than 7 days ago.

EC2 persistent instance retirement scheduled
Last update: March 15, 2021 at 11:46:11 AM UTC-7
us-east-1

Event	EC2 persistent instance retirement scheduled
Start time	March 19, 2021 at 6:35:40 PM UTC-7
Status	Upcoming
End time	March 19, 2021 at 6:36:40 PM UTC-7
Region / Availability Zone	us-east-1
Category	Scheduled change
Affected resources	1

Description

EC2 has detected degradation of the underlying hardware hosting your Amazon EC2 instance associated with this event in the us-east-1 region. Due to this degradation your instance could already be unreachable. We will stop your instance after 2021-03-19 18:36:40 PST. Please take appropriate action before this time.

You can find more information about retirement events scheduled for your EC2 instances in the AWS Management Console https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Events

What will happen to my instance?
Your instance will be stopped after the specified retirement date. You can start it again at any time after it's stopped. Any data on local instance-store volumes will be lost when the instance is stopped or terminated.

Event log

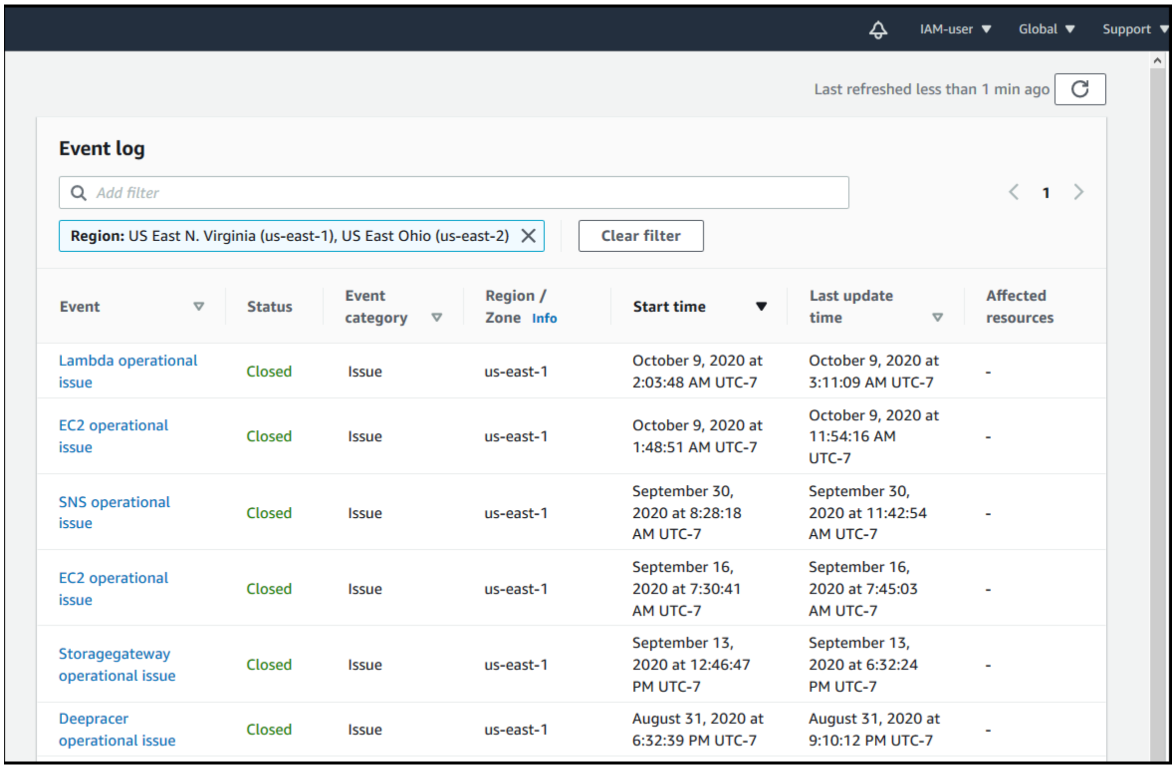
The **Event log** page of the AWS Personal Health Dashboard displays all the AWS Health events that apply to your account. The column layout and behavior is similar to the Dashboard, but the log page includes additional columns and filter options for **Status** and **Event category**.

When you choose an event in the **Event log** list, the **Event details** pane appears with information about the event and resources that are affected by the event. For more information, see Event details pane (p. 9).

You can filter the items by choosing options from the filter list. For example, you can narrow the results by status (closed, open, or upcoming), event category (issue, notification, or scheduled change), Availability Zone, Region, event end time or last update time, AWS service, and so on.

Example : Event log

The following screenshot shows events for only the US East (N. Virginia) and US East (Ohio) Regions.



Event log

Region: US East N. Virginia (us-east-1), US East Ohio (us-east-2)

Event	Status	Event category	Region / Zone Info	Start time	Last update time	Affected resources
Lambda operational issue	Closed	Issue	us-east-1	October 9, 2020 at 2:03:48 AM UTC-7	October 9, 2020 at 3:11:09 AM UTC-7	-
EC2 operational issue	Closed	Issue	us-east-1	October 9, 2020 at 1:48:51 AM UTC-7	October 9, 2020 at 11:54:16 AM UTC-7	-
SNS operational issue	Closed	Issue	us-east-1	September 30, 2020 at 8:28:18 AM UTC-7	September 30, 2020 at 11:42:54 AM UTC-7	-
EC2 operational issue	Closed	Issue	us-east-1	September 16, 2020 at 7:30:41 AM UTC-7	September 16, 2020 at 7:45:03 AM UTC-7	-
Storagegateway operational issue	Closed	Issue	us-east-1	September 13, 2020 at 12:46:47 PM UTC-7	September 13, 2020 at 6:32:24 PM UTC-7	-
Deepracer operational issue	Closed	Issue	us-east-1	August 31, 2020 at 6:32:39 PM UTC-7	August 31, 2020 at 9:10:12 PM UTC-7	-

Events types

There are two types of AWS Health events:

- *Public events* are service events that aren't specific to an AWS account. For example, if there is an issue with Amazon Elastic Compute Cloud (Amazon EC2) in an AWS Region, AWS Health provides information about the event, even if you don't use services or resources in that Region.
- *Account-specific* events are specific to your AWS account or an account in your organization. For example, if there's an issue with an Amazon EC2 instance in a Region that you use, AWS Health provides information about the event and the affected resources in the account.

You can use the following options to identify if an event is public or account-specific:

- In the AWS Personal Health Dashboard, choose the **Affected resources** tab on the **Event log** page. Events with resources are specific to your account. Events without resources are public and are not specific to your account. For more information, see Getting started with the AWS Personal Health Dashboard (p. 7).
- Use the AWS Health API to return the `eventScopeCode` parameter. Events can have the `PUBLIC`, `ACCOUNT_SPECIFIC`, or `NONE` value. For more information, see the DescribeEventDetails operation in the *AWS Health API Reference*.

Event details pane

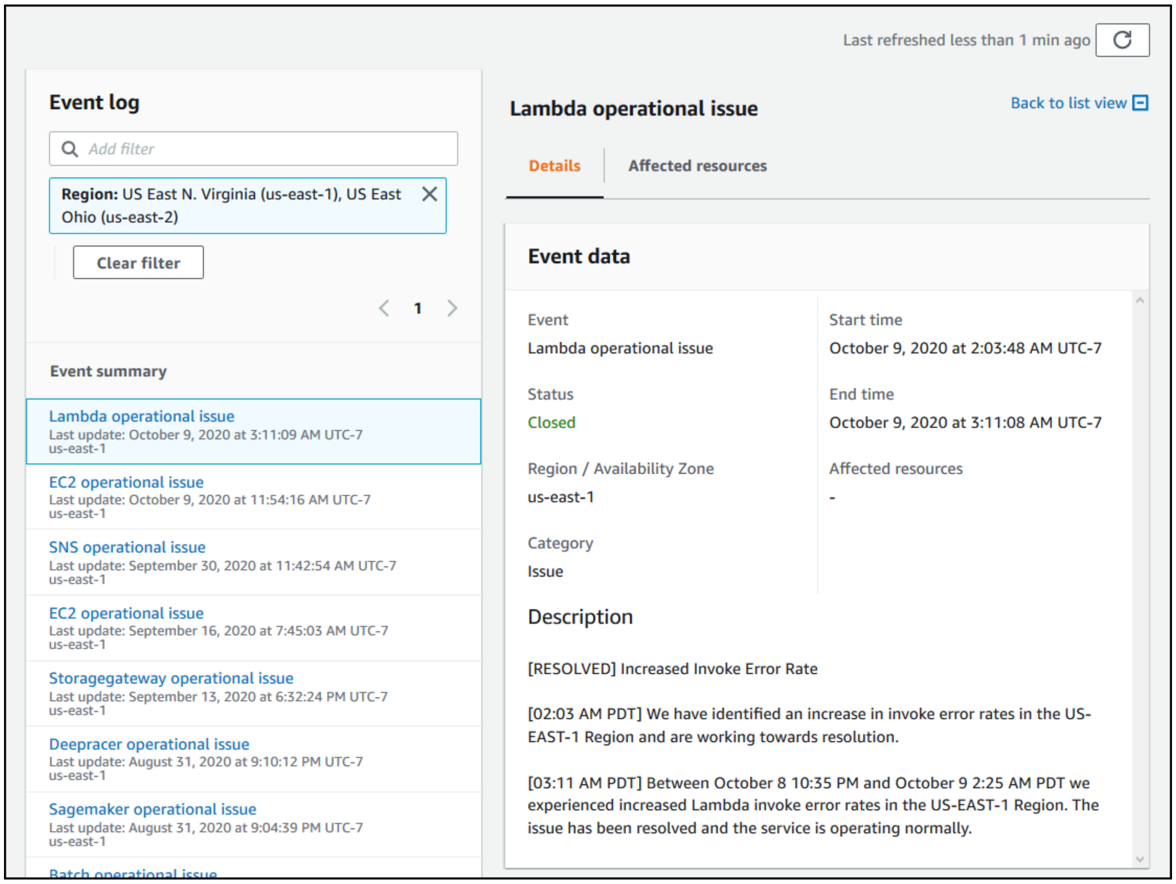
The **Event details** pane has two tabs. The **Details** tab displays a text description of the event and relevant data about the event: the event name, status, Region and Availability Zone, start time, end time, and category. The **Affected resources** tab displays information about any AWS resources that are affected by the event:

- The resource ID (for example, an Amazon EBS volume ID such as `vol-a1b2c34f`) or Amazon Resource Name (ARN), if available or relevant.

You can filter the items that appear in the resources list by choosing options from the filter list. You can narrow the results by resource ID or ARN.

Example : AWS Health event for AWS Lambda

The following screenshot shows an example event for Lambda and a description of the issue.



Last refreshed less than 1 min ago

Event log

Add filter

Region: US East N. Virginia (us-east-1), US East Ohio (us-east-2)

Clear filter

1

Event summary

Lambda operational issue
Last update: October 9, 2020 at 3:11:09 AM UTC-7
us-east-1

EC2 operational issue
Last update: October 9, 2020 at 11:54:16 AM UTC-7
us-east-1

SNS operational issue
Last update: September 30, 2020 at 11:42:54 AM UTC-7
us-east-1

EC2 operational issue
Last update: September 16, 2020 at 7:45:03 AM UTC-7
us-east-1

Storagegateway operational issue
Last update: September 13, 2020 at 6:32:24 PM UTC-7
us-east-1

Deepracer operational issue
Last update: August 31, 2020 at 9:10:12 PM UTC-7
us-east-1

Sagemaker operational issue
Last update: August 31, 2020 at 9:04:39 PM UTC-7
us-east-1

Batch operational issue

Lambda operational issue

Back to list view

Details | Affected resources

Event data

Event Lambda operational issue	Start time October 9, 2020 at 2:03:48 AM UTC-7
Status Closed	End time October 9, 2020 at 3:11:08 AM UTC-7
Region / Availability Zone us-east-1	Affected resources -
Category Issue	

Description

[RESOLVED] Increased Invoke Error Rate

[02:03 AM PDT] We have identified an increase in invoke error rates in the US-EAST-1 Region and are working towards resolution.

[03:11 AM PDT] Between October 8 10:35 PM and October 9 2:25 AM PDT we experienced increased Lambda invoke error rates in the US-EAST-1 Region. The issue has been resolved and the service is operating normally.

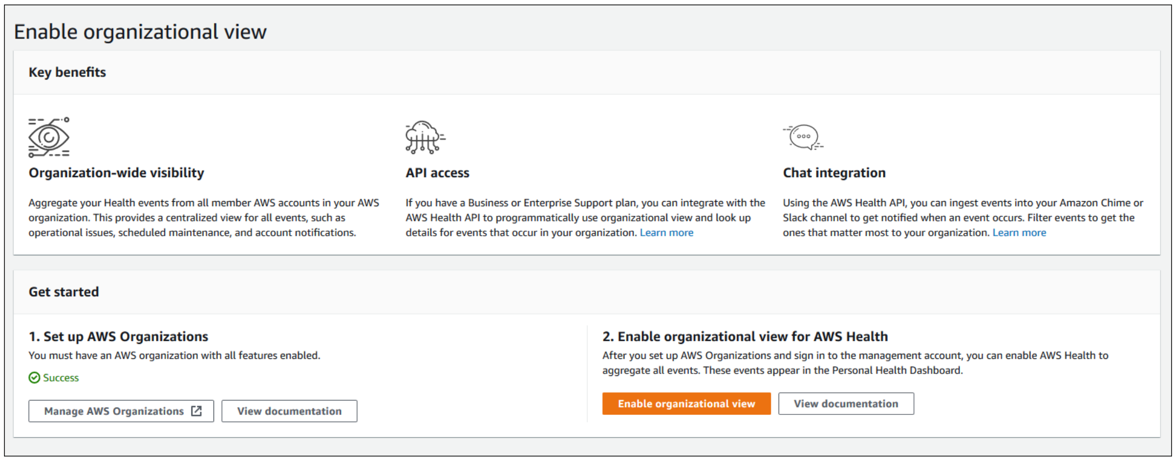
Amazon CloudWatch Events

Use Amazon CloudWatch Events to detect and react to changes for AWS Health events. You can monitor specific AWS Health events that occur in your AWS account, and then set up rules so that you get notified or take action when events change.

You can choose **Amazon CloudWatch Events** to navigate to the CloudWatch Events console. For more information, see Monitoring AWS Health events with Amazon CloudWatch Events (p. 58).

Organizational view

AWS Health integrates with AWS Organizations so that you can view events for all accounts that are part of your organization. This provides you a centralized view for events that appear in your organization. You can use these events to monitor for changes in your resources, services, and applications.



Enable organizational view

Key benefits

Organization-wide visibility

Aggregate your Health events from all member AWS accounts in your AWS organization. This provides a centralized view for all events, such as operational issues, scheduled maintenance, and account notifications.

API access

If you have a Business or Enterprise Support plan, you can integrate with the AWS Health API to programmatically use organizational view and look up details for events that occur in your organization. Learn more

Chat integration

Using the AWS Health API, you can ingest events into your Amazon Chime or Slack channel to get notified when an event occurs. Filter events to get the ones that matter most to your organization. Learn more

Get started

1. Set up AWS Organizations

You must have an AWS organization with all features enabled.

Success

2. Enable organizational view for AWS Health

After you set up AWS Organizations and sign in to the management account, you can enable AWS Health to aggregate all events. These events appear in the Personal Health Dashboard.

For more information, see Aggregating AWS Health events across accounts with organizational view (p. 46).

Alerts for AWS Health events

The AWS Personal Health Dashboard has a bell icon in the console navigation bar with an **Alerts** menu. This feature displays the number of recent AWS Health events that appear on the dashboard in each category. This bell icon appears on several AWS consoles, such as Amazon Elastic Compute Cloud (Amazon EC2), Amazon Relational Database Service (Amazon RDS), AWS Identity and Access Management (IAM), and AWS Trusted Advisor.

Choose the bell icon to see whether your account is affected by recent events. You can then choose an event to navigate to the AWS Personal Health Dashboard for more information.

Accessing the AWS Health API

AWS Health is a RESTful web service that uses HTTPS as a transport and JSON as a message serialization format. Your application code can make requests directly to the AWS Health API. When you use the REST API directly, you must write the necessary code to sign and authenticate your requests. For more information about the AWS Health operations and parameters, see the AWS Health API Reference.

Note
You must have a Business or Enterprise support plan from AWS Support to use the AWS Health API. If you call the AWS Health API from an AWS account that doesn't have a Business or Enterprise support plan, you receive a `SubscriptionRequiredException` error.

You can use the AWS SDKs to wrap the AWS Health REST API calls, which can simplify your application development. You specify your AWS credentials, and these libraries take care of authentication and request signing for you.

AWS Health also provides a AWS Personal Health Dashboard in the AWS Management Console that you can use to view and search for events and affected entities. See Getting started with the AWS Personal Health Dashboard (p. 7).

Endpoints

The AWS Health API follows a multi-Region application architecture and has two regional endpoints in an active-passive configuration. To support active-passive DNS failover, AWS Health provides a single, global endpoint. You can perform a DNS lookup on the global endpoint to determine the active endpoint and corresponding signing AWS Region. This helps you know which endpoint to use in your code, so that you can get the latest information from AWS Health.

When you make a request to the global endpoint, you must specify your AWS access credentials to the regional endpoint that you target and configure the signing for your Region. Otherwise, your authentication might fail. For more information, see Signing AWS Health API requests (p. 17).

The following table represents the default configuration.

Description	Signing Region	Endpoint	Protocol
Active	us-east-1	health.us-east-1.amazonaws.com	HTTPS
Passive	us-east-2	health.us-east-2.amazonaws.com	HTTPS
Global	us-east-1 **Note** This is the signing Region of the current active endpoint.	global.health.amazonaws.com	HTTPS

To determine if an endpoint is the *active endpoint*, do a DNS lookup on the *global endpoint* CNAME, and then extract the AWS Region from the resolved name.

Example : DNS lookup on global endpoint

The following command completes a DNS lookup on the global.health.amazonaws.com endpoint. The command then returns the us-east-1 Region endpoint. This output tells you which endpoint you should use for AWS Health.

```
dig global.health.amazonaws.com | grep CNAME
global.health.amazonaws.com. 10 IN CNAME health.us-east-1.amazonaws.com
```

Tip
Both the active and passive endpoints return AWS Health data. However, the latest AWS Health data is only available from the active endpoint. Data from the passive endpoint will be eventually consistent with the active endpoint. We recommend that you restart any workflows when the active endpoint changes.

Using the high availability endpoint demo

In the following code examples, AWS Health uses a DNS lookup against the global endpoint to determine the active regional endpoint and signing Region. Then, the code restarts the workflow if the active endpoint changes.

Topics

Using the Java demo

Prerequisite

You must install Gradle.

To use the Java example

1. Download the AWS Health high availability endpoint demo from GitHub.
2. Navigate to the demo project `high-availability-endpoint/java` directory.
3. In a command line window, enter the following command.

```
gradle build
```

4. Enter the following commands to specify your AWS credentials.

```
export AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
export AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
export AWS_SESSION_TOKEN="your-aws-token"
```

5. Enter the following command to run the demo.

```
gradle run
```

Example : AWS Health event output

The code example returns the recent AWS Health event in the last seven days in your AWS account. In the following example, the output includes an AWS Health event for the AWS Config service.

```
> Task :run
[main] INFO aws.health.high.availability.endpoint.demo.HighAvailabilityV2Workflow
 - EventDetails(Event=Event(Arn=arn:aws:health:global::event/CONFIG/
AWS_CONFIG_OPERATIONAL_NOTIFICATION/AWS_CONFIG_OPERATIONAL_NOTIFICATION_88a43e8a-
e419-4ca7-9baa-56bcde4dba3,
Service=CONFIG, EventTypeCode=AWS_CONFIG_OPERATIONAL_NOTIFICATION,
 EventTypeCategory=accountNotification, Region=global,
 StartTime=2020-09-11T02:55:49.899Z, LastUpdatedTime=2020-09-11T03:46:31.764Z,
StatusCode=open, EventScopeCode=ACCOUNT_SPECIFIC),
 EventDescription=EventDescription(LatestDescription=As part of our ongoing efforts
 to optimize costs associated with recording changes related to certain ephemeral
 workloads,
AWS Config is scheduled to release an update to relationships modeled within
 ConfigurationItems (CI) for 7 EC2 resource types on August 1, 2021.
Examples of ephemeral workloads include changes to Amazon Elastic Compute Cloud (Amazon
 EC2) Spot Instances, Amazon Elastic MapReduce jobs, and Amazon EC2 Autoscaling.
This update will optimize CI models for EC2 Instance, SecurityGroup, Network Interface,
 Subnet, VPC, VPN Gateway, and Customer Gateway resource types to record direct
 relationships and deprecate indirect relationships.

A direct relationship is defined as a one-way relationship (A->B) between a resource
 (A) and another resource (B), and is typically derived from the Describe API response
 of resource (A).
An indirect relationship, on the other hand, is a relationship that AWS Config infers
 (B->A), in order to create a bidirectional relationship.
For example, EC2 instance -> Security Group is a direct relationship, since security
 groups are returned as part of the describe API response for an EC2 instance.
But Security Group -> EC2 instance is an indirect relationship, since EC2 instances are
 not returned when describing an EC2 Security group.

Until now, AWS Config has recorded both direct and indirect relationships. With
 the launch of Advanced queries in March 2019, indirect relationships can easily be
 answered by running Structured Query Language (SQL) queries such as:

SELECT
 resourceId,
 resourceType
WHERE
 resourceType ='AWS::EC2::Instance'
AND
 relationships.resourceId = 'sg-234213'

By deprecating indirect relationships, we can optimize the information contained within
 a
Configuration Item while reducing AWS Config costs related to relationship changes.
This is especially useful in case of ephemeral workloads where there is a high volume
 of configuration changes for EC2 resource types.

Which resource relationships are being removed?

Resource Type: Related Resource Type
1 AWS::EC2::CustomerGateway: AWS::VPN::Connection
2 AWS::EC2::Instance: AWS::EC2::EIP, AWS::EC2::RouteTable
3 AWS::EC2::NetworkInterface: AWS::EC2::EIP, AWS::EC2::RouteTable
4 AWS::EC2::SecurityGroup: AWS::EC2::Instance, AWS::EC2::NetworkInterface
5 AWS::EC2::Subnet: AWS::EC2::Instance, AWS::EC2::NetworkACL,
 AWS::EC2::NetworkInterface, AWS::EC2::RouteTable
6 AWS::EC2::VPC: AWS::EC2::Instance, AWS::EC2::InternetGateway, AWS::EC2::NetworkACL,
 AWS::EC2::NetworkInterface, AWS::EC2::RouteTable, AWS::EC2::Subnet,
 AWS::EC2::VPNGateway, AWS::EC2::SecurityGroup
7 AWS::EC2::VPNGateway: AWS::EC2::RouteTable, AWS::EC2::VPNConnection

Alternate mechanism to retrieve this relationship information:
```

```
The SelectResourceConfig API accepts a SQL SELECT command, performs the corresponding
 search, and returns resource configurations matching the properties. You can use this
 API to retrieve the same relationship information.
For example, to retrieve the list of all EC2 Instances related to a particular VPC
 vpc-1234abc, you can use the following query:

SELECT
 resourceId,
 resourceType
WHERE
 resourceType ='AWS::EC2::Instance'
AND
 relationships.resourceId = 'vpc-1234abc'

If you have any questions regarding this deprecation plan, please contact AWS Support
 [1]. Additional sample queries to retrieve the relationship information for the
 resources listed above is provided in [2].

[1] https://aws.amazon.com/support
[2] https://docs.aws.amazon.com/config/latest/developerguide/
examplerelationshipqueries.html),
EventMetadata={})
```

Java resources

- For more information, see the Interface HealthClient in the *AWS SDK for Java API Reference* and the source code.
- For more information about the library used in this demo for DNS lookups, see the dnsjava in GitHub.

Using the Python demo

Prerequisite

You must install Python 3.

To use the Python example

1. Download the AWS Health high availability endpoint demo from GitHub.
2. Navigate to the demo project `high-availability-endpoint/python` directory.
3. In a command line window, enter the following commands.

   ```
   pip3 install virtualenv
   virtualenv -p python3 v-aws-health-env
   ```

   Note
   For Python 3.3 and later, you can use the built-in `venv` module to create the virtual environment, instead of installing `virtualenv`. For more information, see venv - Creation of virtual environments on the Python website.

   ```
   python3 -m venv v-aws-health-env
   ```

4. Enter the following command to activate the virtual environment.

   ```
   source v-aws-health-env/bin/activate
   ```

5. Enter the following command to install the dependencies.

```
pip install -r requirements.txt
```

6. Enter the following commands to specify your AWS credentials.

```
export AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
export AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
export AWS_SESSION_TOKEN="your-aws-token"
```

7. Enter the following command to run the demo.

```
python3 main.py
```

Example : AWS Health event output

The code example returns the recent AWS Health event in the last seven days in your AWS account. The following output returns an AWS Health event for an AWS security notification.

```
INFO:botocore.credentials:Found credentials in environment variables.
INFO:root:Details: {'arn': 'arn:aws:health:global::event/SECURITY/
AWS_SECURITY_NOTIFICATION/AWS_SECURITY_NOTIFICATION_0e35e47e-2247-47c4-
a9a5-876544042721',
'service': 'SECURITY', 'eventTypeCode': 'AWS_SECURITY_NOTIFICATION',
 'eventTypeCategory': 'accountNotification', 'region': 'global', 'startTime':
 datetime.datetime(2020, 8, 19, 23, 30, 42, 476000,
tzinfo=tzlocal()), 'lastUpdatedTime': datetime.datetime(2020, 8, 20, 20, 44, 9, 547000,
 tzinfo=tzlocal()), 'statusCode': 'open', 'eventScopeCode': 'PUBLIC'}, description:
{'latestDescription': 'This is the second notice regarding TLS requirements on FIPS
 endpoints.\n\nWe
are in the process of updating all AWS Federal Information Processing Standard (FIPS)
 endpoints across all AWS regions
to Transport Layer Security (TLS) version 1.2 by March 31, 2021 . In order to avoid an
 interruption in service, we encourage you to act now, by ensuring that you connect to
 AWS FIPS endpoints at a TLS version of 1.2.
If your client applications fail to support TLS 1.2 it will result in connection
 failures when TLS versions below 1.2 are no longer supported.\n\nBetween now and March
 31, 2021 AWS will remove TLS 1.0 and TLS 1.1 support from each FIPS endpoint where no
 connections below TLS 1.2 are detected over a 30-day period.
After March 31, 2021 we may deploy this change to all AWS FIPS endpoints, even if there
 continue
to be customer connections detected at TLS versions below 1.2. \n\nWe will provide
 additional updates and reminders on the AWS Security Blog, with a ‘TLS’ tag [1].
 If you need further guidance or assistance, please contact AWS Support [2] or your
 Technical Account Manager (TAM).
Additional information is below.\n\nHow can I identify clients that are connecting with
 TLS
1.0/1.1?\nFor customers using S3 [3], Cloudfront [4] or Application Load Balancer [5]
 you can use
your access logs to view the TLS connection information for these services, and
 identify client
connections that are not at TLS 1.2. If you are using the AWS Developer Tools on your
 clients,
you can find information on how to properly configure your client’s TLS versions by
 visiting Tools to Build on AWS [7] or our associated AWS Security Blog has a link for
 each unique code language [7].\n\nWhat is Transport Layer Security (TLS)?\nTransport
 Layer Security (TLS Protocols) are cryptographic protocols designed to provide secure
 communication across a computer network
[6].\n\nWhat are AWS FIPS endpoints? \nAll AWS services offer Transport Layer Security
 (TLS) 1.2 encrypted endpoints that can be used for all API calls. Some AWS services
 also offer FIPS 140-2 endpoints [9] for customers that require use of FIPS validated
 cryptographic libraries. \n\n[1] https://aws.amazon.com/blogs/security/tag/tls/\n[2]
 https://aws.amazon.com/support\n[3]
https://docs.aws.amazon.com/AmazonS3/latest/dev/LogFormat.html\n[4] https://
docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/AccessLogs.html\n[5]
 https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-
access-logs.html\n[6] https://aws.amazon.com/tools\n[7] https://aws.amazon.com/blogs/
security/tls-1-2-to-become-the-minimum-for-all-aws-fips-endpoints\n[8]
https://en.wikipedia.org/wiki/Transport_Layer_Security\n[9] https://aws.amazon.com/
compliance/fips'}
```

8. When you're finished, enter the following command to deactivate the virtual machine.

```
deactivate
```

Python resources

- For more information about the `Health. Client`, see the AWS SDK for Python (Boto3) API Reference.
- For more information about the library used in this demo for DNS lookups, see the dnspython toolkit and the source code on GitHub.

Signing AWS Health API requests

When you use the AWS SDKs or the AWS Command Line Interface (AWS CLI) to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. For example, if you use the AWS SDK for Java for the previous high availability endpoint demo, you don't need to sign requests yourself.

Java code examples

For more examples on how to use the AWS Health API with the AWS SDK for Java, see this example code (p. 19).

When you make requests, we strongly recommend that you don't use your AWS root account credentials for regular access to AWS Health. You can use the credentials for an IAM user. For more information, see Lock Away Your AWS Account Root User Access Keys in the *IAM User Guide*.

If you don’t use the AWS SDKs or the AWS CLI, then you must sign your requests yourself. We recommend that you use AWS Signature Version 4. For more information, see Signing AWS API Requests in the *AWS General Reference*.

Supported operations in AWS Health

AWS Health supports the following operations for getting information about events that affect an AWS account:

- The event types supported by AWS Health.
- Information about one or more events that match specified filter criteria.
- Information about the entities that are affected by one or more events.
- Categorized counts of events or entities that match specified filter criteria.

All operations are non-mutating. That is, they retrieve data but do not modify it. The following sections summarize the AWS Health operations:

Event types

The DescribeEventTypes operation retrieves event types that match the optional specified filter. An event type is a template definition of an event's AWS service, event type code, and category. An event type and event are similar to a class and object in object-oriented programming. The number of event types supported by AWS Health grows over time.

Events

The DescribeEvents operation retrieves summary information about events that are related to an AWS account. The events can be related to AWS operational issues, scheduled changes to AWS infrastructure, or security and billing notifications. The DescribeEventDetails operation retrieves detailed information about one or more events, such as the AWS service, Region, Availability Zone, event start and end times, and a text description.

Affected entities

The DescribeAffectedEntities operation retrieves information about entities that are affected by one or more events. The results can be filtered by additional criteria, such as status, that might be assigned to AWS resources.

Aggregation

The DescribeEventAggregates operation retrieves a count of the events in each event type category, optionally filtered by other criteria. The DescribeEntityAggregates operation retrieves a count of the entities (resources) that are affected by one or more specified events.

AWS Organizations and Organization View

DescribeEventsForOrganization

DescribeEventsForOrganization returns summary information about events across the AWS Organizations, meeting the specified filter criteria.

DescribeAffectedAccountsForOrganization

DescribeAffectedAccountsForOrganization returns a list of AWS accounts in the AWS Organizations that are affected by the provided event.

DescribeEventDetailsForOrganization

DescribeEventDetailsForOrganization returns detailed information about one or more specified events for one or more accounts in AWS Organizations.

DescribeAffectedEntitiesForOrganization

DescribeAffectedEntitiesForOrganization returns a list of entities that have been affected by one or more events for one or more accounts in your organization, based on the filter criteria.

EnableHealthServiceAccessForOrganization

EnableHealthServiceAccessForOrganization operation grants the AWS Health service permission to interact with AWS Organizations on the customer’s behalf and applies a Service Linked Role to the management account in your organization.

DisableHealthServiceAccessForOrganization

DisableHealthServiceAccessForOrganization operation revokes permission for the AWS Health service to interact with AWS Organizations on the customer's behalf.

DescribeHealthServiceStatusForOrganization

DescribeHealthServiceStatusForOrganization operation provides status information on enabling or disabling AWS Health to work with your organization

For more information about these operations, see the AWS Health API Reference.

Sample Java code for the AWS Health API

The following Java code examples demonstrate how to initialize an AWS Health client and retrieve information about events and entities.

Step 1: Initialize credentials

Valid credentials are required to communicate with the AWS Health API. You can use the key pair of any IAM user associated with the AWS account.

Create and initialize an AWSCredentials instance:

```
AWSCredentials credentials = null;
try {
       credentials = new ProfileCredentialsProvider("default").getCredentials();
} catch (Exception e) {
throw new AmazonClientException(
   "Cannot load the credentials from the credential profiles file. "
   + "Please make sure that your credentials file is at the correct "
   + "location (/home/username/.aws/credentials), and is in valid format.", e);
}
```

Step 2: Initialize an AWS Health API client

Use the initialized credentials object from the previous step to create an AWS Health client:

```
import com.amazonaws.services.health.AWSHealthClient;

AWSHealth awsHealthClient = new AWSHealthClient(credentials);
```

Step 3: Use AWS Health API operations to get event information

DescribeEvents

```
import com.amazonaws.services.health.model.DescribeEventsRequest;
import com.amazonaws.services.health.model.DescribeEventsResult;
import com.amazonaws.services.health.model.Event;
import com.amazonaws.services.health.model.EventFilter;

DescribeEventsRequest request = new DescribeEventsRequest();

EventFilter filter = new EventFilter();
// Filter on any field from the supported AWS Health EventFilter model.
// Here is an example for Region us-east-1 events from the EC2 service.
filter.setServices(singletonList("EC2"));
filter.setRegions(singletonList("us-east-1"));
request.setFilter(filter);
```

```
DescribeEventsResult response = awsHealthClient.describeEvents(request);
List<Event> resultEvents = response.getEvents();

Event currentEvent = null;
for (Event event : resultEvents) {
    // Display result event data; here is a subset.
    System.out.println(event.getArn());
    System.out.println(event.getService());
    System.out.println(event.getRegion());
    System.out.println(event.getAvailabilityZone());
    System.out.println(event.getStartTime());
    System.out.println(event.getEndTime());
 }
```

DescribeEventAggregates

```
import com.amazonaws.services.health.model.DescribeEventAggregatesRequest;
import com.amazonaws.services.health.model.DescribeEventAggregatesResult;
import com.amazonaws.services.health.model.EventAggregate;
import com.amazonaws.services.health.model.EventFilter;

DescribeEventAggregatesRequest request = new DescribeEventAggregatesRequest();
// set the aggregation field
request.setAggregateField("eventTypeCategory");

// filter more on result if needed
EventFilter filter = new EventFilter();
filter.setRegions(singleton("us-east-1"));
request.setFilter(filter);

DescribeEventAggregatesResult response = awsHealthClient.describeEventAggregates(request);

// print event count for each eventTypeCategory
for (EventAggregate aggregate: response.getEventAggregates()) {
    System.out.println("Event Category:" + aggregate.getAggregateValue());
    System.out.println("Event Count:" + aggregate.getCount());
 }
```

DescribeEventDetails

```
import com.amazonaws.services.health.model.DescribeEventDetailsRequest;
import com.amazonaws.services.health.model.DescribeEventDetailsResult;
import com.amazonaws.services.health.model.Event;
import com.amazonaws.services.health.model.EventDetails;


DescribeEventDetailsRequest describeEventDetailsRequest = new
 DescribeEventDetailsRequest();
// set event ARN and local value

describeEventDetailsRequest.setEventArns(singletonList("arn:aws:health:us-
east-1::event/service/eventTypeCode/eventId"));
describeEventDetailsRequest.setLocale("en-US");
filter.setEventArns
DescribeEventDetailsResult describeEventDetailsResult =
 awsHealthClient.describeEventDetails(request);
EventDetails eventDetail = describeEventDetailsResult.getSuccessfulSet().get(0);

// check event-related fields
Event event = eventDetail.getEvent();
System.out.println(event.getService());
System.out.println(event.getRegion());
System.out.println(event.getAvailabilityZone());
System.out.println(event.getStartTime());
System.out.println(event.getEndTime());

// print out event description
System.out.println(eventDetail.getEventDescription().getLatestDescription());
```

DescribeAffectedEntities

```
import com.amazonaws.services.health.model.AffectedEntity;
import com.amazonaws.services.health.model.DateTimeRange;
import com.amazonaws.services.health.model.DescribeAffectedEntitiesRequest;
import
 com.amdescribeEventDetailsRequestazonaws.services.health.model.DescribeAffectedEntitiesResult;

DescribeAffectedEntitiesRequest request = new DescribeAffectedEntitiesRequest();
EntityFilter filter = new EntityFilter();

filter.setEventArns(singletonList("arn:aws:health:us-
east-1::event/service/eventTypeCode/eventId"));

DescribeAffectedEntitiesResult response =
 awsHealthClient.describeAffectedEntities(request);

for (AffectedEntity affectedEntity: response.getEntities()) {
    System.out.println(affectedEntity.getEntityValue());
    System.out.println(affectedEntity.getAwsAccountId());
    System.out.println(affectedEntity.getEntityArn());
 }
```

DescribeEntityAggregates

```
import com.amazonaws.services.health.model.DescribeEntityAggregatesRequest;
import com.amazonaws.services.health.model.DescribeEntityAggregatesResult;
import com.amazonaws.services.health.model.EntityAggregate;

DescribeEntityAggregatesRequest request = new DescribeEntityAggregatesRequest();

request.setEventArns(singletonList("arn:aws:health:us-
east-1::event/service/eventTypeCode/eventId"));

DescribeEntityAggregatesResult response =
 awsHealthClient.describeEntityAggregates(request);

for (EntityAggregate entityAggregate : response.getEntityAggregates()) {
    System.out.println(entityAggregate.getEventArn());
    System.out.println(entityAggregate.getCount());
 }
```

Security in AWS Health

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to AWS Health, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You're also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AWS Health. The following topics show you how to configure AWS Health to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS Health resources.

Topics

Data protection in AWS Health

The AWS shared responsibility model applies to data protection in AWS Health. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.

- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a **Name** field. This includes when you work with AWS Health or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Data encryption

See the following information about how AWS Health encrypts data.

Data encryption refers to protecting data while in-transit (as it travels from the service to your AWS account), and at rest (while it is stored in AWS services). You can protect data in transit using Transport Layer Security (TLS) or at rest using client-side encryption.

AWS Health doesn't record personal identifying information (PII) such as email addresses or customer names in events.

Encryption at rest

All data stored by AWS Health is encrypted at rest.

Encryption in transit

All data sent to and from AWS Health is encrypted in transit.

Key management

AWS Health doesn't support customer-managed encryption keys for data encrypted in the AWS Cloud.

Internetwork traffic privacy

See the following information about how AWS Health works with traffic privacy.

You can enable AWS Health to work with AWS Organizations, so that you can view events from individual AWS accounts that are part of your organization. This feature provides you a centralized view for all AWS Health events, which includes operational issues, scheduled maintenance, and account notifications.

To do so, you must sign in with your organization's management account and enable this feature from the AWS Management Console or use the EnableHealthServiceAccessForOrganization API operation.

For more information, see Aggregating AWS Health events across accounts with organizational view (p. 46).

Identity and access management for AWS Health

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use AWS Health resources. IAM is an AWS service that you can use with no additional charge.

Topics

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in AWS Health.

Service user – If you use the AWS Health service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS Health features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in AWS Health, see Troubleshooting AWS Health identity and access (p. 39).

Service administrator – If you're in charge of AWS Health resources at your company, you probably have full access to AWS Health. It's your job to determine which AWS Health features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with AWS Health, see How AWS Health works with IAM (p. 28).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to AWS Health. To view example AWS Health identity-based policies that you can use in IAM, see AWS Health identity-based policy examples (p. 32).

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see Signing in to the AWS Management Console as an IAM user or root user in the *IAM User Guide*.

You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can also use your company's single sign-on authentication or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the AWS Management Console, use your password with your root user email address or your IAM user name. You can access AWS programmatically using your root user or IAM users access keys. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don't use AWS tools, you must sign the request yourself. Do this using *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 signing process in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.

AWS account root user

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see Managing access keys for IAM users in the *IAM User Guide*. When you generate access keys for an IAM user, make sure you view and securely save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to create an IAM user (instead of a role) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated users and roles in the *IAM User Guide*.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects

in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.

  - **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see Actions, Resources, and Condition Keys for AWS Health in the *Service Authorization Reference*.
  - **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.
  - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see When to create an IAM role (instead of a user) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. You can sign in as the root user or an IAM user, or you can assume an IAM role. When you then make a request, AWS evaluates the related identity-based or resource-based policies. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

AWS Health supports resource-based conditions. You can specify which AWS Health events that users can view. For example, you might create a policy that only allows an IAM user access to specific Amazon EC2 events in the AWS Personal Health Dashboard.

For more information, see Resources (p. 29).

Access control lists

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

AWS Health doesn't support ACLs.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies.

Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How AWS Health works with IAM

Before you use IAM to manage access to AWS Health, you should understand what IAM features are available to use with AWS Health. To get a high-level view of how AWS Health and other AWS services work with IAM, see AWS Services That Work with IAM in the *IAM User Guide*.

Topics

AWS Health identity-based policies

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. AWS Health supports specific actions, resources, and condition keys. To learn about all of the elements that you use in a JSON policy, see IAM JSON Policy Elements Reference in the *IAM User Guide*.

Actions

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

Policy actions in AWS Health use the following prefix before the action: `health:`. For example, to grant someone permission to view detailed information about specified events with the DescribeEventDetails API operation, you include the `heath:DescribeEventDetails` action in the policy.

Policy statements must include an `Action` or `NotAction` element. AWS Health defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as follows.

```
"Action": [
      "health:action1",
      "health:action2"
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Describe`, include the following action.

```
"Action": "health:Describe*"
```

To see a list of AWS Health actions, see Actions Defined by AWS Health in the *IAM User Guide*.

Resources

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

An AWS Health event has the following Amazon Resource Name (ARN) format.

```
arn:${Partition}:health:*::event/service/event-type-code/event-ID
```

For example, to specify the `EC2_INSTANCE_RETIREMENT_SCHEDULED_ABC123-DEF456` event in your statement, use the following ARN.

```
"Resource": "arn:aws:health:*::event/EC2/EC2_INSTANCE_RETIREMENT_SCHEDULED/
EC2_INSTANCE_RETIREMENT_SCHEDULED_ABC123-DEF456"
```

To specify all AWS Health events for Amazon EC2 that belong to a specific account, use the wildcard (*).

```
"Resource": "arn:aws:health:*::event/EC2/*/*"
```

For more information about the format of ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces.

Some AWS Health actions can't be performed on a specific resource. In those cases, you must use the wildcard (*).

```
"Resource": "*"
```

AWS Health API operations can involve multiple resources. For example, the DescribeEvents operation returns information about events that meet a specified filter criteria. This means that an IAM user must have permissions to view this event.

To specify multiple resources in a single statement, separate the ARNs with commas.

```
"Resource": [
      "resource1",
      "resource2"
```

AWS Health supports only resource-level permissions for health events and only for the DescribeAffectedEntities and DescribeEventDetails API operations. For more information, see Resource- and action-based conditions (p. 38).

To see a list of AWS Health resource types and their ARNs, see Resources Defined by AWS Health in the *IAM User Guide*. To learn with which actions you can specify the ARN of each resource, see Actions Defined by AWS Health.

Condition keys

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

AWS Health defines its own set of condition keys and also supports using some global condition keys. To see all AWS global condition keys, see AWS Global Condition Context Keys in the *IAM User Guide*.

The DescribeAffectedEntities and DescribeEventDetails API operations support the `health:eventTypeCode` and `health:service` condition keys.

To see a list of AWS Health condition keys, see Condition Keys for AWS Health in the *IAM User Guide*. To learn with which actions and resources you can use a condition key, see Actions Defined by AWS Health.

Examples

To view examples of AWS Health identity-based policies, see AWS Health identity-based policy examples (p. 32).

AWS Health resource-based policies

Resource-based policies are JSON policy documents that specify what actions a specified principal can perform on the AWS Health resource and under what conditions. AWS Health supports resource-based permissions policies for health events. Resource-based policies let you grant usage permission to other accounts on a per-resource basis. You can also use a resource-based policy to allow an AWS service to access your AWS Health events.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, you must also grant the principal entity permission to access the resource. Grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see How IAM Roles Differ from Resource-based Policies in the *IAM User Guide*.

AWS Health supports only resource-based policies for the DescribeAffectedEntities and DescribeEventDetails API operations. You can specify these actions in a policy to define which principal entities (accounts, users, roles, and federated users) can perform actions on the AWS Health event.

Examples

To view examples of AWS Health resource-based policies, see Resource- and action-based conditions (p. 38).

Authorization based on AWS Health tags

AWS Health doesn't support tagging resources or controlling access based on tags.

AWS Health IAM roles

An IAM role is an entity within your AWS account that has specific permissions.

Using temporary credentials with AWS Health

You can use temporary credentials to sign in with federation, assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS STS API operations such as AssumeRole or GetFederationToken.

AWS Health supports using temporary credentials.

Service-linked roles

Service-linked roles allow AWS services to access resources in other services to complete an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view but not edit the permissions for service-linked roles.

AWS Health supports service-linked roles to integrate with AWS Organizations. The role is named Health_OrganizationsServiceRolePolicy, which allows AWS Health to access health events from other AWS accounts in the organization.

You can use the EnableHealthServiceAccessForOrganization operation to create the service-linked role in the account. However, if you want to disable this feature, you must first call the DisableHealthServiceAccessForOrganization operation. You can then delete the role through the IAM console, IAM API, or AWS Command Line Interface (AWS CLI). For more information, see Using service-linked roles in the *IAM User Guide*.

For more information, see Aggregating AWS Health events across accounts with organizational view (p. 46).

Service roles

This feature allows a service to assume a service role on your behalf. This role allows the service to access resources in other services to complete an action on your behalf. Service roles appear in your IAM account and are owned by the account. This means that an IAM administrator can change the permissions for this role. However, doing so might break the functionality of the service.

AWS Health doesn't support service roles.

AWS Health identity-based policy examples

By default, IAM users and roles don't have permission to create or modify AWS Health resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see Creating Policies on the JSON Tab in the *IAM User Guide*.

Topics

Policy best practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete AWS Health resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started using AWS managed policies** – To start using AWS Health quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see Get started using permissions with AWS managed policies in the *IAM User Guide*.
- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see Grant least privilege in the *IAM User Guide*.
- **Enable MFA for sensitive operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.
- **Use policy conditions for extra security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.

Using the AWS Health console

To access the AWS Health console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS Health resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (IAM users or roles) with that policy.

To ensure that those entities can still use the AWS Health console, you can attach the following AWS managed policy, AWSHealthFullAccess.

The `AWSHealthFullAccess` policy grants an entity full access to the following:

- Enable or disable the AWS Health organizational view feature for all accounts in an AWS organization
- The AWS Personal Health Dashboard in the AWS Health console
- AWS Health API operations and notifications
- View information about accounts that are part of your AWS organization
- View the organizational units (OU) of the management account

Example : AWSHealthFullAccess

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "organizations:EnableAWSServiceAccess",
                "organizations:DisableAWSServiceAccess"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "organizations:ServicePrincipal": "health.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "health:*",
                "organizations:DescribeAccount",
                "organizations:ListAccounts",
                "organizations:ListDelegatedAdministrators",
                "organizations:ListParents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "health.amazonaws.com"
                }
            }
        }
    ]
}
```

Note
You can also use the AWS managed role, `Health_OrganizationsServiceRolePolicy` so that AWS Health can view events for other accounts in your organization. For more information, see Using service-linked roles for AWS Health (p. 41).

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that you're trying to perform.

For more information, see Adding Permissions to a User in the *IAM User Guide*.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Accessing the AWS Personal Health Dashboard and the AWS Health API

The AWS Personal Health Dashboard is available for all AWS accounts. The AWS Health API is available only to accounts with a Business or Enterprise support plan. For more information, see AWS Support.

You can use IAM to create entities (users, groups, or roles), and then give those entities permissions to access the AWS Personal Health Dashboard and the AWS Health API.

By default, IAM users don't have access to the AWS Personal Health Dashboard or the AWS Health API. You give users access to your account's AWS Health information by attaching IAM policies to a single user, a group of users, or a role. For more information, see Identities (Users, Groups, and Roles) and Overview of IAM Policies.

After you create IAM users, you can give those users individual passwords. Then, they can sign in to your account and view AWS Health information by using an account-specific sign-in page. For more information, see How Users Sign In to Your Account.

Note
An IAM user with permissions to view AWS Personal Health Dashboard has read-only access to health information across all AWS services on the account, which can include, but is not limited

to, AWS resource IDs such as Amazon EC2 instance IDs, EC2 instance IP addresses, and general security notifications.
For example, if an IAM policy grants access only to AWS Personal Health Dashboard and the AWS Health API, then the user or role that the policy applies to can access all information posted about AWS services and related resources, even if other IAM policies don't allow that access.

You can use two groups of APIs for AWS Health.

- Individual accounts – You can use the operations such as DescribeEvents and DescribeEventDetails to get information about AWS Health events for your account.
- Organizational account – You can use operations such as DescribeEventsForOrganization and DescribeEventDetailsForOrganization to get information about AWS Health events for accounts that are part of your organization.

For more information about the available API operations, see the AWS Health API Reference.

Individual actions

You can set the `Action` element of an IAM policy to `health:Describe*`. This allows access to the AWS Personal Health Dashboard and AWS Health. AWS Health supports access control to events based on the `eventTypeCode` and service.

Describe access

This policy statement grants access to AWS Personal Health Dashboard and any of the `Describe*` AWS Health API operations. For example, an IAM user with this policy can access the AWS Personal Health Dashboard in the AWS Management Console and call the AWS Health `DescribeEvents` API operation.

Example : Describe access

```
{
  "Version": "2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "health:Describe*"
    ],
    "Resource": "*"
  }]
}
```

Deny access

This policy statement denies access to AWS Personal Health Dashboard and the AWS Health API. An IAM user with this policy can't view the AWS Personal Health Dashboard in the AWS Management Console and can't call any of the AWS Health API operations.

Example : Deny access

```
{
  "Version": "2012-10-17",
  "Statement": [
  {
    "Effect": "Deny",
    "Action": [
      "health:*"
    ],
    "Resource": "*"
  }]
}
```

Organizational view

If you want to enable organizational view for AWS Health, you must allow access to the AWS Health and AWS Organizations actions.

The `Action` element of an IAM policy must include the following permissions:

- `iam:CreateServiceLinkedRole`
- `organizations:EnableAWSServiceAccess`
- `organizations:DescribeAccount`
- `organizations:DisableAWSServiceAccess`
- `organizations:ListAccounts`
- `organizations:ListDelegatedAdministrators`
- `organizations:ListParents`

To understand the exact permissions needed for each APIs, see Actions Defined by AWS Health APIs and Notifications in the *IAM User Guide*.

Note
You must use credentials from the management account for an organization to access the AWS Health APIs for AWS Organizations. For more information, see Aggregating AWS Health events across accounts with organizational view (p. 46).

Allow access to AWS Health organizational view

This policy statement grants access to all AWS Health and AWS Organizations actions that you need for the organizational view feature.

Example : Allow AWS Health organizational view access

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "organizations:EnableAWSServiceAccess",
                "organizations:DisableAWSServiceAccess"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "organizations:ServicePrincipal": "health.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "health:*",
                "organizations:DescribeAccount",
                "organizations:ListAccounts",
                "organizations:ListDelegatedAdministrators",
                "organizations:ListParents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/health.amazonaws.com/
AWSServiceRoleForHealth*"
        }
    ]
}
```

Deny access to AWS Health organizational view

This policy statement denies access to the AWS Organizations actions but allows access to the AWS Health actions for an individual account.

Example : Deny AWS Health organizational view access

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "health:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "organizations:EnableAWSServiceAccess",
                "organizations:DisableAWSServiceAccess"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "organizations:ServicePrincipal": "health.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "organizations:DescribeAccount",
                "organizations:ListAccounts",
                "organizations:ListDelegatedAdministrators",
                "organizations:ListParents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/health.amazonaws.com/
AWSServiceRoleForHealth*"
        }
    ]
}
```

Note

If the user or group that you want to give permissions to already has an IAM policy, you can add the AWS Health-specific policy statement to that policy.

Resource- and action-based conditions

AWS Health supports IAM conditions for the DescribeAffectedEntities and DescribeEventDetails API operations. You can use resource- and action-based conditions to restrict events that the AWS Health API sends to a user, group, or role.

To do so, update the `Condition` block of the IAM policy or set the `Resource` element. You can use String Conditions to restrict access based on certain AWS Health event fields.

You can use the following fields when you specify an AWS Health event in your policy:

- `eventTypeCode`
- `service`

Notes

- The DescribeAffectedEntities and DescribeEventDetails API operations support resource-level permissions. For example, you can create a policy to allow or deny specific AWS Health events.
- The DescribeAffectedEntitiesForOrganization and DescribeEventDetailsForOrganization API operations don't support resource-level permissions.
- For more information, see Actions, resources, and condition keys for AWS Health APIs and Notifications in the *Service Authorization Reference*.

Example : Action-based condition

This policy statement grants access to AWS Personal Health Dashboard and the AWS Health `Describe*` API operations, but denies access to any AWS Health events that relate to Amazon EC2.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "health:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "health:DescribeAffectedEntities",
                "health:DescribeEventDetails"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "health:service": "EC2"
                }
            }
        }
    ]
}
```

Example : Resource-based condition

The following policy has the same effect, but uses the `Resource` element instead.

```
{
  "Version": "2012-10-17",
"Statement": [
{
  "Effect": "Allow",
  "Action": [
    "health:Describe*"
  ],
  "Resource": "*"
},
{
  "Effect": "Deny",
  "Action": [
    "health:DescribeEventDetails",
    "health:DescribeAffectedEntities"
  ],
  "Resource": "arn:aws:health:*::event/EC2/*/*"
}]
}
```

Example : eventTypeCode condition

This policy statement grants access to AWS Personal Health Dashboard and the AWS Health `Describe*` API operations, but denies access to any AWS Health events with the `eventTypeCode` that matches `AWS_EC2_*`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "health:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "health:DescribeAffectedEntities",
                "health:DescribeEventDetails"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "health:eventTypeCode": "AWS_EC2_*"
                }
            }
        }
    ]
}
```

Important
If you call the DescribeAffectedEntities and DescribeEventDetails operations and don't have permission to access the AWS Health event, the `AccessDeniedException` error appears. For more information, see Troubleshooting AWS Health identity and access (p. 39).

Troubleshooting AWS Health identity and access

Use the following information to diagnose and fix common issues that you might encounter when working with AWS Health and IAM.

Topics

I'm not authorized to perform an action in AWS Health

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The `AccessDeniedException` error appears when a user doesn't have permission to use AWS Personal Health Dashboard or the AWS Health API operations.

In this case, the user's administrator must update the policy to allow the user access.

The AWS Health API requires a Business or Enterprise support plan from AWS Support. If you call the AWS Health API from an account that doesn't have a Business or Enterprise support plan, the following error code is returned: `SubscriptionRequiredException`.

I'm not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password. Ask that person to update your policies to allow you to pass a role to AWS Health.

Some AWS services allow you to pass an existing role to that service, instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in AWS Health. However, the action requires the service to have permissions granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform: iam:PassRole
```

In this case, Mary asks her administrator to update her policies to allow her to perform the `iam:PassRole` action.

I want to view my access keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

Important
Do not provide your access keys to a third party, even to help find your canonical user ID. By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see Managing access keys in the *IAM User Guide*.

I'm an administrator and want to allow others to access AWS Health

To allow others to access AWS Health, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in AWS Health.

To get started right away, see Creating your first IAM delegated user and group in the *IAM User Guide*.

I want to allow people outside of my AWS account to access my AWS Health resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether AWS Health supports these features, see How AWS Health works with IAM (p. 28).
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

Using service-linked roles for AWS Health

AWS Health uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS Health. Service-linked roles are predefined by AWS Health and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up AWS Health easier because you don't have to manually add the necessary permissions. AWS Health defines the permissions of its service-linked roles, and unless defined otherwise, only AWS Health can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

Service-linked role permissions for AWS Health

AWS Health uses the service-linked role named `Health_OrganizationsServiceRolePolicy`.

The service-linked role trusts the `health.amazonaws.com` service to assume the role.

The role uses the following permissions policy to allow AWS Health to list accounts in AWS Organizations.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "organizations:ListAccounts",
            "Resource": "*"
        },
        {
            "Sid": "ListAWSServiceAccessForOrganization0",
            "Effect": "Allow",
            "Action": "organizations:ListAWSServiceAccessForOrganization",
            "Resource": "*"
        }
    ]
}
```

Creating a service-linked role for AWS Health

You don't need to manually create a service-linked role. When you call the EnableHealthServiceAccessForOrganization operation, AWS Health creates the service-linked role in the account for you.

Editing a service-linked role for AWS Health

AWS Health doesn't allow you to edit the service-linked role. After you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for AWS Health

If you want to delete this role, you must first call the DisableHealthServiceAccessForOrganization operation. You can then delete the role through the IAM console, IAM API, or AWS Command Line Interface (AWS CLI). For more information, see Using service-linked roles in the *IAM User Guide*.

Logging and monitoring in AWS Health

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Health and your other AWS solutions. AWS provides the following monitoring tools to watch AWS Health, report when something is wrong, and take actions when appropriate:

- *Amazon CloudWatch* monitors your AWS resources and the applications that you run on AWS in real time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify. For example, you can have CloudWatch track CPU usage or other metrics of your Amazon Elastic Compute Cloud (Amazon EC2) instances and automatically launch new instances when needed. For more information, see the Amazon CloudWatch User Guide.
- *Amazon CloudWatch Events* delivers a near-real-time stream of system events that describe changes in AWS resources. CloudWatch Events enables automated event-driven computing. You can write rules that watch for certain events and trigger automated actions in other AWS services when these events happen. For more information, see the Amazon CloudWatch Events User Guide.
- *AWS CloudTrail* captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon Simple Storage Service (Amazon S3) bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.

For more information, see Monitoring AWS Health (p. 71).

Compliance validation for AWS Health

Third-party auditors assess the security and compliance of AWS services as part of multiple AWS compliance programs, such as SOC, PCI, FedRAMP, and HIPAA.

To learn whether AWS Health or other AWS services are in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.

  Note
  Not all services are compliant with HIPAA.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.
- AWS Audit Manager – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in AWS Health

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

AWS Health events are stored and replicated across multiple Availability Zones. This approach ensures that you can access them from the AWS Personal Health Dashboard or the AWS Health API operations. You can view AWS Health events up to 90 days from when they occur.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in AWS Health

As a managed service, AWS Health is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

You use AWS published API calls to access AWS Health through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Configuration and vulnerability analysis in AWS Health

Configuration and IT controls are a shared responsibility between AWS and you, our customer. For more information, see the AWS shared responsibility model.

Security best practices for AWS Health

See the following best practices for working with AWS Health.

Grant AWS Health users minimum possible permissions

Follow the principle of least privilege by using the minimum set of access policy permissions for your users and groups. For example, you might allow an AWS Identity and Access Management (IAM) user access to the AWS Personal Health Dashboard. However, you might not allow that same user to enable or disable access to AWS Organizations.

For more information, see AWS Health identity-based policy examples (p. 32).

View the AWS Personal Health Dashboard

Check your AWS Personal Health Dashboard often to identify events that might affect your account or applications. For example, you might receive an event notification about your resources, such as an Amazon Elastic Compute Cloud (Amazon EC2) instance that needs to be updated.

For more information, see Getting started with the AWS Personal Health Dashboard (p. 7).

Integrate AWS Health with Amazon Chime or Slack

You can integrate AWS Health with your chat tools. This integration lets you and your team get notified about AWS Health events in real time. For more information, see the AWS Health Tools in GitHub.

Monitor for AWS Health events

You can integrate AWS Health with Amazon CloudWatch Events, so that you can create rules for specific events. When CloudWatch Events detects an event that matches your rule, you are notified and can then take action. CloudWatch Events events are Region-specific, so you must configure this service in the Region in which your application or infrastructure resides.

In some cases, the Region for the AWS Health event can't be determined. If that situation occurs, the event appears in the US East (N. Virginia) Region by default. You can set up CloudWatch Events in this Region to ensure that you monitor these events.

For more information, see Monitoring AWS Health events with Amazon CloudWatch Events (p. 58).

Aggregating AWS Health events across accounts with organizational view

By default, you can use AWS Health to view the AWS Health events of a single AWS account. If you use AWS Organizations, you can also view AWS Health events centrally across your organization. This feature provides access to the same information as single account operations. You can use filters to view events in specific AWS Regions, accounts, and services.

You can aggregate events to identify accounts in your organization that are affected by an operational event or get notified for security vulnerabilities. You can then use this information to proactively manage and automate resource maintenance events across your organization. Use this feature to stay informed of upcoming changes to AWS services that might require updates or code changes.

Important

- AWS Health doesn't record events that occurred in your organization before you enabled organizational view. For example, if a member account (111122223333) in your organization received an event for Amazon Elastic Compute Cloud (Amazon EC2) before you enabled this feature, this event won't appear in your organizational view.
- AWS Health events that were sent for accounts in your organization will appear in organizational view as long as the event is available, up to 90 days, even if one or more of those accounts leave your organization.
- Organizational events are available for 90 days before they're deleted. This quota can't be increased.

Prerequisites

Before you use organizational view, you must:

- Be part of an organization with all features enabled.
- Sign in to the management account as an AWS Identity and Access Management (IAM) user or assume an IAM role.

  You can also sign in as the root user (not recommended) in your organization's management account. For more information, see Lock away your AWS account root user access keys in the *IAM User Guide*.
- If you sign in as an IAM user, use an IAM policy that grants access to the AWS Health and Organizations actions, such as the AWSHealthFullAccess policy. For more information, see AWS Health identity-based policy examples (p. 32).

Topics

Organizational view (console)

You can use the AWS Health console to get a centralized view for health events in your AWS organization.

Organizational view is available in the AWS Health console for all AWS Support plans at no additional cost.

Note
If you want to allow users access to this feature in the management account, they must have permissions such as the AWSHealthFullAccess policy. For more information, see AWS Health identity-based policy examples (p. 32).

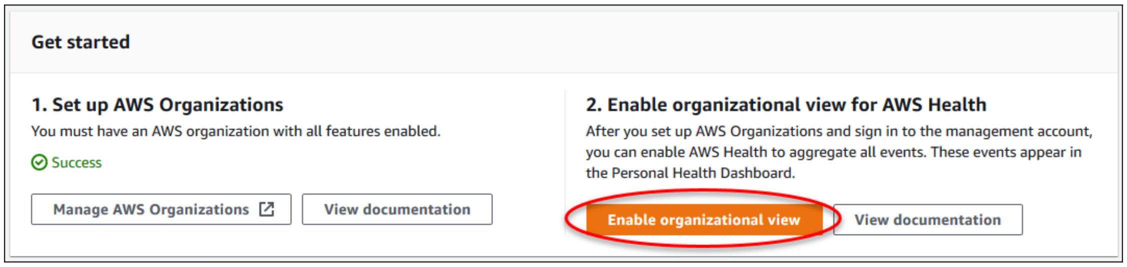
Contents

Enabling organizational view (console)

You can enable organizational view from the AWS Health console. You must sign in to the management account of your AWS organization.

To view the AWS Personal Health Dashboard for your organization

1. Sign in to the AWS Management Console and open the AWS Personal Health Dashboard at https://phd.aws.amazon.com/phd/home.
2. In the navigation pane, choose **Organizational view**, and then choose **Configurations**.
3. On the **Enable organizational view** page, choose **Enable organizational view**.



Get started

1. Set up AWS Organizations
You must have an AWS organization with all features enabled.
Success

2. Enable organizational view for AWS Health
After you set up AWS Organizations and sign in to the management account, you can enable AWS Health to aggregate all events. These events appear in the Personal Health Dashboard.

4. (Optional) If you want to make changes to your AWS organizations, such as creating organizational units (OUs), choose **Manage AWS Organizations**.

   For more information, see Getting started with AWS Organizations in the *AWS Organizations User Guide*.

Notes

- Enabling this feature is an asynchronous process and takes time to complete. Depending on the number of accounts in your organization, it can take several minutes to load the accounts. You can leave and check the AWS Health console later.

- If you have a Business or Enterprise support plan, you can call the DescribeHealthServiceStatusForOrganization API operation to check the status of the process.
- When you enable this feature, the `AWSServiceRoleForHealth_Organizations` service-linked role with the `Health_OrganizationsServiceRolePolicy` policy is applied to the management account in the organization. For more information, see Using service-linked roles for AWS Health (p. 41).

Viewing organizational view events (console)

After you enable organizational view, AWS Health displays health events for all accounts in your organization.

When an account joins your organization, AWS Health automatically adds the account to organizational view. When an account leaves your organization, new events from that account are no longer logged to organizational view. However, existing events remain and you can still query them up to the 90-day limit.

AWS retains the policy data for the account for 90 days from the effective date of the administrator account closure. At the end of the 90 day period, AWS permanently deletes all policy data for the account.

- To retain findings for more than 90 days, you can archive the policies. You can also use a custom action with an EventBridge rule to store the findings in an S3 bucket.
- As long as AWS retains the policy data, when you reopen the closed account, AWS reassigns the account as the service administrator and recovers the service policy data for the account.
- For more information, see Closing an account.

Important
For customers in the AWS GovCloud (US) Regions:

- Before closing your account, back up and then delete your policy data and other account resources. You will no longer have access to them after you close the account.

Note
When you enable this feature, the AWS Health console can display public events from the Service Health Dashboard for the last 7 days. These public events aren't specific to accounts in your organization. Events from the Service Health Dashboard provide public information about the regional availability of AWS services.

You can view organizational view events in the **Dashboard** or the **Event log** page.

Topics

Dashboard

You can use the **Dashboard** page to view events that might affect your AWS infrastructure, such as changes to AWS services and resources that affect your organization.

To view organizational view events in the Dashboard page

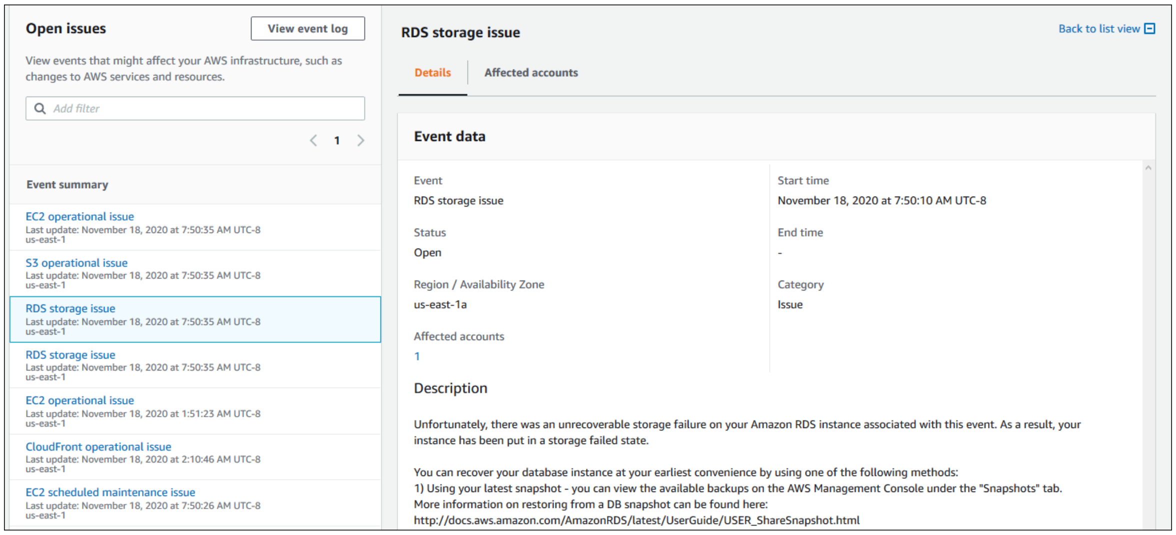
1. Sign in to the AWS Management Console and open the AWS Personal Health Dashboard at https://phd.aws.amazon.com/phd/home.

2. In the navigation pane, choose **Organizational view**, and then choose **Dashboard** to view recent and upcoming events.
3. You can choose the **Open issues**, **Scheduled changes**, or **Other notifications** tab, and then choose the event name.
4. On the **Details** tab, you can review the following information about the event:

   - Event name
   - Status
   - Region / Availability Zone
   - Affected accounts
   - Start time
   - End time
   - Category
   - Description

Example : Dashboard page for organizational view

The following Amazon Relational Database Service (Amazon RDS) event appears in the **Dashboard** page for organizational view and affects one account in the organization.



View events that might affect your AWS infrastructure, such as changes to AWS services and resources.

Event summary

EC2 operational issue
Last update: November 18, 2020 at 7:50:35 AM UTC-8
us-east-1

S3 operational issue
Last update: November 18, 2020 at 7:50:35 AM UTC-8
us-east-1

RDS storage issue
Last update: November 18, 2020 at 7:50:35 AM UTC-8
us-east-1

RDS storage issue
Last update: November 18, 2020 at 7:50:35 AM UTC-8
us-east-1

EC2 operational issue
Last update: November 18, 2020 at 1:51:23 AM UTC-8
us-east-1

CloudFront operational issue
Last update: November 18, 2020 at 2:10:46 AM UTC-8
us-east-1

EC2 scheduled maintenance issue
Last update: November 18, 2020 at 7:50:26 AM UTC-8
us-east-1

RDS storage issue

Event data

Event RDS storage issue	Start time November 18, 2020 at 7:50:10 AM UTC-8
Status Open	End time -
Region / Availability Zone us-east-1a	Category Issue
Affected accounts 1	

Description

Unfortunately, there was an unrecoverable storage failure on your Amazon RDS instance associated with this event. As a result, your instance has been put in a storage failed state.

You can recover your database instance at your earliest convenience by using one of the following methods:
1) Using your latest snapshot - you can view the available backups on the AWS Management Console under the "Snapshots" tab.
More information on restoring from a DB snapshot can be found here:
http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ShareSnapshot.html

Event log

You can also use the **Event log** page to view AWS Health events for organizational view. The column layout and behavior are similar to the **Dashboard** page, except that the **Event log** page includes additional columns and filter options, such as the **Event category**, **Status**, and **Start time**.

To view organizational view events in the Event log page

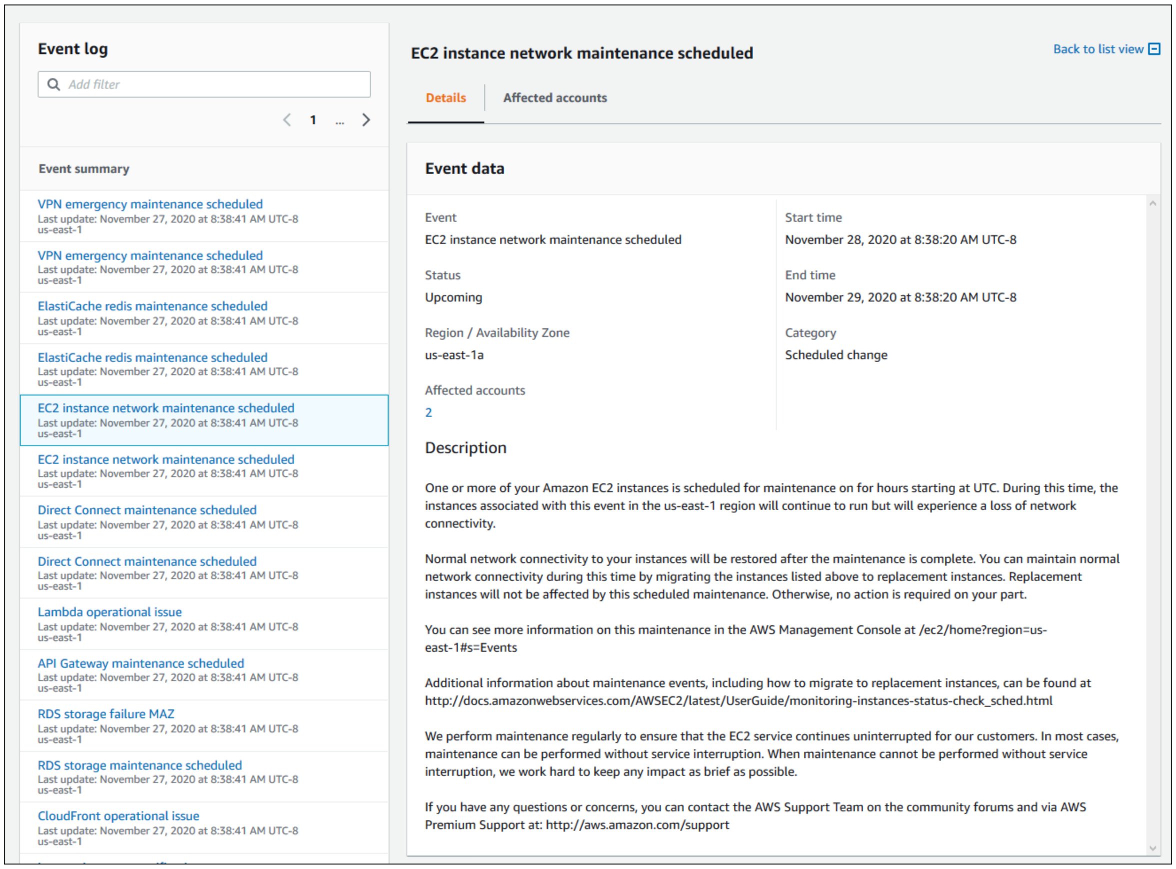
1. Sign in to the AWS Management Console and open the AWS Personal Health Dashboard at https://phd.aws.amazon.com/phd/home.
2. In the navigation pane, choose **Organizational view**, and then choose **Event log**.
3. Under **Event log**, choose the event name. You can review the following information about the event:

   - Event name
   - Status

- Region / Availability Zone
- Affected accounts
- Start time
- End time
- Category
- Description

Example : Event log page for organizational view

The following example Amazon DynamoDB (DynamoDB) event appears in the **Event log** page and affects two accounts in the organization.



Event summary

VPN emergency maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

VPN emergency maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

ElastiCache redis maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

ElastiCache redis maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

EC2 instance network maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

EC2 instance network maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

Direct Connect maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

Direct Connect maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

Lambda operational issue
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

API Gateway maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

RDS storage failure MAZ
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

RDS storage maintenance scheduled
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

CloudFront operational issue
Last update: November 27, 2020 at 8:38:41 AM UTC-8
us-east-1

EC2 instance network maintenance scheduled

Event data

Event	Start time
EC2 instance network maintenance scheduled	November 28, 2020 at 8:38:20 AM UTC-8
Status	End time
Upcoming	November 29, 2020 at 8:38:20 AM UTC-8
Region / Availability Zone	Category
us-east-1a	Scheduled change
Affected accounts	
2	

Description

One or more of your Amazon EC2 instances is scheduled for maintenance on for hours starting at UTC. During this time, the instances associated with this event in the us-east-1 region will continue to run but will experience a loss of network connectivity.

Normal network connectivity to your instances will be restored after the maintenance is complete. You can maintain normal network connectivity during this time by migrating the instances listed above to replacement instances. Replacement instances will not be affected by this scheduled maintenance. Otherwise, no action is required on your part.

You can see more information on this maintenance in the AWS Management Console at /ec2/home?region=us-east-1#s=Events

Additional information about maintenance events, including how to migrate to replacement instances, can be found at http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/monitoring-instances-status-check_sched.html

We perform maintenance regularly to ensure that the EC2 service continues uninterrupted for our customers. In most cases, maintenance can be performed without service interruption. When maintenance cannot be performed without service interruption, we work hard to keep any impact as brief as possible.

If you have any questions or concerns, you can contact the AWS Support Team on the community forums and via AWS Premium Support at: http://aws.amazon.com/support

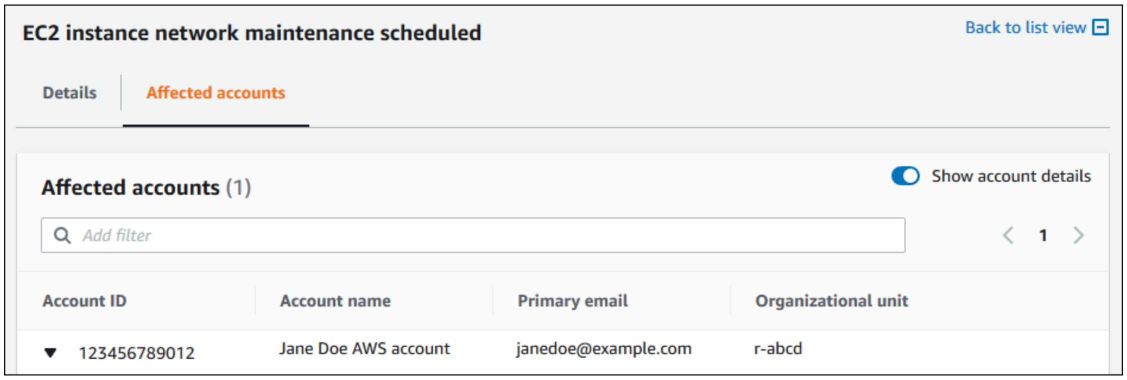
Viewing affected accounts and resources (console)

In the event details for the **Dashboard** and **Event log** pages, you can view the accounts in your organization that are affected by the event and any related resources. For example, if there's an upcoming event for Amazon Elastic Compute Cloud (Amazon EC2) instance maintenance, accounts in your organization that have Amazon EC2 instances can appear in the **Details** tab. You can identify the specific resources and then contact the account owner.

To view affected accounts and resources

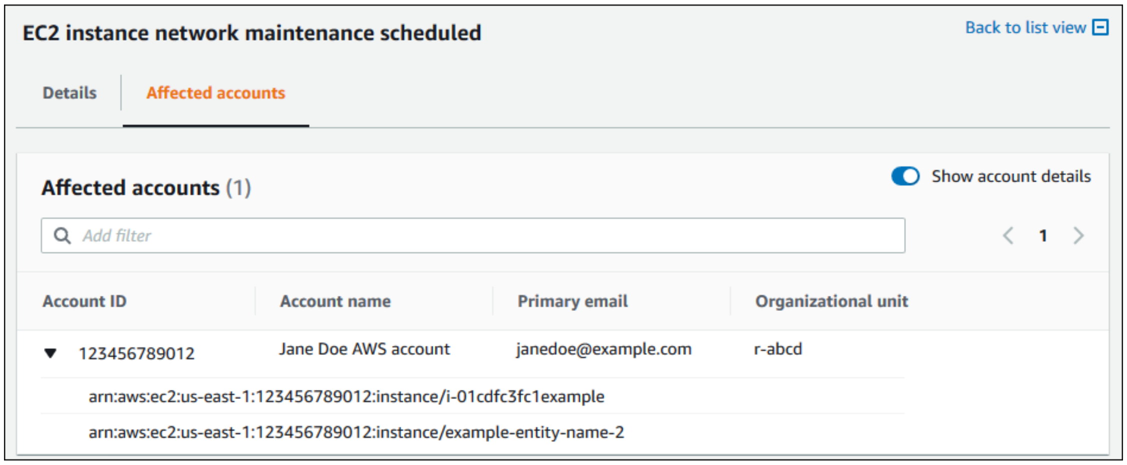
1. In the **Dashboard** or **Event log** page, choose an event that has a value for **Affected accounts**.
2. Choose the **Affected accounts** tab.
3. Choose **Show account details** to view the following information for the accounts:

- Account ID
- Account name
- Primary email
- Organizational unit (OU)



Account ID	Account name	Primary email	Organizational unit
▼ 123456789012	Jane Doe AWS account	janedoe@example.com	r-abcd

4. Expand the account to view the affected resources.



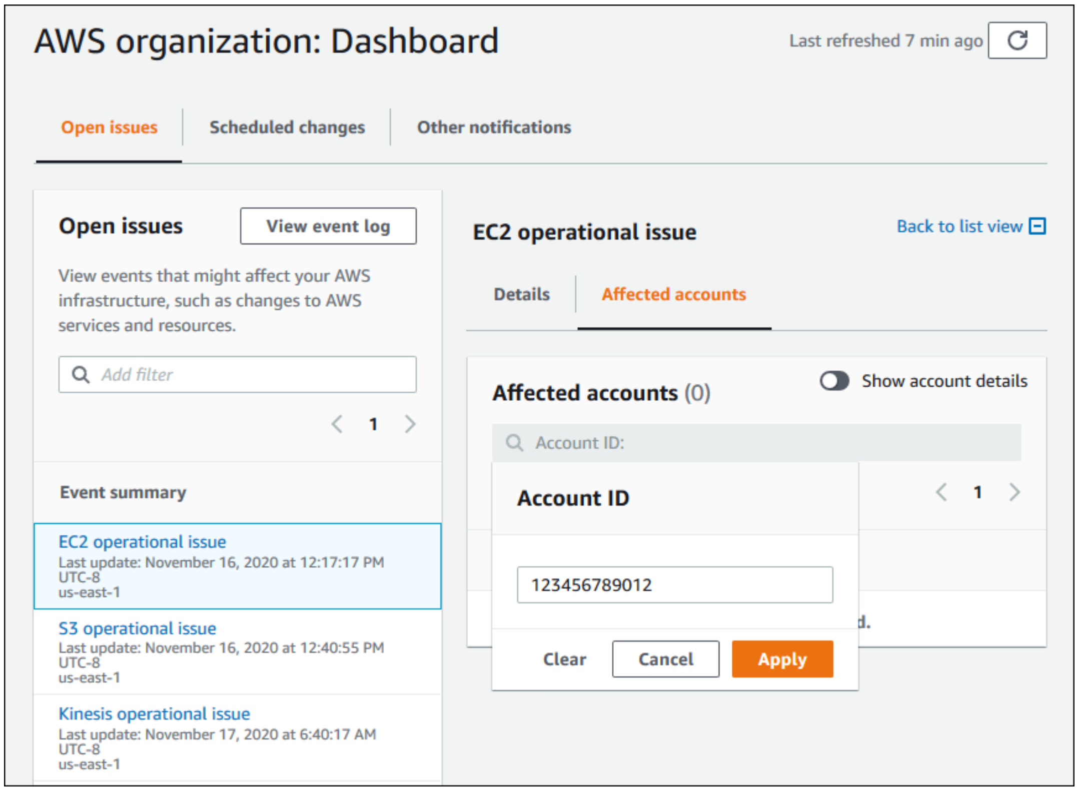
Account ID	Account name	Primary email	Organizational unit
▼ 123456789012	Jane Doe AWS account	janedoe@example.com	r-abcd

arn:aws:ec2:us-east-1:123456789012:instance/i-01cdfc3fc1example
arn:aws:ec2:us-east-1:123456789012:instance/example-entity-name-2

5. If there are more than 10 resources, choose **View all resources** to view them.
6. To filter by account ID for this specific event, do the following:

   a. On the **Affected accounts** tab, choose **Add filter**, choose **Account ID**, and then enter the account ID. You can only enter one account ID at a time.

   b. Choose **Apply**. The account that you entered appears in the list.



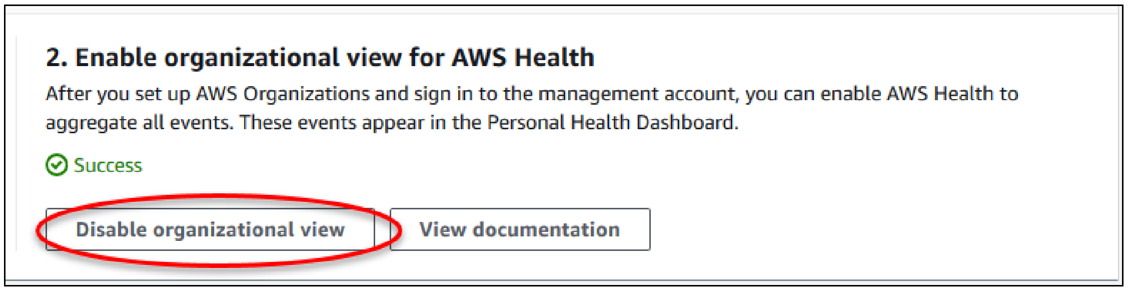
Disabling organizational view (console)

If you don't want to aggregate events for your organization, you can turn off this feature from the management account.

AWS Health stops aggregating events for all other accounts in your organization. You can continue to view previous events from your organization until they're deleted.

To disable organizational view

1. Sign in to the AWS Management Console and open the AWS Personal Health Dashboard at https://phd.aws.amazon.com/phd/home.
2. In the navigation pane, choose **Organizational view**, and then choose **Configurations**.
3. On the **Enable organizational view** page, choose **Disable organizational view**.


2. Enable organizational view for AWS Health
After you set up AWS Organizations and sign in to the management account, you can enable AWS Health to aggregate all events. These events appear in the Personal Health Dashboard.
Success
Disable organizational view
View documentation

After you turn off this feature, AWS Health no longer aggregates events from your organization. However, the service-linked role remains in the management account until you delete it through the AWS

Identity and Access Management (IAM) console, IAM API, or AWS Command Line Interface (AWS CLI). For more information, see Deleting a service-linked role in the *IAM User Guide*.

Organizational view (CLI)

You can also enable the organizational view feature from the AWS Command Line Interface (AWS CLI) instead of the AWS Health console. To use the console, see Enabling organizational view (console) (p. 47).

Note
If you want to allow users access to the management account for the organizational view feature, they must have permissions such as the AWSHealthFullAccess policy. For more information, see AWS Health identity-based policy examples (p. 32).

Topics

Enabling organizational view (CLI)

You can enable organizational view by using the EnableHealthServiceAccessForOrganization API operation.

You can use the AWS Command Line Interface (AWS CLI) or your own code to call this operation.

Note

- You must have a Business or Enterprise support plan to call the AWS Health API.
- You must use the US East (N. Virginia) Region endpoint.

Example

The following AWS CLI command enables this feature from your AWS account. You can use this command from the management account or from an account that can assume the role with the required permissions.

```
aws health enable-health-service-access-for-organization --region us-east-1
```

The following code examples call the EnableHealthServiceAccessForOrganization API operation.

Python

```
import boto3

client = boto3.client('health')

response = client.enable_health_service_access_for_organization()

print(response)
```

Java

You can use the AWS SDK for version Java 2.0 for the following example.

```
import software.amazon.awssdk.services.health.HealthClient;
import software.amazon.awssdk.services.health.HealthClientBuilder;

import software.amazon.awssdk.services.health.model.ConcurrentModificationException;
import
 software.amazon.awssdk.services.health.model.EnableHealthServiceAccessForOrganizationRequest;
import
 software.amazon.awssdk.services.health.model.EnableHealthServiceAccessForOrganizationResponse;
import
 software.amazon.awssdk.services.health.model.DescribeHealthServiceStatusForOrganizationRequest;
import
 software.amazon.awssdk.services.health.model.DescribeHealthServiceStatusForOrganizationResponse;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;

import software.amazon.awssdk.regions.Region;

public class EnableHealthServiceAccessDemo {
    public static void main(String[] args) {
        HealthClient client = HealthClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(
                DefaultCredentialsProvider.builder().build()
            )
            .build();

        try {
            DescribeHealthServiceStatusForOrganizationResponse statusResponse =
 client.describeHealthServiceStatusForOrganization(
                DescribeHealthServiceStatusForOrganizationRequest.builder().build()
            );

            String status = statusResponse.healthServiceAccessStatusForOrganization();
            if ("ENABLED".equals(status)) {
                System.out.println("EnableHealthServiceAccessForOrganization already
 enabled!");
                return;
            }

            client.enableHealthServiceAccessForOrganization(
                EnableHealthServiceAccessForOrganizationRequest.builder().build()
            );

            System.out.println("EnableHealthServiceAccessForOrganization is in
 progress");
        } catch (ConcurrentModificationException cme) {
            System.out.println("EnableHealthServiceAccessForOrganization is already in
 progress. Wait for the action to complete before trying again.");
        } catch (Exception e) {
            System.out.println("EnableHealthServiceAccessForOrganization FAILED: " +
 e);
        }
    }
}
```

For more information, see the AWS SDK for Java 2.0 Developer Guide.

When you enable this feature, the `AWSServiceRoleForHealth_Organizations` service-linked role (SLR) with the `Health_OrganizationsServiceRolePolicy` policy is applied to the management account in the organization.

> **Note**
> Enabling this feature is an asynchronous process and takes time to complete. You can call the DescribeHealthServiceStatusForOrganization operation to check the status of the process.

Viewing organizational view events (CLI)

After you enable this feature, AWS Health starts to record events that affect accounts in the organization. When an account joins your organization, AWS Health automatically adds the account to organizational view.

> **Note**
> AWS Health doesn't record events that occurred in your organization before you enabled organizational view.

When an account leaves your organization, new events from that account are no longer logged to organizational view. However, existing events remain and you can still query them up to the 90-day limit.

AWS retains the policy data for the account for 90 days from the effective date of the administrator account closure. At the end of the 90 day period, AWS permanently deletes all policy data for the account.

- To retain findings for more than 90 days, you can archive the policies. You can also use a custom action with an EventBridge rule to store the findings in an S3 bucket.
- As long as AWS retains the policy data, when you reopen the closed account, AWS reassigns the account as the service administrator and recovers the service policy data for the account.
- For more information, see Closing an account.

> **Important**
> For customers in the AWS GovCloud (US) Regions:
>
> - Before closing your account, back up and then delete your policy data and other account resources. You will no longer have access to them after you close the account.

You can use the AWS Health API operations to return events from organizational view.

Example : Describe organizational view events

The following AWS CLI command returns health events for AWS accounts in your organization.

```
aws health describe-events-for-organization --region us-east-1
```

See the following section for other AWS Health API operations.

Disabling organizational view (CLI)

You can disable organizational view by using the DisableHealthServiceAccessForOrganization API operation.

Example

The following AWS CLI command disables this feature from your account.

```
aws health disable-health-service-access-for-organization --region us-east-1
```

Note
You can also disable the organizational feature by using the Organizations DisableAWSServiceAccess API operation. After you call this operation, AWS Health stops aggregating events for all other accounts in your organization. If you call the AWS Health API operations for organizational view, AWS Health returns an error. AWS Health continues to aggregate health events for your AWS account.

After you disable this feature, AWS Health no longer aggregates events from your organization. However, the service-linked role remains in the management account until you delete it through the AWS Identity and Access Management (IAM) console, IAM API, or AWS CLI. For more information, see Deleting a service-linked Role in the *IAM User Guide*.

AWS Health organizational view API operations

You can use the following AWS Health API operations for organizational view:

- DescribeEventsForOrganization – Returns summary information about events across the organization.
- DescribeAffectedAccountsForOrganization – Returns a list of AWS accounts in the organization that are affected by the specified event.
- DescribeEventDetailsForOrganization – Returns detailed information about the specified events for one or more accounts in the organization.
- DescribeAffectedEntitiesForOrganization – Returns a list of entities that have been affected by one or more events for one or more accounts in an organization.

You can use the following operations to enable or disable AWS Health from working with Organizations:

- EnableHealthServiceAccessForOrganization – Grants AWS Health permission to interact with Organizations and applies the SLR to the management account in the organization.
- DisableHealthServiceAccessForOrganization – Revokes permission for AWS Health to interact with Organizations.
- DescribeHealthServiceStatusForOrganization – Returns status information on whether AWS Health is enabled for your organization.

You must have a Business or Enterprise support plan to call these API operations. If you call the `DescribeEventForOrganization` and `DescribeAffectedAccountsForOrganization` operations from an account that has at least a Business support plan, you can return information about any account in the organization, regardless of the support level of the individual accounts. See the following examples.

Example Example: An organization with accounts that have Business and Developer support plans

- You have three accounts in your organization. The management account has a Business support plan and the other two accounts have a Developer support plan.
- You call the `DescribeEventForOrganization` API operation from the management account or from an account that can assume the role with the required permissions.
- AWS Health returns information for all three accounts.

If you call the `DescribeEventDetailsForOrganization` and `DescribeAffectedEntitiesForOrganization` API operations from an account that has at least a

Business support plan, you can only return information about accounts in the organization that have a Business or Enterprise support level plan.

Example Example: An organization with accounts that have Enterprise, Business, and Developer support plans

- You have five accounts in your organization. The management account has an Enterprise support plan, two accounts have a Business support plan, and two accounts have a Developer support plan.
- You call the `DescribeEventDetailsForOrganization` API operation from the management account.
- AWS Health returns information for only the accounts that have an Enterprise or Business support plan. The accounts that have a Developer support plan appear in the `failedSet` of the response.

Monitoring AWS Health events with Amazon CloudWatch Events

You can use Amazon CloudWatch Events to detect and react to changes for AWS Health events. Then, based on the rules that you create, CloudWatch Events invokes one or more target actions when an event matches the values that you specify in a rule. Depending on the type of event, you can send notifications, capture event information, take corrective action, initiate events, or take other actions. For example, you can use AWS Health to receive email notifications if you have AWS resources in your AWS account that are scheduled for updates, such as Amazon Elastic Compute Cloud (Amazon EC2) instances.

Notes

- This topic uses the CloudWatch Events console to create a rule. You can also use the Amazon EventBridge console to create a rule. For more information, see Creating a rule for an AWS service in the *Amazon EventBridge User Guide*.
- For both services, AWS Health delivers events on a best effort basis. Events are not always guaranteed to be delivered to CloudWatch Events or EventBridge.
- You can create a CloudWatch Events rule to receive notifications for only your account. You can't receive organizational events for other accounts in your AWS Organizations. For more information, see Aggregating AWS Health events across accounts with organizational view (p. 46).

You can choose the following types of targets when using CloudWatch Events as a part of your AWS Health workflow:

- AWS Lambda functions
- Amazon Kinesis Data Streams
- Amazon Simple Queue Service (Amazon SQS) queues
- Built-in targets (CloudWatch alarm actions)
- Amazon Simple Notification Service (Amazon SNS) topics

For example, you can use a Lambda function to pass a notification to a Slack channel when an AWS Health event occurs. Or, you can use Lambda and CloudWatch Events to send custom text or SMS notifications with Amazon SNS when an AWS Health event occurs.

For samples of automation and customized alerts that you can create in response to AWS Health events, see the AWS Health Tools in GitHub.

Topics

About AWS Regions for AWS Health

You must create a CloudWatch Events rule for each Region that you want to receive AWS Health events. If you don't create a rule, you won't receive events. For example, to receive events from the US West (Oregon) Region, you must create a rule for this Region.

Some AWS Health events are not Region-specific and instead are global, such as events sent for AWS Identity and Access Management (IAM). To receive global events, you must create a rule for the US East (N. Virginia) Region.

To receive global events in the AWS GovCloud (US), you must create a rule in the AWS GovCloud (US-West) Region.

About public events for AWS Health

Only AWS Health events that are specific to your AWS account are delivered to CloudWatch Events. For example, this can include events such as a required update to an Amazon EC2 instance and other scheduled change events that might affect your account and resources.

Currently, you can't use CloudWatch Events to return *public* events from the Service Health Dashboard. Events from the Service Health Dashboard provide public information about the Regional availability of a service. These events aren't specific to AWS accounts, so they aren't delivered to CloudWatch Events.

Instead, you can use the AWS Health console and the DescribeEventDetails operation. You can use either option to return information about an event, and then identify if it's a public event from the Service Health Dashboard or an account-specific event that affects your account.

You can use the following options to identify if an event is public or account-specific:

- In the AWS Personal Health Dashboard, choose the **Affected resources** tab on the **Event log** page. Events with resources are specific to your account. Events without resources are public and are not specific to your account. For more information, see Getting started with the AWS Personal Health Dashboard (p. 7).
- Use the AWS Health API to return the `eventScopeCode` parameter. Events can have the `PUBLIC`, `ACCOUNT_SPECIFIC`, or `NONE` value. For more information, see the DescribeEventDetails operation in the *AWS Health API Reference*.

You can also use the AWS Health Service Health Dashboard notifier tool to get notified for public events. For more information, see the aws-health-tools on the GitHub website.

Creating a CloudWatch Events rule for AWS Health

You can create a CloudWatch Events rule to get notified for AWS Health events in your account. Before you create event rules for AWS Health, you should do the following:

- Familiarize yourself with events, rules, and targets in CloudWatch Events. For more information, see What Is Amazon CloudWatch Events? in the *Amazon CloudWatch Events User Guide* and New CloudWatch Events – Track and Respond to Changes to Your AWS Resources.
- Create the target or targets to use in your event rules.

To create a CloudWatch Events rule for AWS Health

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.

2. To change the AWS Region, use the **Region selector** in the upper-right corner of the page. Choose a Region in which you want to track AWS Health events.
3. In the navigation pane, under **Events**, choose **Rules**.
4. Choose **Create rule**, and then under **Event Source**, for **Service Name**, choose **Health**.
5. For **Event Type**, choose one of the following options.
   - Choose **All Events** to create a rule that applies to all AWS services. This rule monitors all events from AWS Health. If you choose this option, you can't specify event type categories or event type codes.
   - Choose **Specific Health events**, **Specific service(s)**, and then choose a service name from the list. This example creates a rule that monitors events for **EC2** so that CloudWatch Events monitors only Amazon EC2 events.
6. If you chose a specific service, choose one of the following options.
   - Choose **Any event type category** to create a rule that applies to all event type categories.
   - Choose **Specific event type category(s)** and then choose a value from the list. This creates a rule that applies only to one event type category, such as **scheduledChange**.

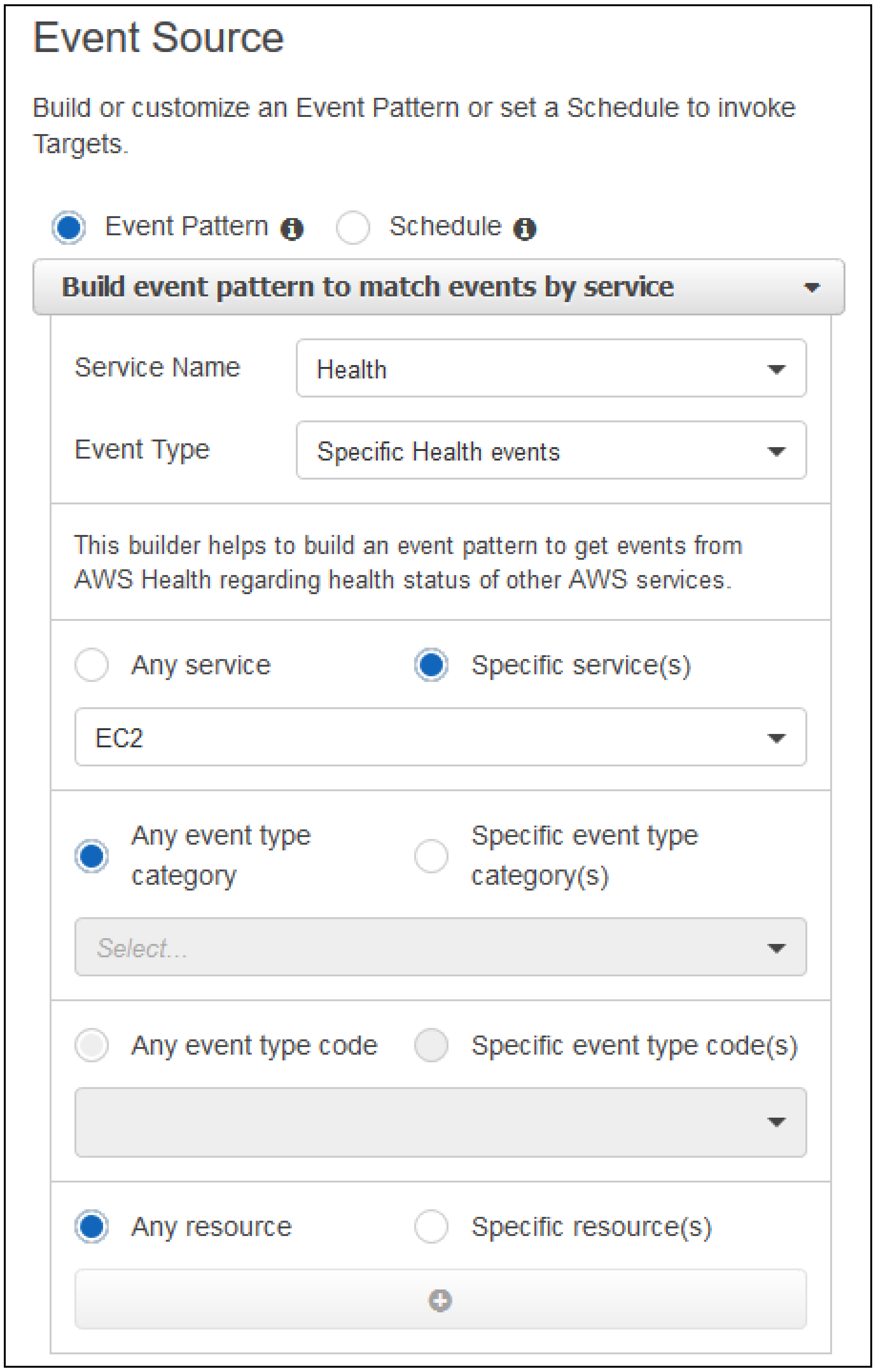
     Tip

     - To monitor all AWS Health events for a specific service, we recommend that you choose **Any event type category** and **Any resource**. This ensures that your rule monitors for any AWS Health events, including any new event type codes, for your specified service. For an example rule, see all Amazon EC2 events (p. 60).
     - You can create a rule to monitor for more than one service or event type category. To do so, you must manually update the event pattern for the rule. For more information, see Creating a rule for multiple services and categories (p. 63).
7. If you chose a specific service and event type category, choose one of the following options for event type codes.
   - Choose **Any event type code** to create a rule that applies to all event type codes.
   - Choose **Specific event type code(s)** and then choose one or more values from the list. This creates a rule that applies only to specific event type codes. For example, if you choose **`AWS_EC2_INSTANCE_STOP_SCHEDULED`** and **`AWS_EC2_INSTANCE_RETIREMENT_SCHEDULED`**, your rule applies only to these events when they occur in your account.
8. Choose one of the following options for affected resources.
   - Choose **Any resource** to create a rule that applies to all resources.
   - Choose **Specific resource(s)** and enter the IDs of one or more resources. For example, you might specify an Amazon EC2 instance ID such as `i-EXAMPLEa1b2c3de4` to monitor for events that affect only this resource.
9.

   Review your rule setup so that it meets your event-monitoring requirements.
10. In the **Targets** area, choose **Add target***.
11. In the **Select target type** list, choose the type of target that you prepared to use with this rule, and then configure any additional options required by that type.
12. Choose **Configure details**.
13. On the **Configure rule details** page, enter a name and description for the rule, and then select the **State** check box to enable the rule as soon as it's created.
14. Choose **Create rule**.

Example : Rule for all Amazon EC2 events

The following example creates a rule so that CloudWatch Events monitors for all Amazon EC2 events, including the event type categories, event codes, and resources.



Event Source

Build or customize an Event Pattern or set a Schedule to invoke Targets.

[✓ Event Pattern] Schedule

Build event pattern to match events by service

Service Name: Health

Event Type: Specific Health events

This builder helps to build an event pattern to get events from AWS Health regarding health status of other AWS services.

Any service [✓ Specific service(s)]

EC2

[✓ Any event type category] Specific event type category(s)

Select...

Any event type code Specific event type code(s)

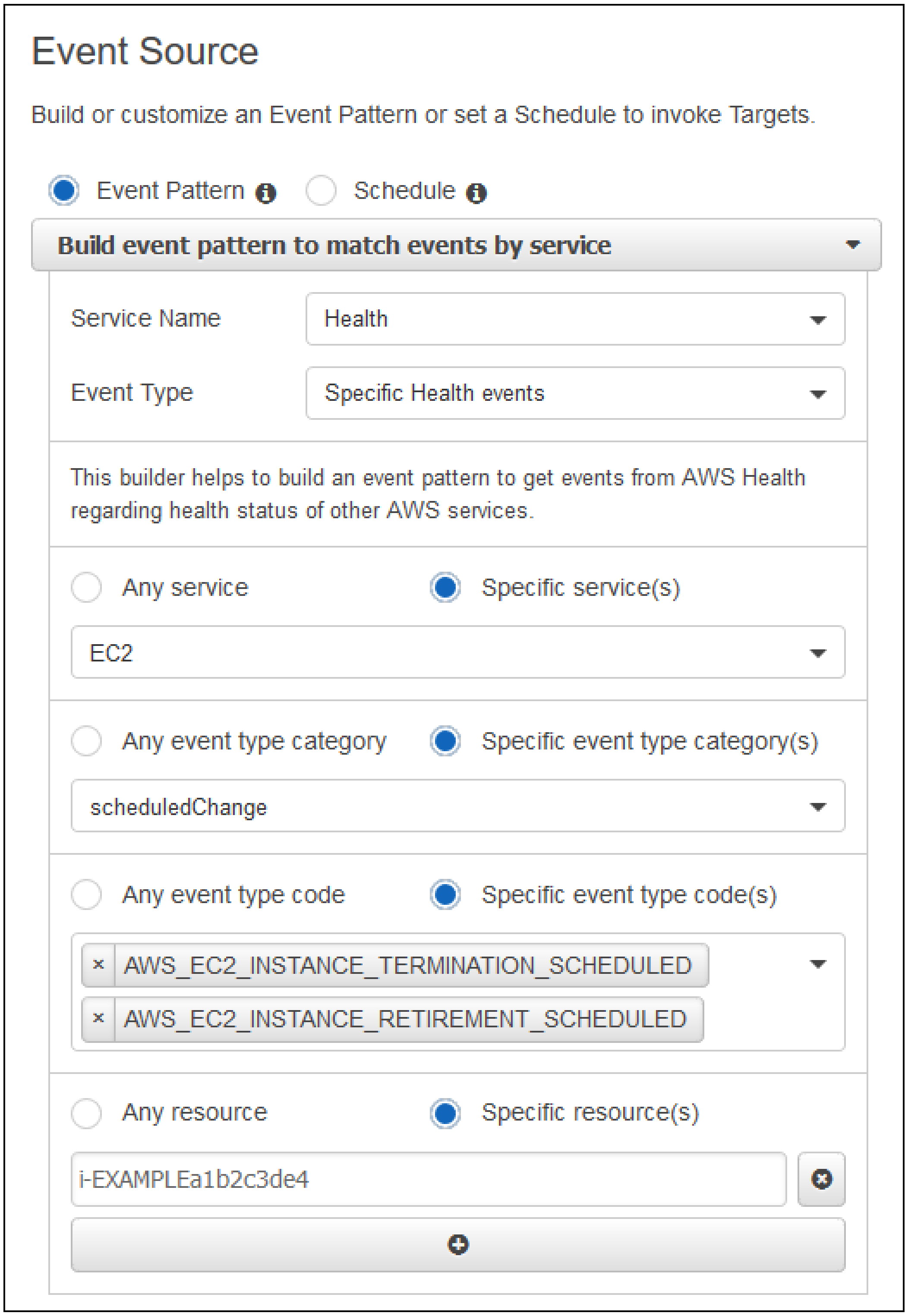
[✓ Any resource] Specific resource(s)

Example : Rule for specific Amazon EC2 events

The following example creates a rule so that CloudWatch Events monitors the following:

- The Amazon EC2 service
- The **scheduledChange** event type category

- The event type codes for `AWS_EC2_INSTANCE_TERMINATION_SCHEDULED` and `AWS_EC2_INSTANCE_RETIREMENT_SCHEDULED`
- The instance with the `i-EXAMPLEa1b2c3de4` ID



Event Source

Build or customize an Event Pattern or set a Schedule to invoke Targets.

[✓ Event Pattern] Schedule

Build event pattern to match events by service

Service Name: Health

Event Type: Specific Health events

This builder helps to build an event pattern to get events from AWS Health regarding health status of other AWS services.

Any service [✓ Specific service(s)]

EC2

Any event type category [✓ Specific event type category(s)]

scheduledChange

Any event type code [✓ Specific event type code(s)]

AWS_EC2_INSTANCE_TERMINATION_SCHEDULED

AWS_EC2_INSTANCE_RETIREMENT_SCHEDULED

Any resource [✓ Specific resource(s)]

i-EXAMPLEa1b2c3de4

Creating a rule for multiple services and categories

The examples in the previous procedure show you how to create a rule for a single service and event type category. You can also create a rule for multiple services and event type categories. This means that you don't have to create a separate rule for each service and category that you want to monitor. To do so, you must edit the event pattern and then enter your changes manually.

You can use any of the following options.

To add services and categories for an existing rule

1. In the CloudWatch Events console, on the **Rules** page, choose the rule name.
2. In the upper-right corner, choose **Actions**, and then choose **Edit**.
3. For **Event Pattern**, enter your changes in the text field.
4. Choose **Configure details**, and then choose **Update rule** to save your changes.

To add services and categories for a new rule

1. Follow the procedure in Creating a CloudWatch Events rule for AWS Health (p. 59) to step 6 (p. 60).
2. Instead of choosing a single service or category from the lists, for **Event Pattern Preview**, choose **Edit**.
3. Enter your changes in the text field and choose **Save**. See the following example pattern (p. 63).
4. Review your event pattern and then follow the rest of the procedure in Creating a CloudWatch Events rule for AWS Health (p. 59) to create your rule.

Use the API or AWS Command Line Interface (AWS CLI)

For a new or existing rule, use the PutRule API operation or the `aws events put-rule` command to update the event pattern. For an example AWS CLI command, see put-rule in the *AWS CLI Command Reference*.

Example Example: Multiple services and event type categories

The following event pattern creates a rule to monitor events for the `issue`, `accountNotification`, and `scheduledChange` event type categories and for three AWS services.

```
{
  "detail": {
    "eventTypeCategory": [
      "issue",
      "accountNotification",
      "scheduledChange"
    ],
    "service": [
      "AUTOSCALING",
      "VPC",
      "EC2"
    ]
  },
  "detail-type": [
    "AWS Health Event"
  ],
  "source": [
    "aws.health"
  ]
}
```

Receiving AWS Health events with AWS Chatbot

You can receive AWS Health events directly in your chat clients, such as Slack and Amazon Chime. Use this event to identify recent AWS service issues that might affect your AWS applications and infrastructure. Then, you can sign in to your AWS Personal Health Dashboard to learn more about the update. For example, if you're monitoring for the `AWS_EC2_INSTANCE_STOP_SCHEDULED` event type in your AWS account, the AWS Health event can appear directly to your Slack channel.

Prerequisites

Before you get started, you must have the following:

- A chat client configured with AWS Chatbot. You can configure Amazon Chime and Slack. For more information, see Getting started with AWS Chatbot in the *AWS Chatbot Administrator Guide*.
- An Amazon SNS topic, that you created and subscribed to. If you already have an SNS topic, you can use an existing one. For more information, see Getting started with Amazon SNS in the *Amazon Simple Notification Service Developer Guide*.

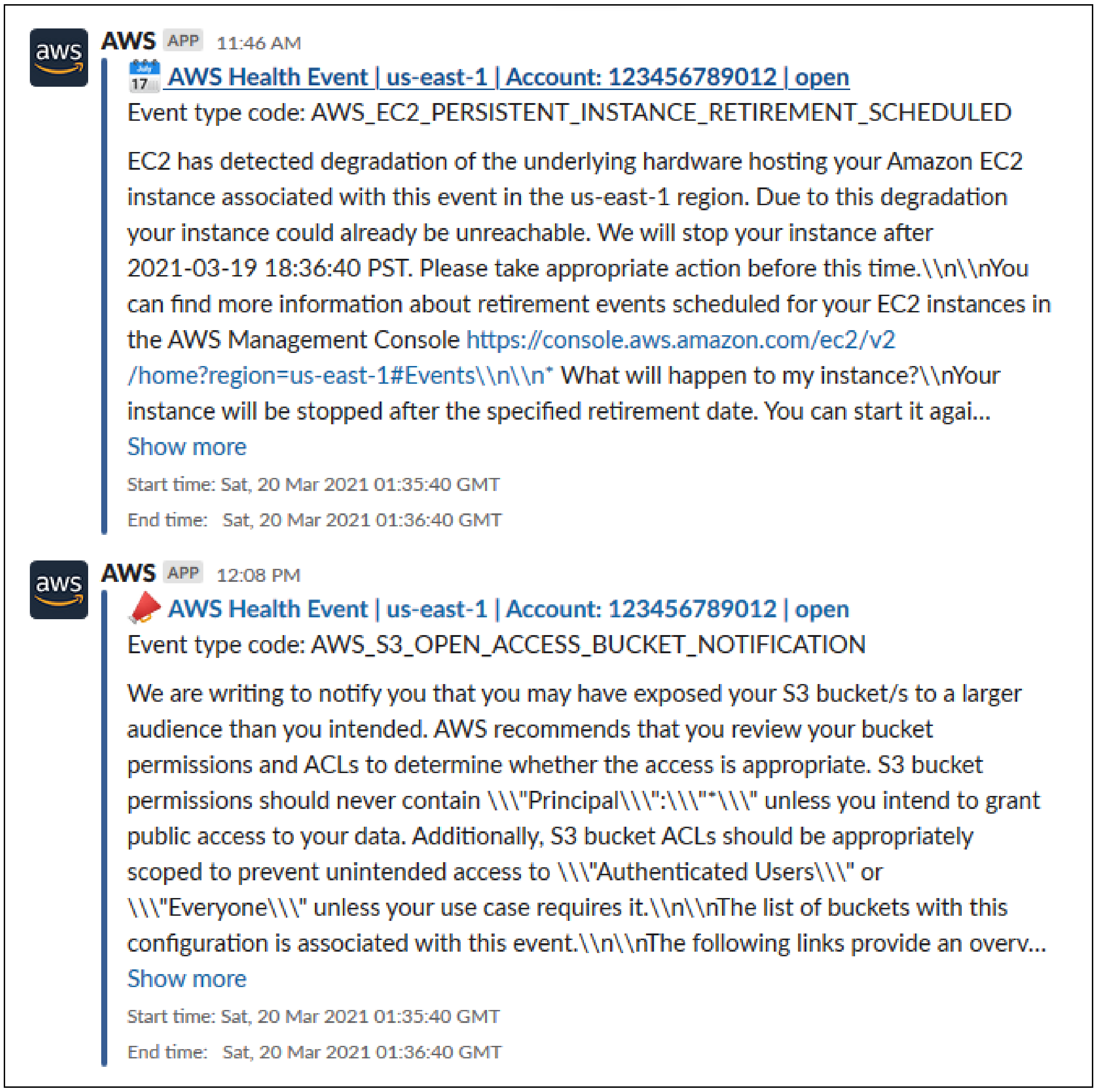
To receive AWS Health events with AWS Chatbot

1. Follow the procedure in Creating a CloudWatch Events rule for AWS Health (p. 59).
   a. When you choose the target in step 11 (p. 60), choose an SNS topic. You will use this same SNS topic in the AWS Chatbot console.
   b. Complete the rest of the procedure to create the rule.
2. Navigate to the AWS Chatbot console.
3. Choose your chat client, such as your Slack channel name, and then choose **Edit**.
4. In the **Notifications - optional** section, for **Topics**, choose the same SNS topic that you specified.
5. Choose **Save**.

   When AWS Health sends an event to CloudWatch Events that matches your rule, the AWS Health event will appear in your chat client.
6. Choose the event name to see more information in your AWS Personal Health Dashboard.

Example : AWS Health events sent to Slack

The following is an example of two AWS Health events for Amazon EC2 and Amazon Simple Storage Service (Amazon S3) in the US East (N. Virginia) Region that appear in the Slack channel.



AWS APP 11:46 AM

AWS Health Event | us-east-1 | Account: 123456789012 | open

Event type code: AWS_EC2_PERSISTENT_INSTANCE_RETIREMENT_SCHEDULED

EC2 has detected degradation of the underlying hardware hosting your Amazon EC2 instance associated with this event in the us-east-1 region. Due to this degradation your instance could already be unreachable. We will stop your instance after 2021-03-19 18:36:40 PST. Please take appropriate action before this time.\\n\\nYou can find more information about retirement events scheduled for your EC2 instances in the AWS Management Console https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Events\\n\\n* What will happen to my instance?\\nYour instance will be stopped after the specified retirement date. You can start it agai...

Show more

Start time: Sat, 20 Mar 2021 01:35:40 GMT

End time: Sat, 20 Mar 2021 01:36:40 GMT

AWS APP 12:08 PM

AWS Health Event | us-east-1 | Account: 123456789012 | open

Event type code: AWS_S3_OPEN_ACCESS_BUCKET_NOTIFICATION

We are writing to notify you that you may have exposed your S3 bucket/s to a larger audience than you intended. AWS recommends that you review your bucket permissions and ACLs to determine whether the access is appropriate. S3 bucket permissions should never contain \\\"Principal\\\":\\\"*\\\" unless you intend to grant public access to your data. Additionally, S3 bucket ACLs should be appropriately scoped to prevent unintended access to \\\"Authenticated Users\\\" or \\\"Everyone\\\" unless your use case requires it.\\n\\nThe list of buckets with this configuration is associated with this event.\\n\\nThe following links provide an overv...

Show more

Start time: Sat, 20 Mar 2021 01:35:40 GMT

End time: Sat, 20 Mar 2021 01:36:40 GMT

Automating actions for Amazon EC2 instances

You can automate actions to respond to scheduled events for your Amazon EC2 instances. When AWS Health sends an event to your AWS account, your CloudWatch Events rule can then invoke targets, such as AWS Systems Manager Automation documents, to automate actions on your behalf.

For example, when an Amazon EC2 instance retirement event is scheduled for an Amazon Elastic Block Store (Amazon EBS)-backed EC2 instance, AWS Health will send the `AWS_EC2_PERSISTENT_INSTANCE_RETIREMENT_SCHEDULED` event type to your AWS Personal Health Dashboard. When your rule detects this event type, you can automate the stop and start of the instance. This way, you don't have to perform these actions manually.

Notes

- This procedure uses the CloudWatch Events console to create a rule. You can also use the EventBridge console to create a rule. For more information, see Creating a rule for an AWS service in the *Amazon EventBridge User Guide*.
- To automate an action for your Amazon EC2 instances, they must be managed by Systems Manager.

For more information, see Automating Amazon EC2 with EventBridge in the *Amazon EC2 User Guide for Linux Instances*.

Prerequisites

You must create an AWS Identity and Access Management (IAM) policy, create an IAM role, and update the trust policy before you can create a rule.

Create an IAM policy

Follow this procedure to create a customer managed policy for your role. This policy gives the role permission to perform actions on your behalf. This procedure uses the JSON policy editor in the IAM console.

To create an IAM policy

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Policies**.
3. Choose **Create policy**.
4. Choose the **JSON** tab.
5. Copy the following JSON and then replace the default JSON in the editor.

   a. In the `Resource` parameter, for the Amazon Resource Name (ARN), enter your AWS account ID.

   b. You can also replace the role name or use the default. This example uses *`AutomationCWRole`*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:StartInstances",
        "ec2:StopInstances",
        "ec2:DescribeInstanceStatus"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "ssm:*"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/AutomationCWRole"
    }
  ]
}
```

6. Choose **Next: Tags**.
7. (Optional) You can use tags as key–value pairs to add metadata to the policy.
8. Choose **Next: Review**.
9. On the **Review policy** page, enter a **Name** such as *`AutomationCWRolePolicy`* and a **Description** (optional).
10. Review the **Summary** page to see the permissions that the policy allows, and then choose **Create policy**.

This policy defines the actions that the role can take. For more information, see Creating IAM policies (console) in the *IAM User Guide*.

Create an IAM role

After you create the policy, you must create an IAM role, and then attach the policy to that role.

To create a role for an AWS service

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Roles**, and then choose **Create role**.
3. For **Select type of trusted entity**, choose **AWS service**.
4. Choose **EC2** for the service that you want to allow to assume this role.
5. Choose **Next: Permissions**.
6. Enter the policy name that you created, such as *`AutomationCWRolePolicy`*, and then select the check box next to the policy.
7. Choose **Next: Tags**.
8. (Optional) You can use tags as key–value pairs to add metadata to the role.
9. Choose **Next: Review**.
10. For **Role name**, enter *`AutomationCWRole`*. This name must be the same name that appears in the ARN of the IAM policy that you created earlier.
11. (Optional) For **Role description**, enter a description for the role.
12. Review the role and then choose **Create role**.

For more information, see Creating a role for an AWS service in the *IAM User Guide*.

Update the trust policy

Follow this procedure to update the trust policy for the role that you created. You must complete this procedure so that you can choose this role in the CloudWatch Events console.

To update the trust policy for the role

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.

2. In the navigation pane, choose **Roles**.
3. In the list of roles in your AWS account, choose the name of the role that you created, such as `AutomationCWRole`.
4. Choose the **Trust relationships** tab, and then choose **Edit trust relationship**.
5. For **Policy Document**, copy the following JSON, and then replace the default policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "events.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

6. Choose **Update Trust Policy**.

For more information, see Modifying a role trust policy (console) in the *IAM User Guide*.

Create a rule for CloudWatch Events

Follow this procedure to create a rule in the CloudWatch Events console so that you can automate the stop and start of EC2 instances that are scheduled for retirement.

To create a rule for CloudWatch Events for Systems Manager automated actions

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, under **Events**, choose **Rules**.
3. Choose **Create rule**, and then under **Event Source**, for **Service Name**, choose **Health**.
4. For **Event Type**, choose **Specific Health events**.
5. Choose **Specific service(s)** and then choose **EC2**.
6. Choose **Specific event type category(s)** and then choose **scheduledChange**.
7. Choose **Specific event types code(s)** and then choose the event type code. For example, for Amazon EC2 EBS-backed instances, choose **`AWS_EC2_PERSISTENT_INSTANCE_RETIREMENT_SCHEDULED`**. For Amazon EC2 instance store-backed instances, choose **`AWS_EC2_INSTANCE_RETIREMENT_SCHEDULED`**.
8. Choose **Any resource**.

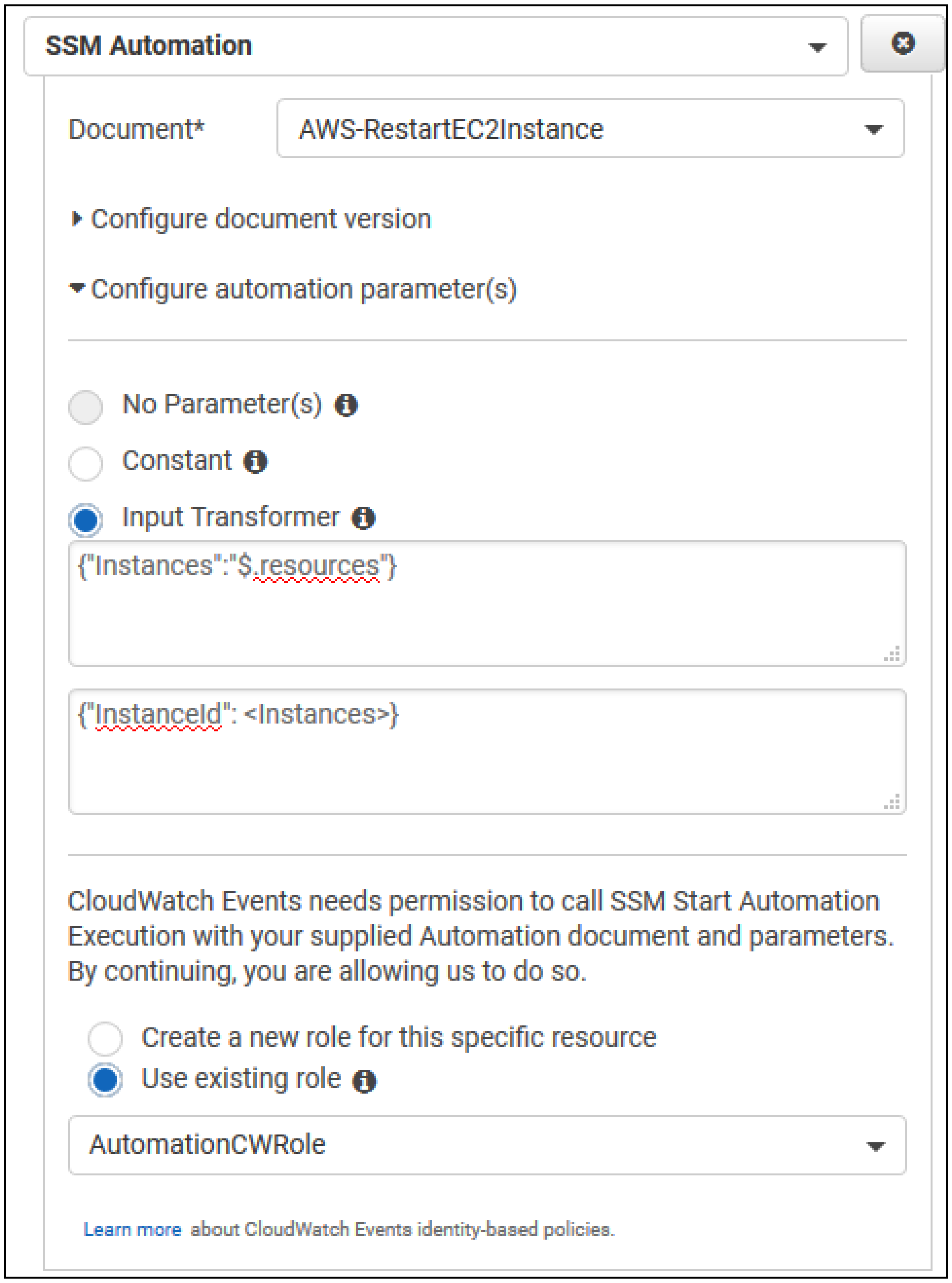
   Your **Event Pattern Preview** might look like the following example.

   Example

```
{
  "source": [
    "aws.health"
  ],
  "detail-type": [
    "AWS Health Event"
  ],
  "detail": {
    "service": [
      "EC2"
    ],
    "eventTypeCategory": [
      "scheduledChange"
    ],
    "eventTypeCode": [
      "AWS_EC2_PERSISTENT_INSTANCE_RETIREMENT_SCHEDULED"
    ]
  }
}
```

9. Add the Systems Manager Automation document target. Under **Targets**, choose **Add target***, and then choose **SSM Automation**.
10. For **Document***, choose **`AWS-RestartEC2Instance`**.
11. Expand the **Configure automation parameters(s)** and then choose **Input Transformer**.
12. For the **Input Path** field, enter `{"Instances":"$.resources"}`.
13. For the second field, enter `{"InstanceId": <Instances>}`.
14. Choose **Use existing role** and then choose your IAM role that you created, such as *`AutomationCWRole`*.

    Your target should look like the following example.



SSM Automation

Document* AWS-RestartEC2Instance

Configure document version

Configure automation parameter(s)

No Parameter(s)

Constant

[✓ Input Transformer]

```
{"Instances":"$.resources"}
```

```
{"InstanceId": <Instances>}
```

CloudWatch Events needs permission to call SSM Start Automation Execution with your supplied Automation document and parameters. By continuing, you are allowing us to do so.

Create a new role for this specific resource

[✓ Use existing role]

AutomationCWRole

Learn more about CloudWatch Events identity-based policies.

Note
If you don't have an existing IAM role with the required EC2 and Systems Manager permissions and trusted relationship, your role won't appear in the list. For more information, see Prerequisites (p. 66).

15. Choose **Configure details**.
16. Enter a name and description for your rule. Keep the **State** option selected.
17. Choose **Create rule**.

Monitoring AWS Health

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Health and your other AWS solutions. AWS provides the following monitoring tools to watch AWS Health, report when something is wrong, and take actions when appropriate:

- Amazon CloudWatch monitors your AWS resources and the applications you run on AWS in real time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify. For more information, see the Amazon CloudWatch User Guide.

  You can use Amazon CloudWatch Events so that you're notified about AWS Health events that might affect your services and resources. For example, if AWS Health publishes an event about your Amazon EC2 instances, you can use these notifications to take action and update or replace your resources as needed. For more information, see Monitoring AWS Health events with Amazon CloudWatch Events (p. 58).
- AWS CloudTrail captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.

Topics

Logging AWS Health API calls with AWS CloudTrail

AWS Health is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Health. CloudTrail captures API calls for AWS Health as events. The calls captured include calls from the AWS Health console and code calls to the AWS Health API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Health. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to AWS Health, the IP address that the request was made from, who made the request, when it was made, and additional details.

To learn more about CloudTrail, including how to configure and enable it, see the AWS CloudTrail User Guide.

AWS Health information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When supported event activity occurs in AWS Health, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for AWS Health, create a *trail*. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

All AWS Health API operations are logged by CloudTrail and are documented in the AWS Health API Reference. For example, calls to the `DescribeEvents`, `DescribeEventDetails`, and `DescribeAffectedEntities` operations generate entries in the CloudTrail log files.

AWS Health supports logging the following actions as events in CloudTrail log files:

- Whether the request was made with root or IAM credentials
- Whether the request was made with temporary security credentials for a role or federated user
- Whether the request was made by another AWS service

For more information, see the CloudTrail userIdentity Element.

You can store your log files in your Amazon S3 bucket for as long as you want. You can also define Amazon S3 lifecycle rules to archive or delete log files automatically. By default, your log files are encrypted with Amazon S3 server-side encryption (SSE).

To be notified upon log file delivery, you can configure CloudTrail to publish Amazon SNS notifications when new log files are delivered. For more information, see Configuring Amazon SNS Notifications for CloudTrail.

You can also aggregate AWS Health log files from multiple AWS regions and multiple AWS accounts into a single Amazon S3 bucket.

For more information, see Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts.

Example: AWS Health log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the DescribeEntityAggregates operation.

```
{
   "Records": [
   {
   "eventVersion": "1.05",
   "userIdentity": {
      "type": "IAMUser",
      "principalId": "AIDACKCEVSQ6C2EXAMPLE",
      "arn": "arn:aws:iam::123456789012:user/JaneDoe",
      "accountId": "123456789012",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "userName": "JaneDoe",
      "sessionContext": {"attributes": {
         "mfaAuthenticated": "false",
         "creationDate": "2016-11-21T07:06:15Z"
      }},
      "invokedBy": "AWS Internal"
     },
   "eventTime": "2016-11-21T07:06:28Z",
   "eventSource": "health.amazonaws.com",
   "eventName": "DescribeEntityAggregates",
   "awsRegion": "us-east-1",
   "sourceIPAddress": "203.0.113.0",
   "userAgent": "AWS Internal",
   "requestParameters": {"eventArns": ["arn:aws:health:us-east-1::event/EBS/
EBS_LOST_VOLUME/EBS_LOST_VOLUME_123"]},
   "responseElements": null,
   "requestID": "05b299bc-afb9-11e6-8ef4-c34387f40bd4",
   "eventID": "e4deb9dc-dbc2-4bdb-8515-73e8abcbc29b",
   "eventType": "AwsApiCall",
   "recipientAccountId": "123456789012"
   }
   ],
   ...
}
```

Document history for AWS Health

The following table describes the documentation for this release of AWS Health.

- **API version:** 2016-08-04
- **Latest documentation update:** July 29, 2021

The following table describes important updates to the AWS Health documentation, beginning in August 28, 2020. You can subscribe to the RSS feed to receive notifications about the updates.

update-history-change	update-history-description	update-history-date
Added documentation (p. 74)	New topic for AWS Health concepts. For more information, see Concepts for AWS Health.	July 29, 2021
Updated documentation for CloudWatch Events (p. 74)	Added a section about how to create a rule for multiple services and event type categories. For more information, see Creating a rule for multiple services and categories.	May 7, 2021
Updated documentation for CloudWatch Events (p. 74)	Updated the section to automate AWS Systems Manager actions for Amazon CloudWatch Events rules. For more information, see Automating actions for Amazon EC2 instances.	April 28, 2021
Updated documentation for CloudWatch Events (p. 74)	Added a section to receive AWS Health events in your chat client. For more information, see Receiving AWS Health events with AWS Chatbot.	March 16, 2021
Updated documentation (p. 74)	The following topics are updated: • Updated the Aggregating AWS Health events topic • Reorganized and updated the Monitor for AWS Health events with Amazon CloudWatch Events topic • Updated the Resource- and action-based conditions section	January 29, 2021

Added the AWS Personal Health Dashboard for organizational view in the AWS Health console	You can use the AWS Health console to enable the organizational view feature. You can then view health events for member accounts in your AWS organization.	December 14, 2020
High availability endpoint demo	You can use the example code to determine the active regional endpoint and signing AWS Region for AWS Health.	October 22, 2020
Updates to the *AWS Health User Guide* (p. 74)	Organization updates and added an RSS feed so that you can subscribe for the latest updates to the AWS Health documentation.	August 28, 2020

Earlier updates

Change	Description	Date
Updated the organizational view topic to include examples.	See Aggregating AWS Health events across accounts with organizational view (p. 46).	June 3, 2020
Security and AWS Health	Added information about security considerations when using AWS Health. See Security in AWS Health (p. 22).	May 5, 2020
Added new section to explain how to use organizational view to events aggregated across all accounts in AWS Organizations.	See Aggregating AWS Health events across accounts with organizational view (p. 46).	December 18, 2019
Added new section "Resource- and Action-based Conditions" to explain Events restrictions vended by the AWS Health API.	See Identity and access management for AWS Health (p. 23).	August 2, 2018
Added a note about the visibility of AWS Health information.	See Identity and access management for AWS Health (p. 23).	August 16, 2017
Service release.	AWS Health released.	December 1, 2016

AWS glossary

For the latest AWS terminology, see the AWS glossary in the *AWS General Reference*.