AWS Application Discovery Service

API Reference

API Version 2015-11-01

AWS Application Discovery Service: API Reference

Table of Contents

Welcome

AWS Application Discovery Service helps you plan application migration projects by automatically identifying servers, virtual machines (VMs), software, and software dependencies running in your on-premises data centers. Application Discovery Service also collects application performance data, which can help you assess the outcome of your migration. The data collected by Application Discovery Service is securely retained in an Amazon-hosted and managed database in the cloud. You can export the data as a CSV or XML file into your preferred visualization tool or cloud-migration solution to plan your migration. For more information, see the Application Discovery Service FAQ.

Application Discovery Service offers two modes of operation.

- **Agentless discovery** mode is recommended for environments that use VMware vCenter Server. This mode doesn't require you to install an agent on each host. Agentless discovery gathers server information regardless of the operating systems, which minimizes the time required for initial on-premises infrastructure assessment. Agentless discovery doesn't collect information about software and software dependencies. It also doesn't work in non-VMware environments. We recommend that you use agent-based discovery for non-VMware environments and if you want to collect information about software and software dependencies. You can also run agent-based and agentless discovery simultaneously. Use agentless discovery to quickly complete the initial infrastructure assessment and then install agents on select hosts to gather information about software and software dependencies.
- **Agent-based discovery** mode collects a richer set of data than agentless discovery by using Amazon software, the AWS Application Discovery Agent, which you install on one or more hosts in your data center. The agent captures infrastructure and application information, including an inventory of installed software applications, system and process performance, resource utilization, and network dependencies between workloads. The information collected by agents is secured at rest and in transit to the Application Discovery Service database in the cloud.

Application Discovery Service integrates with application discovery solutions from AWS Partner Network (APN) partners. Third-party application discovery tools can query the Application Discovery Service and write to the Application Discovery Service database using a public API. You can then import the data into either a visualization tool or cloud-migration solution.

> **Important**
> Application Discovery Service doesn't gather sensitive information. All data is handled according to the AWS Privacy Policy. You can operate Application Discovery Service using offline mode to inspect collected data before it is shared with the service.

Your AWS account must be granted access to Application Discovery Service, a process called *whitelisting*. This is true for AWS partners and customers alike. To request access, sign up for the AWS Application Discovery Service here. We will send you information about how to get started.

This API reference provides descriptions, syntax, and usage examples for each of the actions and data types for the Application Discovery Service. The topic for each action shows the API request parameters and the response. Alternatively, you can use one of the AWS SDKs to access an API that is tailored to the programming language or platform that you're using. For more information, see AWS SDKs.

This guide is intended for use with the *AWS Application Discovery Service User Guide* .

The following are short descriptions of each API action, organized by function.

Managing AWS Application Discovery Agents or the AWS Application Discovery Connector

The Application Discovery Service API includes the following actions to manage agents or the Connector. To learn more about these features of Application Discovery Service, see AWS Application Discovery Service Components:

- *StartDataCollectionByAgentIds*: Instructs the specified agents or the Connector to start collecting data. Application Discovery Service takes several minutes to receive and process data after you initiate data collection.
- *StopDataCollectionByAgentIds*: Instructs the specified agents or the Connector to stop collecting data.
- *DescribeAgents*: Lists agents by ID or lists all agents associated with your user account if you did not specify an agent ID. The output includes agent IDs, IP addresses, media access control (MAC) addresses, agent health, host name where the agent resides, and the version number of each agent.

Querying Configuration Items

A *configuration item* is an IT asset that was discovered in your data center by an agent or the Connector. When you use Application Discovery Service, you can specify filters and query specific configuration items. The service supports Server, Process, and Connection configuration items. This means you can specify a value for the following keys and query your IT assets:

Server

- server.hostName
- server.osName
- server.osVersion
- server.configurationId
- server.agentId

Process

- process.name
- process.configurationId
- server.hostName
- server.osName
- server.osVersion
- server.configurationId
- server.agentId

Connection

- connection.sourceIp
- connection.sourcePort
- connection.destinationIp
- connection.destinationPort

- sourceProcess.configurationId
- sourceProcess.name
- destinationProcess.configurationId
- destinationProcess.name
- sourceServer.configurationId
- sourceServer.hostName
- sourceServer.osName
- sourceServer.osVersion
- destinationServer.configurationId
- destinationServer.hostName
- destinationServer.osName
- destinationServer.osVersion

Application Discovery Service includes the following actions for querying configuration items.

- *DescribeConfigurations*: Retrieves a list of attributes for a specific configuration ID. For example, the output for a *server* configuration item includes a list of attributes about the server, including host name, operating system, number of network cards, etc.
- *ListConfigurations*: Retrieves a list of configuration items according to the criteria you specify in a filter. The filter criteria identify relationship requirements. For example, you can specify filter criteria of process.name with values of *nginx* and *apache*.

Tagging Discovered Configuration Items

You can tag discovered configuration items. Tags are metadata that help you categorize IT assets in your data center. Tags use a *key-value* format. For example, `{"key": "serverType", "value": "webServer"}`.

- *CreateTags*: Creates one or more tags for a configuration items.
- *DescribeTags*: Retrieves a list of configuration items that are tagged with a specific tag. *Or*, retrieves a list of all tags assigned to a specific configuration item.
- *DeleteTags*: Deletes the association between a configuration item and one or more tags.

Exporting Data

You can export data as a CSV file to an Amazon S3 bucket or into your preferred visualization tool or cloud migration solution to help reduce the complexity and time in planning your cloud migration.

- *ExportConfigurations*: Exports all discovered configuration data to an Amazon S3 bucket. Data includes tags and tag associations, processes, connections, servers, and system performance. This API returns an export ID which you can query using the GetExportStatus API.
- *DescribeExportConfigurations*: Gets the status of the data export. When the export is complete, the service returns an Amazon S3 URL where you can download CSV files that include the data.

This document was last published on December 9, 2016.

Actions

The following actions are supported:

CreateTags

Creates one or more tags for configuration items. Tags are metadata that help you categorize IT assets. This API accepts a list of multiple configuration items.

Request Syntax

```
{
   "configurationIds": [ "string" ],
   "tags": [
      {
         "key": "string",
         "value": "string"
      }
   ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

configurationIds (p. 5)

A list of configuration items that you want to tag.

Type: array of Strings

Required: Yes

tags (p. 5)

Tags that you want to associate with one or more configuration items. Specify the tags that you want to create in a *key-value* format. For example:

`{"key": "serverType", "value": "webServer"}`

Type: array of Tag (p. 32) objects

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException

The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException

One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ResourceNotFoundException

The specified configuration ID was not located. Verify the configuration ID and try again.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

DeleteTags

Deletes the association between configuration items and one or more tags. This API accepts a list of multiple configuration items.

Request Syntax

```
{
   "configurationIds": [ "string" ],
   "tags": [
      {
         "key": "string",
         "value": "string"
      }
   ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

configurationIds (p. 7)

A list of configuration items with tags that you want to delete.

Type: array of Strings

Required: Yes

tags (p. 7)

Tags that you want to delete from one or more configuration items. Specify the tags that you want to delete in a *key-value* format. For example:

`{"key": "serverType", "value": "webServer"}`

Type: array of Tag (p. 32) objects

Required: No

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException

The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException

One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ResourceNotFoundException

The specified configuration ID was not located. Verify the configuration ID and try again.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

DescribeAgents

Lists agents or the Connector by ID or lists all agents/Connectors associated with your user account if you did not specify an ID.

Request Syntax

```
{
   "agentIds": [ "string" ],
   "maxResults": number,
   "nextToken": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

agentIds (p. 9)
The agent or the Connector IDs for which you want information. If you specify no IDs, the system returns information about all agents/Connectors associated with your AWS user account.

Type: array of Strings

Required: No

maxResults (p. 9)
The total number of agents/Connectors to return. The maximum value is 100.

Type: Integer

Required: No

nextToken (p. 9)
A token to start the list. Use this token to get the next set of results.

Type: String

Required: No

Response Syntax

```
{
   "agentsInfo": [
      {
         "agentId": "string",
         "agentNetworkInfoList": [
            {
               "ipAddress": "string",
               "macAddress": "string"
            }
         ],
         "connectorId": "string",
         "health": "string",
         "hostName": "string",
         "version": "string"
      }
   ],
   "nextToken": "string"
```

```
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

agentsInfo (p. 9)

Lists agents or the Connector by ID or lists all agents/Connectors associated with your user account if you did not specify an agent/Connector ID. The output includes agent/Connector IDs, IP addresses, media access control (MAC) addresses, agent/Connector health, host name where the agent/Connector resides, and the version number of each agent/Connector.

Type: array of AgentInfo (p. 26) objects

nextToken (p. 9)

The call returns a token. Use this token to get the next set of results.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException

The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException

One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

DescribeConfigurations

Retrieves a list of attributes for a specific configuration ID. For example, the output for a *server* configuration item includes a list of attributes about the server, including host name, operating system, number of network cards, etc.

Request Syntax

```
{
   "configurationIds": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

configurationIds (p. 11)
One or more configuration IDs.
Type: array of Strings
Required: Yes

Response Syntax

```
{
   "configurations": [
      {
         "string" : "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.
The following data is returned in JSON format by the service.

configurations (p. 11)
A key in the response map. The value is an array of data.
Type: array of String to String maps

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException
The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.
HTTP Status Code: 400

InvalidParameterException
One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

DescribeExportConfigurations

Retrieves the status of a given export process. You can retrieve status from a maximum of 100 processes.

Request Syntax

```
{
   "exportIds": [ "string" ],
   "maxResults": number,
   "nextToken": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

exportIds (p. 13)

A unique identifier that you can use to query the export status.

Type: array of Strings

Required: No

maxResults (p. 13)

The maximum number of results that you want to display as a part of the query.

Type: Integer

Required: No

nextToken (p. 13)

A token to get the next set of results. For example, if you specified 100 IDs for `DescribeConfigurationsRequest$configurationIds` but set `DescribeExportConfigurationsRequest$maxResults` to 10, you will get results in a set of 10. Use the token in the query to get the next set of 10.

Type: String

Required: No

Response Syntax

```
{
   "exportsInfo": [
      {
         "configurationsDownloadUrl": "string",
         "exportId": "string",
         "exportRequestTime": number,
         "exportStatus": "string",
         "statusMessage": "string"
      }
   ],
   "nextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

exportsInfo (p. 13)
Returns export details. When the status is complete, the response includes a URL for an Amazon S3 bucket where you can view the data in a CSV file.

Type: array of ExportInfo (p. 29) objects

nextToken (p. 13)
A token to get the next set of results. For example, if you specified 100 IDs for `DescribeConfigurationsRequest$configurationIds` but set `DescribeExportConfigurationsRequest$maxResults` to 10, you will get results in a set of 10. Use the token in the query to get the next set of 10.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException
The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException
One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException
The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ResourceNotFoundException
The specified configuration ID was not located. Verify the configuration ID and try again.

HTTP Status Code: 400

ServerInternalErrorException
The server experienced an internal error. Try again.

HTTP Status Code: 500

DescribeTags

Retrieves a list of configuration items that are tagged with a specific tag. Or retrieves a list of all tags assigned to a specific configuration item.

Request Syntax

```
{
   "filters": [
      {
         "name": "string",
         "values": [ "string" ]
      }
   ],
   "maxResults": number,
   "nextToken": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

filters (p. 15)

You can filter the list using a *key-value* format. You can separate these items by using logical operators. Allowed filters include `tagKey`, `tagValue`, and `configurationId`.

Type: array of TagFilter (p. 33) objects

Required: No

maxResults (p. 15)

The total number of items to return. The maximum value is 100.

Type: Integer

Required: No

nextToken (p. 15)

A token to start the list. Use this token to get the next set of results.

Type: String

Required: No

Response Syntax

```
{
   "nextToken": "string",
   "tags": [
      {
         "configurationId": "string",
         "configurationType": "string",
         "key": "string",
         "timeOfCreation": number,
         "value": "string"
      }
   ]
```

```
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

nextToken (p. 15)

The call returns a token. Use this token to get the next set of results.

Type: String

tags (p. 15)

Depending on the input, this is a list of configuration items tagged with a specific tag, or a list of tags for a specific configuration item.

Type: array of ConfigurationTag (p. 28) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException

The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException

One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ResourceNotFoundException

The specified configuration ID was not located. Verify the configuration ID and try again.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

ExportConfigurations

Exports all discovered configuration data to an Amazon S3 bucket or an application that enables you to view and evaluate the data. Data includes tags and tag associations, processes, connections, servers, and system performance. This API returns an export ID which you can query using the *GetExportStatus* API. The system imposes a limit of two configuration exports in six hours.

Response Syntax

```
{
   "exportId": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

exportId (p. 17)

A unique identifier that you can use to query the export status.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException

The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException

One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

OperationNotPermittedException

This operation is not permitted.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

ListConfigurations

Retrieves a list of configurations items according to the criteria you specify in a filter. The filter criteria identify relationship requirements.

Request Syntax

```
{
   "configurationType": "string",
   "filters": [
      {
         "condition": "string",
         "name": "string",
         "values": [ "string" ]
      }
   ],
   "maxResults": number,
   "nextToken": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

configurationType (p. 18)
A valid configuration identified by the Discovery Service.
Type: String
Valid Values: `SERVER | PROCESS | CONNECTION`
Required: Yes

filters (p. 18)
You can filter the list using a *key-value* format. For example:
`{"key": "serverType", "value": "webServer"}`
You can separate these items by using logical operators.
Type: array of Filter (p. 30) objects
Required: No

maxResults (p. 18)
The total number of items to return. The maximum value is 100.
Type: Integer
Required: No

nextToken (p. 18)
A token to start the list. Use this token to get the next set of results.
Type: String
Required: No

Response Syntax

```
{
   "configurations": [
```

```
      {
         "string" : "string"
      }
   ],
   "nextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

configurations (p. 18)

Returns configuration details, including the configuration ID, attribute names, and attribute values.

Type: array of String to String maps

nextToken (p. 18)

The call returns a token. Use this token to get the next set of results.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException

The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException

One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ResourceNotFoundException

The specified configuration ID was not located. Verify the configuration ID and try again.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

StartDataCollectionByAgentIds

Instructs the specified agents or Connectors to start collecting data.

Request Syntax

```
{
   "agentIds": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

agentIds (p. 20)

The IDs of the agents or Connectors that you want to start collecting data. If you send a request to an agent/Connector ID that you do not have permission to contact, according to your AWS account, the service does not throw an exception. Instead, it returns the error in the *Description* field. If you send a request to multiple agents/Connectors and you do not have permission to contact some of those agents/Connectors, the system does not throw an exception. Instead, the system shows `Failed` in the *Description* field.

Type: array of Strings

Required: Yes

Response Syntax

```
{
   "agentsConfigurationStatus": [
      {
         "agentId": "string",
         "description": "string",
         "operationSucceeded": boolean
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

agentsConfigurationStatus (p. 20)

Information about agents or the Connector that were instructed to start collecting data. Information includes the agent/Connector ID, a description of the operation performed, and whether or not the agent/Connector configuration was updated.

Type: array of AgentConfigurationStatus (p. 25) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException

The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException

One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

StopDataCollectionByAgentIds

Instructs the specified agents or Connectors to stop collecting data.

Request Syntax

```
{
   "agentIds": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 34).

The request accepts the following data in JSON format.

agentIds (p. 22)

The IDs of the agents or Connectors that you want to stop collecting data.

Type: array of Strings

Required: Yes

Response Syntax

```
{
   "agentsConfigurationStatus": [
      {
         "agentId": "string",
         "description": "string",
         "operationSucceeded": boolean
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

agentsConfigurationStatus (p. 22)

Information about agents or the Connector that were instructed to stop collecting data. Information includes the agent/Connector ID, a description of the operation performed, and whether or not the agent/Connector configuration was updated.

Type: array of AgentConfigurationStatus (p. 25) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 36).

AuthorizationErrorException

The AWS user account does not have permission to perform the action. Check the IAM policy associated with this account.

HTTP Status Code: 400

InvalidParameterException

One or more parameters are not valid. Verify the parameters and try again.

HTTP Status Code: 400

InvalidParameterValueException

The value of one or more parameters are either invalid or out of range. Verify the parameter values and try again.

HTTP Status Code: 400

ServerInternalErrorException

The server experienced an internal error. Try again.

HTTP Status Code: 500

Data Types

The AWS Discovery Service API contains several data types that various actions use. This section describes each data type in detail.

Note
The order of each element in a data type structure is not guaranteed. Applications should not assume a particular order.

The following data types are supported:

AgentConfigurationStatus

Information about agents or Connectors that were instructed to start collecting data. Information includes the agent/Connector ID, a description of the operation, and whether or not the agent/Connector configuration was updated.

Contents

agentId

The agent/Connector ID.

Type: String

Required: No

description

A description of the operation performed.

Type: String

Required: No

operationSucceeded

Information about the status of the `StartDataCollection` and `StopDataCollection` operations. The system has recorded the data collection operation. The agent/Connector receives this command the next time it polls for a new command.

Type: Boolean

Required: No

AgentInfo

Information about agents or Connectors associated with the user's AWS account. Information includes agent/Connector IDs, IP addresses, media access control (MAC) addresses, agent/Connector health, hostname where the agent/Connector resides, and agent version for each agent.

Contents

agentId
The agent/Connector ID.
Type: String
Required: No

agentNetworkInfoList
Network details about the host where the agent/Connector resides.
Type: array of AgentNetworkInfo (p. 27) objects
Required: No

connectorId
The ID of the Connector.
Type: String
Required: No

health
The health of the agent/Connector.
Type: String
Valid Values: `HEALTHY | UNHEALTHY | RUNNING | UNKNOWN | BLACKLISTED | SHUTDOWN`
Required: No

hostName
The name of the host where the agent/Connector resides. The host can be a server or virtual machine.
Type: String
Required: No

version
The agent or Connector version.
Type: String
Required: No

AgentNetworkInfo

Network details about the host where the agent/Connector resides.

Contents

ipAddress

The IP address for the host where the agent/Connector resides.

Type: String

Required: No

macAddress

The MAC address for the host where the agent/Connector resides.

Type: String

Required: No

ConfigurationTag

Tags for a configuration item. Tags are metadata that help you categorize IT assets.

Contents

configurationId
The configuration ID for the item you want to tag. You can specify a list of keys and values.
Type: String
Required: No

configurationType
A type of IT asset that you want to tag.
Type: String
Valid Values: `SERVER | PROCESS | CONNECTION`
Required: No

key
A type of tag to filter on. For example, *serverType*.
Type: String
Required: No

timeOfCreation
The time the configuration tag was created in Coordinated Universal Time (UTC).
Type: Timestamp
Required: No

value
A value to filter on. For example *key* = *serverType* and *value* = *web server*.
Type: String
Required: No

ExportInfo

Information regarding the export status of the discovered data. The value is an array of objects.

Contents

configurationsDownloadUrl
A URL for an Amazon S3 bucket where you can review the configuration data. The URL is displayed only if the export succeeded.
Type: String
Required: No

exportId
A unique identifier that you can use to query the export.
Type: String
Required: Yes

exportRequestTime
The time the configuration data export was initiated.
Type: Timestamp
Required: Yes

exportStatus
The status of the configuration data export. The status can succeed, fail, or be in-progress.
Type: String
Valid Values: `FAILED | SUCCEEDED | IN_PROGRESS`
Required: Yes

statusMessage
Helpful status messages for API callers. For example: Too many exports in the last 6 hours. Export in progress. Export was successful.
Type: String
Required: Yes

Filter

A filter that can use conditional operators.

Contents

condition

A conditional operator. The following operators are valid: EQUALS, NOT_EQUALS, CONTAINS, NOT_CONTAINS. If you specify multiple filters, the system utilizes all filters as though concatenated by *AND*. If you specify multiple values for a particular filter, the system differentiates the values using *OR*. Calling either *DescribeConfigurations* or *ListConfigurations* returns attributes of matching configuration items.

Type: String

Required: Yes

name

The name of the filter. The following filter names are allowed for `SERVER` configuration items.

Server

- `server.hostName`
- `server.osName`
- `server.osVersion`
- `server.configurationid`
- `server.agentid`

The name of the filter. The following filter names are allowed for `PROCESS` configuration items.

Process

- `process.configurationid`
- `process.name`
- `server.configurationid`
- `server.hostName`
- `server.osName`
- `server.osVersion`
- `server.agentId`

The name of the filter. The following filter names are allowed for `CONNECTION` configuration items.

Connection

- `connection.sourceIp`
- `connection.destinationIp`
- `connection.destinationPort`
- `sourceProcess.configurationId`
- `sourceProcess.name`
- `sourceProcess.commandLine`
- `destinationProcess.configurationId`
- `destinationProcess.name`
- `sourceServer.configurationId`
- `sourceServer.hostName`
- `sourceServer.osName`
- `sourceServer.osVersion`
- `sourceServer.agentId`

- `destinationServer.configurationId`
- `destinationServer.hostName`
- `destinationServer.osName`
- `destinationServer.osVersion`
- `destinationServer.agentId`

Type: String

Required: Yes

values

A string value that you want to filter on. For example, if you choose the `destinationServer.osVersion` filter name, you could specify `Ubuntu` for the value.

Type: array of Strings

Required: Yes

Tag

Metadata that help you categorize IT assets.

Contents

key
A type of tag to filter on.
Type: String
Required: Yes

value
A value for a tag key to filter on.
Type: String
Required: Yes

TagFilter

The name of a tag filter. Valid names are: `tagKey`, `tagValue`, `configurationId`.

Contents

name
A name of a tag filter.
Type: String
Required: Yes

values
Values of a tag filter.
Type: array of Strings
Required: Yes

Common Parameters

The following table lists the parameters that all actions use for signing Signature Version 4 requests. Any action-specific parameters are listed in the topic for that action. To view sample requests, see Examples of Signed Signature Version 4 Requests or Signature Version 4 Test Suite in the *Amazon Web Services General Reference*.

Action
The action to be performed.

Type: string

Required: Yes

Version
The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm
The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: `AWS4-HMAC-SHA256`

Required: Conditional

X-Amz-Credential
The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: *access_key*/*YYYYMMDD*/*region*/*service*/aws4_request.

For more information, see Task 2: Create a String to Sign for Signature Version 4 in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: `20120325T120000Z`.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see Handling Dates in Signature Version 4 in the *Amazon Web Services General Reference*.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service. For a list of services that support AWS Security Token Service, go to Using Temporary Security Credentials to Access AWS in *Using Temporary Security Credentials*.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see Task 1: Create a Canonical Request For Signature Version 4 in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

Common Errors

This section lists the common errors that all actions return. Any action-specific errors are listed in the topic for the action.

IncompleteSignature
The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure
The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction
The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId
The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

InvalidParameterCombination
Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue
An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter
The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString
The query string contains a syntax error.

HTTP Status Code: 404

MissingAction
The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken
The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter
A required parameter for the specified action is not supplied.

HTTP Status Code: 400

OptInRequired
The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired
The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable
The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

Throttling
The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError
The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400