akamai
techdocs

Terraform Datastream

EOL: v0.8 - 4.1

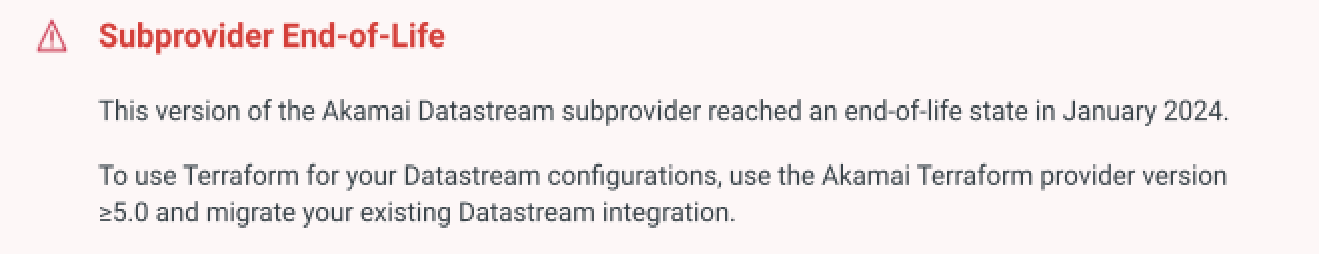
June 13, 2024

Version history

Version	Date	Description
1.8.0	October 25, 2021	Initial release of Datastream subprovider documentation.
Guides		
1.9.0	December 6, 2021	Added `datastream` data source output block to main guide.
2.3.0	August 25, 2022	Product name clean up in main guide.
4.0.0	December 4, 2023	Fixed broken link in main guide.
4.1.0	June 15, 2023	Fixed broken link in main guide.
	June 21, 2023	Clarified information around data sets in the migration guide.
	June 23, 2023	Fixed broken list in data set table in the migration guide.
	July 5, 2023	Added migration guide.
	September 13, 2023	Fixed broken link in migration guide.
Resource		
1.9.0	December 6, 2021	Fixed argument nesting and added security notes.
3.5.0 3.6.0 4.0.0 4.1.0	July 5, 2023	Added create time banner and fixed broken link.
	August 04, 2023	Updated attribute section of the `datastream` resource to reflect actual standard output.
	August 11, 2023	Clarified language in the create time banner.
	August 28, 2023	Updated link in create time banner.
	January 28, 2024	Updated time in the create time banner.
	June 12, 2024	Clarified `active` argument conditions.
Data Sources		
1.9.0	December 6, 2021	Fixed attribute nesting in Dataset fields.
4.1.0	July 5, 2023	Fixed broken link in Activation history, Dataset fields, and Data streams.
	August 10, 2023	Fixed broken link in Data streams.

Contents

Create data streams



⚠ **Subprovider End-of-Life**

This version of the Akamai Datastream subprovider reached an end-of-life state in January 2024.

To use Terraform for your Datastream configurations, use the Akamai Terraform provider version ≥5.0 and migrate your existing Datastream integration.

With this subprovider, you can create data streams for your properties to provide scalable, low latency streaming of data in raw form. You can use raw data logs to find details about specific incidents, search the logs for instances using a specific IP address, or analyze the patterns of multiple attacks.

You can configure your data stream to bundle and push logs to a destination for storing, monitoring, and analytical purposes. Each data stream supports only one endpoint to send log data to, either:

- Amazon S3. Provides cloud object storage for your data. For more information, see Getting started with Amazon S3.
- Azure Storage. Provides object storage for data objects that is highly available, secure, durable, scalable, and redundant. See Azure Blob Storage.
- Custom HTTPS endpoint. Send log data gathered by your stream to an HTTPS endpoint of your choice.
- Datadog. Provides monitoring for servers, databases, and services, allows for stacking and aggregating metrics. See Datadog.
- Google Cloud Storage. Provides a cloud-based storage with low latency, high durability, and worldwide accessibility. See Google Cloud Storage.
- Oracle Cloud. Provides a scalable, cloud-based storage using the S3-compatible API connectivity option to store data. See Oracle Cloud.
- Splunk. Provides an interface for advanced metrics, monitoring, and data analysis. See Splunk.
- Sumo Logic. Provides advanced analytics for your log files. See Sumo Logic.

Prerequisites

DataStream collects performance logs against selected delivery properties and streams them to configured destinations. To create a stream, you need to have at least one existing property within the group and contract you want the stream to collect logs for. A stream can start collecting logs only if the referenced properties have the `datastream` behavior enabled in their rule tree and are active on the production network.

Use the akamai_property and akamai_property_activation resources to create and activate or import delivery properties.

DataStream workflow

- Get the group ID. You need your group ID to create a data stream.
- Get the data set fields. Choose the data set fields that you want to monitor in your logs in a stream configuration.
- Create a data stream. Create a new stream to collect logs for associated properties and send that data to a connector.
- Add a DataStream rule to a property. Copy the returned JSON snippet into the rule tree configuration.
- Activate the property version. After you modify the rule tree, activate the changed property on the production network.
- Activate the stream. Activate the latest version of a stream to start collecting and sending logs to a destination.
- View activation history. Check a history of activation status changes for all versions of a stream.
- Delete a stream. If you don't need the log data anymore, you can delete a stream.

Get the group ID

When setting up streams, you need to get the Akamai `group_id`.

The DataStream module supports both ID formats, either with or without the `grp_` prefix. For more information about prefixes, see the ID prefixes section of the Property Manager API (PAPI) documentation.

Get the data set fields

Use the `akamai_datastream_dataset_fields` data source to view the data set fields available within the template. Store the `dataset_field_id` values of the fields you want to receive in logs.

Create a data stream

To monitor and gain real-time access to delivery performance, create a new stream or import an existing one using the `akamai_datastream` resource. You can associate up to 100 properties with a single stream and specify a data set that you want this stream to deliver. For each property, you can create up to 3 streams to specify different data sets that you want to receive about your application, and send it to the destinations of your choice.

-> Note Data stream activation might be time-consuming, so set the `active` flag to `false` until you completely finish the setup.

Once you set up the `akamai_datastream` resource, run `terraform apply`. Terraform shows an overview of changes, so you can still go back and modify the configuration, or confirm to proceed. See Command: apply.

Add a DataStream rule to a property

To start collecting logs for properties in a stream, you need to enable the DataStream behavior in each property that is part of any stream. You can't receive logs from properties with a disabled DataStream behavior even if they're part of active data streams.

The `terraform apply` command returns a `papi_json` attribute with a JSON-encoded rule for DataStream. Use the output block to declare the rule value exported by a subprovider, for example:

hcl

```hcl
output "datastream_rule" {
    value = data.akamai_datastream.example_stream.papi_json
```

An example of the JSON-encoded rule for DataStream:

JSON

```json
{
    "name": "Datastream Rule",
    "children": [],
    "behaviors": [
        {
            "name": "datastream",
            "options": {
                "streamType": "LOG",
                "logEnabled": true,
                "logStreamName": 7050,
                "samplingPercentage": 100
            }
        }
    ],
    "criteria": [],
    "criteriaMustSatisfy": "all"
}
```

Copy this snippet to the rule tree files in properties you created the stream for. You can also create a new `.json` file with the snippet and insert it to the property rule tree by adding `"#include:example-file.json"` under the `children` array. See [Referencing sub-files from a template]() for more information.

If you wish to customize how your data stream is handled, see the [`datastream` behavior in the PAPI documentation]().

Activate the property version

Use the [`akamai_property_activation`]() resource to activate the modified property version on the production network. You can only stream logs for active properties with the DataStream behavior enabled. Run `terraform apply` again to implement the changes.

Activate the data stream version

Once you've made all the modifications in your data stream, set the `active` flag in the `akamai_datastream` resource to `true` and run `terraform apply`. This operation takes approximately 90 minutes.

The moment a stream goes active and the `DataStream` behavior is enabled in your property, it starts collecting and sending logs to a destination. If you want to stop receiving these logs, you can deactivate a stream at any time by setting the flag back to `false`.

View activation history

Use the `akamai_datastream_activation_history` data source to get detailed information about activation status changes for a version of a stream.

Delete a stream

To delete a stream, remove the `akamai_datastream` resource and all the dependencies from your Terraform configuration. If you want to delete an `active` stream, the provider automatically deactivates it first. If you want to delete a stream with a pending status, either `activating` or `deactivating`, the provider waits until the status becomes stable and proceeds with the operation.

Deleting a stream means that you can't activate this stream again, and that you stop receiving logs for the properties that this stream monitors.

Activation history

akamai_datastream_activation_history

Use the `akamai_datastream_activation_history` data source to list detailed information about the activation status changes for all versions of a stream.

Example

This example returns the activation history for a provided stream ID.

```hcl
data "akamai_datastream_activation_history" ds {
  stream_id = 12345
}

output "ds_history_stream_id" {
  value = data.akamai_datastream_activation_history.ds.stream_id
}

output "ds_history_activations" {
  value = data.akamai_datastream_activation_history.ds.activations
}
```

Argument reference

The data source supports this argument:

- `stream_id` - (Required) A stream's unique identifier.

Attributes reference

This data source returns these attributes:

- `activations` - Detailed information about an activation status change for a version of a stream, including:
  - `created_by` - The user who activated or deactivated the stream.
  - `created_date` - The date and time of an activation status change.
  - `stream_id` - A stream's unique identifier.
  - `stream_version_id` - A stream version's unique identifier.
  - `is_active` - Whether the version of the stream is active.

Dataset fields

akamai_datastream_dataset_fields

Use the akamai_datastream_dataset_fields data source to list groups of data set fields available in the template.

Example

This example returns data set fields for a default template.

```hcl
data "akamai_datastream_dataset_fields" "fields" {}
```

Argument reference

The data source supports this argument:

- template_name - (Optional) The name of the data set template you use in your stream configuration. Currently, EDGE_LOGS is the only available data set template and the default value for this argument.

Attributes reference

This data source returns these attributes:

- fields - A group of data set fields available in a template, including:
    - dataset_group_name - The name of the data set group.
    - dataset_group_description - Additional information about the data set group.
    - dataset_fields - A list of data set fields available within the data set group, including:
        - dataset_field_description - Additional information about the data set field.
        - dataset_field_id - Unique identifier for the field.
        - dataset_field_json_key - The JSON key for the field in a log line.
        - dataset_field_name - The name of the data set field.

Data streams

akamai_datastreams

Use the `akamai_datastreams` data source to list details about the DataStream configuration.

Example usage

This example returns stream data for a specific group ID:

hcl

```hcl
locals {
  concrete_stream = [for stream in data.akamai_datastreams.stream_list_in_group.streams :
  stream if stream.stream_name == "Concrete stream name"][0]
}

data "akamai_datastreams" "stream_list_in_group" {
  group_id = "1234"
}
```

Argument reference

The data source supports this argument:

- `group_id` - (Optional) Unique identifier of the group that can access the product.

Attributes reference

This data source returns these attributes:

- `streams` - Returns the latest versions of the stream configurations for all groups within your account. You can use the `group_id` parameter to view the latest versions of all configurations in a specific group.
    - `activation_status` - The activation status of the stream. These are possible values: `ACTIVATED`, `DEACTIVATED`, `ACTIVATING`, `DEACTIVATING`, or `INACTIVE`. See the Activate a stream and Deactivate a stream operations.
    - `archived` - Whether the stream is archived.
    - `connectors` - The connector where the stream sends logs.
    - `contract_id` - Identifies the contract that the stream is associated with.
    - `created_by` - The user who created the stream.
    - `created_date` - The date and time when the stream was created in this format: `14-07-2020 07:07:40 GMT`.

- `current_version_id` - Identifies the current version of the stream.
- `errors` - Objects that may indicate stream failure errors. Learn more about Errors.
    - `detail` - A message informing about the status of the failed stream.
    - `title` - A descriptive label for the type of error.
    - `type` - Identifies the error type, either `ACTIVATION_ERROR` or `UNEXPECTED_SYSTEM_ERROR`. In case of these errors, contact support for assistance before continuing.
- `group_id` - Identifies the group where the stream is created.
- `group_name` - The group name where the stream is created.
- `properties` - List of properties associated with the stream.
    - `property_id` - The identifier of the property.
    - `property_name` - The descriptive label for the property.
- `stream_id` - A stream's unique identifier.
- `stream_name` - The name of the stream.
- `stream_type_name` - Specifies the type of the data stream. `Logs - Raw` is the only stream type name currently available.
- `stream_version_id` - A stream version's unique identifier.

DataStream

akamai_datastream

Akamai constantly gathers log entries from thousands of edge servers around the world. You can use the `akamai_datastream` resource to capture these logs and deliver them to a connector of your choice at low latency. A connector, also known as a destination, represents a third-party configuration where you want to send your stream's log files to. For each stream, you can only set one connector.

When creating a stream, you select properties to associate with the stream, data set fields to monitor in logs, and a destination to send these logs to. You can also decide whether to activate the stream on making the request. Only active streams collect and send logs to their destinations.

Example

```hcl
resource "akamai_datastream" "stream" {
    active             = true
    config {
        delimiter          = "SPACE"
        format             = "STRUCTURED"
        frequency {
            time_in_sec = 30
        }
        upload_file_prefix = "pre"
        upload_file_suffix = "suf"
    }
    contract_id        = "C-0N7RAC7"
    dataset_fields_ids = [
        1002, 1005, 1006
    ]
    email_ids          = [
        "example@example.com",
        "example2@example.com"
    ]
    group_id           = 12345
    property_ids       = [
        100011011
    ]
    stream_name        = "Test data stream"
    stream_type        = "RAW_LOGS"
    template_name      = "EDGE_LOGS"

    s3_connector {
        access_key        = "ACC35k3YT2ll1H4dXWx5itGhpc7FlSbvvOvky10"
        bucket            = "example.bucket.com"
        connector_name    = "S3Destination"
        path              = "log/edgelogs"
        region            = "ap-south-1"
        secret_access_key = "SKTACC3K3YAKIA6DK7TD"
    }
}
```

Argument reference

The resource supports these arguments:

- `active` - (Required) Whether to activate the data stream when the resource is applied.
    - `true` activates the data stream upon `terraform apply`.
    - `false` creates or updates a data stream without an activation.
- `config` - (Required) Provides information about the log line configuration, log file format, names of log files sent, and file delivery. The argument includes these sub-arguments:
    - `delimiter` - (Optional) A delimiter that you want to use to separate data set fields in the log lines. Currently, `SPACE` is the only available delimiter. This field is required for the `STRUCTURED` log file `format`.
    - `format` - (Required) The format in which you want to receive log files, either `STRUCTURED` or `JSON`. When `delimiter` is present in the request, `STRUCTURED` is the mandatory format.
    - `frequency` - (Required) How often you want to collect logs from each uploader and send them to a destination.
        - `time_in_sec` - (Required) The time in seconds after which the system bundles log lines into a file and sends it to a destination. `30` or `60` are the possible values.
    - `upload_file_prefix` - (Optional) The prefix of the log file that you want to send to a destination. It's a string of at most 200 characters. If unspecified, defaults to `ak`.
    - `upload_file_suffix` - (Optional) The suffix of the log file that you want to send to a destination. It's a static string of at most 10 characters. If unspecified, defaults to `ds`.
- `contract_id` - (Required) Identifies the contract that has access to the product.
- `dataset_fields_ids` - (Required) Identifiers of the data set fields within the template that you want to receive in logs. The order of the identifiers define how the value for these fields appears in the log lines. See Data set parameters.
- `email_ids` - (Optional) A list of email addresses you want to notify about activations and deactivations of the stream.
- `group_id` - (Required) Identifies the group that has access to the product and this stream configuration.
- `property_ids` - (Required) Identifies the properties that you want to monitor in the stream. Note that a stream can only log data for active properties.
- `stream_name` - (Required) The name of the stream.
- `stream_type` - (Required) The type of stream that you want to create. Currently, `RAW_LOGS` is the only possible stream type.
- `template_name` - (Required) The name of the data set template available for the product that you want to use in the stream. Currently, `EDGE_LOGS` is the only data set template available.

- s3_connector - (Optional) Specify details about the Amazon S3 connector in a stream. When validating this connector, DataStream uses the provided access_key and secret_access_key values and saves an akamai_write_test_2147483647.txt file in your Amazon S3 folder. You can only see this file if validation succeeds, and you have access to the Amazon S3 bucket and folder that you're trying to send logs to. The argument includes these sub-arguments:
    - access_key - (Required) Secret. The access key identifier that you use to authenticate requests to your Amazon S3 account. See Managing access keys (AWS API).
    - bucket - (Required) The name of the Amazon S3 bucket. See Working with Amazon S3 Buckets.
    - connector_name - (Required) The name of the connector.
    - path - (Required) The path to the folder within your Amazon S3 bucket where you want to store your logs. See Amazon S3 naming conventions.
    - region - (Required) The AWS region where your Amazon S3 bucket resides. See Regions and Zones in AWS.
    - secret_access_key - (Required) Secret. The secret access key identifier that you use to authenticate requests to your Amazon S3 account.
- azure_connector - (Optional) Specify details about the Azure Storage connector configuration in a data stream. Note that currently DataStream supports only streaming data to block objects. The argument includes these sub-arguments:
    - access_key - (Required) Secret. Either of the access keys associated with your Azure Storage account. See View account access keys in Azure.
    - account_name - (Required) Specifies the Azure Storage account name.
    - connector_name - (Required) The name of the connector.
    - container_name - (Required) Specifies the Azure Storage container name.
    - path - (Required) The path to the folder within the Azure Storage container where you want to store your logs. See Azure blob naming conventions.
- datadog_connector - (Optional) Specify details about the Datadog connector in a stream, including:
    - auth_token - (Required) Secret. The API key associated with your Datadog account. See View API keys in Datadog.
    - compress logs - (Optional) Enables GZIP compression for a log file sent to a destination. If unspecified, this defaults to false.
    - connector_name - (Required) The name of the connector.
    - service - (Optional) The service of the Datadog connector. A service groups together endpoints, queries, or jobs for the purposes of scaling instances. See View Datadog reserved attribute: service.
    - source - (Optional) The source of the Datadog connector. See View Datadog reserved attribute: source.
    - tags - (Optional) The tags of the Datadog connector. See View Datadog tags.
    - url - (Required) The Datadog endpoint where you want to store your logs. See View Datadog logs endpoint.

- `splunk_connector` - (Optional) Specify details about the Splunk connector in your stream. Note that currently DataStream supports only endpoint URLs ending with `collector/raw`. The argument includes these sub-arguments:
    - `compress_logs` - (Optional) Enables GZIP compression for a log file sent to a destination. If unspecified, this defaults to `true`.
    - `connector_name` - (Required) The name of the connector.
    - `event_collector_token` - (Required) Secret. The Event Collector token associated with your Splunk account. See View usage of Event Collector token in Splunk.
    - `url` - (Required) The raw event Splunk URL where you want to store your logs.
- `gcs_connector` - (Optional) Specify details about the Google Cloud Storage connector you can use in a stream. When validating this connector, DataStream uses the private access key to create an `Akamai_access_verification_<timestamp>.txt` object file in your GCS bucket. You can only see this file if the validation process is successful, and you have access to the Google Cloud Storage bucket where you are trying to send logs. The argument includes these sub-arguments:
    - `bucket` - (Required) The name of the storage bucket you created in your Google Cloud account. See Bucket naming conventions.
    - `connector_name` - (Required) The name of the connector.
    - `path` - (Optional) The path to the folder within your Google Cloud bucket where you want to store logs. See Object naming guidelines.
    - `private_key` - (Required) Secret. The contents of the JSON private key you generated and downloaded in your Google Cloud Storage account.
    - `project_id` - (Required) The unique ID of your Google Cloud project.
    - `service_account_name` - (Required) The name of the service account with the storage.object.create permission or Storage Object Creator role.
- `https_connector`- (Optional) Specify details about the custom HTTPS endpoint you can use as a connector for a stream, including:
    - `authentication_type` - (Required) Either `NONE` for no authentication, or `BASIC`. For basic authentication, provide the `user_name` and `password` you set in your custom HTTPS endpoint.
    - `compress_logs` - (Optional) Whether to enable GZIP compression for a log file sent to a destination. If unspecified, this defaults to `false`.
    - `connector_name` - (Required) The name of the connector.
    - `password` - (Optional) Secret. Enter the password you set in your custom HTTPS endpoint for authentication.
    - `url` - (Required) Enter the secure URL where you want to send and store your logs.
    - `user_name` - (Optional) Secret. Enter the valid username you set in your custom HTTPS endpoint for authentication.
- `sumologic_connector` - (Optional) Specify details about the Sumo Logic connector in a stream, including:
    - `collector_code` - (Required) Secret. The unique HTTP collector code of your Sumo Logic `endpoint`.

    - `compress_logs` - (Optional)Enables GZIP compression for a log file sent to a destination. If unspecified, this defaults to `true`.
    - `connector_name` - (Required) The name of the connector.
    - `endpoint` - (Required) The Sumo Logic collection endpoint where you want to send your logs. You should follow the `https://<SumoEndpoint>/receiver/v1/http` format and pass the collector code in the `collectorCode` argument.
- `oracle_connector`- (Optional) Specify details about the Oracle Cloud Storage connector in a stream. When validating this connector, DataStream uses the provided `access_key` and `secret_access_key` values and tries to save an `Akamai_access_verification_<timestamp>.txt` file in your Oracle Cloud Storage folder. You can only see this file if the validation process is successful, and you have access to the Oracle Cloud Storage bucket and folder that you're trying to send logs to.
    - `access_key` - (Required) Secret. The access key identifier that you use to authenticate requests to your Oracle Cloud account. See Managing user credentials in OCS.
    - `bucket` - (Required) The name of the Oracle Cloud Storage bucket. See Working with Oracle Cloud Storage buckets.
    - `connector_name` - (Required) The name of the connector.
    - `namespace` - (Required) The namespace of your Oracle Cloud Storage account. See Understanding Object Storage namespaces.
    - `path` - (Required) The path to the folder within your Oracle Cloud Storage bucket where you want to store your logs.
    - `region` - (Required) The Oracle Cloud Storage region where your bucket resides. See Regions and availability domains in OCS.
    - `secret_access_key` - (Required) Secret. The secret access key identifier that you use to authenticate requests to your Oracle Cloud account.

Attributes reference

There is no default standard output as the attribute values are sensitive, but you can get your data stream's ID from the last line of the process log.

Shell

```
akamai_datastream.my_datastream: Creation complete after 9s [id=12345]
```

Notice

Akamai secures and delivers digital experiences for the world's largest companies. Akamai's Intelligent Edge Platform surrounds everything, from the enterprise to the cloud, so customers and their businesses can be fast, smart, and secure. Top brands globally rely on Akamai to help them realize competitive advantage through agile solutions that extend the power of their multi-cloud architectures. Akamai keeps decisions, apps, and experiences closer to users than anyone — and attacks and threats far away. Akamai's portfolio of edge security, web and mobile performance, enterprise access, and video delivery solutions is supported by unmatched customer service, analytics, and 24/7/365 monitoring. To learn why the world's top brands trust Akamai, visit www.akamai.com, blogs.akamai.com, or @Akamai on Twitter. You can find our global contact information at www.akamai.com/locations.

Akamai is headquartered in Cambridge, Massachusetts in the United States with operations in more than 57 offices around the world. Our services and renowned customer care are designed to enable businesses to provide an unparalleled Internet experience for their customers worldwide. Addresses, phone numbers, and contact information for all locations are listed on www.akamai.com/locations.

While precaution has been taken in the preparation of this document, Akamai Technologies, Inc. assumes no responsibility for errors, omissions, or for damages resulting from the use of the information herein. The information in this document is subject to change without notice. Without limitation of the foregoing, if this document discusses a product or feature in beta or limited availability, such information is provided with no representation or guarantee as to the matters discussed, as such products/features may have bugs or other issues.

Akamai and the Akamai wave logo are registered trademarks or service marks in the United States (Reg. U.S. Pat. & Tm. Off). Akamai Intelligent Edge Platform is a trademark in the United States. Products or corporate names may be trademarks or registered trademarks of other companies and are used only for explanation and to the owner's benefit, without intent to infringe.

Published June 13, 2024