Attribute Sync

Introduction

Attribute sync enables users to be linked to attributes that can be used for inventory management, campaign analysis and targeting.
These associations are retrieved when an ad is requested.

Consent

As the data is associated with a user ID, consent is required. When you push data to Adhese via attribute-sync, we assume that the sender has validated the consent. Do not send us data for which no consent has been given.

We do validate consent on the ad-request side. Ad requests without a userID or consent will not be linked to data pushed via attribute-sync.

Pushing data

Authentication

The first step in pushing data is to retrieve a valid JSON Web Token (JWT) from our Keycloak server. We use the ROPC flow, but this may change in the future. You will be provided with login credentials for an account that is specifically used for attribute-sync.

An example request to retrieve a JWT token

```
curl --location 'https://auth.we.adhese.org/realms/user-sync/protocol/openid-connect/token' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'client_id=user-sync-app' \
--data-urlencode 'username=[username]' \
--data-urlencode 'password=[password]' \
--data-urlencode 'grant_type=password'
```

This token can be used multiple times before it expires.

IP-filtering

Upon request, we can enable IP-filtering on the attribute-sync endpoint. If IP-filtering is enabled, we will only accept data from a predefined list of IP addresses.

Uploading data

The uploading of data actually happens via the https://ads-[customer].adhese.com/usersync/attributesync endpoint (POST with json payload).

Ensure that the access token from the prior steps is used as a bearer token in the `Authorization` header.

```
{
  "reportDateTimeValid": "2026-02-11T17:11:00Z",
  "entries": [
    {
      "identifier": {
        "name": "DIGIMON_ID",
        "value": "userId1"
      },
      "attributes": [
        {
          "name": "firstAttribute",
          "value": [
            "known_app_user",
            "example_value"
          ]
        },
        {
          "name": "secondAttribute",
          "value": [
            "known_app_user2",
            "known_test_trait_pageview_less_then_2"
          ]
        }
      ]
    },
    {
      "identifier": {
        "name": "DIGIMON_ID",
        "value": "foobar"
      },
      "attributes": [
        {
          "name": "firstAttribute",
          "value": [
            "known_app_user",
```

```
            "another_example_value"
          ]
        }
      ]
    }
  ]
}
```

Above you can see an example payload.

reportDateTimeValid	Optional	For how long should we store the data from this request? If this is not specified, we default to seven days in the future. By default, we limit the validity to 30 days.
entries	Mandatory	Each user is represented by one entry. We accept up to 500 entries per request. If you need to store data for more than 500 users, please submit multiple requests.
entries.identifier.value	Mandatory	This field represents the userId of this entry.
entries.identifier.name	Mandatory	Some customers have different types of userIDs (e.g. an actual userID and a deviceID) and want to be able to store different data for the same ID if it came from a different source. You can think of this as a namespace for your userIDs. These can be arbitrary, but must be used consistently. Please inform our support agents which ones you plan to use. Even if you only use a single source of user IDs, you still need to choose a value.
entries.attributes	Mandatory but can be empty	Can be left empty for deleting data. See below.
entries.attributes.name	Mandatory	The attribute for which you want to store values for the specific userId.
entries.attributes.values	Mandatory	The values you want to store with the attribute for the specific userId.

Each time we receive data via attribute-sync for a specific user, the latest request overwrites any previous requests for that user, even if certain attributes were included in an earlier request but not in the latest one.

This has the side effect that you can delete all attribute-sync data associated with a user by leaving the entries.attributes field empty.

As attribute-sync is intended for text-based data, we cannot accept data containing Unicode control characters.

Using the data

Cooperate with Support to configure which targets correspond to which userIDs and which attributes correspond to which targets. Once this is complete, ad requests with consent and a user ID will automatically receive additional targets from the data you pushed, even if the push occurred less than a second ago.

What data to store

As all data pushed via attribute-sync must be accessible immediately, it is stored in memory. For this reason, please be mindful of the data you store.

- Only store attributes that you actually plan on using
- Do not make attributes or attribute values overly verbose; they do not need to be human-readable
- Keep userIds to a reasonable length

Revision #13
Created 18 October 2024 07:45:26 by Casper Steuperaert
Updated 10 March 2026 13:14:01 by Ron Van Maanen