

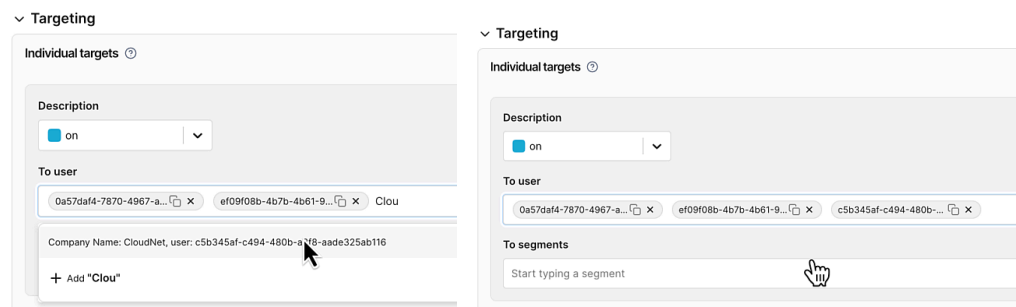
Setting Up Identity Type-Ahead and Hover Tooltips

Overview

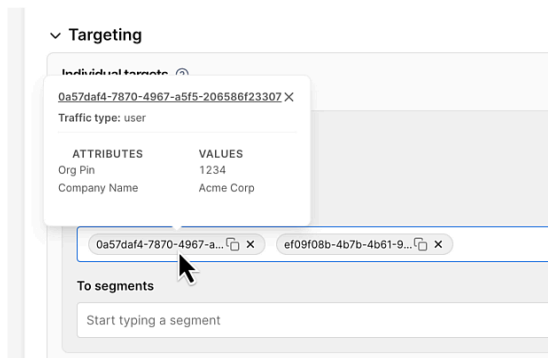
This document describes how to set up **Identity Type-Ahead** and **Identity Tooltips** in Harness Feature Management & Experimentation (FME).

What you get:

- **Key Type-Ahead:** When adding individual keys to targeting rules, type the leading characters of an attribute value to see matching keys from your identity data (case-sensitive, leading characters only - does not match mid-string)



- **Identity Tooltips:** Hover over any key to see its associated attribute values



Two-step setup:

1. Define attributes in Traffic Type (admin screen)
2. Upload identity data (keys + attribute values) via API

Setup Flow

STEP 1: DEFINE ATTRIBUTES
(Admin Settings - One Time)

Navigate to Traffic Type:

↳ FME Settings → View [to right of project name] → Traffic Types

Add Attributes:

↳ [On Row of Traffic Type] Click "View / Edit attributes"

↳ Click Actions > Create an attribute

↳ Define: name = "company_name", type = STRING

↳ Save attribute

↳ Repeat for other attributes:

↳ company_short_code (STRING)

↳ account_number (STRING)

↳ region (STRING)

✅ Attributes defined (required for tooltips to display correctly)

STEP 2: UPLOAD IDENTITY DATA (API - Periodic or Continuous)

Upload via API:

- └─ **Option A: Use Helper Tool (Recommended)**
 - └─ Build CSV File (use friendly column names):
 - └─ Format: key, shortname1, shortname2, ...
 - └─ Example:

```
key,company,short_code,account,region
a3f7b9c2-...,Acme Corporation,ACME,ACC-98765,us-west
b8e4c3d7-...,TechStart Inc,TCHS,ACC-12345,eu-central
c1d9e5f3-...,Global Systems Ltd,GLSY,ACC-54321,ap-southeast
d7f2a4b8-...,DataFlow Corp,DTFL,ACC-67890,us-east
```
 - └─ Extract fme_identities_upload_tool.zip
 - └─ Create config JSON file mapping CSV columns to formal attributes
 - └─ Run: `python3 fme_bulk_identities.py --csv identities.csv --config config.json`
 - └─ Tool converts CSV to JSON and calls API automatically
- └─ **Option B: Direct API Call**
 - └─ POST `https://api.split.io/internal/api/v2/trafficTypes/{traffic-type-id}/environments/{environment-id}/identities`
 - └─ Headers: `Authorization: Bearer ADMIN_API_KEY,`
`Content-Type: application/json`
 - └─ Body: JSON array with "key" and "values" fields
(overwrites all attributes)
 - └─ See API Reference section for full example
- └─ Can be automated via script/pipeline

Repeat for Multiple Environments (Optional):

- |
- | ↳ Upload same or different data to:
- | | ↳ Production environment
- | | ↳ Staging environment
- | | ↳ Development environment
- |
- |
- | ↓

☒ **ENABLED: Key Type-Ahead in Individual Targets**

When adding individual keys:

- Type "Acme" in "Add individually" field
- Dropdown shows matching identities:
 - "company_name: Acme Corporation, account: a3f7b9c2-4d1e-4a8b-9c3f..."
 - "+ Add 'Acme'" (ignore this - adds literal string, not the key)
- Click the matching identity result (first item)
- Type-ahead matches leading characters (case-sensitive)
- Does not match mid-string (e.g., "Corp" won't find "Acme Corporation")

☒ **ENABLED: Identity Inspector Tooltips**

When hovering over a key:

- Shows: a3f7b9c2-4d1e-4a8b-9c3f-7e6d5a4b3c2d
- Tooltip displays:

Key: a3f7b9c2-4d1e-4a8b-9c3f-7e6d5a4b3c2d

Attributes:

- company_name: Acme Corporation
- company_short_code: ACME
- account_number: ACC-98765
- region: us-west

API Reference

Bulk Upload (Save Identities)

Endpoint:

```
None
POST
https://api.split.io/internal/api/v2/trafficTypes/{traffic-type-id}/environments/{environment-id}/identities
```

Headers:

```
None
Authorization: Bearer ADMIN_API_KEY
Content-Type: application/json
```

Request Body (JSON):

```
JSON
[
  {
    "key": "a3f7b9c2-4d1e-4a8b-9c3f-7e6d5a4b3c2d",
    "values": {
      "company_name": "Acme Corporation",
      "company_short_code": "ACME",
      "account_number": "ACC-98765",
      "region": "us-west"
    }
  },
  {
    "key": "b8e4c3d7-2a9f-4b7c-8d2e-6f5a3b9c8d1e",
    "values": {
      "company_name": "TechStart Inc",
      "company_short_code": "TCHS",
      "account_number": "ACC-12345",
      "region": "eu-central"
    }
  }
]
```

Important: This bulk endpoint **OVERWRITES** all **attributes** for each key. Any attributes not included in the request will be deleted.

Example cURL:

Shell

```
curl -v -X POST \
  -H 'Content-Type: application/json' \
  -H 'Authorization: Bearer ADMIN_API_KEY' \
  -d '[
    {
      "key": "a3f7b9c2-4d1e-4a8b-9c3f-7e6d5a4b3c2d",
      "values": {
        "company_name": "Acme Corporation",
        "company_short_code": "ACME",
        "account_number": "ACC-98765",
        "region": "us-west"
      }
    }
  ]' \

https://api.split.io/internal/api/v2/trafficTypes/{traffic-type-id}/environments/{environment-id}/identities
```

Single Identity Update (Save Identity)

Endpoint:

None

PUT

<https://api.split.io/internal/api/v2/trafficTypes/{traffic-type-id}/environments/{environment-id}/identities/{key}>

Headers:

None

[Authorization: Bearer ADMIN_API_KEY](#)

```
Content-Type: application/json
```

Request Body (JSON):

```
JSON
{
  "values": {
    "company_name": "Acme Corporation",
    "company_short_code": "ACME"
  }
}
```

Important: This single-key endpoint **PRESERVES unspecified attributes**. Only the attributes included in the request will be updated.

Example cURL:

```
Shell
curl -v -X PUT \
  -H 'Content-Type: application/json' \
  -H 'Authorization: Bearer ADMIN_API_KEY' \
  -d '{
    "values": {
      "company_name": "Acme Corporation Updated"
    }
  }' \

https://api.split.io/internal/api/v2/trafficTypes/{traffic-type-id}/environment
s/{environment-id}/identities/a3f7b9c2-4d1e-4a8b-9c3f-7e6d5a4b3c2d
```

Using the Helper Tool

A Python utility (`fme_bulk_identities.py`) is included in `fme_identities_upload_tool.zip`, which converts CSV files to the required JSON format and handles API calls automatically.

Usage:

Shell

```
python3 fme_bulk_identities.py \
  --csv identities.csv \
  --config config.json \
  [--batch-size 1000] \
  [--dry-run]
```

CSV Format (use friendly column names):

None

```
key,company,short_code,account,region
a3f7b9c2-4d1e-4a8b-9c3f-7e6d5a4b3c2d,Acme Corporation,ACME,ACC-98765,us-west
b8e4c3d7-2a9f-4b7c-8d2e-6f5a3b9c8d1e,TechStart Inc,TCHS,ACC-12345,eu-central
c1d9e5f3-8b2a-4c6d-9e3f-5a7b4c8d2e1f,Global Systems
Ltd,GLSY,ACC-54321,ap-southeast
```

Config Format (JSON):

JSON

```
{
  "api_base_url": "https://api.split.io/internal/api/v2",
  "traffic_type_id": "user",
  "environment_id": "your-env-id",
  "admin_api_key": "your-api-key",
  "attribute_mapping": {
    "company": "company_name",
    "short_code": "company_short_code",
    "account": "account_number",
    "region": "region"
  }
}
```

What the mapping does:

- CSV column `company` → FME attribute `company_name`
- CSV column `short_code` → FME attribute `company_short_code`
- CSV column `account` → FME attribute `account_number`
- CSV column `region` → FME attribute `region`

This allows you to use friendly names in your CSV while mapping to the exact attribute names defined in FME.

The tool will convert your CSV to JSON and call the API endpoint automatically. Use `--dry-run` to preview the API payload without actually uploading.

FAQ

Q: Do I need to upload identity data to every environment?

A: Not required, but recommended for best UX:


- **Production:** Upload real customer data (scrubbed PII)
- **Staging:** Upload realistic test data
- **Development:** Upload small sample dataset

Each environment's identity data is independent.

Q: Can I update identity data after initial upload?

A: Yes, but be aware of how each API endpoint behaves:

Bulk POST endpoint (Save Identities - used by default in helper tool):

-  **REPLACES all attributes** for each key
- Matching keys → **Overwrites all attributes** (clears those not in the request)
- New keys → Adds to identity table
- Missing keys → Not deleted (must delete separately via API)

Single PUT endpoint (Save Identity - used with `--patch-mode` in helper tool):

-  **Preserves unspecified attributes**

- Matching keys → Updates or adds only the attributes provided
 - Existing attributes not in the request → Remain unchanged
 - Best for partial updates without affecting other attributes
-

Q: How often should I upload identity data?

A: Depends on use case:

- **Daily batch:** Upload via helper tool in automated pipeline (scheduled)
 - **Ad-hoc:** Manual upload via helper tool (testing/demos)
 - **Continuous:** Re-upload when identity data changes (automated pipeline)
-

Appendix: Helper Tool Advanced Features

The `fme_bulk_identities.py` utility provides robust functionality for managing identity data uploads:

Upload Modes

Bulk Mode (Default - POST endpoint):

Shell

```
python3 fme_bulk_identities.py --csv identities.csv --config config.json
```

- Uses the **Save Identities** (plural) bulk endpoint
- **OVERWRITES all attributes** for each key in the CSV
- Any attributes not in the CSV will be **deleted from those keys**
- Keys not in the CSV remain unchanged
- **Makes a single API call per batch** (default: 1000 identities per call)
- Fast and efficient for large datasets
- Best for: Complete identity data replacement

Patch Mode (PUT endpoint per identity):

Shell

```
python3 fme_bulk_identities.py --csv identities.csv --config config.json  
--patch-mode
```

- Uses the **Save Identity** (singular) endpoint for each key
- **PRESERVES unspecified attributes**
- Only updates the attributes included in the CSV
- **Makes one API call per identity** (one call per CSV row)
- ⚠️ **Rate limit:** api.split.io endpoints default to 20 calls per 10 seconds
- Use wisely for small datasets or when partial updates are critical
- Best for: Partial updates to existing identity data (small batches)

Options

Batch Size Control:

```
Shell
--batch-size 500
```

- Controls how many identities are sent per API call (bulk mode only)
- Default: 1000
- Adjust if hitting API rate limits in --patch-mode or timeouts in either mode

Dry Run:

```
Shell
--dry-run
```

- Preview what would be uploaded without making actual API calls
- Shows sample payloads and warnings about overwrite behavior
- Always recommended before first real upload

Error Handling

The utility provides:

- **Row validation:** Detects CSV format errors before uploading
- **Transformation errors:** Reports which rows couldn't be mapped
- **API errors:** Shows HTTP errors with response details
- **Progress tracking:** Shows upload progress for large datasets
- **Summary report:** Final counts of successes and failures

Example output:

None


 Loading config from: config.json

 Loading CSV from: identities.csv

 Found 5000 rows in CSV


 Transforming CSV rows to Split identities...

 Successfully transformed 5000 identities

 BULK MODE: Uploading 5000 identities in 5 batch(es)...

 WARNING: Will OVERWRITE all attributes for each key

 Batch 1/5 (1000 identities)...

 Successfully uploaded 1000 identities

...

=====
 UPLOAD SUMMARY
=====

Total rows in CSV: 5000

Valid identities: 5000

Transformation errors: 0

Successfully uploaded: 5000

Failed to upload: 0

When to Use Each Mode

Scenario	Mode	Reason
Initial identity data load	Bulk (default)	Fastest, no existing data to preserve
Complete data refresh	Bulk (default)	Want to remove old attributes
Add new attribute to all keys	Patch (<code>--patch-mode</code>)	Preserve existing attributes
Update subset of attributes	Patch (<code>--patch-mode</code>)	Don't want to delete unspecified attributes
Testing/preview before upload	Either + <code>--dry-run</code>	Verify payload without making changes