hop-client Documentation

SCiMMA

Jul 29, 2020

CONTENTS

CHAPTER

ONE

USER'S GUIDE

1.1 Installation

You can install hop-client either via pip, conda, or from source.

To install with pip:

```
pip install -U hop-client
```

To install with conda, you must use the channel from the SCiMMA Anaconda organization:

```
conda install --channel scimma hop-client
```

To install from source:

```
tar -xzf hop-client-x.y.z.tar.gz
cd hop-client-x.y.z
python setup.py install
```

1.2 Quickstart

1.2.1 Reading messages

The hop client supports a python-based API for reading messages from a stream, as follows:

```
from hop import stream

with stream.open("kafka://hostname:port/topic", "r", format="json") as s:
    for idx, msg in s:
        print(msg)
```

This block will hang forever, listening to new messages and processing them as they arrive. By default, this will only process new messages since the connection was opened. The `start_at` option lets you control where in the stream you can start listening from. For example, if you'd like to listen to all messages stored in a topic, you can do:

```
with stream.open("kafka://hostname:port/topic", "r", format="json", start_at="latest
↪") as s:
    for idx, msg in s:
        print(msg)
```

1.2.2 Writing messages

We can also publish messages to a topic, as follows:

```
from hop import stream

with stream.open("kafka://hostname:port/topic", "w", format="json") as s:
    s.write({"my": "message"})
```

1.2.3 Using the CLI

Publish a GCN

```
hop publish kafka://hostname:port/gcn mygcn.gcn3
```

An example RFC 822 formatted GCN circular (`example.gcn3`) is provided in `tests/data`.

Client configuration properties can be passed to `hop publish` via `-X property=value` or in a configuration file specified by `-F <config-file>`, mimicking the behavior of `kafkacat`. This can be used to connect to a Kafka broker with SSL authentication enabled, for example.

Consume a GCN

```
hop subscribe kafka://hostname:port/gcn mygcn.gcn3 -e
```

Configuration properties can be passed in a manner identical to `hop publish` above.

1.3 Commands

hop-client provides a command line interface for various tasks:

`hop publish`: parse and publish GCN circulars

1.3.1 `hop publish`

This command allows a user to parse an RFC 833 formatted GCN circular and publish JSON-formatted GCNs via Kafka.

1.4 Stream

1.4.1 The Stream Object

The Stream object allows a user to connect to a Kafka broker and read in a variety of alerts, such as GCN circulars. It also allows one to specify default settings used across all streams opened from the Stream instance.

Let's open up a stream and show the Stream object in action:

```
from hop import Stream

stream = Stream(format="json")
with stream.open("kafka://hostname:port/topic", "r") as s:
    for idx, msg in s:
        print(msg)
```

A common use case is to not specify any defaults, so a shorthand is provided for using one:

```
from hop import stream

with stream.open("kafka://hostname:port/topic", "r") as s:
    for _, msg in s:
        print(msg)
```

You can also configure the open stream handle with various options, including a timeout, a progress bar, and a message limit:

```
with stream.open("kafka://hostname:port/topic", "r") as s:
    for _, msg in s(timeout=10, limit=20):
        print(msg)
```

CHAPTER

TWO

API REFERENCE

2.1 hop-client API

2.1.1 hop.cli

2.1.2 hop.io

2.1.3 hop.publish

2.1.4 hop.subscribe

2.1.5 hop.models

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search