alerthub-bulk-tools 利用ガイド

はじめに

本ツール `alerthub-bulk-tools` は、Kompira AlertHub (以下 AlertHub)にリソースを一括で設定登録するためのツールです。

このツールは、ルールとトリガーの一括登録に対応しています。

次のような流れでルールやトリガーをまとめて登録することが出来ます。

1. AlertHub への接続設定を行う
2. AlertHub の画面で、テンプレートとなるルール/トリガーを設定する
3. `download-rule` / `download-trigger` コマンドを実行し、テンプレートとするルール/トリガーの設定をファイルとしてダウンロードする
4. ダウンロードしたファイルを編集して、テンプレートファイルを作成する
5. テンプレートファイルに埋め込むデータをCSVファイルとして作成する
6. `register-rule` / `register-trigger` コマンドを実行し、ルール/トリガーをまとめて登録する

AlertHub への接続設定

本ツールを利用するためには、AlertHub へ接続するためのスペースIDとAPIトークンが必要になります。
デフォルトでは `alethub-bulk-tools` はコマンドを実行する際のカレントディレクトリにある `config.toml` からこれらの情報を読み込むので、コマンドを実行する前にこの `config.toml` を編集し設定してください。

`config.toml` は次のような形式になります。

```
[kcloud]
space = "<スペースID>"
token = "<APIトークン>"

[proxy]
host = "<プロキシサーバアドレス>:<ポート番号>"
user = "<ユーザ名>"
password = "<パスワード>"
```

スペースIDはURL(`https://<スペースID>.cloud.kompira.jp`)から取得します。

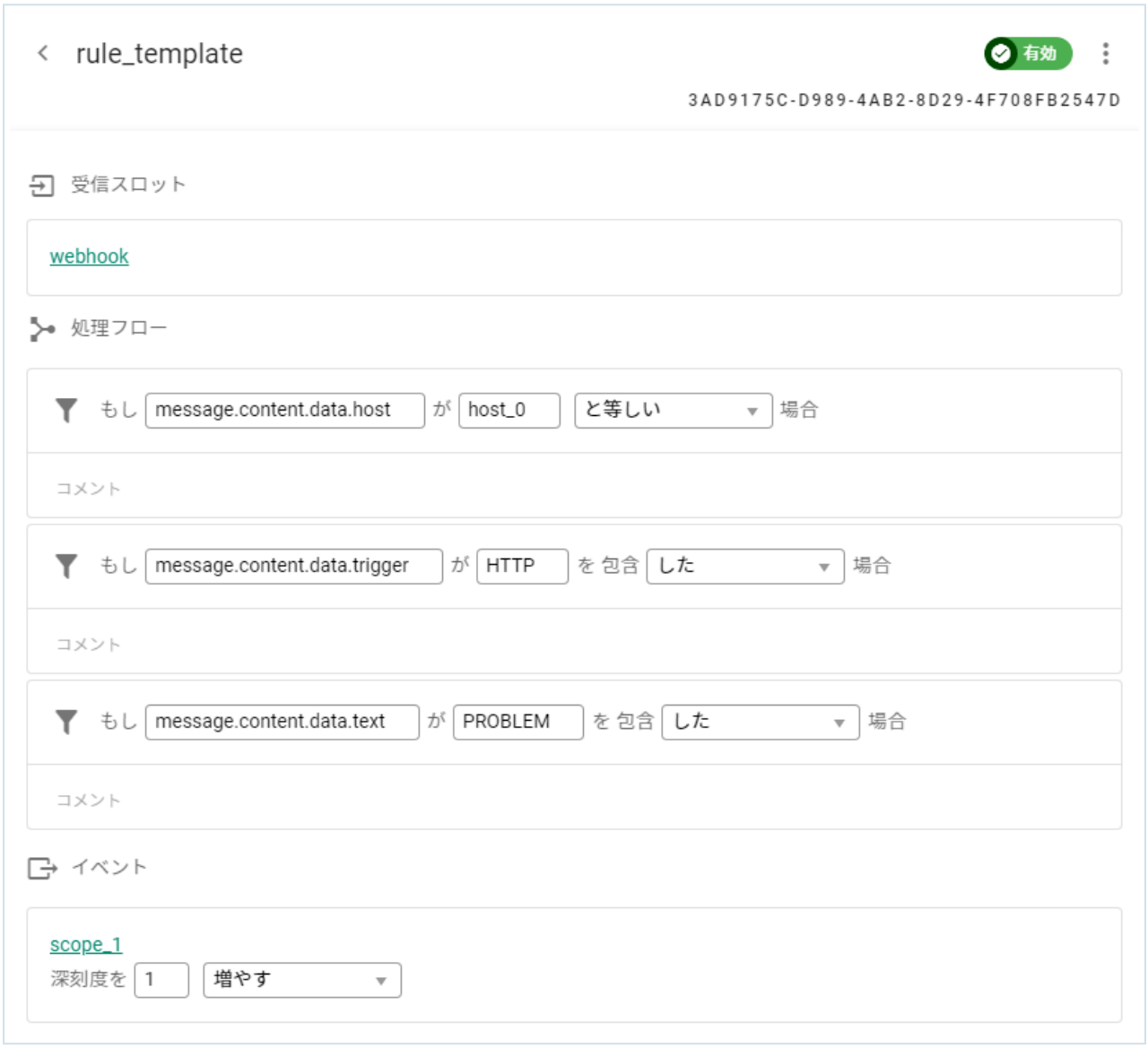
APIトークンはKompiraCloudの設定画面の `API Token` の項から発行してください。

`[proxy]` 以降の設定は、プロキシサーバを経由してKompiraCloudにアクセスする場合に記載してください。
プロキシサーバを経由しない場合は、記載の必要はありません。

ルール / トリガーの事前設定

本ツールでは、既存のルール / トリガーの設定をテンプレートとして一括登録を行うため、
テンプレートの元となるルール / トリガーの設定が必要となります。

ここでは、以降の説明のために以下の様なルールを設定するものとします。



ルール / トリガーのダウンロード

download-rule / download-trigger

ルール/トリガーの設定のダウンロードは、`download-rule` / `download-trigger` コマンドで行います。

それぞれ、コマンドプロンプトで `alerthub-bulk-tools.exe` が配置されているディレクトリに移動し、次のように実行してください。

```
$ alerthub-bulk-tools download-rule <ルールID>
$ alerthub-bulk-tools download-trigger <トリガーID> --scope <スコープID>
```

`<ルールID>` 、 `<トリガーID>` はそれぞれを画面表示した際に右上に表示される値です。
クリックすることで ID がクリップボードへコピーされます。

`download-trigger` の実行には、トリガーIDだけでなくトリガーが属するスコープのIDも必要なことに注意してください。

ダウンロードされた設定は、それぞれ作業ディレクトリ以下のファイル `rule-<ルールID>.toml` / `trigger-<ルールID>.toml` に保存されます。

前項で作成したルールはダウンロードすると以下の様な内容となります。

```
disabled = false
displayName = "rule_template"
receiveSlotId = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"

[eventParams.xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx]
value = 1
method = "increase"

[[flow.steps]]
operatorId = "ae45218d-db7a-4533-9f94-34c2b3c9467f"
comment = ""
parameters.F1 = "message.content.data.host"
parameters.O1 = "equal to"
parameters.S1 = "host_0"

[[flow.steps]]
operatorId = "79a610df-854f-45cf-91b6-5b3597683315"
comment = ""
parameters.F1 = "message.content.data.trigger"
parameters.O1 = "does"
parameters.S1 = "HTTP"

[[flow.steps]]
operatorId = "79a610df-854f-45cf-91b6-5b3597683315"
comment = ""
parameters.F1 = "message.content.data.text"
parameters.O1 = "does"
parameters.S1 = "PROBLEM"
```

テンプレートファイルの作成

テンプレートファイルはTOMLファイルを生成するMustacheテンプレートとして記述します。

`download-rule` / `download-trigger` コマンドによって生成されるTOMLファイルを編集し、補完したい部分を置き換えることでテンプレートファイルを作成できます。

例えば、ルールの登録のためのテンプレートは次のような内容になります。
この例では、「ルールの表示名称」「深刻度を変化させる対象のスコープ」「条件判断に使用するホスト名文字列」「条件判断に使用する監視内容文字列」を組み替えて登録できるようなテンプレートとしています。

```
disabled = false
displayName = "rule_{{Index}}"
receiveSlotId = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"

[eventParams.{{Data.scopeid}}]
value = 1
method = "increase"

[[flow.steps]]
operatorId = "ae45218d-db7a-4533-9f94-34c2b3c9467f"
comment = ""
parameters.F1 = "message.content.data.host"
parameters.O1 = "equal to"
parameters.S1 = "{{Data.hostname}}"

[[flow.steps]]
operatorId = "79a610df-854f-45cf-91b6-5b3597683315"
comment = ""
parameters.F1 = "message.content.data.trigger"
parameters.O1 = "does"
parameters.S1 = "{{Data.trigger}}"

[[flow.steps]]
operatorId = "79a610df-854f-45cf-91b6-5b3597683315"
comment = ""
parameters.F1 = "message.content.data.text"
parameters.O1 = "does"
parameters.S1 = "PROBLEM"
```

`{{` と `}}` で囲まれている部分が補完される部分です。
補完には `Data` と `Index` の２つの変数を使用できます。

変数	説明
Data	データファイルから入力される値、`Data.key` という形で `key` フィールドの値を参照できる
Index	補完に使われているデータのインデックス(1から始まる)

データファイルの作成

データファイルはCSVファイルとして記述します。
データファイルの文字コードは `UTF-8` および `Shift-JIS` が利用できます。

例えば、上のテンプレートの補完に使われるデータファイルは次のような内容になります。

```
scopeid,hostname,trigger
06986c8a-xxxx-xxxx-xxxx-aef88c0beeaf,host_1,HTTP
3c5d129f-xxxx-xxxx-xxxx-c0054362885e,host_2,HTTP
339b4dc3-xxxx-xxxx-xxxx-ca1caa14ceef,host_3,HTTP
3525ee8e-xxxx-xxxx-xxxx-e4025f63e173,host_4,HTTP
21ba4381-xxxx-xxxx-xxxx-ebf7783c12f4,host_5,HTTP
06986c8a-xxxx-xxxx-xxxx-aef88c0beeaf,host_1,PING
3c5d129f-xxxx-xxxx-xxxx-c0054362885e,host_2,PING
339b4dc3-xxxx-xxxx-xxxx-ca1caa14ceef,host_3,PING
3525ee8e-xxxx-xxxx-xxxx-e4025f63e173,host_4,PING
21ba4381-xxxx-xxxx-xxxx-ebf7783c12f4,host_5,PING
```

1行目は各列のデータのフィールドを表します。
テンプレート上でデータにアクセスするために使用するので**必ず付記してください**。

2行目以降の行はリソースの作成に使われるデータとして解釈されます。
各行それぞれのデータによってテンプレートが保管され、1行につき1つのリソースが作成されます。

例として、上のテンプレートを3行目(`3c5d129f` から始まる行)のデータで補完した結果は次のようになります。

```
disabled = false
displayName = "rule_2"
receiveSlotId = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"

[eventParams.3c5d129f-xxxx-xxxx-xxxx-c0054362885e]
value = 1
method = "increase"

[[flow.steps]]
operatorId = "ae45218d-db7a-4533-9f94-34c2b3c9467f"
comment = ""
parameters.F1 = "message.content.data.host"
parameters.O1 = "equal to"
parameters.S1 = "host_2"

[[flow.steps]]
operatorId = "79a610df-854f-45cf-91b6-5b3597683315"
comment = ""
parameters.F1 = "message.content.data.trigger"
parameters.O1 = "does"
parameters.S1 = "HTTP"

[[flow.steps]]
operatorId = "79a610df-854f-45cf-91b6-5b3597683315"
comment = ""
parameters.F1 = "message.content.data.text"
parameters.O1 = "does"
parameters.S1 = "PROBLEM"
```

なお、TOMLファイルのどのフィールドにどういったデータが入るかは、ダウンロードしたTOMLファイルを参考に判断してください。

ルール / トリガーの一括登録

register-rule / register-trigger

ルール/トリガーの登録は、`register-rule` / `register-trigger`コマンドで行います。

それぞれ、コマンドプロンプトで `alerthub-bulk-tools.exe` が配置されているディレクトリに移動し、以下のように実行してください。

データファイルの文字コードが `UTF-8` の場合

```
$ alerthub-bulk-tools register-rule -t <テンプレートファイルへのパス> -d <データファイルへのパス>
$ alerthub-bulk-tools register-trigger -t <テンプレートファイルへのパス> -d <データファイルへのパス>
```

データファイルの文字コードが `Shift-JIS` の場合

```
$ alerthub-bulk-tools register-rule -t <テンプレートファイルへのパス> -d <データファイルへのパス>  --sjis-data
$ alerthub-bulk-tools register-trigger -t <テンプレートファイルへのパス> -d <データファイルへのパス>  --sjis-data
```

作成したデータの件数分以下の出力がされれば実行は完了です。
以下はルールを登録した際の1データ分の出力例です。

```
Render template with <n>th data
POST Request: https://<スペースID>.cloud.kompira.jp/api/apps/alerthub/rules
Register rule: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```