Part 1
st2 コマンドチュートリアル

大山 裕泰 / DMM.com ラボ

2017/11/30 InternetWeek 2017 運用自動化ハンズオン ～StackStormで実践するインフラ運用革命～

免責事項

- 第三者の製品・サービスについて、特定の製造者やサービス提供者につき、製品やサービスを評価するものではありません。
- 当社事例は、あくまで当社事案であり各社のシステム・サービス要件等によって、機能、パフォーマンスその他の面で該当しない場合があります。
- 本プレゼンテーションは当社構築した StackStorm に関する技術者の現時点での感想に基づいています。
- 無断複製・転載を禁じます。

目的

StackStorm の CLI コマンド (st2) の使い方を習得する

実施内容

- Level-1 : ログイン
- Level-2 : get, list による情報収集
- Level-3 : pack の取得
- Level-4 : pack の設定
- Level-5 : アクションの実行
- Level-6 : Rule の記述
- Appendix

Level-1: ログイン

認証機構 (st2auth) に対してユーザ認証を行いアクセストークンを取得する

Level-1: ログイン

Usage:

```
$ st2 login <ユーザ名>
```

e.g.

```
vagrant@st2:~$ st2 login st2admin
Password:
Logged in as st2admin
```

Note:

- 認証を行った結果、アクセストークンが ~/.st2/token-<username> に格納される
  (以降アクセストークンは 24 時間有効)

Level-1: ログイン

課題

‘st2 login’ コマンドで StackStorm にログインしてください

Level-1: ログイン

課題

‘st2 login’ コマンドで StackStorm にログインしてください

解答

```
$ st2 login st2admin
```

Level-2: get, list による情報収集

pack, action, rule などの一覧・詳細情報を取得する

Level-2: get, list による情報収集

Usage:

```
$ st2 <target> {get|list}
```

e.g.

```
vagrant@st2:~$ st2 pack list
+---------+---------+-------------------------------------
| ref     | name    | description
+---------+---------+-------------------------------------
| chatops | chatops | ChatOps integration pack
| core    | core    | Pack containing basic actions.
| default | default | Pack where all the resources which
|         |         | a pack specified get saved.
| linux   | linux   | Generic linux actions
| packs   | packs   | Pack containing pack management fu
| st2     | st2     | StackStorm pack management
+---------+---------+-------------------------------------
vagrant@st2:~$
```

pack	rule	sensor
action	keys	role
trigger	execution	trace
webhook	timer	…

[<target> の一覧]

Level-2: get, list による情報収集

課題

アクション ‘action’ の一覧を取得してください

Level-2: get, list による情報収集

課題

アクション ‘action’ の一覧を取得してください

解答

```
$ st2 action list
```

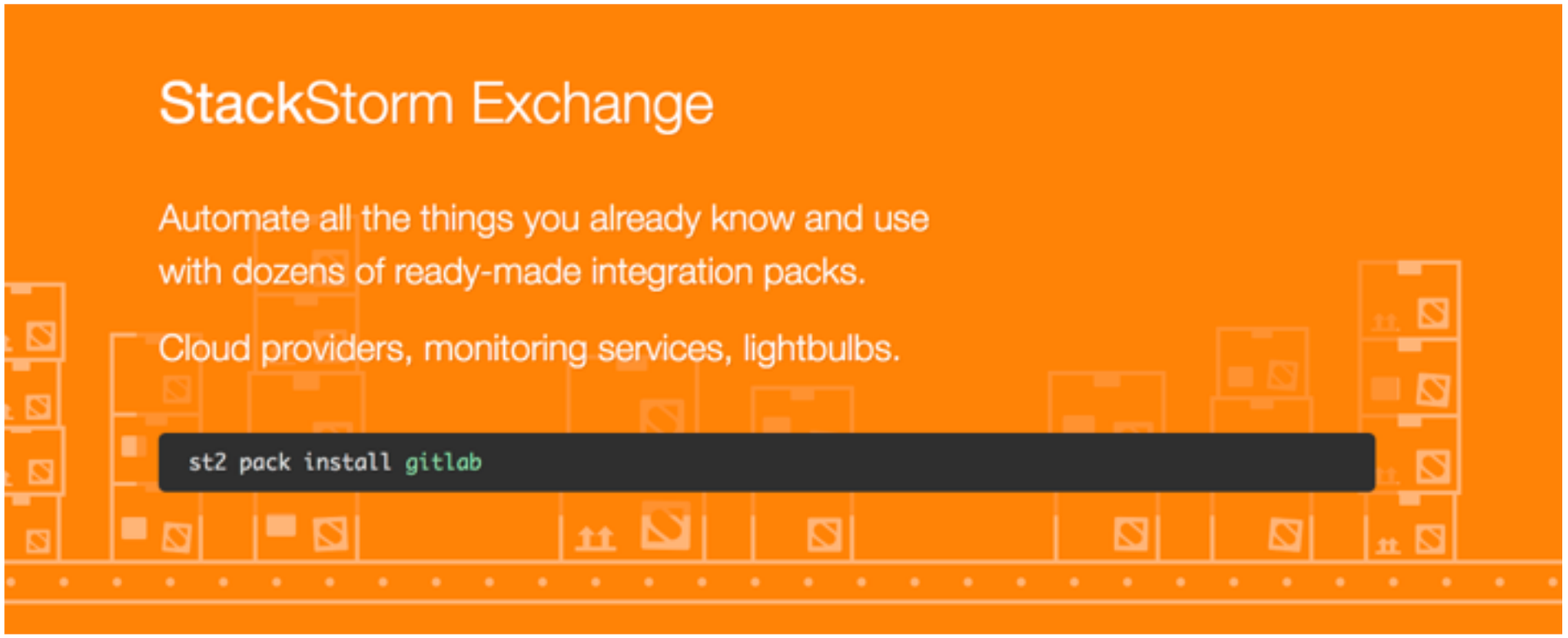
Level-3: pack の取得

Exchange から pack をインストールする

Level-3: pack の取得

StackStorm Exchange

- 100 以上のサードパーティ製の StackStorm pack が登録
- st2 コマンドから任意の pack を取得可能



抜粋：StackStorm Exchange

Level-3: pack の取得

Usage:

```
$ st2 pack install <pack 名>
```

e.g.

```
vagrant@st2:~$ st2 pack install slack

For the "slack" pack, the following content will be registered:

rules     |  0
sensors   |  1
triggers  |  0
actions   |  113
aliases   |  0

Installation may take a while for packs with many items.

        [ succeeded ] download pack
        [ succeeded ] make a prerun
        [ succeeded ] install pack dependencies
        [ succeeded ] register pack

+-------------+-----------------------------------------------------+
| Property    | Value                                               |
+-------------+-----------------------------------------------------+
| name        | slack                                               |
| description | st2 content pack containing slack integrations      |
| version     | 0.6.2                                               |
| author      | StackStorm, Inc.                                    |
+-------------+-----------------------------------------------------+
vagrant@st2:~$
```

Level-3: pack の取得

課題

pack “slack” をインストールしてください

Level-3: pack の取得

課題

pack “slack” をインストールしてください

解答

```
$ st2 pack install slack
```

Level-4: pack の設定

インストールした pack の設定を行う

Level-4: pack 設定

Usage:

```
$ st2 pack config <pack 名>
```

設定スキーマ (config.schema.yaml) で記述されたルールに従い、
インタラクティブな設定入力が行える

e.g.

```
vagrant@st2:~$ st2 pack config sensu
ssl (boolean) [n]:
host: localhost
pass (secret): ******
port (integer) [4567]:
user: admin
---
Do you want to preview the config in an editor before saving? [y]:
---
Do you want me to save it? [y]: y
+----------+-----------------------------+
| Property | Value                       |
+----------+-----------------------------+
| id       | 59f6f6f6bda19007c403722e    |
| pack     | sensu                       |
| values   | {                           |
|          |     "ssl": false,           |
|          |     "host": "localhost",    |
|          |     "pass": "********",     |
|          |     "port": 4567,           |
|          |     "user": "admin"         |
|          | }                           |
+----------+-----------------------------+
vagrant@st2:~$
```

Level-4: pack 設定

課題

pack “slack” の設定を行ってください

設定パラメータ

post_message_action.**username**	bot-<あなたのお名前>
post_message_action.**webhook_url**	<チャットで共有>
post_message_action.**channel**	test-public
admin.**organization**	internetweek-st2-hands-on

(他はデフォルト値を設定してください)

Level-4: pack 設定

課題

pack “slack” の設定を行ってください

解答

```
$ st2 pack config slack
```

設定結果：

```
vagrant@st2:~$ cat /opt/stackstorm/configs/slack.yaml
action_token: null
admin:
  admin_token: ''
  attempts: 1
  auto_join_channels: null
  organization: internetweek-st2
  set_active: true
post_message_action:
  channel: test1
  icon_emoji: ':panda_face:'
  username: bot-HiroyasuOHYAMA
  webhook_url: https://hooks.slack.com/services/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
sensor:
  strip_formatting: false
  token: ''
vagrant@st2:~$
```

Level-5: アクションの実行

pack に登録されたアクションを実行する

Usage:

```
$ st2 run <pack>.<action> [argument …]
```

1. st2 に登録されているアクションの一覧を確認

```
$ st2 action list --pack slack
```

2. アクションのパラメータを確認

```
$ st2 action get slack.post_message
```

3. アクションを実行

```
$ st2 run slack.post_message message='Hello, StackStorm!'
```

課題

slack にメッセージを出力してください

設定パラメータ

出力チャンネル	#test-<Your Name>
出力メッセージ	“level-5 subject clear!”

(他はデフォルト値を設定してください)

課題

slack にメッセージを出力してください

解答

```
$ st2 run slack.post_message \
> channel="#iw2017-st2-handson" \
> message="level-5 subject clear!"
```

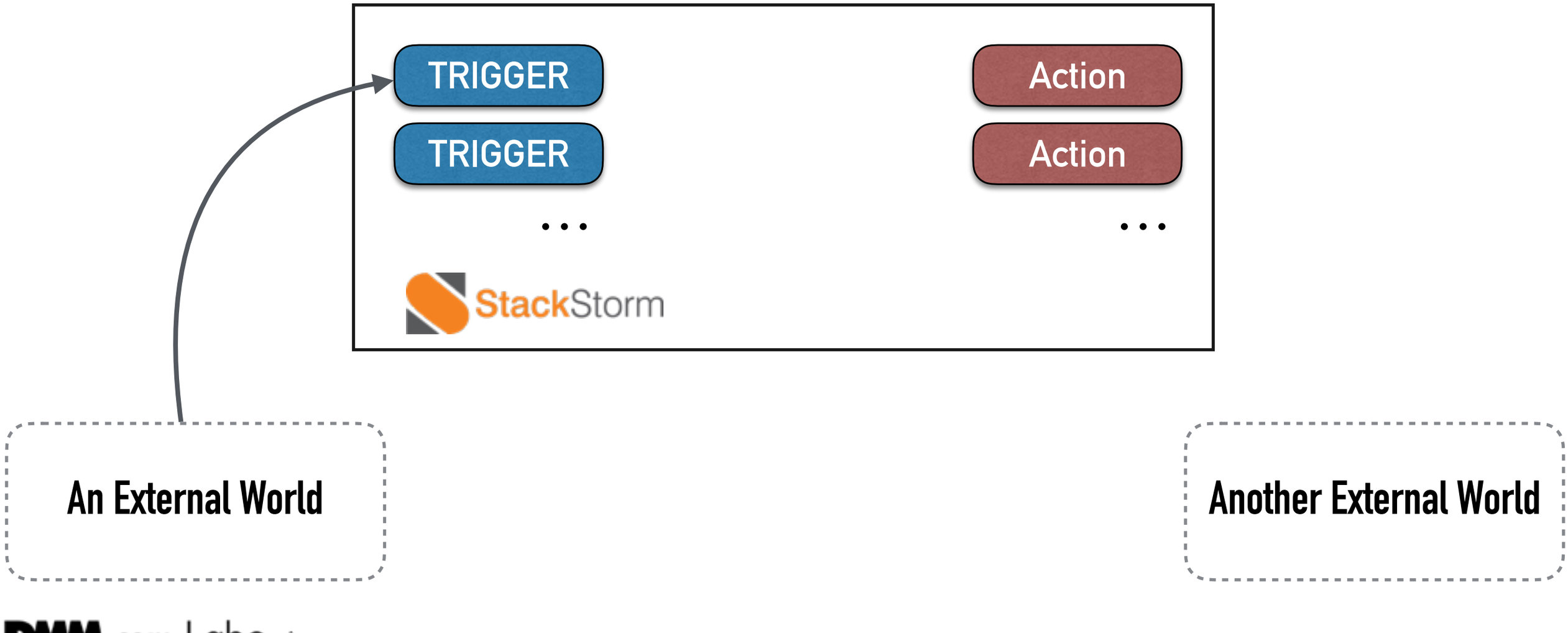

Level-6: Rule の記述

検知したイベントに応じて
実行するアクションを設定する

Level-6: Rule の記述

Rule とは

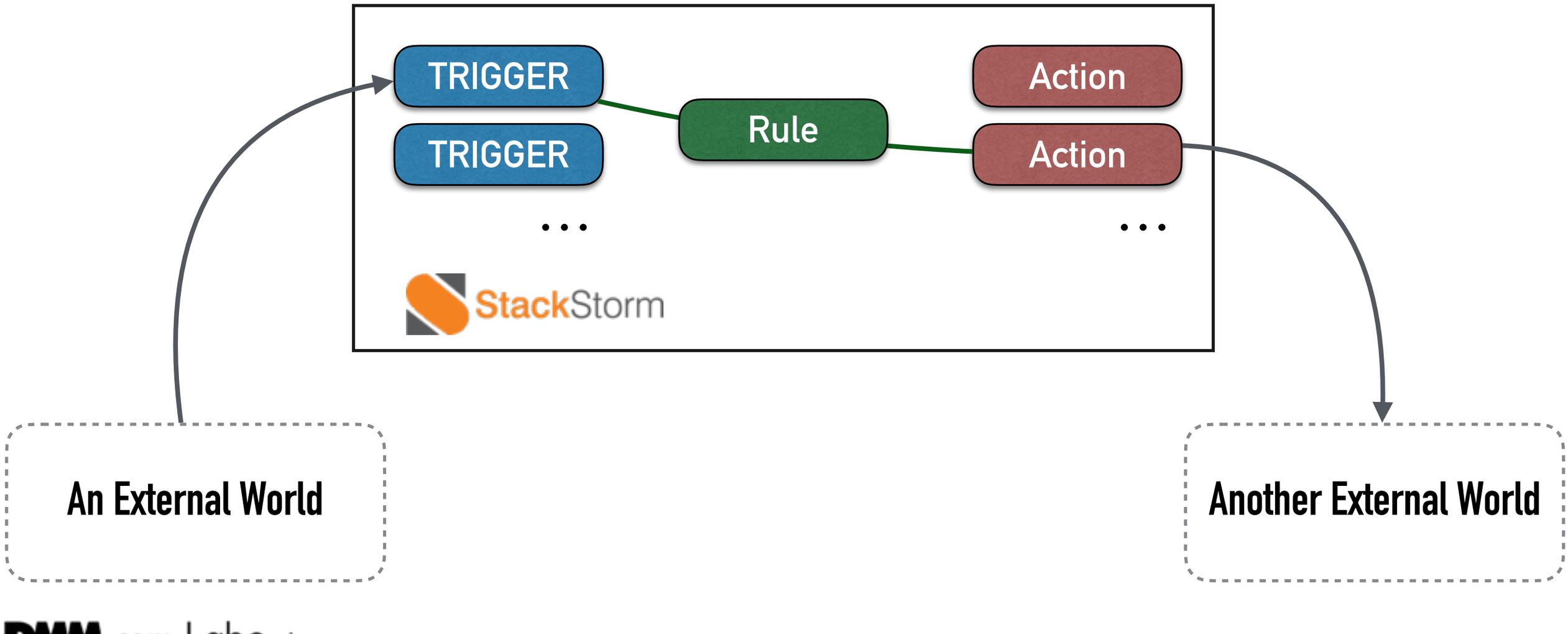
- Trigger と Action を紐付ける
- Action を実行する条件の設定・パラメータの変換を行う



Level-6: Rule の記述

Rule とは

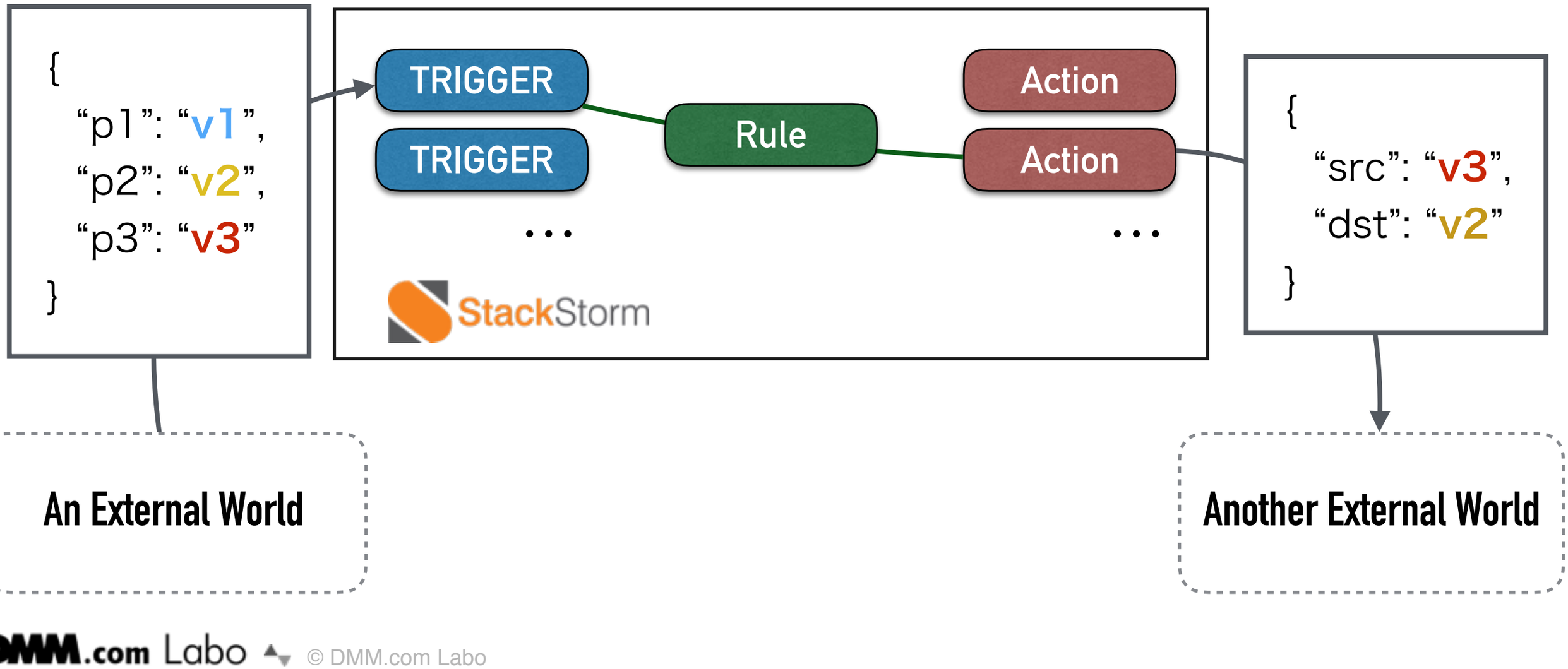
- Trigger と Action を紐付ける
- Action を実行する条件の設定・パラメータの変換を行う



Level-6: Rule の記述

Rule とは

- Trigger と Action を紐付ける
- Action を実行する条件の設定・パラメータの変換を行う



定期処理ルールの設定

st2.core.IntervalTimer を利用し 60s 間隔でアクションを実行

Rule 定義ファイル

Rule を登録

```
vagrant@st2:~$ cat rule_interval.yaml
---
name: "rule_interval"
pack: "example"
description: ""
enabled: true

trigger:
    type: "core.st2.IntervalTimer"
    parameters:
        unit: seconds
        delta: 60

action:
    ref: "slack.post_message"
    parameters:
        message: "Hello: {{ trigger }}"
        channel: test12345
vagrant@st2:~$ st2 rule create rule_interval.yaml
```

ルールの無効化・削除

```
vagrant@st2:~$ st2 rule list
+------------------------+---------+-----------------------------------------------------------------------------+---------+
| ref                    | pack    | description                                                                 | enabled |
+------------------------+---------+-----------------------------------------------------------------------------+---------+
| chatops.notify         | chatops | Notification rule to send results of action executions to stream for chatops | True    |
| example.rule_interval  | example |                                                                             | True    |
| example.rule_webhook   | example |                                                                             | True    |
+------------------------+---------+-----------------------------------------------------------------------------+---------+
vagrant@st2:~$
```

登録されたルールを無効化

```
$ st2 rule disable example.rule_interval
```

登録されたルールを削除

```
$ st2 rule delete example.rule_interval
```

Appendix: WebUI

Appendix: WebUI

GUI からも操作可能

https://< VM の IP アドレス>:8443/

Appendix: WebAPI を作成

Webhook API を作成し

HTTP リクエストをハンドリング

appendix) Webhook を作成

st2.core.webhook を利用 HTTP リクエストをハンドリング

Action の実行条件の記述

new

```
$ cat rule_webhook.yaml
name: "rule_webhook"
pack: "examples"
description: ""
enabled: true

trigger:
  type: "core.st2.webhook"
  parameters:
    url: "iw2017-st2"

criteria:
  trigger.body.name:
    pattern: "admin"
    type: "equals"

action:
  ref: slack.post_message
  parameters:
    message: "Message: {{trigger.body.message}}"
    icon_emoji: ":stuck_out_tongue_closed_eyes:"

$ st2 rule create rule_webhook.yaml
```

ex2) Webhook を作成

st2.core.webhook を利用 HTTP リクエストをハンドリング

```
$ curl -k https://localhost/api/v1/webhooks/iw2017-st2 \
> -H 'Content-Type: application/json' \
> -H ‘X-Auth-Token: <アクセストークン>' \
> -d '{"name": "admin", "message": "Hello, st2!"}'
```

$ cat rule_webhook.yaml

```
    url: "iw2017-st2"

criteria:
  trigger.body.name:
    pattern: "admin"
    type: "equals"

action:
  ref: slack.post_message
  parameters:
    message: "Message: {{trigger.body.message}}"
    icon_emoji: ":stuck_out_tongue_closed_eyes:"

$ st2 rule create rule_webhook.yaml
```

ex2) Webhook を作成

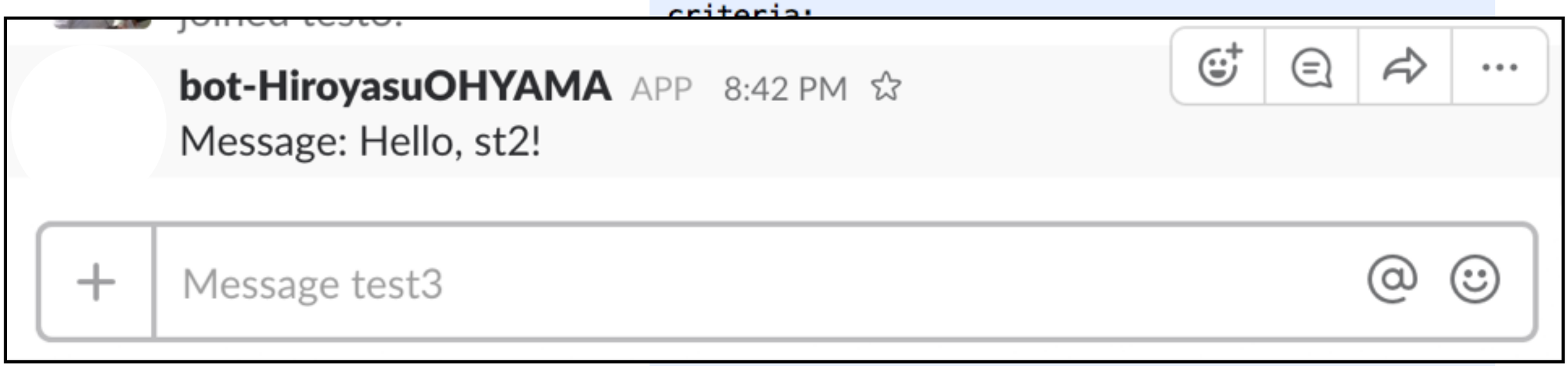
st2.core.webhook を利用 HTTP リクエストをハンドリング

```
$ cat rule_webhook.yaml
```

```
$ curl -k https://localhost/api/v1/webhooks/iw2017-st2 \
> -H 'Content-Type: application/json' \
> -H 'X-Auth-Token: <アクセストークン>' \
> -d '{"name": "admin", "message": "Hello, st2!"}'
```

```
    url: "iw2017-st2"

criteria:
```




```
$ st2 rule create rule_webhook.yaml
```

ex2) Webhook を作成

st2.core.webhook を利用 HTTP リクエストをハンドリング

```
$ cat rule_webhook.yaml
name: "rule_webhook"
pack: "examples"
description: ""
enabled: true

trigger:
  type: "core.st2.webhook"
  parameters:
    url: "iw2017-st2"

criteria:
  trigger.body.name:
```

```
$ curl -k https://localhost/api/v1/webhooks/iw2017-st2 \
> -H 'Content-Type: application/json' \
> -H 'X-Auth-Token: <アクセストークン>' \
> -d '{"name": "admin", "message": "Hello, st2!"}'
```

```
$ st2 rule create rule_webhook.yaml
```

ex2) Webhook を作成

st2.core.webhook を利用 HTTP リクエストをハンドリング

```
$ curl -k https://localhost/api/v1/webhooks/iw2017-st2 \
> -H 'Content-Type: application/json' \
> -H ‘X-Auth-Token: <アクセストークン>' \
> -d '{"name": "admin", "message": "Hello, st2!"}'
```

$ cat rule_webhook.yaml

```
    url: "iw2017-st2"

criteria:
  trigger.body.name:
    pattern: "admin"
    type: "equals"

action:
  ref: slack.post_message
  parameters:
    message: "Message: {{trigger.body.message}}"
    icon_emoji: ":stuck_out_tongue_closed_eyes:"

$ st2 rule create rule_webhook.yaml
```

ex2) Webhook を作成

st2.core.webhook を利用 HTTP リクエストをハンドリング

```
$ curl -k https://localhost/api/v1/webhooks/iw2017-st2 \
> -H 'Content-Type: application/json' \
> -H ‘X-Auth-Token: <アクセストークン>' \
> -d '{"name": "admin", "message": "Hello, st2!"}'
```

```
$ cat rule_webhook.yaml
```

```
    url: "iw2017-st2"

criteria:
  trigger.body.name:
    pattern: "admin"
    type: "equals"

action:
  ref: slack.post_message
  parameters:
    message: "Message: {{trigger.body.message}}"
    icon_emoji: ":stuck_out_tongue_closed_eyes:"
```

```
$ st2 rule create rule_webhook.yaml
```

ex2) Webhook を作成

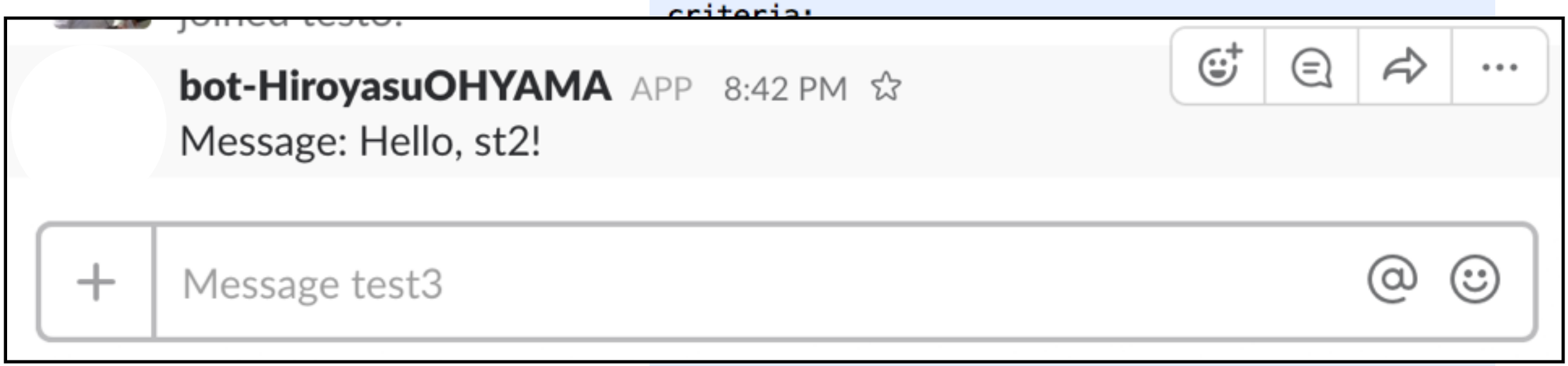
st2.core.webhook を利用 HTTP リクエストをハンドリング

```
$ curl -k https://localhost/api/v1/webhooks/iw2017-st2 \
> -H 'Content-Type: application/json' \
> -H ‘X-Auth-Token: <アクセストークン>' \
> -d '{"name": "admin", "message": "Hello, st2!"}'
```

```
$ cat rule_webhook.yaml
```

```
    url: "iw2017-st2"

criteria:
```




```
$ st2 rule create rule_webhook.yaml
```

まとめ

- 特に利用頻度の高いコマンドの使い方を紹介しました。
- さらに詳しく知りたい方は：**CLI Reference - StackStorm Document** をご参照ください。