
err-stackstorm

Release 2.1.4

Aug 14, 2020

Contents:

1	Getting Started	3
1.1	Overview	3
1.2	Features	3
1.3	Compatibility	4
1.4	Supported Chat Backends	4
2	Quick Start	7
3	Installation	9
3.1	Setting up Errbot's Webserver	9
3.2	Update Errbot's config.py	10
3.3	Installing err-stackstorm	11
4	Configuration	13
4.1	General	14
4.2	ST2 ChatOps Configuration	14
4.3	Authentication	15
4.4	Advanced Options	16
4.5	Locale	17
4.6	Reference	18
5	Authentication	19
5.1	Standalone	19
5.2	Client-Side	20
6	Authorisation	25
6.1	Errbot Access Control List	25
7	Action Aliases	27
7.1	Overview	27
7.2	Creating a simple action alias	27
7.3	Usage	28
7.4	Advanced Formatting	28
8	Troubleshooting	31
8.1	Basic Checks	31
8.2	F.A.Q.	36

9 Project	37
9.1 Community	37
9.2 Contributing	37
10 Indices and tables	39

err-stackstorm is a community project to bring StackStorm ChatOps to Errbot. No commercial support is provided by StackStorm.

- *Overview*
- *Features*
- *Compatibility*
- *Supported Chat Backends*

1.1 Overview

err-stackstorm is a community project to bring StackStorm [ChatOps](#) to [Errbot](#). No commercial support is provided by StackStorm. *err-stackstorm* exposes StackStorm's [Action Aliases](#) to your chat environment, where you and your team can manage aspects of infrastructure and code.

The objectives for *err-stackstorm* project are:

1. Emulate hubot-stackstorm features.
2. Provide a Python friendly ChatOps solution.
3. Respect StackStorm's enterprise offering.
4. Maintain the same high quality as the StackStorm project.
5. Collaborate with the StackStorm community to evolve ChatOps features.

1.2 Features

err-stackstorm communicates directly with the StackStorm API from with an *errbot* instance.

- List action-alias help dynamically. When StackStorm action-aliases are updated, they are immediately available in the err-stackstorm output. Filtering by pack name and keyword can be used when looking for help.
- Access-Control Lists based on any combination of chat username, command and room. ACLs are defined in the errbot configuration file.
- Associate StackStorm user credentials with chat usernames. Client-Side authenticate lets err-stackstorm dynamically map chat user accounts with StackStorm authenticated users. Credentials are passed via an out of band authentication web page.
- Trigger action-aliases directly from the bot.
- Support for multiple chat backends, as provided by errbot.
- Customise plugin prefix.
- Customise StackStorm route key to allow more than one bot to be connected to a single StackStorm instance.
- Docker image available to get up and running quickly and easily.
- Python based using modern 3.6 features.

1.3 Compatibility

The plugin has been developed and tested against the below software combinations. Because you might be running different Python or Errbot versions, the below are the optimal combinations:

Table 1: Ideal Combination of Versions

err-stackstorm	Python	Errbot	Stackstorm Client
2.1	3.6+	6.0.0	not used
2.0	3.4	5.2.0	2.10
1.4	3.4	5.1.2	2.5
1.3	3.4	5.1.2	2.5
1.2	3.4	5.0	2.2
1.1	3.4	4.3	2.2
1.0	2.7	3.x	2.1

1.4 Supported Chat Backends

Errbot provides official support for most major chat back-ends and many more chat back-ends are available through unofficial plugins.

Table 2: Supported Chat Backends

Backend	Mode value	Support Type
Slack	slack	Integrated
Text	text	Integrated
XMPP	xmpp	Integrated
Mattermost	mattermost	Plugin
Rocket Chat	aoikrocketchaterrbot	Plugin
Gitter	gitter	Plugin
Discord	discord	Plugin

Backend support provides a minimum set of operations such as *connect* and *authentication* methods along with ways to *identify* and *send messages* to users/rooms.

Advanced formatting may not be available on all backends since additional code would be required in *err-stackstorm* to translate Stackstorm's Action Aliases *extra* parameter.

Backends that currently support nice (extra) formatting:

- Slack
- XMPP

CHAPTER 2

Quick Start

If you are familiar with Errbot and Stackstorm, this guide will get you up and running in no time. For in-depth information, refer to *Installation* and *Configuration*

1. Enable Errbot's webserver:

```
!repos install https://github.com/tkit/errbot-plugin-webserverconfiguration
```

2. Paste the sample configuration below in Errbot's `config.py` file adjusting the URLs to match your Stackstorm instance and set up one of the authentication methods:

```
STACKSTORM = {
    'auth_url': 'https://your.stackstorm.com/auth/v1',
    'api_url': 'https://your.stackstorm.com/api/v1',
    'stream_url': 'https://your.stackstorm.com/stream/v1',
    'route_key': 'errbot',
    'plugin_prefix': 'st2',
    'verify_cert': True,
    'secrets_store': 'cleartext',
    'api_auth': {
        'user': {
            'name': 'my_username',
            'password': "my_password",
        },
        'token': "<User token>",
        'apikey': '<API Key>'
    },
    'rbac_auth': {
        'standalone': {},
    },
    'timer_update': 900, # Unit: second. Interval to check the user token is still
    ↪ valid.
}
```

3. Install `err-stackstorm`:

```
!repos install https://github.com/nzlosh/err-stackstorm.git
```

4. Install ChatOps pack on Stackstorm and copy [this rule](#) to it.
5. Edit the `chatops/actions/post_message.yaml` file and replace `chatops` with `errbot`.
6. Set up an [action alias](#) on Stackstorm. See `action_alias` for more details.
7. Sending `!st2help` to your bot will list the available Stackstorms's aliases.
8. Aliases can be run like this: `!st2 run date on 192.168.5.1`

Important: `st2chatops` does not have to be running on your StackStorm instance. This plugin *replaces* `st2chatops`.

- *Setting up Errbot's Webserver*
 - *Permanently enabling Errbot's webserver*
 - *Temporarily enabling Errbot's webserver*
- *Update Errbot's config.py*
- *Installing err-stackstorm*

Installation of the err-stackstorm plugin can be performed from within a running Errbot instance. Ensure Errbot is up and running before attempting to install the plugin.

Note: See the Errbot installation documentation [here](#) for instructions on how to setup Errbot with the chat backend of your choice.

Important: These instructions assume a running instance of StackStorm is already in place. See the [official StackStorm documentation](#) for details.

3.1 Setting up Errbot's Webserver

err-stackstorm operates by listening for StackStorm's ChatOps messages and delivers to the chat backend of your choice. In order for StackStorm to be able to reach Errbot, we need to enable Errbot's [webserver plugin](#).

3.1.1 Permanently enabling Errbot's webserver

In order to permanently enable the webserver plugin, send the following command to Errbot:

```
!repos install https://github.com/tkit/errbot-plugin-webserverconfiguration
```

Warning: In production environments it is recommended to place a reverse-proxy like [nginx](#) or [Apache](#) in front of Errbot.

3.1.2 Temporarily enabling Errbot's webserver

If you don't want to permanently enable the webserver or perhaps simply wants to run a quick test, send the following command to Errbot:

```
!plugin config Webserver {'HOST': '0.0.0.0', 'PORT': 3141, 'SSL': {'enabled': False,  
↪ 'host': '0.0.0.0', 'port': 3142, 'certificate': '', 'key': ''}}
```

The variables must be adjusted to match the operating environment in which Errbot is running. See [Errbot documentation](#) for further configuration information.

Note: The command above is only applied for the current runtime and may not persist after the errbot process is restarted.

3.2 Update Errbot's config.py

Paste the sample configuration below in Errbot's `config.py` file adjusting the URLs to match your Stackstorm instance and set up one of the authentication methods.

```
STACKSTORM = {  
    'auth_url': 'https://your.stackstorm.com/auth/v1',  
    'api_url': 'https://your.stackstorm.com/api/v1',  
    'stream_url': 'https://your.stackstorm.com/stream/v1',  
    'route_key': 'errbot',  
    'plugin_prefix': 'st2',  
    'verify_cert': True,  
    'secrets_store': 'cleartext',  
    'api_auth': {  
        'user': {  
            'name': 'my_username',  
            'password': "my_password",  
        },  
        'token': "<User token>",  
        'apikey': '<API Key>'  
    },  
    'rbac_auth': {  
        'standalone': {},  
    },  
    'timer_update': 900, # Unit: second. Interval to check the user token is still_  
↪ valid.  
}
```

See [Configuration](#) for in-depth explanation.

3.3 Installing err-stackstorm

Confirm Errbot is configured to install plugin dependencies. The below line should be present in Errbot's *config.py*:

```
AUTOINSTALL_DEPS = True
```

This line ensures that Errbot's will attempt to automatically install the requirements of any plugin you may ask it to install.

Now install err-stackstorm:

```
!repos install https://github.com/nzlosh/err-stackstorm.git
```

The plugin will fail to install if any errors are encountered. This is often caused by configuration errors in Errbot's *config.py*.

You can confirm that is installed by typing:

```
!repos list
```

The list should contain the `nzlosh/err-stackstorm` repo.

- *General*
- *ST2 ChatOps Configuration*
- *Authentication*
 - *Username/Password*
 - *User Token*
 - *API Key*
 - *Secrets Store*
 - * *Cleartext*
- *Advanced Options*
 - *Route Key*
 - * *Example*
 - *Plugin Prefix*
 - * *Example*
 - *Session TTL*
 - *User Token TTL*
- *Locale*
- *Reference*

4.1 General

err-stackstorm configuration is found in Errbot's *config.py* file.

Note: If you followed the Errbot setup documentation this file will have been created by downloading a template. If this file has not already been created, please create it following the [Errbot's instructions](#).

Here's a sample *err-stackstorm* configuration:

```
STACKSTORM = {
    "auth_url": "https://your.stackstorm.com/auth/v1",
    "api_url": "https://your.stackstorm.com/api/v1",
    "stream_url": "https://your.stackstorm.com/stream/v1",
    "route_key": "errbot",
    "plugin_prefix": "st2",
    "verify_cert": True,
    "secrets_store": "cleartext",
    "session_ttl": 3600,
    "user_token_ttl": 86400,
    "api_auth": {
        "user": {
            "name": "my_username",
            "password": "my_password",
        },
        "token": "<User token>",
        "apikey": "<API Key>"
    },
    "rbac_auth": {
        "standalone": {},
    },
    "timer_update": 900, # Unit: second. Interval to check the user token is still_
    ↪ valid.
}
```

4.2 ST2 ChatOps Configuration

StackStorm's [ChatOps pack](#) has to be installed and a notify rule file added to the pack.

The notify rule must be placed in `/<st2>/packs/chatops/rules`. You can find the rule file necessary for *err-stackstorm* [here](#). Just copy-and-paste this file and save it as `notify_errbot.yaml`.

Edit the `/<st2>/packs/chatops/actions/post_message.yaml` file and replace *chatops* with *errbot*:

```
route:
    default: "errbot"
```

Note: See Route Key below for information on customising the route.

4.3 Authentication

err-stackstorm must have valid credentials to use StackStorm's API. Stackstorm's authentication is possible with:

- Username and password
- User token
- API key

See [Stackstorm's Authentication docs](#) for more details on how to generate credentials.

4.3.1 Username/Password

Using a username and password will allow *err-stackstorm* to renew the user token once it expires.

Note: If you specify both username/password and a User Token, the User Token will be used and the *username/password* will be ignored.

4.3.2 User Token

To avoid using the username/password pair in a configuration file, it's possible to supply a User Token as generated by StackStorm when a username/password is authenticated successfully.

If using an User Token *err-stackstorm* will no longer have access to the Stackstorm's API once it expires.

Note: When the token expires, a new one must be generated and updated in `config.py` which in turn requires Errbot to be restarted. This form of authentication is the least practical for production environments.

4.3.3 API Key

API Key support has been included since StackStorm v2.0.

When an API Key is provided, all other authentication settings are ignored and the API Key is used.

If using an API Key, *err-stackstorm* will no longer have access to the Stackstorm's API once it expires.

Note: It is considered a mistake to supply a token or username/password pair when using the API Key.

4.3.4 Secrets Store

The secrets store is used by *err-stackstorm* to cache StackStorm API credentials. The available backends are:

- cleartext

Cleartext

The cleartext store maintains the cache in memory and does not encrypt the contents to disk. It **does not** protect the stored secrets in memory.

4.4 Advanced Options

4.4.1 Route Key

The route key is used by err-stackstorm to inform StackStorm where to send result notifications for action-aliases. StackStorm sends notification events via the stream interface that are marked with the route key. Err-stackstorm filters these events using the route key and will handle any events that match its configured route key.

By altering the route key, it is possible to have multiple instances of err-stackstorm that are connected to the same StackStorm instance. This would allow for configurations where StackStorm is available on multiple chat backends.

Example

A StackStorm instance has 2 err-stackstorm instances. The first err-stackstorm instance uses the *errbot-slack* route key while the second instance uses the *errbot-discord* route key. Both instances have the same plugin prefix and expose the same action-aliases. This would mean the command *!st2 pack list* could be run on discord and the result notification would be routed to the *errbot-discord* err-stackstorm instance.

4.4.2 Plugin Prefix

By default the plugin prefix is set to *st2*. The plugin prefix serves to prevent action-aliases collisions with errbot's native plugins commands. It is possible to customise the plugin prefix to use a more appropriate naming scheme for the environment err-stackstorm is running.

Aside from cosmetic value, customising the plugin prefix can allow for multiple err-stackstorm instances to occupy the same chat channel. This would be achieved by setting a unique plugin-prefix per instance.

Note: Always use strings that do not conflict with existing errbot commands.

Example

In the case of multiple instances of a StackStorm and err-stackstorm pair, say 1 per data centre or 1 per region, it would be possible to assign a unique plugin prefix per instance. Let's say there were 2 pairs with the plugin-prefix of *dc1* and *dc2*. Both bots could occupy *#automation* channel and users could trigger the pack list action-alias in data centre #1 by calling *!dc1 pack list* or get available action-aliases from data centre #2 by calling *!dc2help*.

4.4.3 Session TTL

The session time to live is an internal timeout for err-stackstorm. It is used to set the maximum lifetime an err-stackstorm session is permitted to exist. Once the session expires the user will need to re-authenticate before being able to execute action-alias commands. An err-stackstorm session ttl should be equal to or less than the user token ttl.

When session time to live expires, err-stackstorm will report to the user that the session is no longer valid and they should reauthenticate.

4.4.4 User Token TTL

The user token ttl is used to set the maximum life time a StackStorm User Token will be permitted to exist. Once the user token has expired the user will need to re-authenticate before being able to execute action-alias commands.

When a user token time to live expires, err-stackstorm will report it as an error communicating with the StackStorm API. It would be more user-friendly to ensure the session ttl expires before the user token ttl.

Note: The user token ttl *must* **be** equal to or lower than the StackStorm API <https://docs.stackstorm.com/authentication.html?highlight=ttl#usage>. By default StackStorm's token ttl is set to 24 hours, but the value can be increased through *st2.conf*. If user_token_ttl is greater than the StackStorm API token ttl value, err-stackstorm will fail to fetch a valid API token and not function correctly.

4.5 Locale

Errbot uses the system's locale for handling text. If you're getting unicode errors like this:

```
UnicodeEncodeError: 'ascii' codec can't encode character '\xe9' in position 83:
↳ ordinal not in range(128)
```

Make sure the systems locale is configured for unicode encoding. In the example below, the machine has been set to English (en) New Zealand (NZ) with utf-8 encoding (.UTF8).

```
# locale
LANG=en_NZ.UTF8
LANGUAGE=
LC_CTYPE="en_NZ.UTF8"
LC_NUMERIC="en_NZ.UTF8"
LC_TIME="en_NZ.UTF8"
LC_COLLATE="en_NZ.UTF8"
LC_MONETARY="en_NZ.UTF8"
LC_MESSAGES="en_NZ.UTF8"
LC_PAPER="en_NZ.UTF8"
LC_NAME="en_NZ.UTF8"
LC_ADDRESS="en_NZ.UTF8"
LC_TELEPHONE="en_NZ.UTF8"
LC_MEASUREMENT="en_NZ.UTF8"
LC_IDENTIFICATION="en_NZ.UTF8"
LC_ALL=en_NZ.UTF8
```

4.6 Reference

Option	Description
auth_url	StackStorm's authentication url end point. Used to authenticate credentials against StackStorm.
api_url	StackStorm's API url end point. Used to execute action aliases received from the chat back-end.
stream_url	StackStorm's Stream url end point. Used to received ChatOps notifications.
verify_cert	Default is <i>True</i> . Verify the SSL certificate is valid when using https end points. Applies to all end points.
route_key	Default is <i>errbot</i> . The name of the route to bot will listen for and submit action-alias executions with.
plugin_prefix	Default is <i>st2</i> . Text used to prefix action-alias commands with to avoid name collisions between StackStorm Action-Aliases and Errbot plugin commands.
api_auth.user.name	Errbot's username to authenticate with StackStorm.
api_auth.user.password	Errbot's password to authenticate with StackStorm.
api_auth.token	Errbot's user token to authenticate with StackStorm. Used instead of a username/password pair.
api_auth.apikey	Errbot API key to authenticate with StackStorm. Used instead of a username/password pair or user token.
timer_update	Unit: seconds. Default: 60. Interval for err-stackstorm to the user token is valid.
rbac_auth.standalone	Standalone authentication.
rbac_auth.clientside	Clientside authentication, a chat user will supply StackStorm credentials to err-stackstorm via an authentication page.
rbac_auth.clientside.url	Url to the authentication web page.
session_ttl	Unit: seconds. Default: 3600. The time to live for a authentication session.
user_token_ttl	Unit: seconds. Default: 86400. The time to live for a StackStorm user token.
secrets_store.cleartext	Use the in-memory store.

- *Standalone*
 - *Configuration*
- *Client-Side*
 - *Configuration*
 - * *NGINX*
 - * *err-stackstorm configuration*
 - * *Authenticating*

Important: Do not configure multiple authentication methods at the same time.

5.1 Standalone

This is the default authentication method where *err-stackstorm* uses its own credentials for all calls to the StackStorm API. All action-aliases issued by chat service users execute the underlying workflows with *err-stackstorm* credentials. Any Role Based Access Control policies can only be applied to the bot user which gives limited control.

5.1.1 Configuration

An empty dictionary in the *standalone* key is all that is required to maintain *err-stackstorm*'s default authentication method.

```
"rbac_auth": {  
  "standalone": {}  
}
```

5.2 Client-Side

Note: This implementation is specific to err-stackstorm.

err-stackstorm provides a way to associate the chat service user account with a StackStorm username/password or API token. Once a chat user is associated with their StackStorm credentials, any action-alias will be executed using the associated StackStorm credentials.

This method allows for finer control as StackStorm's Role Based Access Control can be defined per-user.

Chat users create a new authentication session with *err-stackstorm* by calling *session_start* with a secret word, it can be anything the user wishes. A **UUID** is generated for the session and the chat user is invited to follow a URL to an authentication page hosted by Errbot.

Important: For security reasons, the UUID is used only once and is consumed when the page is accessed. Any subsequent attempts to access the authentication page using the same link will result in an error.

The login page must be protected by TLS encryption and ideally require an SSL client certificate. The login page should not be exposed directly to the internet, but have a reverse proxy such as nginx placed between it and any external service consumers.

The user enters their StackStorm credentials via the login page which *err-stackstorm* will validate against the StackStorm API. If the credentials are valid, the user token or api key will be cached by *err-stackstorm* and the supplied credentials discarded.

5.2.1 Configuration

To configure the Client-Side authentication method, one needs to take steps to setup both Nginx and Errbot. Nginx is used to serve static web content for the authentication web page and Errbot serves as the API backend for authentication calls.

NGINX

Note: This example is provided as a guide only. You are expected to know and understand how to configure nginx for your environment. It is outside of the scope of this documentation to go over nginx's configuration and explain SSL certificates.

Important: This example does not show how to secure access using SSL certificates. It is **highly recommended** to use SSL for production environments.

To help understand the example below, the following conditions are assumed:

- the nginx server is running *on the same host* as errbot.
- the host's fully qualified domain name is `my_host.my_fqdn`.
- nginx listens to standard SSL-enabled port: 443.
- Errbot's webserver listens on the ip address 127.0.0.1 TCP port 3141 without ssl enabled.
- Errbot's plugins directory is `/data/errbot/plugins`
- `err-stackstorm` is installed in `/data/errbot/plugins/nzlosh/err-stackstorm`.
- The SSL certificate and private key used by nginx are called `errbot.crt` and `errbot.key`.

```
server {
    listen      my_host.my_fqdn:443 ssl;
    server_name my_host.my_fqdn;

    ssl on;

    ssl_certificate      /etc/ssl/errbot.crt;
    ssl_certificate_key  /etc/ssl/errbot.key;
    ssl_session_cache    shared:SSL:10m;
    ssl_session_timeout  5m;
    ssl_protocols        TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers           EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH:ECDHE-RSA-
↪AES128-GCM-SHA384:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA128:DHE-RSA-
↪AES128-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-GCM-SHA128:ECDHE-RSA-
↪AES128-SHA384:ECDHE-RSA-AES128-SHA128:ECDSA-RSA-AES128-SHA:ECDSA-RSA-AES128-SHA:DHE-
↪RSA-AES128-SHA128:DHE-RSA-AES128-SHA128:DHE-RSA-AES128-SHA:DHE-RSA-AES128-SHA:ECDSA-
↪RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA384:AES128-GCM-SHA128:AES128-
↪SHA128:AES128-SHA128:AES128-SHA:AES128-SHA:DES-CBC3-SHA:HIGH:!aNULL:!eNULL:!EXPORT:!
↪DES:!MD5:!PSK:!RC4;
    ssl_prefer_server_ciphers on;

    index index.html index.htm;

    access_log      /var/log/nginx/ssl-errbot.access.log combined;
    error_log       /var/log/nginx/ssl-errbot.error.log;

    add_header      Front-End-Https on;
    add_header      X-Content-Type-Options nosniff;

    location /login/ {
        proxy_pass      http://127.0.0.1:3141$request_uri;
        proxy_read_timeout  90;
        proxy_connect_timeout 90;
        proxy_redirect    off;

        proxy_set_header  Host my_host.my_fqdn;
        proxy_set_header  X-Real-IP $remote_addr;
        proxy_set_header  X-Forwarded-For $proxy_add_x_forwarded_for;

        proxy_set_header  Connection '';
        chunked_transfer_encoding off;
        proxy_buffering off;
        proxy_cache off;
        proxy_set_header  Host my_host.my_fqdn;
    }
}
```

(continues on next page)

(continued from previous page)

```
location / {  
    root      /data/errbot/plugins/nzlosh/err-stackstorm/html/;  
    index     index.html index.htm;  
}
```

After successfully setting up nginx, the client side authentication url would be `https://my_host.my_fqdn:443/`.

err-stackstorm configuration

A url is required to correctly configure client-side authentication for ChatOps. This URL is Errbot's authentication endpoint that you have just set up.

```
"rbac_auth": {  
    "clientside": {  
        "url": "https://<hostname>:<port>/"  
    }  
},
```

Authenticating

Once the client side authentication is setup, you should be able to trigger the authentication process with `!session_start my_secret_word` which will return a url to complete the login processes. This is how the page looks like:

Err-stackstorm Authentication



Associate your
StackStorm token/key with your chat
account.

▼ Username/password

▶ User Token

▶ API Key

Association failed

Waiting for user credential
submission.

- *Errbot Access Control List*

6.1 Errbot Access Control List

Errbot comes with native Access Control List support. It can be configured to constrain command execution by grouping `command`, `channel` and `user`. Glob patterns can be used in each field to provide flexibility in ACL definitions.

As an example, a StackStorm instance has an automatic package upgrade workflow. Its progress can be viewed by executing the action alias: `apu stats <role>`, which is defined as shown below:

```
| action_ref      | st2_apu.apu_status |
| formats        | [                  |
|                | {                  |
|                |     "representation": [
|                |         "apu status {{role}}"
|                |     ],
|                |     "display": "apu status <role>"
|                | }                  |
|                | ]                  |
```

The Errbot ACL configuration below allows `@user1` to view the status of the upgrade, but *not to start/stop* the upgrade process, which are other action-aliases that are triggered with `st2 apu ...`)

```
ACL_SQUAD_INFRA = ["@admin1", "@admin2", "@admin3", "@admin4"]
ACL_APU_USERS = ['@user1']
ACL_EVERYONE = ["*"]
ACCESS_CONTROLS = {
    'whoami': {
        'allowrooms': ['@bot_user'],
```

(continues on next page)

(continued from previous page)

```
    'allowusers': ACL_EVERYONE
  },
  'st2 apu status*':{
    'allowrooms': ['#channel'],
    'allowusers': ACL_SQUAD_INFRA + ACL_APU_USERS
  },
  'st2 apu*':{
    'allowrooms': ['#channel'],
    'allowusers': ACL_SQUAD_INFRA
  },
}
```

Getting the correct usernames to fill into `allowusers` or `denyusers` isn't always obvious. Use `errbot's !whoami` command to get the correct username for use within ACL definitions. The *nick* value is what should be used in the configuration in the case of Slack.

Warning: UI interface names do not always match with the internal nickname/username. `!whoami` is a surefire way of retrieving the correct username.

On a small scale it's possible to use the `!whoami` command to get the correct user account name but for large installations it'd make more sense to use pre-defined patterns.

Action Aliases

Note: This section assumes you have little to no familiarity with Stackstorm. If that's not the case, you may skip it.

7.1 Overview

[Action Aliases](#) is a Stackstorm feature that helps *exposing actions to the bot*. Their main purpose is to provide a *simple human readable* representation of actions, very useful in text-based interfaces, notable ChatOps.

Generally speaking, you can expose any action to the bot. You can list them using `st2 action-alias list`.

7.2 Creating a simple action alias

Let's create a simple action alias to demonstrate *err-stackstorm*. Aliases are deployed via packs so let's create one quickly:

```
mkdir -p /<st2path>/packs/errtest/aliases
```

Create a file inside the `aliases` folder named `run_remote.yaml` and paste the yaml defined below:

```
---
name: "local_shell_cmd"
action_ref: "core.local"
description: "Execute a command on a remote host via SSH."
formats:
  - "run {{cmd}}"
result:
  format: "operation completed {~} {{ execution.result }}"
```

After that, ask Stackstorm to reload its configuration:

```
st2ctl reload --register-all
```

Note: Errbot will automatically refresh its action alias list when you add or remove aliases on Stackstorm. Type `!st2help` to your bot on the chat to list available StackStorm commands.

In this example `local_shell_cmd` is an alias for the `core.local` action. If you want to run a command against a remote host, you could have used the `core.remote` action.

The supported format for the alias is specified in the `formats` field. A single alias can support multiple formats for the same action. The result will then be returned to *err-stackstorm* and Errbot will propagate that back to your chat backend.

7.3 Usage

Once the alias is setup, talk to your bot and type:

```
!st2 run date
```

The bot will answer your request using the `result.format` definition:

```
operation completed {~} Sun Jul 7 02:08:58 PDT 2019
```

7.4 Advanced Formatting

See also:

Don't forget to check ST2's official documentation on [Action Aliases](#).

Here's an example of an Alias that runs a command on a list of remote hosts and outputs the results nicely formatted on **Slack**.

```
---
name: "remote_shell_cmd"
action_ref: "core.remote"
description: "Execute a command on a remote host via SSH."
formats:
  - "run {{cmd}} on {{hosts}}"
result:
  format: |
    Ran command *{{execution.parameters.cmd}}* on *{{ execution.result | length }}*
    ↳hosts.

    {% for host in execution.result %}
    \*Host\*: {{host}} {{ ":white_check_mark:" if execution.result[host].stdout
    ↳else "x:" }}
    \*\*\*{{ execution.result[host].stdout or execution.result[host].stderr or "No
    ↳result"}}\*\*\*
    {% endfor %}
```

The alias above will format the execution output per host once it gathers the results. The sheer amount backticks and escaping are due to particularities between Errbot and Slack - this may not work in another chat backend.

This is how the bot will answer you on Slack:

Ran command *date* on 2 hosts.

Host: 192.168.0.150 

Sun Jul 7 02:08:58 PDT 2019

Host: 192.168.0.151 

Sun Jul 7 02:08:59 PDT 2019

Important: Advanced formatting may not be available to all chat backends since each backend requires specific code to translate ST2's *extra* parameter.

- *Basic Checks*
 - *Is the Errbot process running?*
 - *Is the Errbot webhook listening?*
 - *Is the Errbot machine able to communicate with the StackStorm end points?*
 - *Are the Errbot authentication credentials for StackStorm correct?*
 - * *Username/password*
 - * *User Token*
 - * *API Key*
 - *Is Errbot connected correctly to the chat back-end?*
 - *Is the StackStorm ChatOps pack installed and configured correctly?*
 - *Are events being sent via the StackStorm Stream?*
 - *Are the events seen in the errbot logs using errbot as their route?*
- *F.A.Q.*
 - *running errbot returns command not found*
 - *st2chatops is not running*

8.1 Basic Checks

8.1.1 Is the Errbot process running?

Check if an instance of Errbot is running on the host:

```
# ps faux | grep errbo[t]
root      158707  0.1  0.0 2922228 59640 pts/21  Sl+   Aug14   2:29   |   \_ /opt/errbot/
↳ bin/python3 /opt/errbot/bin/errbot -c /data/errbot/etc/config.py
```

8.1.2 Is the Errbot webhook listening?

Check Errbot's internal web server is listening on the correct interface. Use the PID from the previous command to filter the output to find which port the errbot process is listening on. You can use `ss`:

```
# ss -tlnp | grep 158707
LISTEN    0      128          *:3141         :*              users: (("errbot",158707,21))
```

or `Netstat`:

```
# netstat -tlnp | grep 158707
tcp        0      0 0.0.0.0:3141         0.0.0.0:*       LISTEN      ↳ 158707/python3
```

If the result is empty, errbot is not listening on any ports. Refer to the Webserver section in [Installation](#).

8.1.3 Is the Errbot machine able to communicate with the StackStorm end points?

From the errbot machine perform a `curl` to the StackStorm endpoint:

```
curl http://<stackstorm_host>/api/v1/rules
```

8.1.4 Are the Errbot authentication credentials for StackStorm correct?

Here's you can test that the authentication is working.

Username/password

A successful username/password authentication is shown below:

```
$ st2 auth errbot
Password:
+-----+-----+
| Property | Value |
+-----+-----+
| user     | errbot |
| token    | 10342978da134ae5bbb7dc94d2ba9c08 |
| expiry   | 2017-09-29T14:31:20.799212Z |
+-----+-----+
```

If the username and password are valid and correctly entered in errbot's configuration file, errbot will be authorised to interact with StackStorm's API/Stream end points.

User Token

You can test the user token from the configuration using the `st2` command and passing the token with `-t` argument.

Important: Make sure no environment variables are set that could provide a valid token or api key already.

```
$ st2 action-alias list -t 10342978da134ae5bbb7dc94d2ba9c08
```

ref	enabled	pack	description
packs.pack_get	True	packs	Get information about installed StackStorm pack.
packs.pack_install	True	packs	Install/upgrade StackStorm packs.
packs.pack_search	True	packs	Search for packs in StackStorm Exchange and other directories.
packs.pack_show	True	packs	Show information about the pack

If a list of action aliases are shown, the token is valid.

API Key

Confirm the api key has been created and still registered with StackStorm by using it with the st2 command:

```
$ st2 apikey list --api-key ZzVk3DEBZ4FiZmMEmDBkM2x5ZmM5jWZkZWZjZjZmMZEWYzQwZD2iYzUyM2RhYTkTNMYmNDYNODIOOTYwMzE20A
```

id	user	metadata
586e6deadbeef66deadbeef6	errbot	{u'used_by': u'errbot api access'}

8.1.5 Is Errbot connected correctly to the chat back-end?

How to test if the bot is connected to the chat back-end is dependant on the back-end. The simplest way is to send a message to the bot user account requesting the built-in help.

For example, using a slack client the following command would be used `/msg @bot_name !help`. The bot should respond with its help text:

```
bot [11:01 AM]
_All commands_

*Backup*
_Backup related commands._
• *.backup* - Backup everything.
*ChatRoom*
```

(continues on next page)

(continued from previous page)

```

_This is a basic implementation of a chatroom_
• *.room join* - Join (creating it first if needed) a chatroom.
• *.room occupants* - List the occupants in a given chatroom.
• *.room invite* - Invite one or more people into a chatroom.
• *.room topic* - Get or set the topic for a room.

```

8.1.6 Is the StackStorm ChatOps pack installed and configured correctly?

err-stackstorm requires the ChatOps pack to be installed. To confirm it is installed, use the `st2 cli`.

```

$ st2 pack list
+-----+-----+-----+-----+
| ref          | name          | description          | version |
+-----+-----+-----+-----+
| chatops      | chatops       | ChatOps integration pack | 0.2.0  |
+-----+-----+-----+-----+

```

Confirm the `notify_errbot.yaml` is inside the `chatops/rules` directory:

```
$ cat /opt/stackstorm/packs/chatops/rules/notify_errbot.yaml
```

You should see a YAML like the one below:

```

---
name: "notify-errbot"
pack: "chatops"
enabled: true
description: "Notification rule to send results of action executions to stream for_
↳ chatops"
trigger:
  type: "core.st2.generic.notifytrigger"
criteria:
  trigger.route:
    pattern: "errbot"
    type: "equals"
action:
  ref: chatops.post_result
  parameters:
    channel: "{{ trigger.data.source_channel }}"
    user: "{{ trigger.data.user }}"
    execution_id: "{{ trigger.execution_id }}"

```

The rule should be available using command `st2 rule get chatops.notify-errbot`

```

+-----+-----+
| Property | Value |
+-----+-----+
| id       | 5a6b1abc5b3a0f0f5bcd54e7 |
| uid      | rule:chatops:notify-errbot |
| ref      | chatops.notify-errbot |
+-----+-----+

```

(continues on next page)

(continued from previous page)

pack	chatops	
name	notify-errbot	
description	Notification rule to send results of action executions to stream for chatops	
enabled	True	
action	{	
	"ref": "chatops.post_result",	
	"parameters": {	
	"user": "{{trigger.data.user}}",	
	"execution_id": "{{trigger.execution_id}}",	
	"channel": "{{trigger.data.source_channel}}"	
	}	
	}	
criteria	{	
	"trigger.route": {	
	"pattern": "errbot",	
	"type": "equals"	
	}	
	}	
tags		
trigger	{	
	"type": "core.st2.generic.notifytrigger",	
	"ref": "core.st2.generic.notifytrigger",	
	"parameters": {}	
	}	
type	{	
	"ref": "standard",	
	"parameters": {}	
	}	

8.1.7 Are events being sent via the StackStorm Stream?

From the errbot host connect to the StackStorm stream endpoint and watch for events emitted as actions are executed by StackStorm:

```
curl -s -v -H 'Accept: text/event-stream' -H 'X-Auth-Token: 10342978da134ae5bbb7dc94d2ba9c08' http://<stackstorm_host>/stream/v1/stream
```

The correct URL will depend on your StackStorm installation, the URL must corresponds to <https://api.stackstorm.com/stream/v1/stream/>

8.1.8 Are the events seen in the errbot logs using errbot as their route?

To see the events in the log, the debug level `BOT_LOG_LEVEL = logging.DEBUG` will need to be added to errbot's configuration file `config.py`.

If events are configured correctly, logs will be shown like this (`st2.announcement__errbot`):

```
17:04:12 DEBUG    root          Dispatching st2.announcement__errbot_
↳event, 990 bytes...
17:04:12 DEBUG    lib.st2pluginapi      *** Errbot announcement event detected!
↳***
st2.announcement__errbot event, 990 bytes
```

If the announcement event is showing as:

```
2018-01-26 15:51:55,246 DEBUG      sseclient      Dispatching st2.  
↪announcement__chatops event, 508 bytes...
```

This indicates that the route wasn't set to errbot, refer to *Configuration*.

8.2 F.A.Q.

8.2.1 running errbot returns command not found

If Errbot was installed in a python virtual environment, make sure the virtual environment is activated correctly.

8.2.2 st2chatops is not running

This is okay, err-stackstorm **replaces** st2chatops.

9.1 Community

You can find users of *err-stackstorm* on:

- [Gitter](#)
- [Slack](#) on #err-stackstorm channel.

There's also the comprehensive *Troubleshooting* section to help guide you through possible break/fix scenarios.

9.2 Contributing

If you think you've found a bug or need a new feature [open an issue](#) on the [github repository](#).

If you want to contribute to the *err-stackstorm* project, there are plenty of improvements to be made, contact [nzlosh](#) via chat or email to discuss how you can get involved.

CHAPTER 10

Indices and tables

- `genindex`
- `modindex`
- `search`