auton Documentation

Release 0.2.23

Adrien Delle Cave

Jan 20, 2023

Contents

auton is a free and open-source, we develop it to run programs and command-lines on remote servers through HTTP protocol. There are two programs, auton for client side and autond for server side. auton is just a helper to transform command-lines into HTTP protocol, it is able to transform basic arguments, file arguments and environment variables. For example, you can use auton from CI/CD to run on remote servers, you just need to configure your endpoints:

- ansible
- curl
- terraform

You can also use auton if you need to execute a new version of a software but you can't install it on a legacy server or tests programs execution.

CHAPTER 1

Quickstart

Using autond in Docker

```
docker-compose up -d
```

See docker-compose.yml

CHAPTER 2

Installation

2.1 autond for server side

```
pip install autond
```

2.2 auton for client side

```
pip install auton
```

CHAPTER 3

Environment variables

3.1 autond

Variable	Description	Default
AUTOND_CONFIG	Configuration file contents(e.g. `export AUTOND_CONFIG="$(cat auton.yml)"`)	
AUTOND_LOGFILE	Log file path	/var/log/autond/daemon.log
AUTOND_PIDFILE	autond pid file path	/run/auton/autond.pid
AUTON_GROUP	auton group	auton or root
AUTON_USER	auton user	auton or root

3.2 auton

Variable	Description	Default
AUTON_AUTH_USER	user for authentication	
AUTON_AUTH_PASSWD	password for authentication	
AUTON_ENDPOINT	name of endpoint	
AUTON_LOGFILE	Log file path	/var/log/auton/auton.log
AUTON_NO_RETURN_CODE	Do not exit with return code if present	False
AUTON_UID	auton job uid	random uuid
AUTON_URI	autond URI(s)(e.g. http://auton-01.example.org:8666,http://auton-02.example.org:8666)	

CHAPTER 4

Autond configuration

See configuration example etc/auton/auton.yml.example

4.1 Endpoints

In this example, we declared three endpoints: ansible-playbook-ssh, ansible-playbook-http, curl. They used subproc plugin.

```
endpoints:
  ansible-playbook-ssh:
    plugin: subproc
    config:
      prog: ansible-playbook
      timeout: 3600
      args:
        - '/etc/ansible/playbooks/ssh-install.yml'
        - '--tags'
        - 'sshd'
      become:
        enabled: true
      env:
        DISPLAY_SKIPPED_HOSTS: 'false'
  ansible-playbook-http:
    plugin: subproc
    config:
      prog: ansible-playbook
      timeout: 3600
      args:
        - '/etc/ansible/playbooks/http-install.yml'
        - '--tags'
        - 'httpd'
      become:
        enabled: true
```

(continues on next page)

(continued from previous page)

```
      env:
        DISPLAY_SKIPPED_HOSTS: 'false'
  curl:
    plugin: subproc
    config:
      prog: curl
      timeout: 3600
```

4.2 Authentication

To enable authentication, you must add `auth_basic` and `auth_basic_file` lines in section `general`:

```
auth_basic:      'Restricted'
auth_basic_file: '/etc/auton/auton.passwd'
```

Use `htpasswd` to generate `auth_basic_file`:

```
htpasswd -c -s /etc/auton/auton.passwd foo
```

And you have to add for each modules route `auth: true`:

```
modules:
  job:
    routes:
      run:
        handler:   'job_run'
        regexp:    '^run/(?P<endpoint>[^\/]+)/(?P<id>[a-z0-9][a-z0-9\-]{7,63})$'
        safe_init: true
        auth:      true
        op:        'POST'
      status:
        handler:   'job_status'
        regexp:    '^status/(?P<endpoint>[^\/]+)/(?P<id>[a-z0-9][a-z0-9\-]{7,63})$'
        auth:      true
        op:        'GET'
```

Use section `users` to specify users allowed by endpoint:

```
ansible-playbook-ssh:
  plugin: subproc
  users:
    maintainer: true
    bob: true
  config:
    prog: ansible-playbook
    timeout: 3600
    args:
      - '/etc/ansible/playbooks/ssh-install.yml'
      - '--tags'
      - 'sshd'
    become:
      enabled: true
    env:
      DISPLAY_SKIPPED_HOSTS: 'false'
```

4.3 Plugin subproc

subproc plugin executes programs with python `subprocess`.

Predefined AUTON environment variables during execution:

Variable	Description
AUTON	Mark the job is executed in AUTON environment
AUTON_JOB_TIME	Current time in local time zone
AUTON_JOB_GMTIME	Current time in GMT
AUTON_JOB_UID	Current job uid passed from client
AUTON_JOB_UUID	Unique ID of the current job

Use keyword `prog` to specify program path:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
```

Use keyword `workdir` to change the working directory:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      workdir: somedir/
```

Use keyword `search_paths` to specify paths to search `prog`:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      search_paths:
        - /usr/local/bin
        - /usr/bin
        - /bin
```

Use section `become` to execute with an other user:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      become:
        enabled: true
        user: foo
```

Use keyword `timeout` to raise an exception after n seconds (default: 60 seconds):

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      timeout: 3600
```

Use section `args` to define arguments always present:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      args:
        - '-s'
        - '-4'
```

Use keyword `disallow-args` to disable arguments from client:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      args:
        - '-vvv'
        - 'https://example.com'
      disallow-args: true
```

Use section `argfiles` to define arguments files always present:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      argfiles:
        - arg: '--key'
          filepath: /tmp/private_key
        - arg: '-d@'
          filepath: /tmp/data
```

Use keyword `disallow-argfiles` to disable arguments files from client:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      argfiles:
        - arg: '--key'
          filepath: /tmp/private_key
        - arg: '-d@'
          filepath: /tmp/data
      disallow-argfiles: true
```

Use section `env` to define environment variables always present:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      env:
        HTTP_PROXY: http://proxy.example.com:3128/
        HTTPS_PROXY: http://proxy.example.com:3128/
```

Use keyword `disallow-env` to disable environment variables from client:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      env:
        HTTP_PROXY: http://proxy.example.com:3128/
        HTTPS_PROXY: http://proxy.example.com:3128/
      disallow-env: true
```

Use section `envfiles` to define environment variables files always present:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      envfiles:
        - somedir/foo.env
        - somedir/bar.env
```

Use keyword `disallow-envfiles` to disable environment files from client:

```
endpoints:
  curl:
    plugin: subproc
    config:
      prog: curl
      envfiles:
        - somedir/foo.env
        - somedir/bar.env
      disallow-envfiles: true
```

CHAPTER 5

Auton command-lines

5.1 endpoint curl examples:

Get URL https://example.com:

```
auton --endpoint curl --uri http://localhost:8666 -a 'https://example.com'
```

Get URL https://example.com with auton authentication:

```
auton --endpoint curl --uri http://localhost:8666 --auth-user foo
--auth-passwd bar -a 'https://example.com'
```

Add environment variable HTTP_PROXY:

```
auton --endpoint curl --uri http://localhost:8666 -a 'https://example.com' -e
'HTTP_PROXY=http://proxy.example.com:3128/'
```

Import already declared environment variable with argument –imp-env:

```
HTTPS_PROXY=http://proxy.example.com:3128/ auton --endpoint curl --uri http://
localhost:8666 -a 'https://example.com' --imp-env HTTPS_PROXY
```

Load environment variables from local files:

```
auton --endpoint curl --uri http://localhost:8666 -a 'https://example.com'
--load-envfile foo.env
```

Tell to autond to load environment variables files from its local fs:

```
auton --endpoint curl --uri http://localhost:8666 -a 'https://example.com'
--envfile /etc/auton/auton.env
```

Add multiple autond URIs for high availability:

```
auton --endpoint curl --uri http://localhost:8666 --uri http://localhost:8667
-a 'https://example.com'
```

Add arguments files to send local files:

```
auton --endpoint curl --uri http://localhost:8666 -A '--cacert=cacert.pem' -a
'https://example.com'
```

Add multiple arguments:

```
auton --endpoint curl --uri http://localhost:8666 --multi-args '-vvv -u
foo:bar https://example.com' --multi-argsfiles '-d@=somedir/foo.txt -d@=bar.
txt --cacert=cacert.pem'
```

Get file contents from stdin with -:

```
cat foo.txt | auton --endpoint curl --uri http://localhost:8666 --multi-args
'-vvv -u foo:bar sftp://example.com' --multi-argsfiles '--key=private_key.pem
--pubkey=public_key.pem -T=-'
```