caso Documentation

Release

Spanish National Research Council (CSIC)

February 24, 2017

Contents

1 Installation 3

2 Configuration 5

3 Usage 9

cASO is a pluggable extractor of Cloud Accounting Usage Records from an OpenStack installation. cASO gets usage information from nova or ceilometer APIs and can generate valid output for Apel SSM or logstash.

Contents:

CHAPTER 1

Installation

Pre-requisites

If you are planning to use `cASO` for generating accounting records for EGI, you will need a valid APEL/SSM configuration. Follow the documentation available at the EGI FedCloud wiki

Installation

The best way to install cASO and have the most up to date version is using the repositories and packages provided in the EGI AppDB:

https://appdb.egi.eu/store/software/caso

Manual installation

At the command line:

```
$ pip install caso
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv caso
$ pip install caso
```

CentOS 6

On CentOS 6, you can use Software Collections to install Python 2.7:

```
$ yum -y install centos-release-SCL
$ yum -y install python27
```

There are also some dependencies of the packages used by `cASO` that need to be installed (`gcc`, `libffi-devel` and `openssl-devel`):

```
$ yum -y install gcc libffi-devel openssl-devel
```

You can then install `pip` for that version of Python and use that to install `cASO`:

```
$ scl enable python27 bash
$ easy_install-2.7 pip
$ pip install caso
$ exit    # this terminates bash with the SCL python2.7
```

In this case you can later on use `caso-extract` with the following command line:

```
$ scl enable python27 caso-extract
```

Alternatively, if you want to use a virtualenv:

```
$ scl enable python27 bash
$ virtualenv caso
$ . caso/bin/activate
$ pip install caso
$ exit    # this terminates bash with the SCL python2.7
```

Running from the virtualenv:

```
$ scl enable python27 caso/bin/caso-extract
```

CHAPTER 2

Configuration

OpenStack Configuration

Publishing benchmark information

Starting with the V0.4 of the accounting record it is possible to publish benchmark information. In order to do so, you need to add this information to the flavor properties and configure caso to retrieve this information. There are two different values that need to be added to the flavor:

- The benchmark name, indicated with the `benchmark_name` flavor property.
- The benchmark value, indicated with the `benchmark_value` flavor property.

So, if you are using HEPSPEC06 and the benchmark value is `99` for the flavor `m1.foo` you should set this as follows:

```
openstack flavor set --property benchmark_name="HEPSPEC06" --property benchkark_value=99 m1.foo
```

Using different keys

If you do not want to use the default flavor properties `benchmark_name` and `benchkark_value` (for example because you are using different benchmark types and values) you can specify which properties `cASO` should look for by using the `benchmark_name_key` `benchkark_value_key` in the configuration file.

User credentials

The user configured in the previous section has to be a member of each of the tenants (another option is to convert that user in an administrator, but the former option is a safer approach) for which it is extracting the accounting. Otherwise, `cASO` will not be able to get the usages and will fail:

```
openstack role create accounting
openstack user create --password <password> accounting
# For each of the tenants, add the user with the accounting role
openstack role add --user accounting --project <project> accounting
```

Also, this user needs access to Keystone so as to extract the users information.

- If you are using the V2 identity API, you have to give admin rights to the `accounting` user, editing the `/etc/keystone/policy.json` file and replacing the line:

```
    "admin_required": "role:admin or is_admin:1 or",
```

with:

```
"admin_required": "role:admin or is_admin:1 or role:accounting",
```

- If you are using the V3 identity API you can grant the user just the rights for listing the users adding the appropriate rules in the `/etc/keystone/policy.json`.

cASO configuration

`cASO` uses a config file (default at `/etc/caso/caso.conf`) with several sections. A sample file is available at `etc/caso/caso.conf.sample`.

`[DEFAULT]` section

The `[DEFAULT]` section configures the basic behavior of `cASO`. The sample config file (`/etc/caso/caso.conf.sample`) includes a description of every option. You should check at least the following options:

- `extractor` (default value: `nova`), specifies which extractor to use for getting the data. The following APIs are supported: `ceilometer` and `nova`. Both should generate equivalent information.
- `site_name` (default value: `<None>`). Name of the site as defined in GOCDB.
- `service_name` (default value: `$site_name`). Name of the service within a site. This is used if you have several endpoints within your site.
- `tenants` (list value, default empty). List of the tenants to extract records from.
- `messengers` (list, default: `caso.messenger.noop.NoopMessenger`). List of the messengers to publish data to. Valid messenges are:
  - `caso.messenger.ssm.SSMMessengerV02` for publishing APEL V0.2 records.
  - `caso.messenger.ssm.SSMMessengerV04` for publishing APEL V0.4 records.
  - `caso.messenger.logstash.LogstashMessenger` for publishing to Logstash.
- `mapping_file` (default: `/etc/caso/voms.json`). File containing the mapping from VOs to local tenants as configured in Keystone-VOMS, in the form:

```
{
    "VO": {
        "tenants": ["foo", "bar"],
    }
}
```

- `benchmark_name_key` and `benchmark_value_key`. These two configuration options are used by `cASO` to retrieve the benchmark information form the OpenStack flavors.

`[keystone_auth]` section

This section is used to specify the authentication credentials to be used to connect to the OpenStack APIs. cASO leverages the OpenStack keystoneauth library for authentication, so that it is possible to use any authentication plugin that is available there (so starting on version 1.0 of cASO it is possible to use the Keystone V3 API).

Important: You need to specify the `auth_type` that you want to use (normally `v3password` is a good choice.

For an exhaustive list of available plugins please refer to the keystoneauth documentation.

`[ssm]` section

Options defined here configure the SSM messenger. There is only one option at the moment:

- `output_path` (default: `/var/spool/apel/outgoing/openstack`), directory to put the generated SSM records. APEL/SSM should be configured to take records from that directory.

`[logstash]` section

Options defined here configure the logstash messenger. Available options:

- `host` (default: `localhost`), host of Logstash server.
- `port` (default: `5000`), Logstash server port.

CHAPTER 3

Usage

command line

cASO provides the caso-extract command to generate new records from your OpenStack deployment. caso-extract -h will show a complete list of available arguments.

Use the --extract_from argument to specify the date from when the records should be extracted. If no value is set, then cASO will extract the records from the last run. If equal to "None", then extract records from the beggining of time. If not time zone is specified, UTC will be used.

Important: If you are running an OpenStack Nova version lower than Kilo there is a bug in its API, making impossible to paginate over deleted results.

Since nova is limiting the results to 1000 by default, if you are expecting more than 1000 results you will get just the last 1000. This is important if you are publishing data for the first time, or if you are republishing all your accounting). If this is your case, adjust the osapi_max_limit to a larger value in /etc/nova/nova.conf.

Running as a cron job

The best way of running cASO is via a cron job like the following:

```
10 * * * * caso-extract
```

Migration from OSSSM

If you had a previous installation of osssm, you can migrate to cASO following these steps:

1. Remove the previous osssm installation (e.g. remove apel-ssm-openstack rpm).
2. Remove any cron jobs related to osssm.extract or osssm.push, a single cron job as described above is enough. You should keep the cron job that executes ssmsend, this is still needed to send the records to the accounting database.