cerberus Documentation

Release 0.1

Eurogiciel

July 28, 2015

Contents

1 Introduction **1**

2 Installation **3**
2.1 Cerberus installation and configuration 3

3 Architecture **5**
3.1 Cerberus' Architecture 5

4 API References **9**
4.1 Cerberus REST API (root) 9
4.2 Cerberus REST API (v1) 11

5 Plugin development **21**
5.1 Writing plugins 21

6 Indices and tables **23**

HTTP Routing Table **25**

CHAPTER 1

Introduction

Cerberus is a Security As A Service project aimed at integrating security tools inside Openstack.

Cerberus offers a framework to integrate **security components** (scanners of vulnerabilities, behavior analysis, IPS, IDS, SIEM) in order to propagate changes of the platform to them and to collect security reports and security alarms.

CHAPTER 2

Installation

2.1 Cerberus installation and configuration

2.1.1 Install from source

There is no release of Cerberus as of now, the installation can be done from the git repository.

Retrieve and install Cerberus :

```
git clone git://git.openstack.org/stackforge/cerberus
cd cerberus
python setup.py install
```

This procedure installs the `cerberus` python library and a few executables:

- `cerberus-api`: API service
- `cerberus-agent`: Task management service

Install a sample configuration file :

```
mkdir /etc/cerberus
cp etc/cerberus/cerberus.conf.sample /etc/cerberus/cerberus.conf
```

2.1.2 Configure Cerberus

Edit `/etc/cerberus/cerberus.conf` to configure Cerberus.

The following shows the basic configuration items:

```
[DEFAULT]
verbose = True
log_dir = /var/log/cerberus

rabbit_host = RABBIT_HOST
rabbit_userid = openstack
rabbit_password = RABBIT_PASSWORD

[auth]
username = cerberus
password = CERBERUS_PASSWORD
tenant = service
region = RegionOne
```

```
url = http://localhost:5000/v2.0

[keystone_authtoken]
username = cerberus
password = CERBERUS_PASSWORD
project_name = service
region = RegionOne
auth_url = http://localhost:5000/v2.0
auth_plugin = password

[database]
connection = mysql://cerberus:CERBERUS_DBPASS@localhost/cerberus
```

2.1.3 Setup the database and storage backend

MySQL/MariaDB is the recommended database engine. To setup the database, use the `mysql` client:

```
mysql -uroot -p << EOF
CREATE DATABASE cerberus;
GRANT ALL PRIVILEGES ON cerberus.* TO 'cerberus'@'localhost' IDENTIFIED BY 'CERBERUS_DBPASS';
EOF
```

Run the database synchronisation scripts:

```
cerberus-dbsync upgrade
```

Init the storage backend:

```
cerberus-storage-init
```

2.1.4 Setup Keystone

Cerberus uses Keystone for authentication.

To integrate Cerberus to Keystone, run the following commands (as OpenStack administrator):

```
keystone user-create --name cerberus --pass CERBERUS_PASS
keystone user-role-add --user cerberus --role admin --tenant service
```

Create the `Security` service and its endpoints:

```
keystone service-create --name Cerberus --type security
keystone endpoint-create --service-id SECURITY_SERVICE_ID \
    --publicurl http://localhost:8300 \
    --adminurl http://localhost:8300 \
    --internalurl http://localhost:8300
```

2.1.5 Start Cerberus

Start the API and processing services :

```
cerberus-api --config-file /etc/cerberus/cerberus.conf
cerberus-agent --config-file /etc/cerberus/cerberus.conf
```

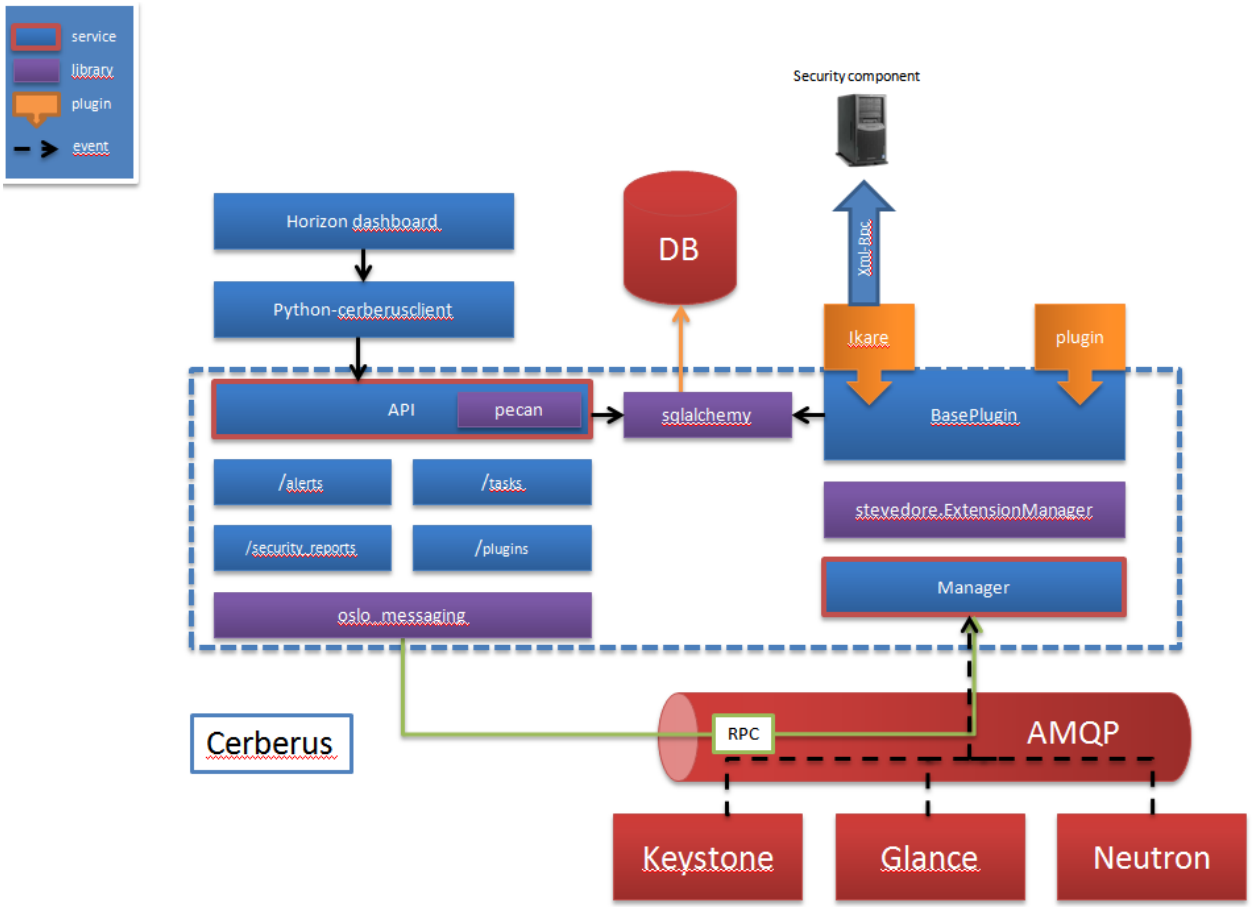
CHAPTER 3

Architecture

3.1 Cerberus' Architecture

Cerberus can be cut in two big parts:

- API
- Manager



3.1.1 Cerberus' API

The API is a REST server documented later.

3.1.2 Cerberus' manager

Cerberus is easy to extend thanks to a plugin system.

The manager has some functions:

- it loads `plugins`
- it manages `tasks`
- it stores `security reports` and `security alarms` in database

3.1.3 Plugins

Plugins are created to communicate with a particular security component. They are defined by their:

- unique identifier (uuid)
- name
- version
- provider
- type (scanner, SIEM...)

Plugins can subscribe to events sent on the notification topic Cerberus' manager listens on. For example, this can be useful to automatically configure a tool if a project has been created or if a certain role is granted to an user. Plugins may also implement some functions that the manager calls through `tasks`.

3.1.4 Tasks

Cerberus manages tasks. In order to create a task, you need to call the Cerberus' API by passing some information:

- The name of the task
- The plugin uuid handling the task
- The method to call on this plugin
- The type (periodic or not, default is not)
- The period if the task is periodic (for now, period is in seconds only)
- Persistent (True/False, conditional): tell Cerberus you want this task to be

stored in database (useful if the manager handling the task is shut down)

The tasks may be stopped/started. As such, they have a state (running or not).

3.1.5 Security reports

Cerberus stores security reports provided by the security components. These security reports have a predefined schema and Cerberus stores the following information:

- The uuid of the security report

- The uuid of the plugin
- The report identifier
- The Openstack's component identifier (e.g: an instance id, a network id)
- The component type (e.g: instance, network)
- The component name
- The Openstack's project identifier
- The ticket identifier (see sticks)
- The title
- The description
- The security rating
- The vulnerabilities
- The number of vulnerabilities
- The date of the last report

Security reports may be retrieved by their uuid.

3.1.6 Security alarms

Cerberus stores security alarms provided by the security components such as SIEM. These security alarms have a predefined schema and Cerberus stores the following information:

- The uuid of the alarm
- The uuid of the plugin
- The alarm identifier
- The Openstack's component identifier (e.g: an instance id, a network id)
- The Openstack's project identifier
- The ticket identifier (see sticks)
- The timestamp (date when the notification has been received on oslo bus)
- The summary
- The severity
- The status (e.G: new)
- The description

Security alarms may be retrieved by their uuid.

3.1.7 Module loading and extensions

Cerberus manager makes use of stevedore to load extensions dynamically.

CHAPTER 4

API References

4.1 Cerberus REST API (root)

GET /
Return the version list

Return type list(APIVersion)

type APILink
API link description.

Data samples:

Json

```
{
    "href": "http://127.0.0.1:8888/v1",
    "rel": "self",
    "type": "text/html"
}
```

XML

```
<value>
  <type>text/html</type>
  <rel>self</rel>
  <href>http://127.0.0.1:8888/v1</href>
</value>
```

href

Type unicode

URL of the link.

rel

Type unicode

Relationship with this link.

type

Type unicode

Type of link.

type APIMediaType

Media type description.

Data samples:

Json

```
{
    "base": "application/json",
    "type": "application/vnd.openstack.sticks-v1+json"
}
```

XML

```
<value>
  <base>application/json</base>
  <type>application/vnd.openstack.sticks-v1+json</type>
</value>
```

base

Type unicode

Base type of this media type.

type

Type unicode

Type of this media type.

type APIVersion

API Version description.

Data samples:

Json

```
{
    "id": "v1",
    "links": [
        {
            "href": "http://127.0.0.1:8888/v1",
            "rel": "self",
            "type": "text/html"
        }
    ],
    "media_types": [
        {
            "base": "application/json",
            "type": "application/vnd.openstack.sticks-v1+json"
        }
    ],
    "status": "STABLE",
    "updated": "2014-08-11T16:00:00Z"
}
```

XML

```
<value>
  <id>v1</id>
  <status>STABLE</status>
  <updated>2014-08-11T16:00:00Z</updated>
```

```
        <links>
          <item>
            <type>text/html</type>
            <rel>self</rel>
            <href>http://127.0.0.1:8888/v1</href>
          </item>
        </links>
        <media_types>
          <item>
            <base>application/json</base>
            <type>application/vnd.openstack.sticks-v1+json</type>
          </item>
        </media_types>
      </value>
```

id

Type unicode

ID of the version.

links

Type list(APILink)

List of links to API resources.

media_types

Type list(APIMediaType)

Types accepted by this API.

status

Type Enum(EXPERIMENTAL, STABLE)

Status of the version.

updated

Type unicode

Last update in iso8601 format.

4.2 Cerberus REST API (v1)

4.2.1 Plugins

GET /v1/plugins
Get a list of plugins loaded by Cerberus manager

Return a list of plugins loaded by Cerberus manager

Raises HTTPServiceUnavailable: an error occurred in Cerberus Manager or the service is unavailable HTTPNotFound: any other error

Return type `PluginResourceCollection`

GET /v1/plugins/ (*uuid*)
Get details of a specific plugin whose identifier is uuid

Parameters

- **uuid** (unicode) – the identifier of the plugin

Return details of a specific plugin

Raises HTTPServiceUnavailable: an error occurred in Cerberus Manager or the service is unavailable HTTPNotFound: Plugin is not found. Also any other error

Return type PluginResource

type PluginResource

Type describing a plugin.

Data samples:

Json

```
{
    "id": 1,
    "methods": [
        "method_1",
        "method_2"
    ],
    "name": "some_plugin",
    "provider": "some_provider",
    "subscribed_events": [
        "image.update"
    ],
    "tool_name": "some_tool",
    "type": "scanner",
    "uuid": "063d4206-5afc-409c-a4d1-c2a469299d37",
    "version": "2015.1"
}
```

XML

```
<value>
  <name>some_plugin</name>
  <id>1</id>
  <uuid>063d4206-5afc-409c-a4d1-c2a469299d37</uuid>
  <methods>
    <item>method_1</item>
    <item>method_2</item>
  </methods>
  <version>2015.1</version>
  <provider>some_provider</provider>
  <subscribed_events>
    <item>image.update</item>
  </subscribed_events>
  <type>scanner</type>
  <tool_name>some_tool</tool_name>
</value>
```

description

Type unicode

Description of the plugin.

id

Type integer

Id of the plugin.

methods

Type list(unicode)

Hook methods.

name

Type unicode

Name of the plugin.

provider

Type unicode

Provider of the plugin.

subscribed_events

Type list(unicode)

Subscribed events of the plugin.

tool_name

Type unicode

Tool name of the plugin.

type

Type unicode

Type of the plugin.

uuid

Type unicode

Uuid of the plugin.

version

Type unicode

Version of the plugin.

4.2.2 Security alarms

GET /v1/security_alarms
Get stored security alarms.

Return list of security alarms

Raises HTTPNotFound: Any database error

Return type `SecurityAlarmResourceCollection`

GET /v1/security_alarms/{id}/ (*alarm_id*)
Get security alarm in db

Return a security alarm

Raises HTTPNotFound: Alarm not found or any database error

Return type `SecurityAlarmResource`

PUT /v1/security_alarms/{id}/ (*alarm_id*) **/tickets**
Modify the ticket id associated to a security alarm in db.

Parameters

- **ticket_id** – the ticket_id to store in db.

Raises HTTPNotFound: Alarm not found or any database error

type SecurityAlarmResource
Representation of a security alarm.

Data samples:

Json

```
{
    "alarm_id": "fea4b170-ed46-4a50-8b91-ed1c6876be7d",
    "component_id": "4b75699f7a9649438932bebdbf9711e0",
    "description": "Apache suffered an attack by brute force. Thousands of attempts to log f
    "id": 1,
    "plugin_id": "927c8435-f81f-468a-92cb-ebb08ed0fad2",
    "project_id": "e845a1f2004847e4ac14cb1732a2e75f",
    "severity": "critical",
    "status": "new",
    "summary": "Several attempts to log failed",
    "timestamp": "2015-03-24T09:50:50.577840"
}
```

XML

```
<value>
  <id>1</id>
  <plugin_id>927c8435-f81f-468a-92cb-ebb08ed0fad2</plugin_id>
  <alarm_id>fea4b170-ed46-4a50-8b91-ed1c6876be7d</alarm_id>
  <timestamp>2015-03-24T09:50:50.577840</timestamp>
  <status>new</status>
  <severity>critical</severity>
  <project_id>e845a1f2004847e4ac14cb1732a2e75f</project_id>
  <component_id>4b75699f7a9649438932bebdbf9711e0</component_id>
  <summary>Several attempts to log failed</summary>
  <description>Apache suffered an attack by brute force. Thousands of attempts to log failed
</value>
```

alarm_id

Type unicode

Associated alarm id.

component_id

Type unicode

Component id.

description

Type unicode

Description.

id

Type integer

Security alarm id.

plugin_id

Type unicode

Associated plugin id.

project_id

Type unicode

Associated project id.

severity

Type unicode

Severity.

status

Type unicode

Status.

summary

Type unicode

Summary.

ticket_id

Type unicode

Associated ticket id.

timestamp

Type datetime

creation date.

4.2.3 Security reports

GET /v1/security_reports
Get stored security reports.

Return list of security reports

Raises HTTPNotFound: Any database error

Return type `SecurityReportResourceCollection`

GET /v1/security_reports/{id}/ (*uuid*)
Get security report in db.

Return a security report

Raises HTTPNotFound: Report not found or any database error

Return type `SecurityReportResource`

DELETE /v1/security_reports/{id}/ (*uuid*)
Delete the security report stored in db.

Raises HTTPNotFound: Report not found or any database error

PUT /v1/security_reports/{id}/ (*uuid*) **/tickets**
Modify the ticket id associated to a security report in db.

Parameters

- **ticket_id** – the ticket_id to store in db.

Raises HTTPNotFound: Report not found or any database error

type SecurityReportResource
Representation of a security report.

Data samples:

Json

```
{
    "component_id": "a1d869a1-6ab0-4f02-9e56-f83034bacfcb",
    "component_name": "openstack-server",
    "component_type": "instance",
    "description": "security report",
    "last_report_date": "2015-05-06T16:19:29",
    "plugin_id": "063d4206-5afc-409c-a4d1-c2a469299d37",
    "project_id": "510c7f4ed14243f09df371bba2561177",
    "report_id": "fea4b170-ed46-4a50-8b91-ed1c6876be7d",
    "security_rating": 7.4,
    "title": "Security report",
    "uuid": "8a8608a2-681f-44e3-8298-7d8b5039b5b9",
    "vulnerabilities": "{\"443\": {\"archived\": \"false\", \"protocol\": \"tcp\", \"family\
    "vulnerabilities_number": "2"
}
```

XML

```
<value>
  <uuid>8a8608a2-681f-44e3-8298-7d8b5039b5b9</uuid>
  <plugin_id>063d4206-5afc-409c-a4d1-c2a469299d37</plugin_id>
  <report_id>fea4b170-ed46-4a50-8b91-ed1c6876be7d</report_id>
  <component_id>a1d869a1-6ab0-4f02-9e56-f83034bacfcb</component_id>
  <component_type>instance</component_type>
  <component_name>openstack-server</component_name>
  <project_id>510c7f4ed14243f09df371bba2561177</project_id>
  <title>Security report</title>
  <description>security report</description>
  <security_rating>7.4</security_rating>
  <vulnerabilities>{"443": {"archived": "false", "protocol": "tcp", "family": "Web Servers",
  <vulnerabilities_number>2</vulnerabilities_number>
  <last_report_date>2015-05-06T16:19:29</last_report_date>
</value>
```

component_id

Type unicode

Associated component id.

component_name

Type unicode

Component name.

component_type

Type unicode

Component type.

description

Type unicode

Description.

last_report_date

Type datetime

Last report date.

plugin_id

Type unicode

Associated plugin id.

project_id

Type unicode

Associated project id.

report_id

Type unicode

Associated report id provided by plugin.

security_rating

Type float

Security rating.

ticket_id

Type unicode

Associated ticket id.

title

Type unicode

Title of report.

uuid

Type unicode

Security report id.

vulnerabilities

Type unicode

Vulnerabilities.

vulnerabilities_number

Type integer

Total of Vulnerabilities.

4.2.4 Tasks

GET /v1/tasks
List tasks handled by Cerberus Manager.

Return list of tasks

Raises HTTPServiceUnavailable: an error occurred in Cerberus Manager or the service is unavailable

Return type `TaskResourceCollection`

GET /v1/tasks
Get details of a task

Return task details

Raises HTTPNotFound: task is not found

Return type `TaskResource`

POST /v1/tasks
Create a task

Return task details

Raises HTTPBadRequest

Return type `TaskResource`

DELETE /v1/tasks
Delete a task

Raises HTTPNotFound: task does not exist

POST /v1/tasks/{id}/actions/force_delete
Force delete task

Raises HTTPNotFound: task is not found

POST /v1/tasks/{id}/actions/start
Start task

Raises HTTPBadRequest: task not found or impossible to start it

POST /v1/tasks/{id}/actions/stop
Stop task

Raises HTTPBadRequest: task not found or impossible to stop it

type TaskResource
Representation of a task.

Data samples:

Json

```
{
    "id": "4820cea8-e88e-463b-ae1f-6bbde009cc93",
    "name": "some_task",
    "period": 3,
    "persistent": true,
    "plugin_id": "063d4206-5afc-409c-a4d1-c2a469299d37",
    "state": "running",
    "type": "recurrent"
}
```

XML

```
<value>
  <name>some_task</name>
  <period>3</period>
  <state>running</state>
  <id>4820cea8-e88e-463b-ae1f-6bbde009cc93</id>
  <plugin_id>063d4206-5afc-409c-a4d1-c2a469299d37</plugin_id>
  <type>recurrent</type>
  <persistent>true</persistent>
</value>
```

id

Type integer

Associated task id.

method

Type unicode

Hook methods.

name

Type unicode

Name of the task.

period

Type integer

Period if periodic.

persistent

Type bool

If task must persist.

plugin_id

Type unicode

Associated plugin id.

state

Type unicode

Running or not.

type

Type unicode

Type of the task.

CHAPTER 5

Plugin development

5.1 Writing plugins

This documentation gives you some clues on how to write a new plugin for Cerberus if you wish to integrate a security component which is not covered by an existing plugin.

5.1.1 Cerberus manager

The cerberus manager is implemented in `cerberus/manager.py`. The cerberus manager loads all plugins defined in the namespace `cerberus.plugins`. It is also responsible of tasks management.

5.1.2 Plugins

Cerberus manager makes use of stevedore to load extensions dynamically. Plugins can:

- subscribe to notifications sent through AMQP.
- define a callable method (@webmethod) which will be invoked either once or periodically thanks to a task.

Notifications

Plugins must implement the method `process_notification(self, ctxt, publisher_id, event_type, payload, metadata)`: which receives an event message the plugin subscribed to.

For example, the `test_plugin` plugin listens to one event:

- image.update

Tasks

For a plugin to be invoked through a task, it must implement a method with decorator `@webmethod`

For example, the `test_plugin` plugin defines one callable method:

- get_security_reports

Adding new plugins

Cerberus needs to be easy to extend and configure so it can be tuned for each installation. A plugin system based on setuptools entry points makes it easy to add new monitors in the agents. In particular, Cerberus uses Stevedore, and you should put your entry point definitions in the entry_points.txt file of your Cerberus egg. Alternatively, you can put your entry point definitions in the setup.cfg file before installing Cerberus Installing a plugin automatically activates it the next time the cerberus manager starts.

CHAPTER 6

Indices and tables

- *genindex*
- *search*

/

GET /, 9

/v1

GET /v1/plugins, 11
GET /v1/plugins/(uuid), 11
GET /v1/security_alarms, 13
GET /v1/security_alarms/{id}/(alarm_id),
13
GET /v1/security_reports, 15
GET /v1/security_reports/{id}/(uuid),
15
GET /v1/tasks, 18
POST /v1/tasks, 18
POST /v1/tasks/{id}/actions/force_delete,
18
POST /v1/tasks/{id}/actions/start, 18
POST /v1/tasks/{id}/actions/stop, 18
PUT /v1/security_alarms/{id}/(alarm_id)/tickets,
14
PUT /v1/security_reports/{id}/(uuid)/tickets,
15
DELETE /v1/security_reports/{id}/(uuid),
15
DELETE /v1/tasks, 18

A

alarm_id (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 14
APILink (webservice type), 9
APIMediaType (webservice type), 9
APIVersion (webservice type), 10

B

base (cerberus.api.root.APIMediaType attribute), 10

C

component_id (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 14
component_id (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 16
component_name (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 16
component_type (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 16

D

description (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 12
description (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 14
description (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17

H

href (cerberus.api.root.APILink attribute), 9

I

id (cerberus.api.root.APIVersion attribute), 11
id (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 12
id (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 14
id (cerberus.api.v1.datamodels.task.TaskResource attribute), 19

L

last_report_date (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
links (cerberus.api.root.APIVersion attribute), 11

M

media_types (cerberus.api.root.APIVersion attribute), 11
method (cerberus.api.v1.datamodels.task.TaskResource attribute), 19
methods (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 13

N

name (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 13
name (cerberus.api.v1.datamodels.task.TaskResource attribute), 19

P

period (cerberus.api.v1.datamodels.task.TaskResource attribute), 19
persistent (cerberus.api.v1.datamodels.task.TaskResource attribute), 19
plugin_id (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 15
plugin_id (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
plugin_id (cerberus.api.v1.datamodels.task.TaskResource attribute), 19
PluginResource (webservice type), 12
project_id (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 15
project_id (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
provider (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 13

R

rel (cerberus.api.root.APILink attribute), 9

report_id (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
S
security_rating (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
SecurityAlarmResource (webservice type), 14
SecurityReportResource (webservice type), 16
severity (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 15
state (cerberus.api.v1.datamodels.task.TaskResource attribute), 19
status (cerberus.api.root.APIVersion attribute), 11
status (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 15
subscribed_events (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 13
summary (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 15
T
TaskResource (webservice type), 18
ticket_id (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 15
ticket_id (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
timestamp (cerberus.api.v1.datamodels.security_alarm.SecurityAlarmResource attribute), 15
title (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
tool_name (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 13
type (cerberus.api.root.APILink attribute), 9
type (cerberus.api.root.APIMediaType attribute), 10
type (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 13
type (cerberus.api.v1.datamodels.task.TaskResource attribute), 19
U
updated (cerberus.api.root.APIVersion attribute), 11
uuid (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 13
uuid (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
V
version (cerberus.api.v1.datamodels.plugin.PluginResource attribute), 13
vulnerabilities (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17
vulnerability_number (cerberus.api.v1.datamodels.security_report.SecurityReportResource attribute), 17