ODSC Documentation

Release 1.0.0

Ramiro Blázquez / Daniel Garcia (cr0hn)

May 22, 2017

Contents

1 Index 3

2 ODSC in a few words 7

3 Why? 9

4 Contributing 11

5 License 13

Project site	https://github.com/bbva/odsc
Issues	https://github.com/bbva/odsc/issues/
Documentation	https://odsc.readthedocs.org/
DockerHub	https://hub.docker.com/r/bbvalabs/odsc/
Authors	Ramiro Blázquez / Daniel Garcia (cr0hn)
Latest Version	1.0.0
Python versions	3.5 or above

CHAPTER 1

Index

Quick Start

Install using Docker Compose

ODSC use various services (like Redis or PostgresSQL) that you need to install and configure. So the most easly wat to desploy ODSC is using *docker-compose*:

1. **Download the project code**

```
> git clone https://github.com/BBVA/ODSC.git /tmp/odsc
```

2. **Go to the source code installation**

```
> cd /tmp/odsc
```

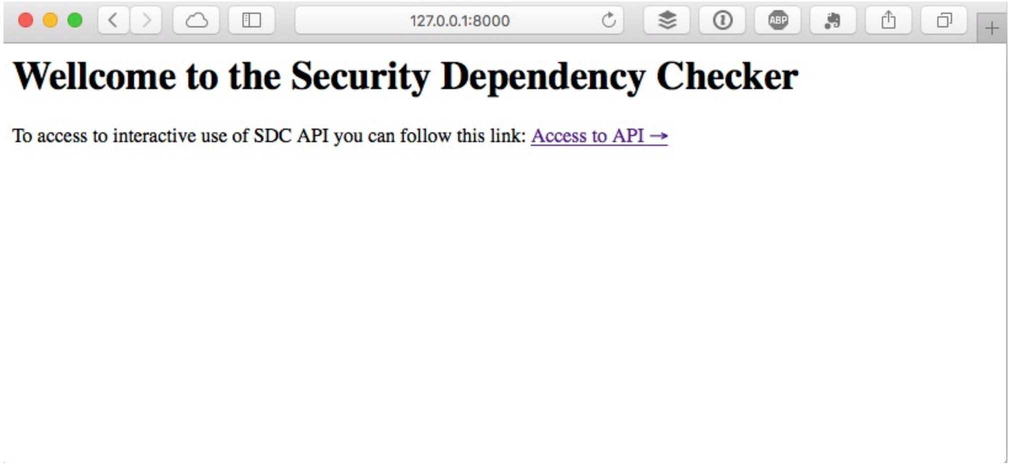
3. **Run ‘Docker compose‘ to deploy the service:**

```
> docker-compose build & docker-compose up
```

4. **Use the project**

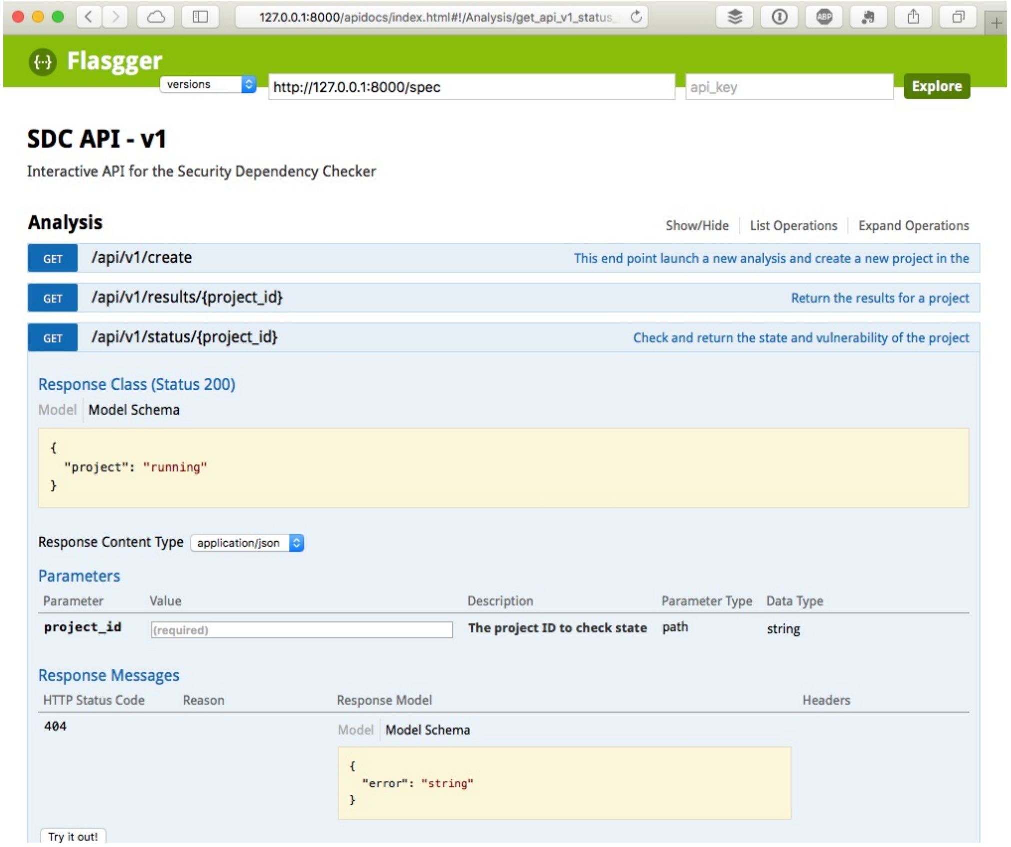
4.1. *Using web browser*

Open your web browser to the address: http://127.0.0.1:



Wellcome to the Security Dependency Checker

To access to interactive use of SDC API you can follow this link: Access to API →


127.0.0.1:8000/apidocs/index.html#!/Analysis/get_api_v1_status_
Flasgger
versions
http://127.0.0.1:8000/spec
api_key
Explore
Show/Hide
List Operations
Expand Operations
Try it out!

SDC API - v1

Interactive API for the Security Dependency Checker

Analysis

GET /api/v1/create — This end point launch a new analysis and create a new project in the

GET /api/v1/results/{project_id} — Return the results for a project

GET /api/v1/status/{project_id} — Check and return the state and vulnerability of the project

Response Class (Status 200)

Model | Model Schema

```
{
  "project": "running"
}
```

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
project_id	(required)	The project ID to check state	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
404		Model | Model Schema { "error": "string" }	

4.2. *Using in console with CURL*

Very easy, after deploy the service, you only need to run a simple *curl* in a console:

```
> curl http://127.0.0.1/api/v1/project/create?lang=nodejs&repo=https://
↪github.com/ramirobg94/QuizCore
{project: ñaskdjflasjfklas}
```

Now check the status:

```
> curl http://127.0.0.1/api/v1/project/status/
{scan_status: "running"}
```

And, when finished, get results:

```
> curl http://127.0.0.1/api/v1/project/results/
```

Advanced deploy & config

Manual launch

ODSC is written in Python, using Flask and Celery. The service is composed by two parts:

- Rest API (Flask)
- Analyzers (Celery processes)

We'll need to launch the two processes individually.

Note: Pay attention to install the tools used by the analyzers before launch the service!

see *Binaries required*

Launch Celery

```
> cd odsc
> celery -A run:celery worker -l INFO
```

Launch web application

```
> cd odsc
> gunicorn -w 4 run:app
```

Binaries required

ODSC uses various software to launch the analysis. To do that, you need to install:

- retire: https://www.npmjs.com/package/retire
- OWASP Dependency Check: https://www.owasp.org/index.php/OWASP_Dependency_Check

Environment vars

You can customize the deployments setting some environments variables

- REDIS: Redis addr. Default: redis://localhost:6379
- CELERY_BACKEND. Celery backend used. Usually a database. Default: redis://localhost:6379

- CELERY_BROKER_URL: Broker used for distribute tasks. Default: redis://localhost:6379
- SQLALCHEMY_DATABASE_URI: Database DSN where store the results. Default: postgresql+pg8000://postgres:password@localhost/vulnerabilities
- ADDITIONAL_BINARY_PATHS: If your *retire* or *odsc* are not in the default system PATH, you can add more paths. Default: ":/usr/local/bin/"

Binaries required

ODSC uses various software to launch the analysis. To do that, you need to install:

- **Retire**: https://www.npmjs.com/package/retire
- **OWASP Dependency Check**: https://www.owasp.org/index.php/OWASP_Dependency_Check

Plugin development

TODO

CHAPTER 2

ODSC in a few words

Is a meta tool to analyze the security Issues in third party libraries used in your project.

CHAPTER 3

Why?

There're **many different tools for analyze third party vulnerabilities** for many languages, **but there're not a unique tool that works well for all of them**.

This implies that:

1. you need to locate the language specific tool,
2. learn how use it,
3. transform the results format to one you understand

We're created **this project to simplify this process** and you can focused only in the important: your project.

ODSC is a tool that **can choose the most suitable security tools for each languages and return unified and in friendly way**: a JSON format.

CHAPTER 4

Contributing

Any collaboration is welcome!

There're many tasks to do.You can check the Issues and send us a Pull Request.

CHAPTER 5

License

This project is distributed under MIT license