ATTACK DEFENSE

by PentesterAcademy

Name	Safety: Python Dependency Check
URL	https://attackdefense.com/challengedetails?cid=2257
Type	DevOps Basics: Software Component Analysis

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Challenge Description

Safety is a tool used to check dependencies of Python projects. This is to make sure that the project will have no security vulnerabilities.

A Kali CLI machine (kali-cli) is provided to the user with Safety installed on it. The source code for three sample web applications is provided in the home directory of the root user.

Objective: Utilise the safety tool to run checks on the dependencies installed for security vulnerabilities.

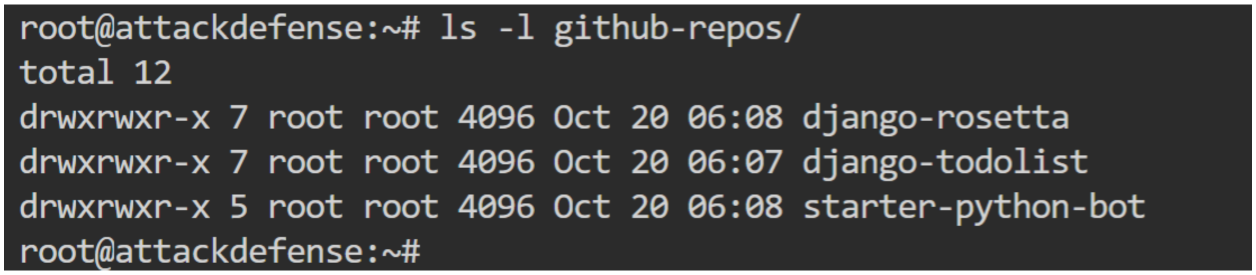
Instructions:

- The source code of web applications is provided at /root/github-repos

Solution

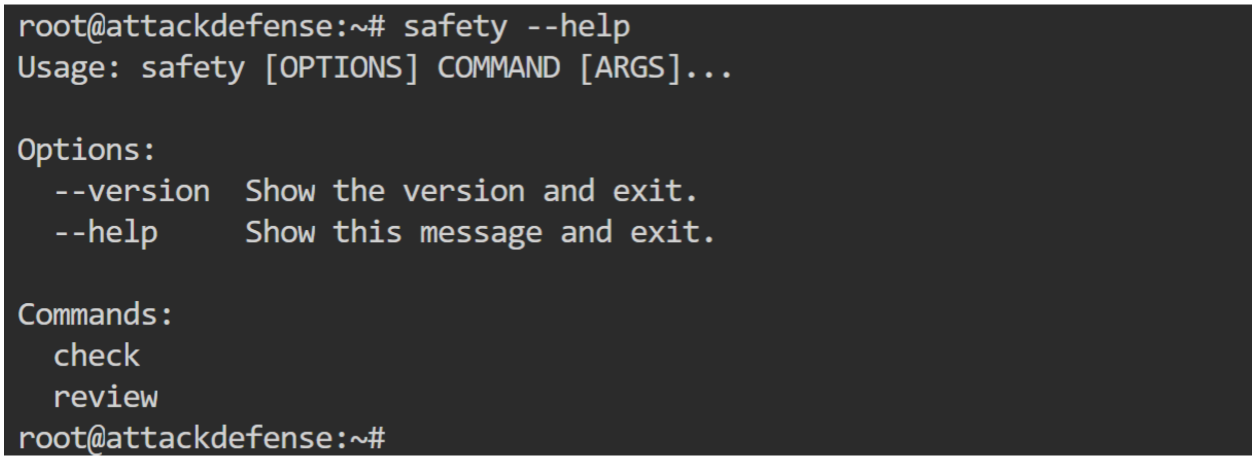
Step 1: Check the provided web applications.

Command: ls -l github-repos/


```
root@attackdefense:~# ls -l github-repos/
total 12
drwxrwxr-x 7 root root 4096 Oct 20 06:08 django-rosetta
drwxrwxr-x 7 root root 4096 Oct 20 06:07 django-todolist
drwxrwxr-x 5 root root 4096 Oct 20 06:08 starter-python-bot
root@attackdefense:~#
```

Step 2: Check the available options of the safety tool.

Command: safety --help

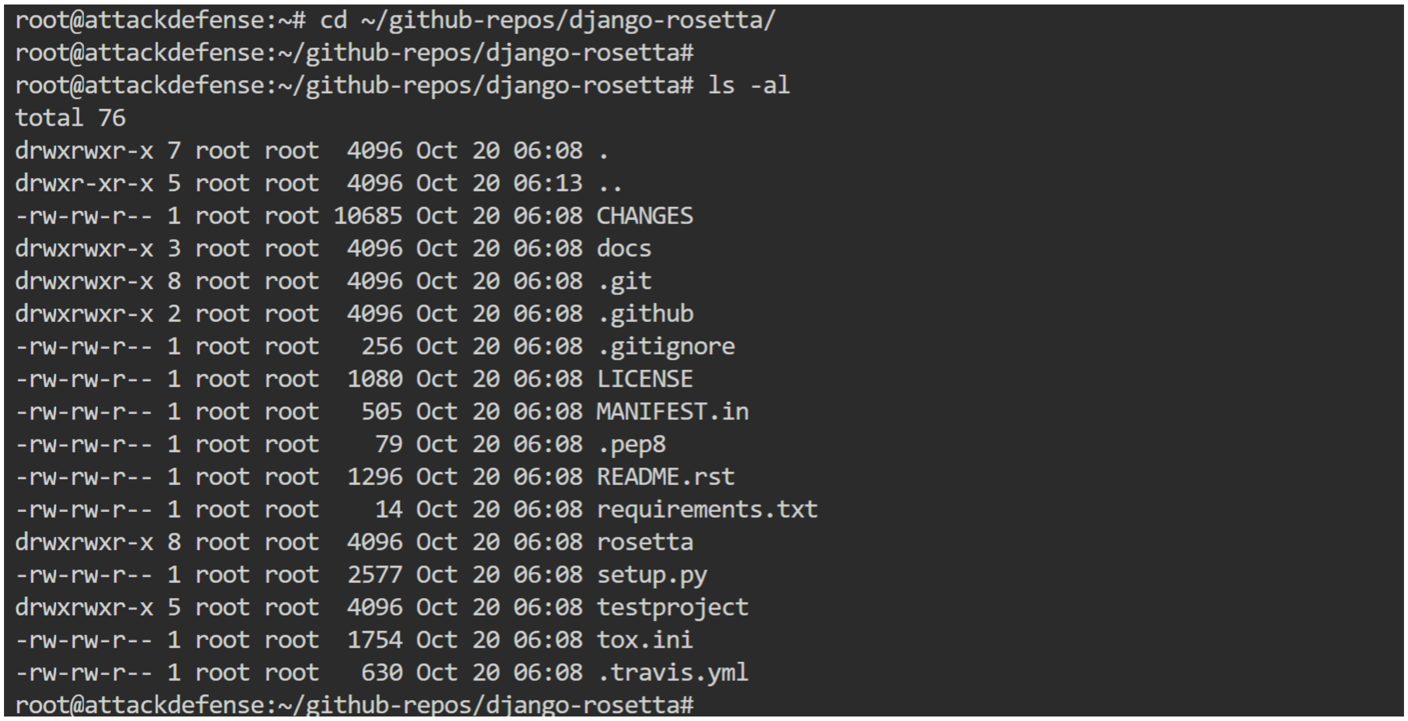

```
root@attackdefense:~# safety --help
Usage: safety [OPTIONS] COMMAND [ARGS]...

Options:
  --version  Show the version and exit.
  --help     Show this message and exit.

Commands:
  check
  review
root@attackdefense:~#
```

We will take one example at a time and run the tool on that.

Example 1: django-rosetta

Step 1: Change to the django-rosetta directory and check its contents.

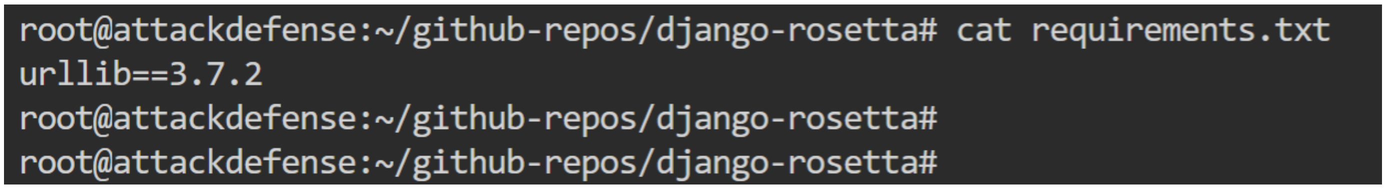
Commands:
cd ~/github-repos/django-rosetta/
ls -al


```
root@attackdefense:~# cd ~/github-repos/django-rosetta/
root@attackdefense:~/github-repos/django-rosetta#
root@attackdefense:~/github-repos/django-rosetta# ls -al
total 76
drwxrwxr-x 7 root root  4096 Oct 20 06:08 .
drwxr-xr-x 5 root root  4096 Oct 20 06:13 ..
-rw-rw-r-- 1 root root 10685 Oct 20 06:08 CHANGES
drwxrwxr-x 3 root root  4096 Oct 20 06:08 docs
drwxrwxr-x 8 root root  4096 Oct 20 06:08 .git
drwxrwxr-x 2 root root  4096 Oct 20 06:08 .github
-rw-rw-r-- 1 root root   256 Oct 20 06:08 .gitignore
-rw-rw-r-- 1 root root  1080 Oct 20 06:08 LICENSE
-rw-rw-r-- 1 root root   505 Oct 20 06:08 MANIFEST.in
-rw-rw-r-- 1 root root    79 Oct 20 06:08 .pep8
-rw-rw-r-- 1 root root  1296 Oct 20 06:08 README.rst
-rw-rw-r-- 1 root root    14 Oct 20 06:08 requirements.txt
drwxrwxr-x 8 root root  4096 Oct 20 06:08 rosetta
-rw-rw-r-- 1 root root  2577 Oct 20 06:08 setup.py
drwxrwxr-x 5 root root  4096 Oct 20 06:08 testproject
-rw-rw-r-- 1 root root  1754 Oct 20 06:08 tox.ini
-rw-rw-r-- 1 root root   630 Oct 20 06:08 .travis.yml
root@attackdefense:~/github-repos/django-rosetta#
```

The safety tool will look for “requirements.txt” file in the project directory and check the dependencies in it.

Step 2: Check the content present in the “requirements.txt” file.

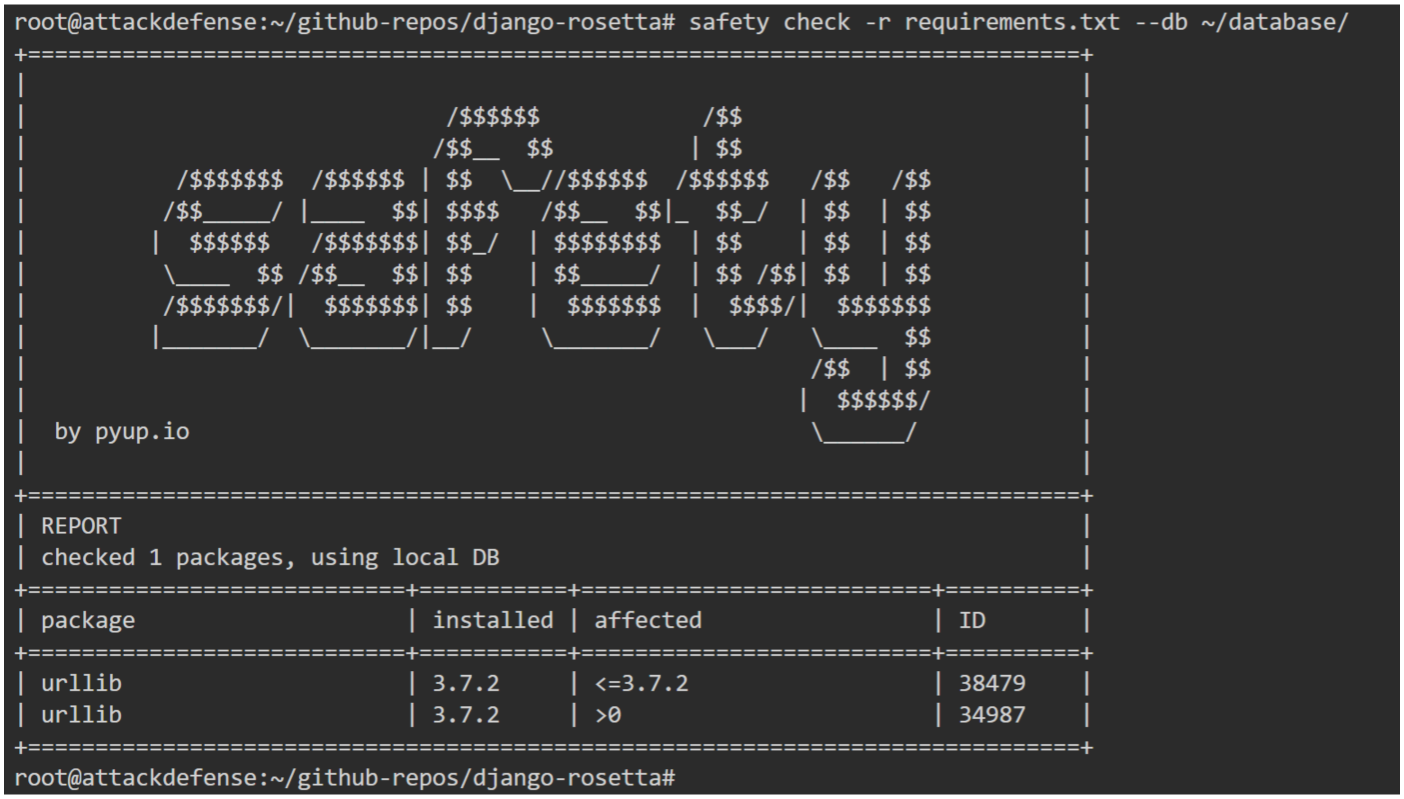
Command: cat requirements.txt


```
root@attackdefense:~/github-repos/django-rosetta# cat requirements.txt
urllib==3.7.2
root@attackdefense:~/github-repos/django-rosetta#
root@attackdefense:~/github-repos/django-rosetta#
```

The file contains the dependency name and version, i.e urllib version 3.7.2

Step 3: Run the safety command with the check mode while passing the database path which is located in the home directory of the user and the requirements.txt file.

Command: safety check -r requirements.txt --db ~/database/



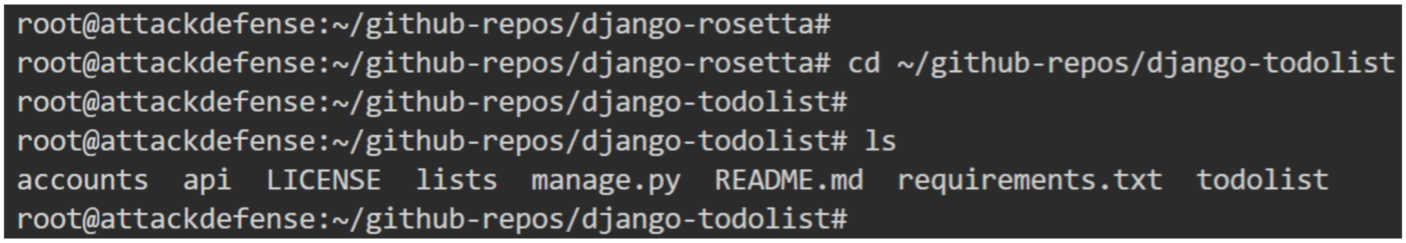
```
root@attackdefense:~/github-repos/django-rosetta# safety check -r requirements.txt --db ~/database/
+==============================================================================+
|                                                                              |
|                               /$$$$$$            /$$                         |
|                              /$$__  $$          | $$                         |
|           /$$$$$$$  /$$$$$$ | $$  \__//$$$$$$  /$$$$$$   /$$   /$$           |
|          /$$_____/ |____  $$| $$$$   /$$__  $$|_  $$_/  | $$  | $$           |
|         |  $$$$$$   /$$$$$$$| $$_/  | $$$$$$$$  | $$    | $$  | $$           |
|          \____  $$ /$$__  $$| $$    | $$_____/  | $$ /$$| $$  | $$           |
|          /$$$$$$$/|  $$$$$$$| $$    |  $$$$$$$  |  $$$$/|  $$$$$$$           |
|         |_______/  \_______/|__/     \_______/   \___/   \____  $$           |
|                                                          /$$  | $$           |
|                                                         |  $$$$$$/           |
|  by pyup.io                                             \______/            |
|                                                                              |
+==============================================================================+
| REPORT                                                                       |
| checked 1 packages, using local DB                                           |
+============================+===========+==========================+==========+
| package                    | installed | affected                 | ID       |
+============================+===========+==========================+==========+
| urllib                     | 3.7.2     | <=3.7.2                  | 38479    |
| urllib                     | 3.7.2     | >0                       | 34987    |
+==============================================================================+
root@attackdefense:~/github-repos/django-rosetta#
```

Safety tool has identified that urllib version 3.7.2 or below is vulnerable.

Example 2: django-todolist

Step 1: Change to the django-todolist directory and check its contents.

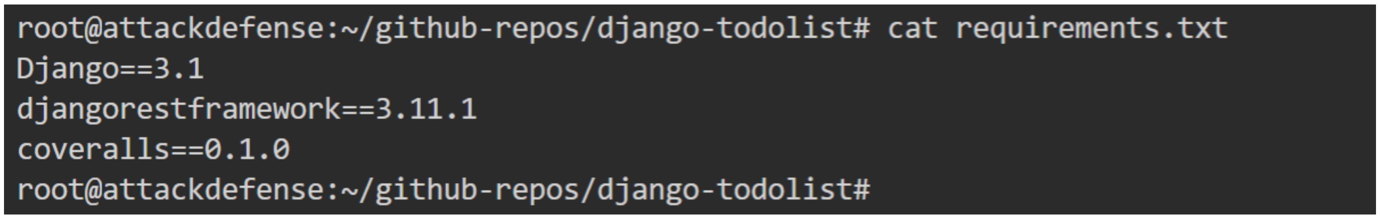
Commands:
cd ~/github-repos/django-todolist
ls



```
root@attackdefense:~/github-repos/django-rosetta#
root@attackdefense:~/github-repos/django-rosetta# cd ~/github-repos/django-todolist
root@attackdefense:~/github-repos/django-todolist#
root@attackdefense:~/github-repos/django-todolist# ls
accounts  api  LICENSE  lists  manage.py  README.md  requirements.txt  todolist
root@attackdefense:~/github-repos/django-todolist#
```

Step 2: Run the following command to check the content present in the "requirements.txt" file.

Command: cat requirements.txt

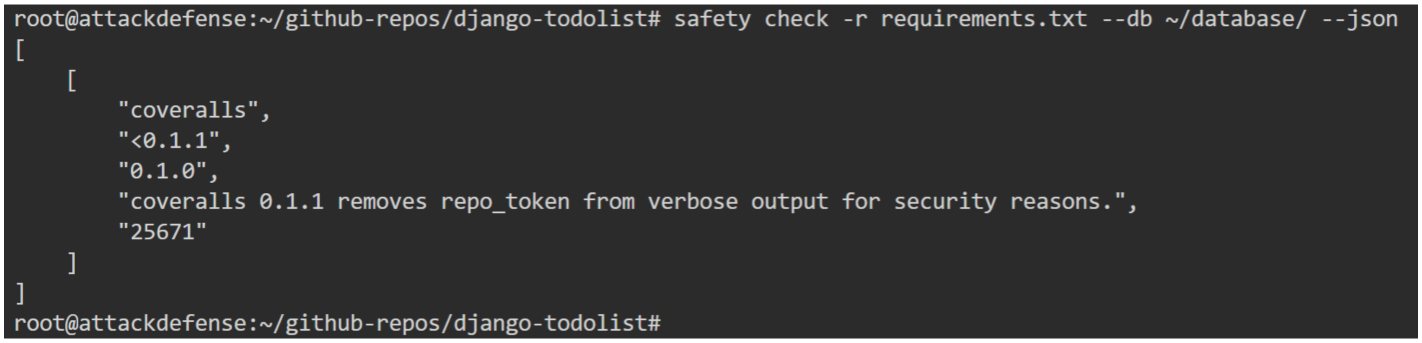


```
root@attackdefense:~/github-repos/django-todolist# cat requirements.txt
Django==3.1
djangorestframework==3.11.1
coveralls==0.1.0
root@attackdefense:~/github-repos/django-todolist#
```

The file contains the dependency name and version, i.e Django version 3.1, djangorestframework version 3.11.1 and coveralls version 0.1.0

Step 3: Run the safety command with the check mode while passing the database path which is located in the home directory of the user and the requirements.txt file and output will be in JSON format.

Command: safety check -r requirements.txt --db ~/database/ --json



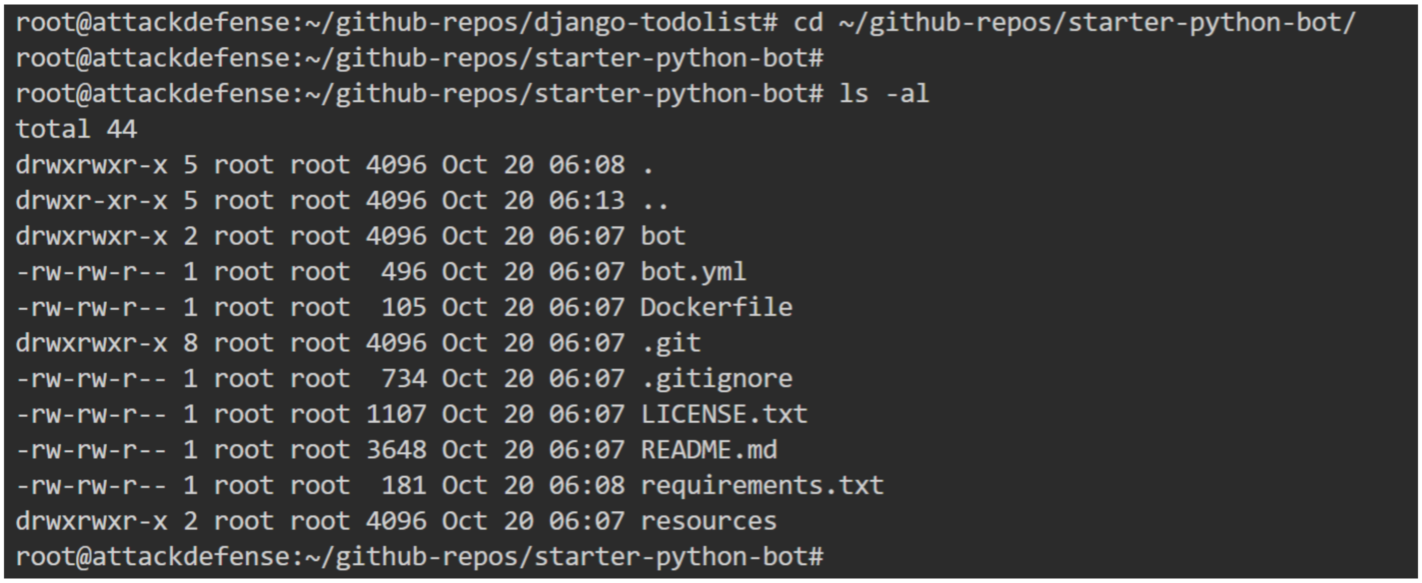
```
root@attackdefense:~/github-repos/django-todolist# safety check -r requirements.txt --db ~/database/ --json
[
    [
        "coveralls",
        "<0.1.1",
        "0.1.0",
        "coveralls 0.1.1 removes repo_token from verbose output for security reasons.",
        "25671"
    ]
]
root@attackdefense:~/github-repos/django-todolist#
```

Safety tool has identified that coveralls version 0.1.1 or below is vulnerable to sensitive data disclosure

Example 3: Python Serverless Boilerplate

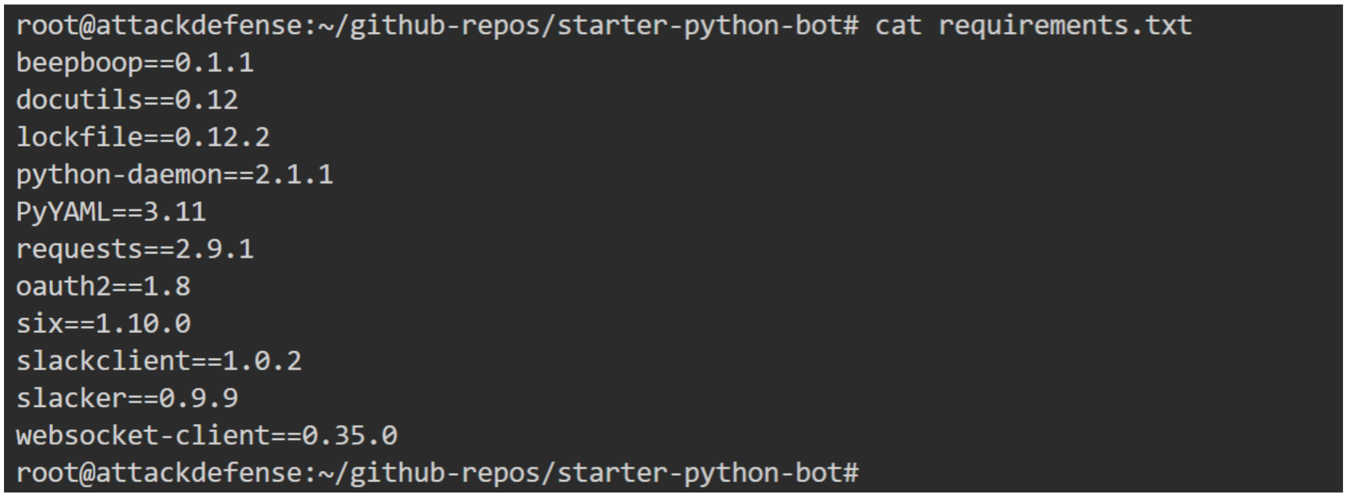
Step 1: Change to the python-serverless-boilerplate directory and check its contents.

Commands:
cd ~/github-repos/starter-python-bot/
ls -al


```
root@attackdefense:~/github-repos/django-todolist# cd ~/github-repos/starter-python-bot/
root@attackdefense:~/github-repos/starter-python-bot#
root@attackdefense:~/github-repos/starter-python-bot# ls -al
total 44
drwxrwxr-x 5 root root 4096 Oct 20 06:08 .
drwxr-xr-x 5 root root 4096 Oct 20 06:13 ..
drwxrwxr-x 2 root root 4096 Oct 20 06:07 bot
-rw-rw-r-- 1 root root  496 Oct 20 06:07 bot.yml
-rw-rw-r-- 1 root root  105 Oct 20 06:07 Dockerfile
drwxrwxr-x 8 root root 4096 Oct 20 06:07 .git
-rw-rw-r-- 1 root root  734 Oct 20 06:07 .gitignore
-rw-rw-r-- 1 root root 1107 Oct 20 06:07 LICENSE.txt
-rw-rw-r-- 1 root root 3648 Oct 20 06:07 README.md
-rw-rw-r-- 1 root root  181 Oct 20 06:08 requirements.txt
drwxrwxr-x 2 root root 4096 Oct 20 06:07 resources
root@attackdefense:~/github-repos/starter-python-bot#
```

Step 2: Run the following command to check the content present in the “requirements.txt” file.

Command: cat requirements.txt


```
root@attackdefense:~/github-repos/starter-python-bot# cat requirements.txt
beepboop==0.1.1
docutils==0.12
lockfile==0.12.2
python-daemon==2.1.1
PyYAML==3.11
requests==2.9.1
oauth2==1.8
six==1.10.0
slackclient==1.0.2
slacker==0.9.9
websocket-client==0.35.0
root@attackdefense:~/github-repos/starter-python-bot#
```

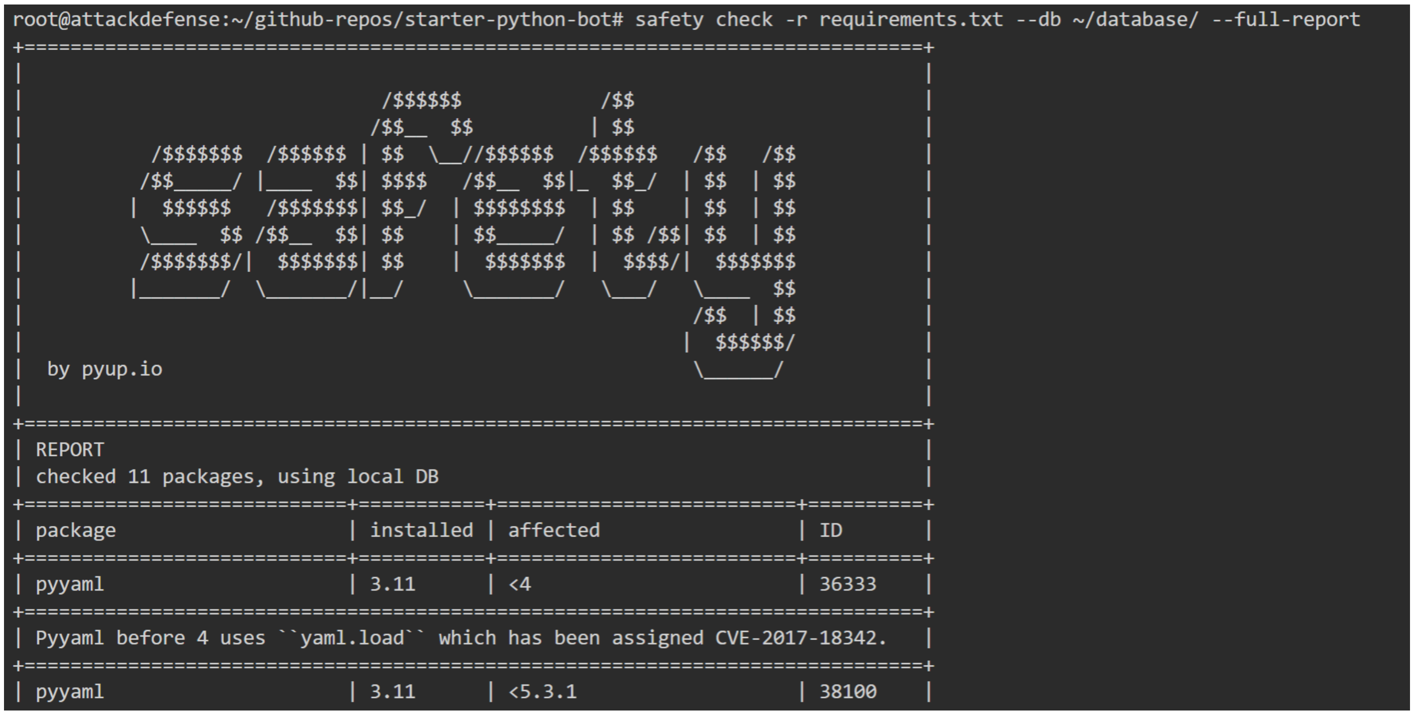
The file contains the dependency name and version, The list is given below

- Beepboop version 0.1.1
- Docutils version 0.12
- Lockfile version 0.12.2
- Python-daemon version 2.1.1

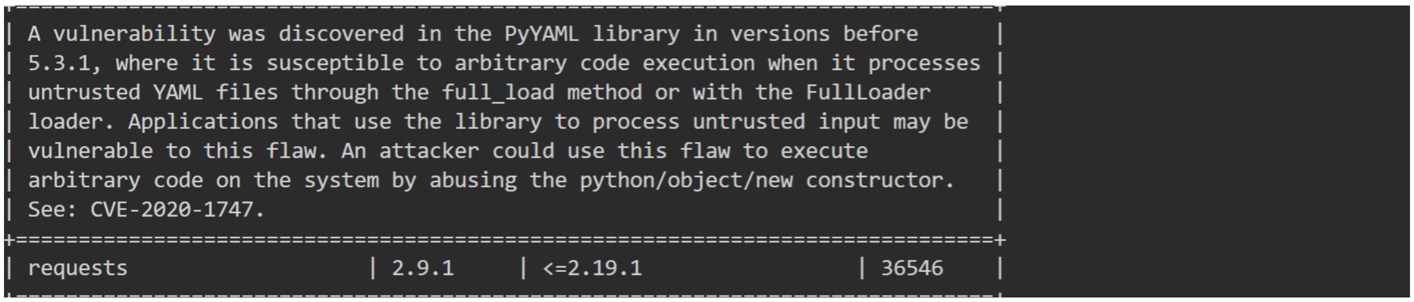
- Pyyaml version 3.11
- Requests version 2.9.1
- Oauth2 version 1.8
- Six version1.10.0
- Slackclient version 1.0.2
- Slacker version 0.9.9
- Websocket-client version 0.35.0

Step 3: Run the safety command with the check mode while passing the database path which is located in the home directory of the user and the requirements.txt file and generate a report.

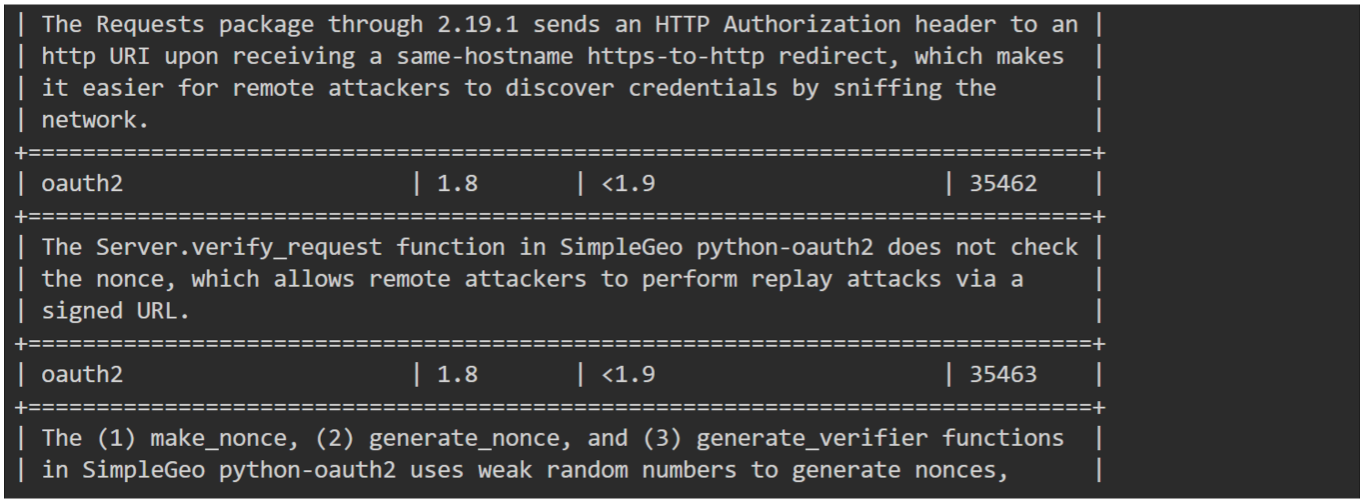
Command: safety check -r requirements.txt --db ~/database/ --full-report



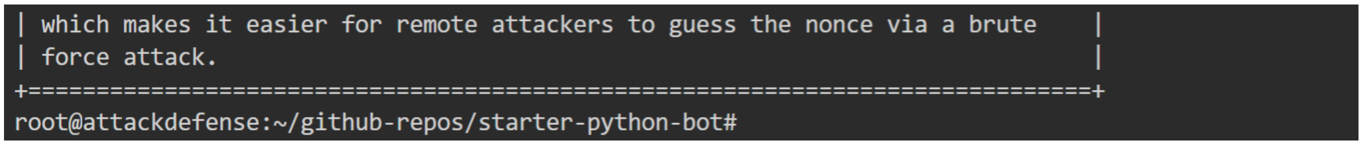
```
root@attackdefense:~/github-repos/starter-python-bot# safety check -r requirements.txt --db ~/database/ --full-report
+==============================================================================+
|                                                                              |
|                               /$$$$$$            /$$                         |
|                              /$$__  $$          | $$                         |
|           /$$$$$$$  /$$$$$$ | $$  \__//$$$$$$  /$$$$$$   /$$   /$$           |
|          /$$_____/ |____  $$| $$$$   /$$__  $$|_  $$_/  | $$  | $$           |
|         |  $$$$$$   /$$$$$$$| $$_/  | $$$$$$$$  | $$    | $$  | $$           |
|          \____  $$ /$$__  $$| $$    | $$_____/  | $$ /$$| $$  | $$           |
|          /$$$$$$$/|  $$$$$$$| $$    |  $$$$$$$  |  $$$$/|  $$$$$$$           |
|         |_______/  \_______/|__/     \_______/   \___/   \____  $$           |
|                                                          /$$  | $$           |
|                                                         |  $$$$$$/           |
|  by pyup.io                                             \______/            |
|                                                                              |
+==============================================================================+
| REPORT                                                                       |
| checked 11 packages, using local DB                                          |
+============================+===========+==========================+==========+
| package                    | installed | affected                 | ID       |
+============================+===========+==========================+==========+
| pyyaml                     | 3.11      | <4                       | 36333    |
+==============================================================================+
| Pyyaml before 4 uses ``yaml.load`` which has been assigned CVE-2017-18342.   |
+==============================================================================+
| pyyaml                     | 3.11      | <5.3.1                   | 38100    |
```



```
+------------------------------------------------------------------------------+
| A vulnerability was discovered in the PyYAML library in versions before      |
| 5.3.1, where it is susceptible to arbitrary code execution when it processes |
| untrusted YAML files through the full_load method or with the FullLoader     |
| loader. Applications that use the library to process untrusted input may be  |
| vulnerable to this flaw. An attacker could use this flaw to execute          |
| arbitrary code on the system by abusing the python/object/new constructor.   |
| See: CVE-2020-1747.                                                          |
+==============================================================================+
| requests                   | 2.9.1     | <=2.19.1                 | 36546    |
+------------------------------------------------------------------------------+
```



```
| The Requests package through 2.19.1 sends an HTTP Authorization header to an |
| http URI upon receiving a same-hostname https-to-http redirect, which makes  |
| it easier for remote attackers to discover credentials by sniffing the      |
| network.                                                                     |
+==============================================================================+
| oauth2                      | 1.8       | <1.9                      | 35462    |
+==============================================================================+
| The Server.verify_request function in SimpleGeo python-oauth2 does not check |
| the nonce, which allows remote attackers to perform replay attacks via a     |
| signed URL.                                                                  |
+==============================================================================+
| oauth2                      | 1.8       | <1.9                      | 35463    |
+==============================================================================+
| The (1) make_nonce, (2) generate_nonce, and (3) generate_verifier functions  |
| in SimpleGeo python-oauth2 uses weak random numbers to generate nonces,      |
```



```
| which makes it easier for remote attackers to guess the nonce via a brute    |
| force attack.                                                                |
+==============================================================================+
root@attackdefense:~/github-repos/starter-python-bot#
```

Safety has identified multiple vulnerabilities in the python project such as

- **Pyyaml:** the package version 4 or below was vulnerable to CVE-2017-18342
- **Requests:** the package version 2.19.1 or below was vulnerable to data disclosure via network sniffing
- **Oauth2:** the package version 1.9 or below is vulnerable to brute force attack.

Learnings

Perform Software Component Analysis with safety utility.