ATTACK DEFENSE

by PentesterAcademy

Name	Graudit: Hunting Sensitive Information
URL	https://www.attackdefense.com/challengedetails?cid=2051
Type	DevSecOps: Static Code 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

Graudit is a script with a set of signatures that is used to perform different scans to find sensitive information in the source code.

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

Objective: Scan the source code using Graudit utility and find the security issues!

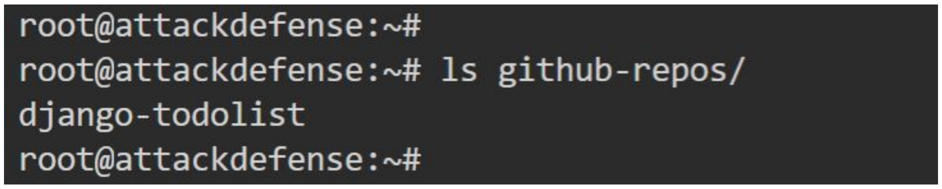
Instructions:

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

Solution

Step 1: Check the provided web application

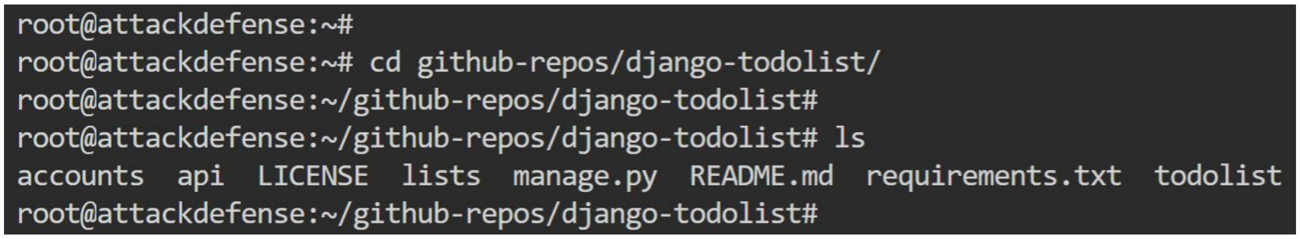
Command: ls github-repos/



```
root@attackdefense:~#
root@attackdefense:~# ls github-repos/
django-todolist
root@attackdefense:~#
```

Step 3: Change to the django-todolist directory and check its contents.

Commands:
cd github-repos/django-todolist
ls

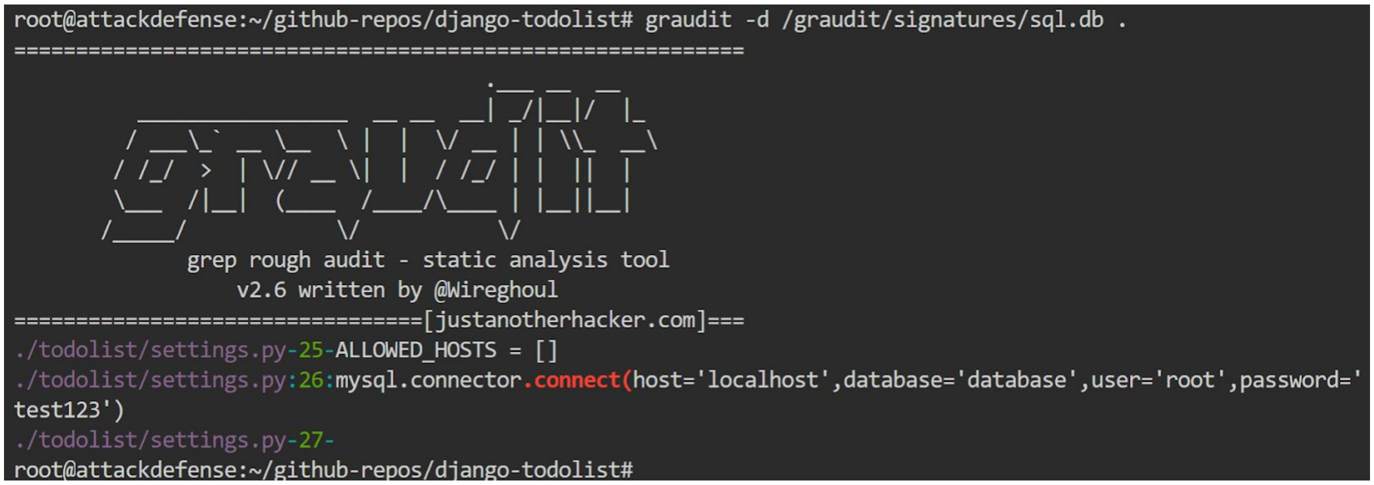


```
root@attackdefense:~#
root@attackdefense:~# cd github-repos/django-todolist/
root@attackdefense:~/github-repos/django-todolist#
root@attackdefense:~/github-repos/django-todolist# ls
accounts  api  LICENSE  lists  manage.py  README.md  requirements.txt  todolist
root@attackdefense:~/github-repos/django-todolist#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

Example 1: Find Sensitive information based on SQL queries.

Step A: Run the graudit tool and pass the SQL signature database to the tool.

Command: graudit -d /graudit/signatures/sql.db .



```
root@attackdefense:~/github-repos/django-todolist# graudit -d /graudit/signatures/sql.db .
===========================================================
                                           .___ __  __
          _________________  __ __  __| _/|__|/  |_
         / ___\_  __ \__  \ |  |  \/ __ | |  \   __\
        / /_/  >  | \// __ \|  |  / /_/ | |  ||  |
        \___  /|__|  (____  /____/\____ | |__||__|
       /_____/            \/           \/
              grep rough audit - static analysis tool
                  v2.6 written by @Wireghoul
=================================[justanotherhacker.com]===
./todolist/settings.py-25-ALLOWED_HOSTS = []
./todolist/settings.py:26:mysql.connector.connect(host='localhost',database='database',user='root',password='
test123')
./todolist/settings.py-27-
root@attackdefense:~/github-repos/django-todolist#
```

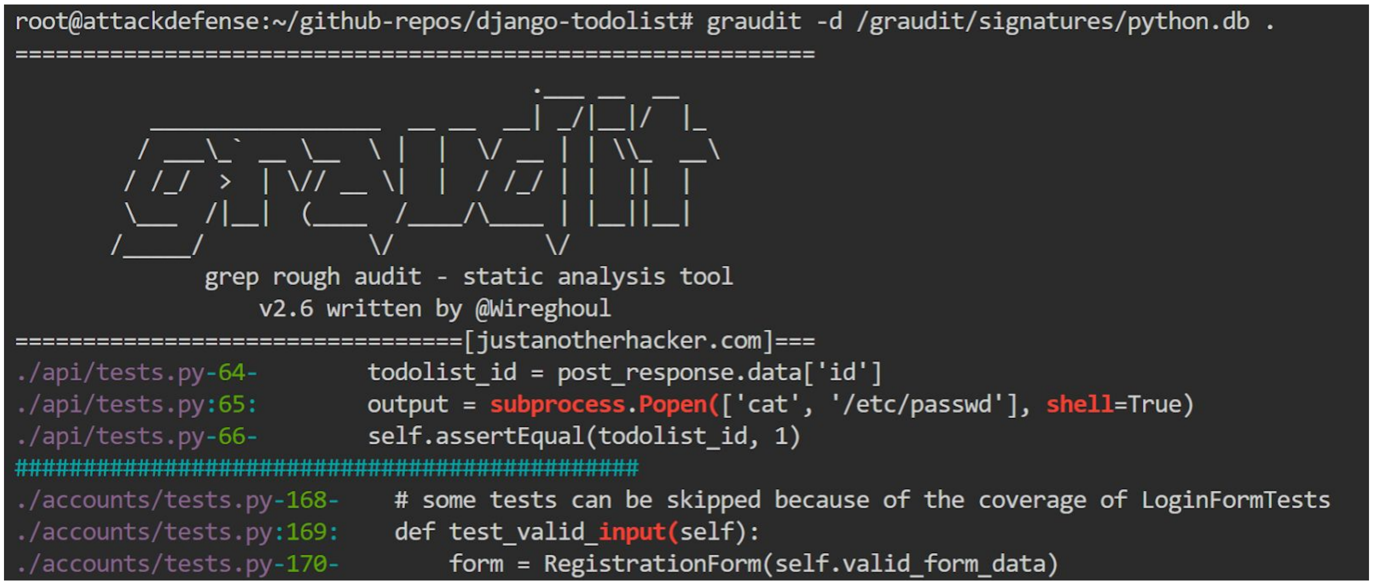
Issues Detected:

- Found Database Credentials

Example 2: Find Sensitive information using Python Signature

Step A: Run the graudit tool and pass the Python signature database to the tool

Command: graudit -d /graudit/signatures/python.db .



```
root@attackdefense:~/github-repos/django-todolist# graudit -d /graudit/signatures/python.db .
===========================================================
                                    .___ __  __
          _________________  __ __  __| _/|__|/  |_
         / ___\_` __ \__  \ |  |  \/ __ | | \\_  __\
        / /_/  >  | \// __ \|  |  / /_/ | |  ||  |
        \___  /|__|  (____  /____/\____ | |__||__|
       /_____/            \/           \/
              grep rough audit - static analysis tool
                  v2.6 written by @Wireghoul
=================================[justanotherhacker.com]===
./api/tests.py-64-          todolist_id = post_response.data['id']
./api/tests.py:65:          output = subprocess.Popen(['cat', '/etc/passwd'], shell=True)
./api/tests.py-66-          self.assertEqual(todolist_id, 1)
##############################################
./accounts/tests.py-168-    # some tests can be skipped because of the coverage of LoginFormTests
./accounts/tests.py:169:    def test_valid_input(self):
./accounts/tests.py-170-        form = RegistrationForm(self.valid_form_data)
```

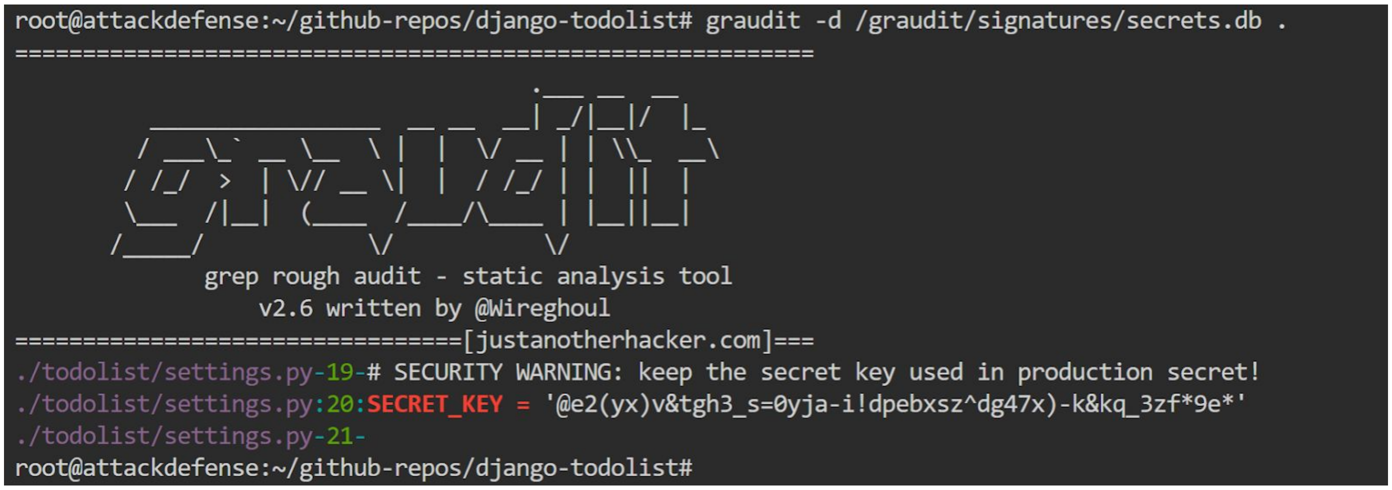
Issues Detected

- Entry to read sensitive data from /etc/passwd file

Example 3: Find sensitive information using the secret keys based signature

Step A: Run the graudit tool and pass the secrets signature database to the tool

Command: graudit -d /graudit/signatures/secrets.db .



```
root@attackdefense:~/github-repos/django-todolist# graudit -d /graudit/signatures/secrets.db .
===========================================================
                                      .___ __  __
          _________________  __ __  __| _/|__|/  |_
         / ___\_  __ \__  \ |  |  \/ __ | | \\_  __\
        / /_/  >  | \// __ \|  |  / /_/ | |  ||  |
        \___  /|__|  (____  /____/\____ | |__||__|
       /_____/            \/           \/
              grep rough audit - static analysis tool
                  v2.6 written by @Wireghoul
=================================[justanotherhacker.com]===
./todolist/settings.py-19-# SECURITY WARNING: keep the secret key used in production secret!
./todolist/settings.py:20:SECRET_KEY = '@e2(yx)v&tgh3_s=0yja-i!dpebxsz^dg47x)-k&kq_3zf*9e*'
./todolist/settings.py-21-
root@attackdefense:~/github-repos/django-todolist#
```

Issues Detected

- Secret Key found

Learnings

Perform Static code analysis using the Graudit tool.