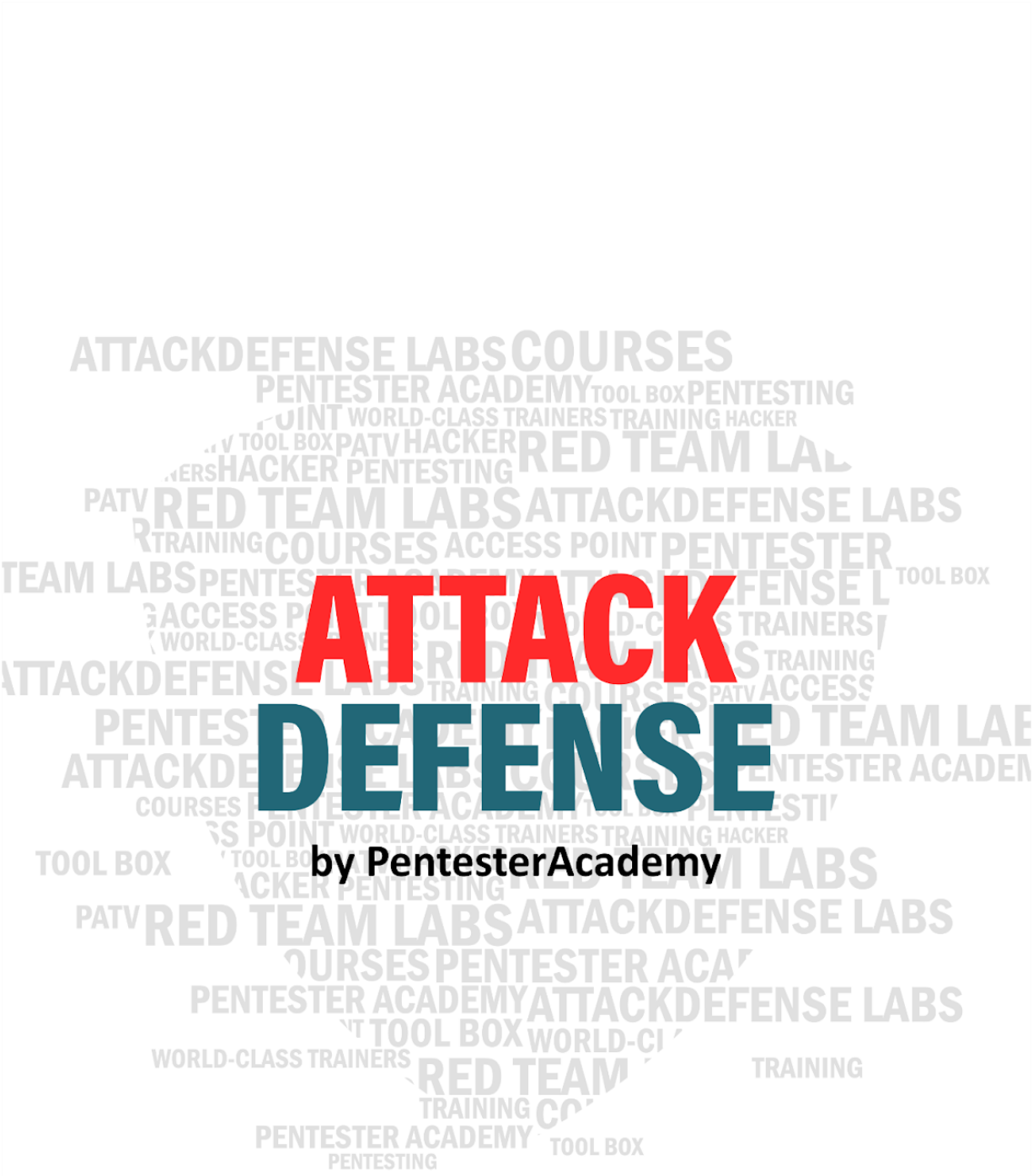
ATTACK
DEFENSE
by PentesterAcademy

Name	GitSecrets: Finding Hardcoded Credentials
URL	https://www.attackdefense.com/challengedetails?cid=2047
Type	DevSecOps Basics: Sensitive Information Scan

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

GitSecrets tool scans the source code for sensitive information (e.g. SSH keys, AWS tokens) before pushing it to a Git repository. It is triggered by pre-commit hooks.

A Kali CLI machine (kali-cli) is provided to the user with GitSecrets 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 with GitSecrets tool and find the sensitive information!.

Instructions:

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

Solution

Step 1: Check the provided web application.

Command: ls -l github-repos

```
root@attackdefense:~# ls -l github-repos/
total 4
drwxrwxr-x 7 root root 4096 Sep 15 08:04 django-todolist
root@attackdefense:~#
```

Step 2: Add the git configuration in order to add and commit repositories locally.

Commands:
git config --global user.name "root"
git config --global user.email root@server.xyz

```
root@attackdefense:~#
root@attackdefense:~# git config --global user.name "root"
root@attackdefense:~# git config --global user.email root@server.xyz
root@attackdefense:~#
```

Step 3: Navigate to the django-todolist directory and install git secrets in the project directory.

Commands:
cd github-repos/django-todolist/
git-secrets --install

```
root@attackdefense:~# cd github-repos/django-todolist/
root@attackdefense:~/github-repos/django-todolist# git-secrets --install
✓ Installed commit-msg hook to .git/hooks/commit-msg
✓ Installed pre-commit hook to .git/hooks/pre-commit
✓ Installed prepare-commit-msg hook to .git/hooks/prepare-commit-msg
root@attackdefense:~/github-repos/django-todolist#
```

The pre-commit hooks are installed in the project directory.

Step 4: Create a custom rule for GitSecrets to check for the “password” keyword in the string.

Commands:
git-secrets --add 'password\s*=\s*.+'

```
root@attackdefense:~/github-repos/django-todolist#
root@attackdefense:~/github-repos/django-todolist#
root@attackdefense:~/github-repos/django-todolist# git-secrets --add 'password\s*=\s*.+'
root@attackdefense:~/github-repos/django-todolist#
```

Step 5: Run the git commit command to commit the code.

Commands:
git add .
git commit -m "Added"

```
root@attackdefense:~/github-repos/django-todolist# git add .
root@attackdefense:~/github-repos/django-todolist# git commit -m "Added"
accounts/forms.py:18:    password = forms.CharField(
accounts/forms.py:30:        password = self.cleaned_data.get('password')
accounts/forms.py:49:    password = forms.CharField(
accounts/forms.py:64:        password = self.cleaned_data.get('password')
accounts/views.py:15:                password=request.POST['password']
accounts/views.py:36:                password=request.POST['password']
api/tests.py:13:        self.client.login(username='test', password='test')
api/tests.py:31:        self.client.login(username='admin', password='admin')
api/tests.py:43:        self.client.login(username='test', password='test')
api/tests.py:137:        self.client.login(username='test', password='test')
lists/tests.py:25:        self.client.login(username='test', password='test')

[ERROR] Matched one or more prohibited patterns
```

Issues Detected

- Hardcoded credentials found in the source code.

Learnings

Perform sensitive information scanning with GitSecrets utility.