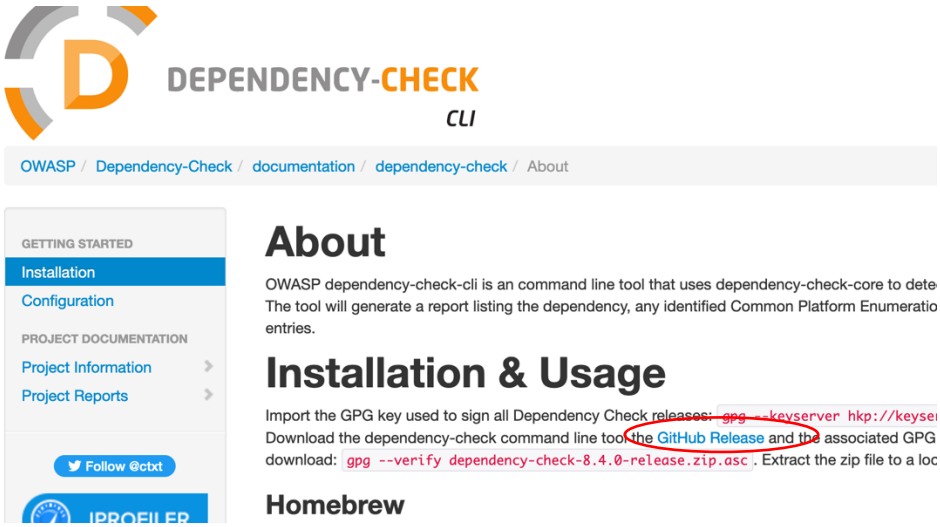
Lab8： SCA and CI/CD Integration

I. SCA Basic

A. OWASP Dependency-Check

1. Install OWASP Dependency-Check
   https://jeremylong.github.io/DependencyCheck/dependency-check-cli/index.html



2. cd ~
   unzip dependency-check-8.4.0-release.zip
   mkdir check
   cd check
   git clone https://github.com/webpwnized/mutillidae.git
   cd ..
3. **Check the Mutillidae**
   ../dependency-check/bin/dependency-check.sh --scan mutillidae
   firefox dependency-check-report.html

Summary

Display: Showing All Dependencies (click to show less)

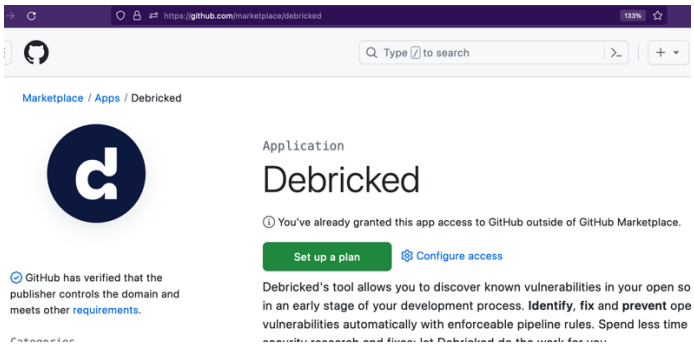
Dependency	Vulnerability IDs	Package	Highest Severity	CVE Count	Confidence	Evidence Count
bookmark-site.js				0		0
commented-php-meter-script.txt.zip				0		2
content-security-policy.js				0		0
ddsmoothmenu-init.js				0		0
ddsmoothmenu.js				0		0
file5.exe	cpe:2.3:a:file_project:file:5:*:*:*:*:*:*:*		HIGH	9	High	4
file8.zip	cpe:2.3:a:file_project:file:8:*:*:*:*:*:*			0	High	4

4. **Practice:** Pease try to clone the dvpwa project from GitHub and check it (https://github.com/anxolerd/dvpwa.git)

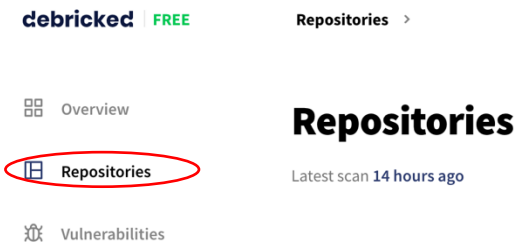
B. Debricked

1. Register the Debricked through GitHub

Note that you must install app from **GitHub Marketplace**

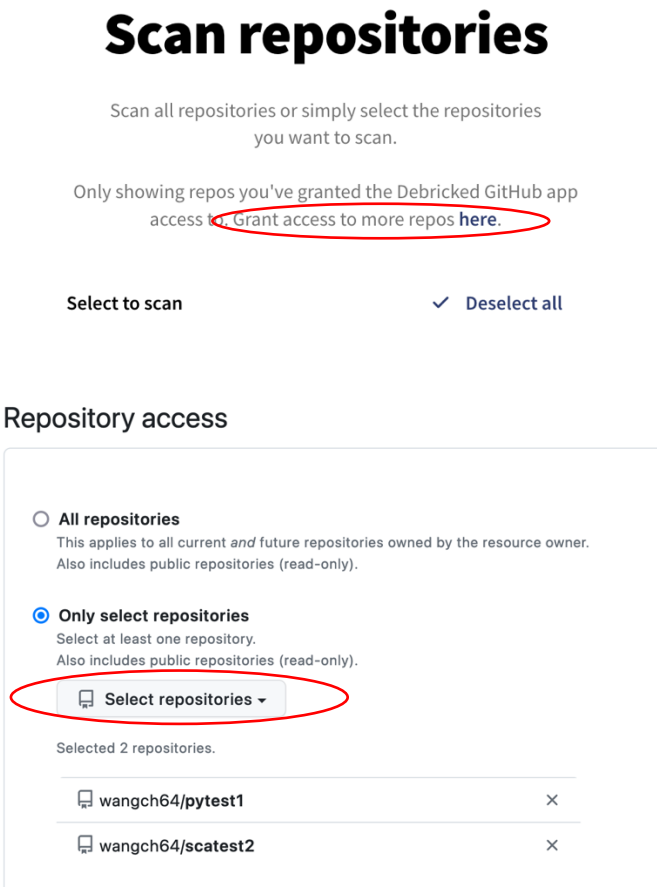


Login ~

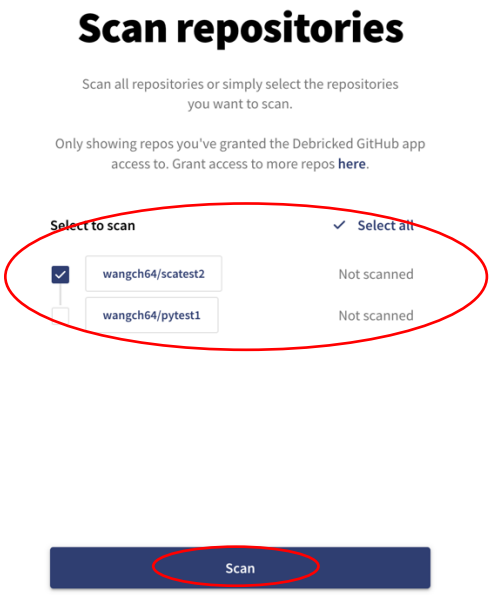


Press "+NEW" => repositories

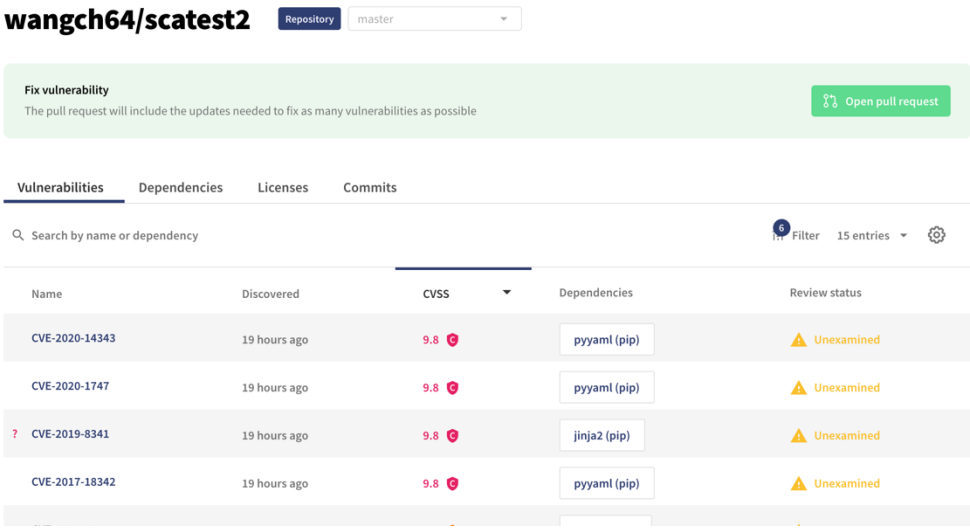
2. You must **grant access to more repos**



3. You can select the repo(s) and run **Scan**



4. See the results



II. SCA with CI/CD Integration

A. GitHub Actions

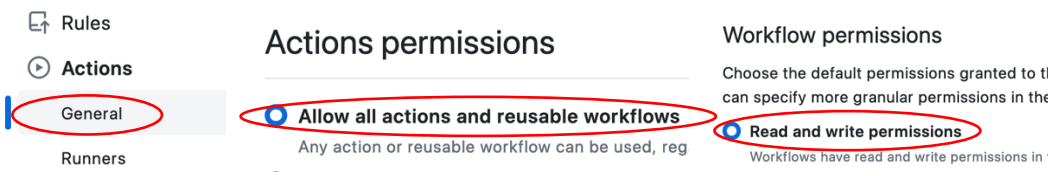
1. pip package safety

ref: https://escape.tech/blog/application-security-101-part-1-software-component-analysis/

create a new repository (named **scatest3**) imported from:

git@github.com:anxolerd/dvpwa.git

Note that you should change Settings=> Actions => General to change **Actions permissions**



2. git clone git@github.com:wangch64/scatest3.git
3. **create actions**

   cd scatest3

   mkdir .github

   cd .github

   mkdir workflows

   cd workflows

   vi sca3.yml

```
on: [push]

jobs:
  dependency_analysis:
    runs-on:   ubuntu-latest
    name: Test dependencies for security flaws
    steps:
      - uses: actions/checkout@v2
      - name:
        run: pip3 install safety && safety check
       shell: bash
```

```
git add .
git commit -m "commit"
git push -u origin master
```

see the results

```
run                                                                    3s
1  ▸ Run pip3 install safety && safety check
4  Defaulting to user installation because normal site-packages is not writeable
5  Collecting safety
6    Downloading safety-2.3.5-py3-none-any.whl (57 kB)
7       ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 57.5/57.5 KB 2.2 MB/s eta 0:00:00
8  Requirement already satisfied: setuptools>=19.3 in /usr/lib/python3/dist-packages (from
   safety) (59.6.0)
9  Requirement already satisfied: Click>=8.0.2 in /usr/lib/python3/dist-packages (from
   safety) (8.0.3)
10 Requirement already satisfied: requests in /usr/lib/python3/dist-packages (from safety)
   (2.25.1)
11 Collecting packaging<22.0,>=21.0
12   Downloading packaging-21.3-py3-none-any.whl (40 kB)
13      ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 40.8/40.8 KB 4.7 MB/s eta 0:00:00
14 Collecting dparse>=0.6.2
15   Downloading dparse-0.6.3-py3-none-any.whl (12 kB)
16 Collecting ruamel.yaml>=0.17.21
17   Downloading ruamel.yaml-0.17.35-py3-none-any.whl (112 kB)
```

B. Debricked with CI/CD

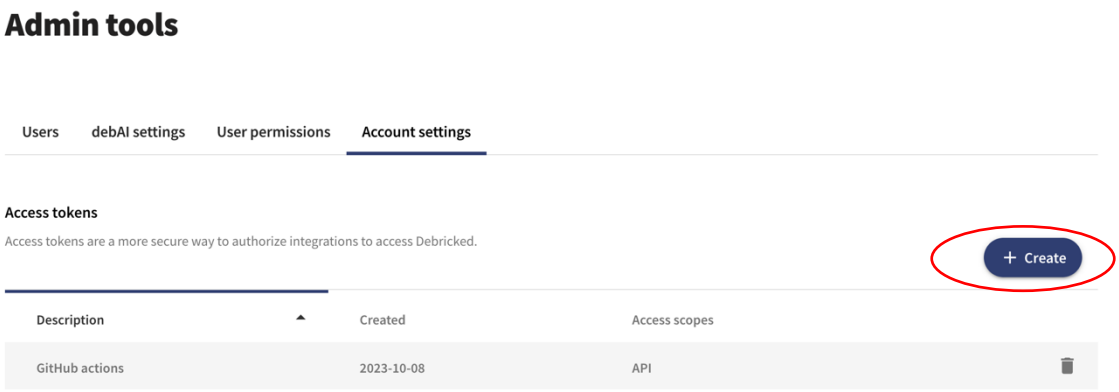
create a new repository (named **scatest4**) imported from:
https://github.com/romanoroth/GitHubDevSecOps.git

cd scatest4
cd .github
mv workflows workold
mkdir workflows
cd workflows
vi sca4.yml

ref: https://github.com/marketplace/actions/debricked-vulnerability-scan

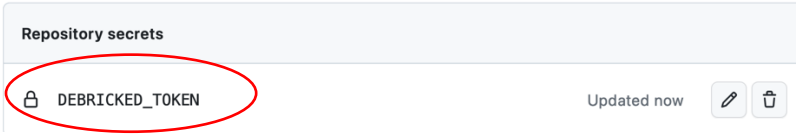
---

Create api-token for Debricked
Admin tools => Account Setting =>Create



In Github => Settings => Secrets and variables =>Actions =>New repository secret

Neame: DEBRICKED_TOKEN



```
git add .
git commit -m "commit"
git push -u origin main
```

See the results

```
vulnerabilities-scan
succeeded now in 1m 11s                                   Search logs
76  |   | net.minidev:json-smart (Maven):
77  |   | ----------------------------------------
78  |   | * URL: https://debricked.com/app/en/dependency/20163?repositoryId=73838
79  |   | * Vulnerabilities:
80  |   | * CVE-2023-1370
81  |   |   * URL: https://debricked.com/app/en/vulnerability/266817?repositoryId=73838&commitId=2108688
82  |   |   * CVSS2: 0
83  |   |   * CVSS3: 7.5
84  |   | * CVE-2021-27568
```

C. Snyk SCA with CI/CD (Python Example)

Test python project: **scatest3**
From GitHub marketplace serach Snyk actions=> Python 3.10

Add snyk.ymal to ~/scatest3/.github/workflows
Uploading Snyk scan results to GitHub Code Scanning

```
name: Example workflow for Python using Snyk
on: push
jobs:
  security:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@master
      - name: Run Snyk to check for vulnerabilities
        uses: snyk/actions/python-3.10@master
        continue-on-error: true # To make sure that SARIF upload gets called
        env:
```

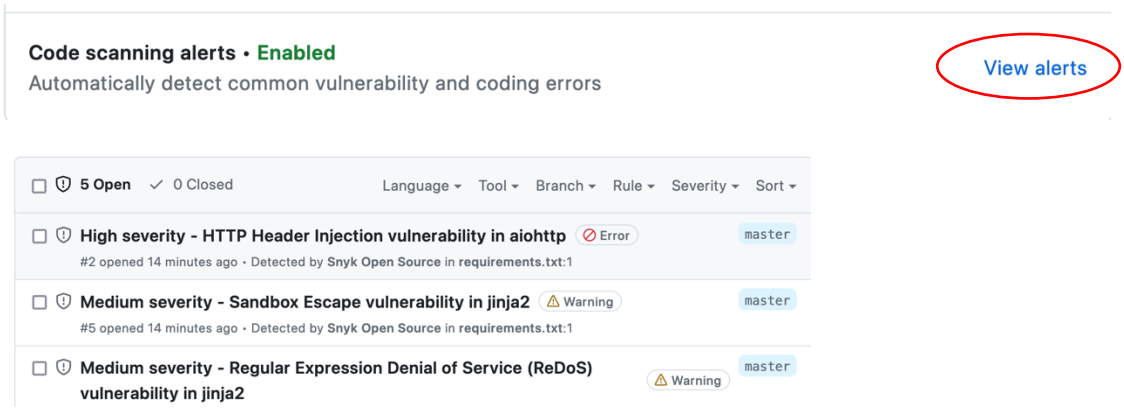
```
        SNYK_TOKEN: ${{ secrets.SNYK_TOKEN }}
      with:
        args: --sarif-file-output=snyk.sarif
    - name: Upload result to GitHub Code Scanning
      uses: github/codeql-action/upload-sarif@v2
      with:
        sarif_file: snyk.sarif
```

```
git add .
git commit -m "commit"
git push -u origin master
```

See the results:

```
Run Snyk to check for vulnerabilities                                  1m 29s
 9
10  Testing /github/workspace...
11
12  Tested 18 dependencies for known issues, found 5 issues, 13 vulnerable paths.
13
14
15  Issues to fix by upgrading dependencies:
16
17    Upgrade aiohttp@3.5.3 to aiohttp@3.8.5 to fix
18    ✗ Open Redirect [Medium Severity][https://security.snyk.io/vuln/SNYK-PYTHON-AIOHTTP-
    1079232] in aiohttp@3.5.3
19      introduced by aiohttp@3.5.3 and 2 other path(s)
20    ✗ HTTP Request Smuggling [Medium Severity][https://security.snyk.io/vuln/SNYK-PYTHON-
    AIOHTTP-5798483] in aiohttp@3.5.3
21      introduced by aiohttp@3.5.3 and 2 other path(s)
22    ✗ HTTP Header Injection [High Severity][https://security.snyk.io/vuln/SNYK-PYTHON-
    AIOHTTP-1584144] in aiohttp@3.5.3
23      introduced by aiohttp@3.5.3 and 2 other path(s)
24
25    Upgrade jinja2@2.10 to jinja2@2.11.3 to fix
```

In "Security"



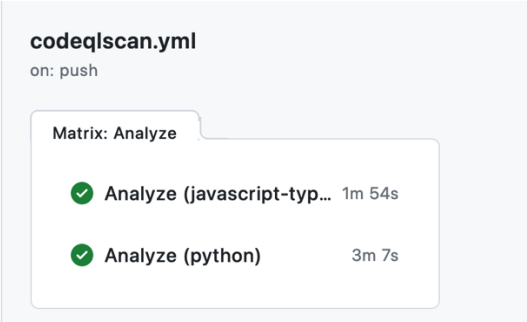
D. CodeQL SAST with CI/CD (Python and Javascript Example)

Test python project: **scatest3**
Settings -> Code security and analysis -> Code scanning
Set up CodeQL (Advanced)

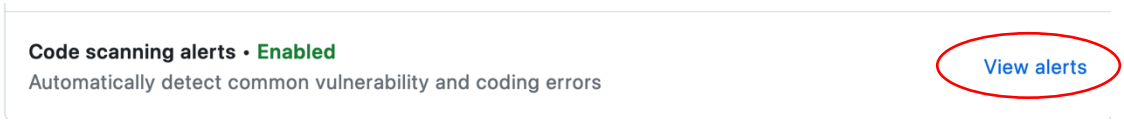
Copy the yml file (do not save it or commit it) to your local machine named: codeqlscan.yml

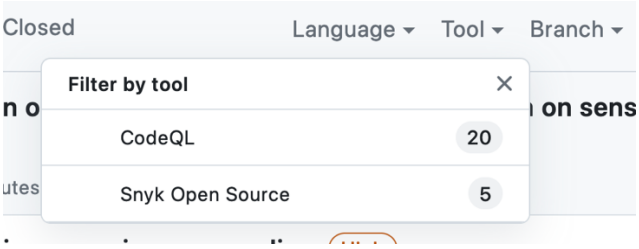
```
git add .
git commit -m "commit"
git push -u origin master
```

See the results:



In "Security"





Select Language -> Python:

See the vulnerability

sqli/dao/**user.py**:41

```
38              return User.from_raw(await cur.fetchone())
39
40      def check_password(self, password: str):
41          return self.pwd_hash == md5(password.encode('utf-8')).hexdigest()
```

Sensitive data (password) is used in a hashing algorithm (MD5) that is insecure for password hashing, since it is not a computationally expensive hash function.

CodeQL Show paths

E.