Continuous Integration Pipeline - Using Vulnerable Pygoat Web Application

Introduction

In this project, a DevSecOps CI pipeline was developed using GitHub Actions, focusing on enhancing the security of the development lifecycle. This pipeline integrated Pygoat, a deliberately vulnerable application was used who code was further analyzed with Bandit, a Python security linter, through GitHub Actions. Additionally, Docker Scout was employed for container image scanning, ensuring the security of the containerized applications.

The artifacts generated from these processes were then uploaded to Splunk for further analysis, providing a centralized platform for comprehensive security assessment. This project demonstrated the integration of security tools within the CI pipeline, aiming to identify and mitigate vulnerabilities early in the development process.

The project serves as a practical example of implementing DevSecOps principles, showcasing the importance of incorporating security checks into the continuous integration process. Through the use of GitHub Actions, Bandit, Docker Scout, and Splunk, the project achieved a balance between development efficiency and security rigor.

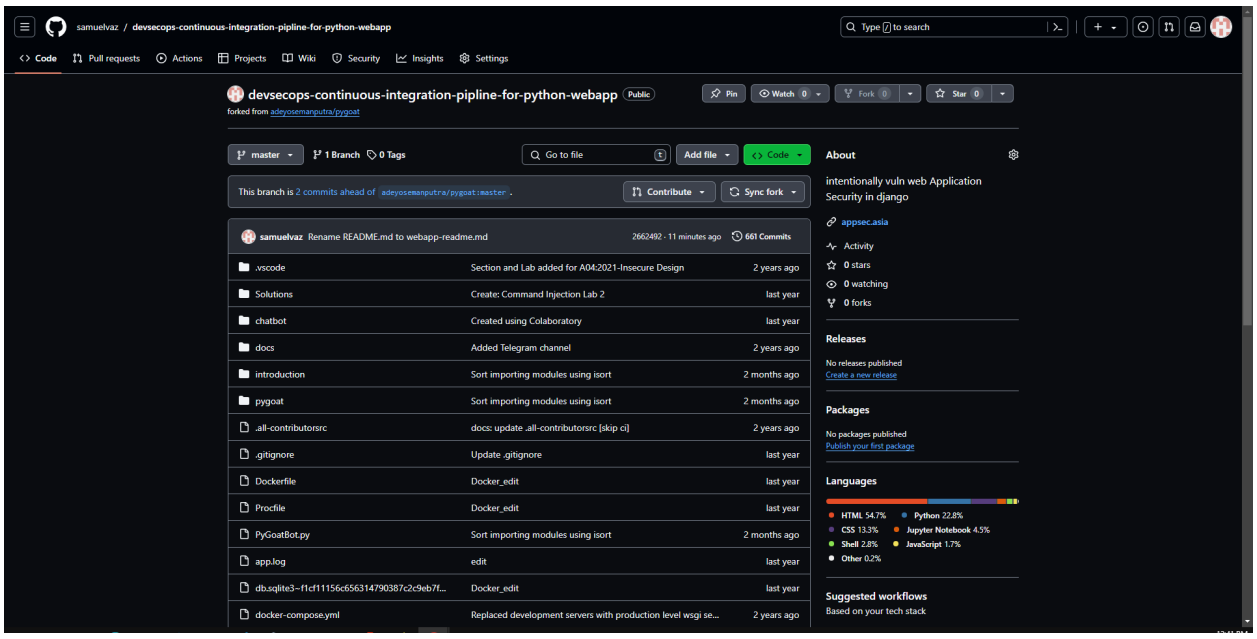
Tools Used

Following tools were used in order to implement the Continuous Integration Pipeline

1. GitHub Actions
2. Bandit
3. Docker Scout
4. Splunk

Forking Pygoat Web App

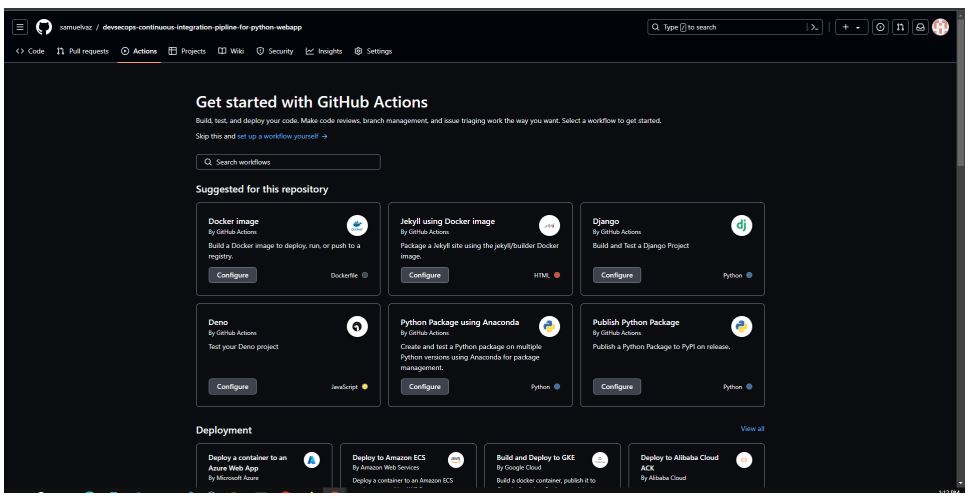
- The vulnerable web application that is used to analyze: https://github.com/adeyosemanputra/pygoat

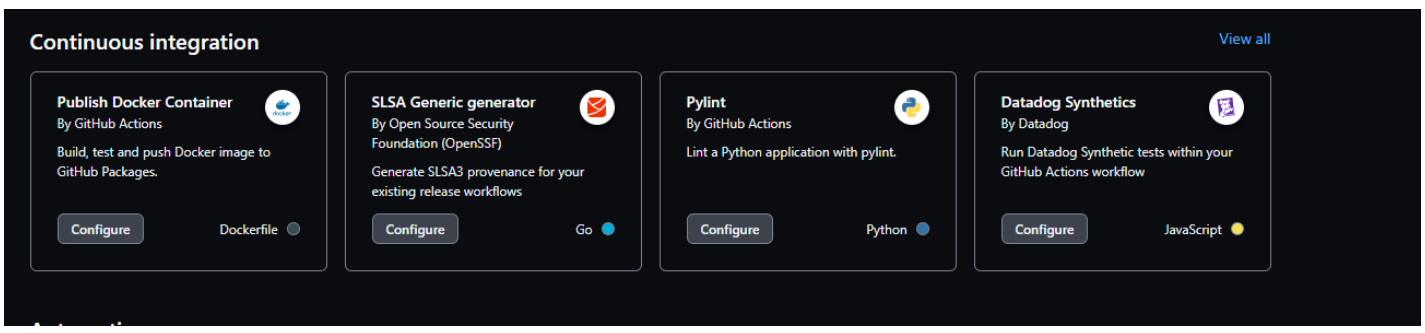


GitHub Actions

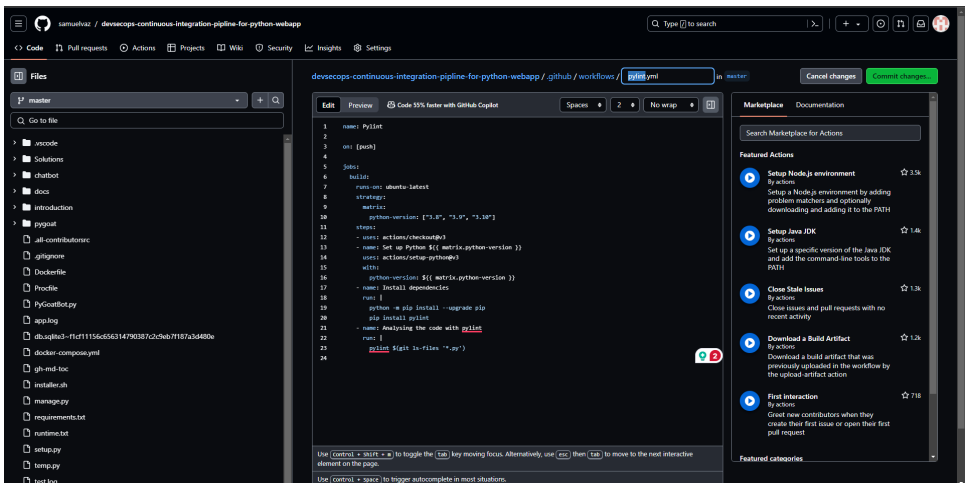
Step 1: Configure PyLint

- Head over to GitHub action, and select Pylint by Github actions for Continuous Integration



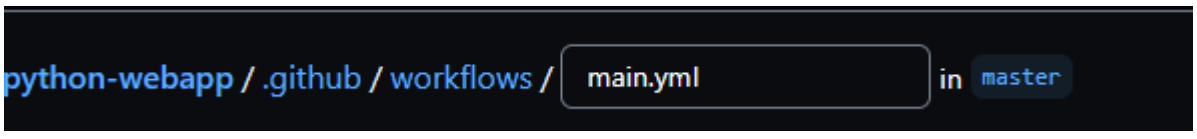


Click `configure` and this should create a `.github/workflows/pylint.yml` file in your GitHub repository.



Step 2: Rename YAML

- Rename `pylint.yml` to `main.yml`



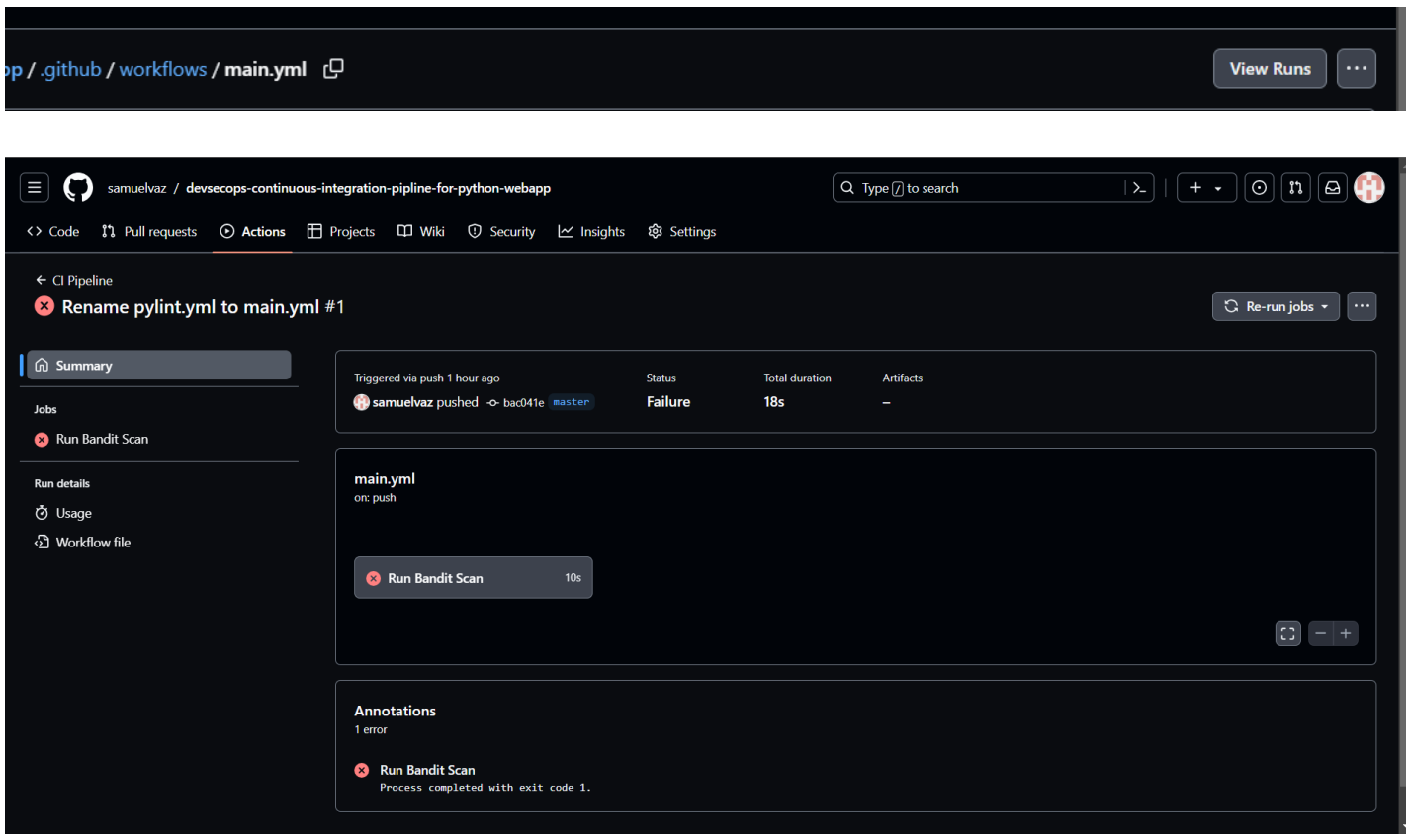
Step 3: Creating Bandit scan

- Bandit is a tool used for automated security testing, specifically for Static Application Security Testing (SAST), to identify vulnerabilities in software code during the development phase.
- Configure edit `main.yml` as below

```
name: CI Pipeline

on: [push]

jobs:
```

```
  sast_scan:
    name: Run Bandit Scan
    runs-on: ubuntu-latest

    steps:
    - name: Checkout Code
        #replace 4.1.2 with the latest or any other version of your choice
      uses: actions/checkout@v4.1.2

    - name: Set up Python
      uses: actions/setup-python@v5.1.0
      with:
        python-version: 3.8

    - name: Install Bandit #installing bandit on ubuntu latest machine
      run: pip install bandit

    - name: Run Bandit Scan
      run: bandit -ll -ii -r .
```

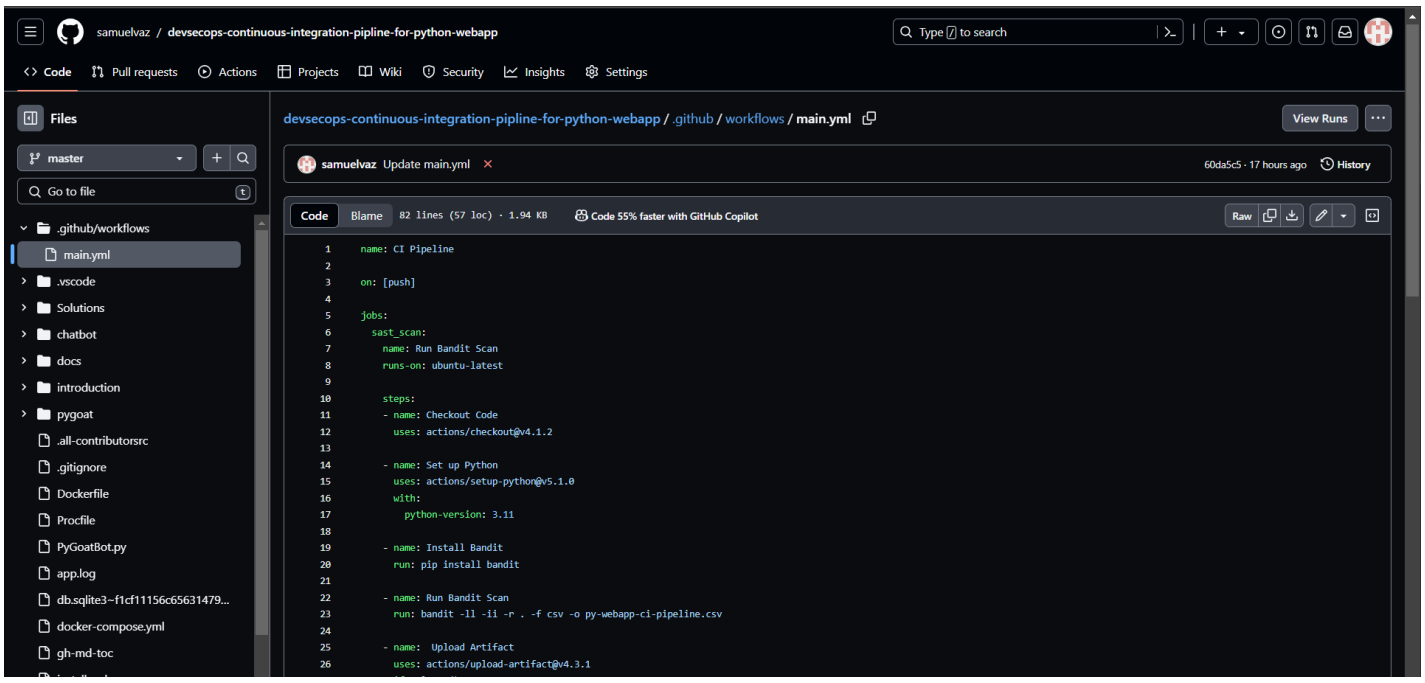
- This bandit scan consists of scanning the code repo and listing only medium and high severity with medium and high confidence.
- `-ll` for medium and high severity level
- `-ii` for a medium and high confidence level
- `-r .` mean scanning current directory recursively
- Commit the changes and click on `View Runs`



- Failed Bandit scan means the scan has encountered vulnerabilities in the code.

Step 4: Creating a output file for all the findings

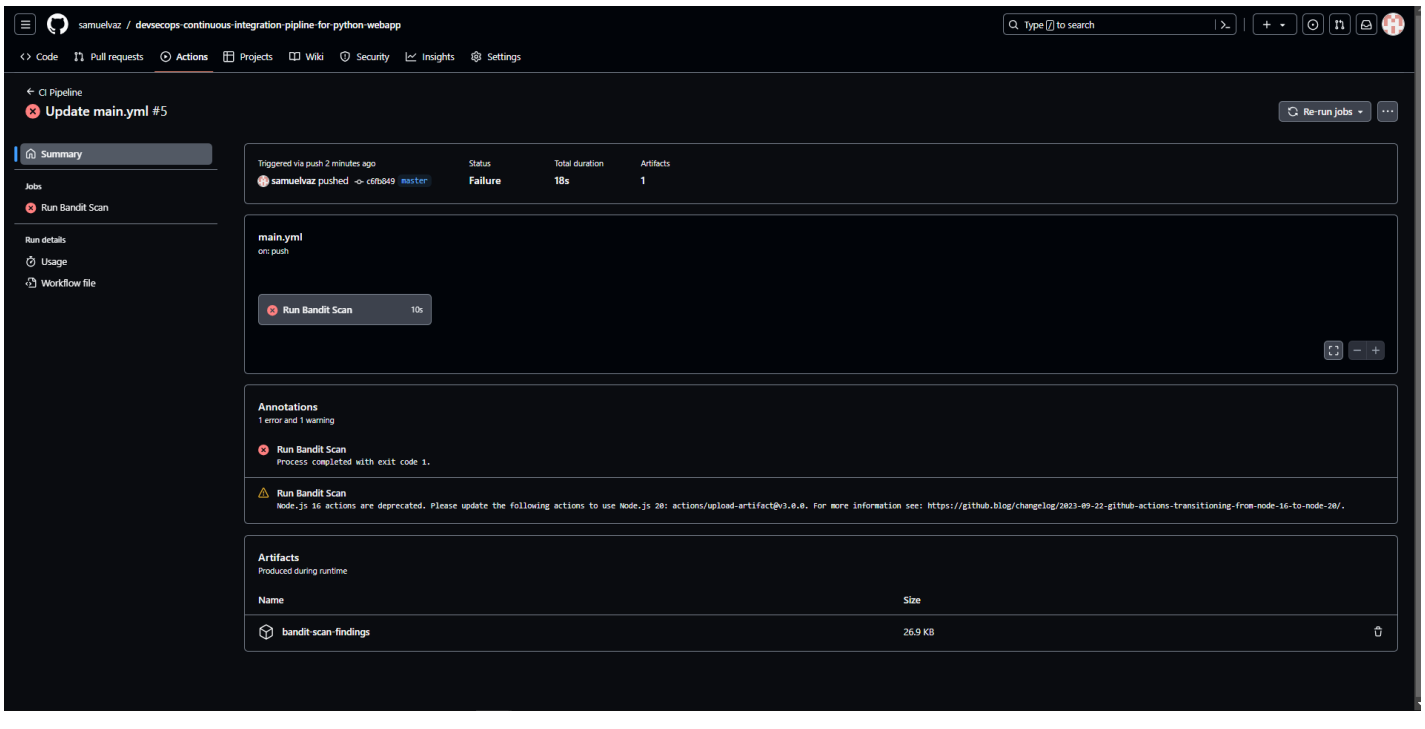
- For all the bandit scan findings, create output files that can be used for visualizing.
- Using following bandit command to generate output `.csv` file




```
name: CI Pipeline
on: [push]

jobs:
  sast_scan:
    name: Run Bandit Scan
    runs-on: ubuntu-latest

    steps:
    - name: Checkout Code
      uses: actions/checkout@v4.1.2

    - name: Set up Python
      uses: actions/setup-python@v5.1.0
      with:
        python-version: 3.13

    - name: Install Bandit
      run: pip install bandit

    - name: Run Bandit Scan
      run: bandit -ll -ii -r . -f csv -o py-webapp-ci-pipeline.csv

    - name:  Upload Artifact
      uses: actions/upload-artifact@v3.0.0
      if: always()
      with:
        name: bandit-scan-findings
        path: py-webapp-ci-pipeline.csv
```

Step 5: Bandit Scan Output File

- After committing the updated `YAML` code the Bandit scan generated a `.json` file



- The scan has generated an artifact named `'bandit-scan-findings'` which is a machine consumeable CSV file that can be used to further analyze the findings.

Container Image Scanning

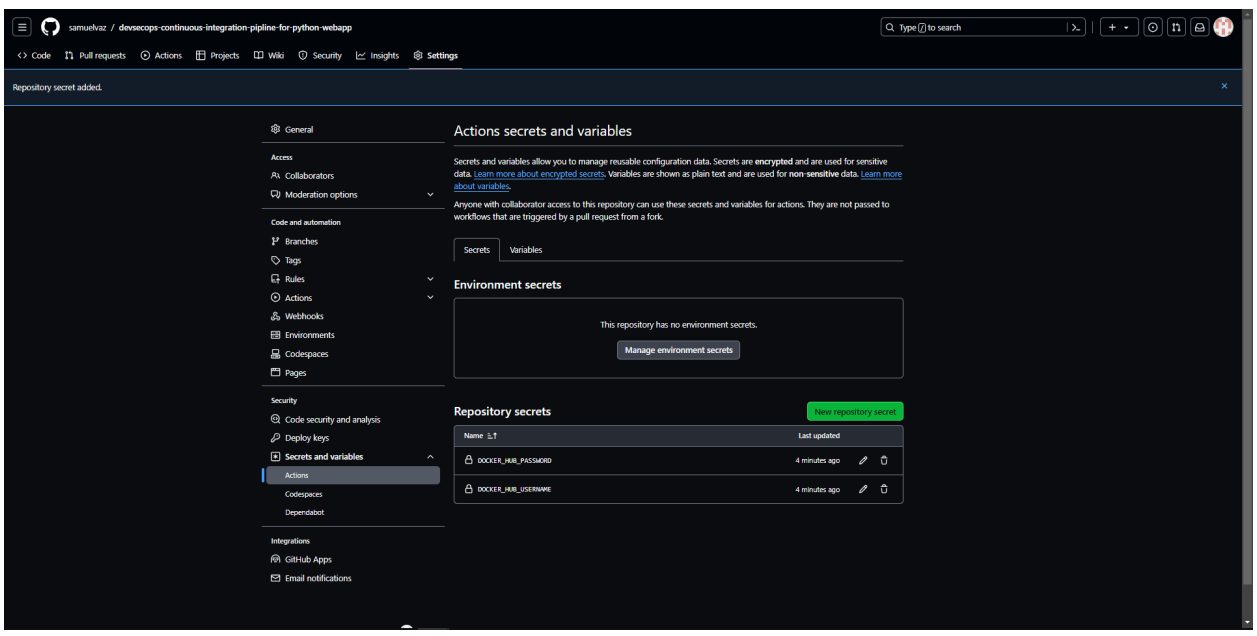
Step 1: Creating Job for Container Image Scanning

- Creating Job in the `main.yml` for scanning the container image
- Below `YAML` code consist the changes made in the previous `YAML` code

```yaml
name: CI Pipeline

on: [push]

jobs:
  sast_scan:
    name: Run Bandit Scan
    runs-on: ubuntu-latest

    steps:
    - name: Checkout Code
      uses: actions/checkout@v4.1.2

    - name: Set up Python
      uses: actions/setup-python@v5.1.0
      with:
        python-version: 3.11

    - name: Install Bandit
      run: pip install bandit

    - name: Run Bandit Scan
      run: bandit -ll -ii -r . -f csv -o py-webapp-ci-pipeline.csv

    - name:  Upload Artifact
      uses: actions/upload-artifact@v4.3.1
      if: always()
      with:
        name: bandit-scan-findings
        path: py-webapp-ci-pipeline.csv
```

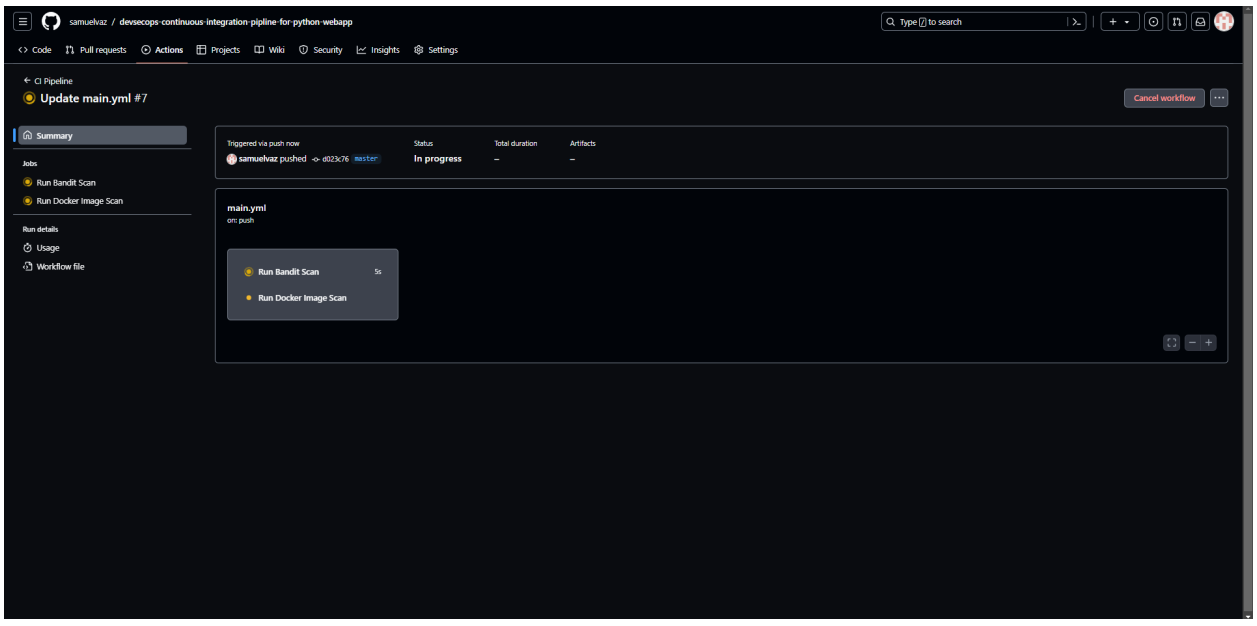
```
image_scan:
 name: Build Image and Run Image Scan
 runs-on: ubuntu-latest

 steps:
 - name: Checkout code
   uses: actions/checkout@v4.1.2

 - name: Set up Docker
   uses: crazy-max/ghaction-setup-docker@v3.1.0

 - name: Build Docker Image
   run: docker build -f Dockerfile -t myapp:latest .

 - name: Docker Scout Scan
   run: |
     curl -fsSL https://raw.githubusercontent.com/docker/scout-cli/main/install.sh -o insta
ll-scout.sh
       sh install-scout.sh
       echo ${{ secrets.DOCKER_HUB_PASSWORD }} | docker login -u ${{ secrets.DOCKER_HUB_USERN
AME }} --password-stdin
       docker scout quickview
       docker scout cves
```

- `${{secrets.docker_hub_password}}` and `${{secrets.docker_hub_username}}` are github repository secrets



Step 2: Committing the changes

- After the updated `main.yml` is committed two jobs are triggered
  - Bandit Scan
  - Docker Image Scan

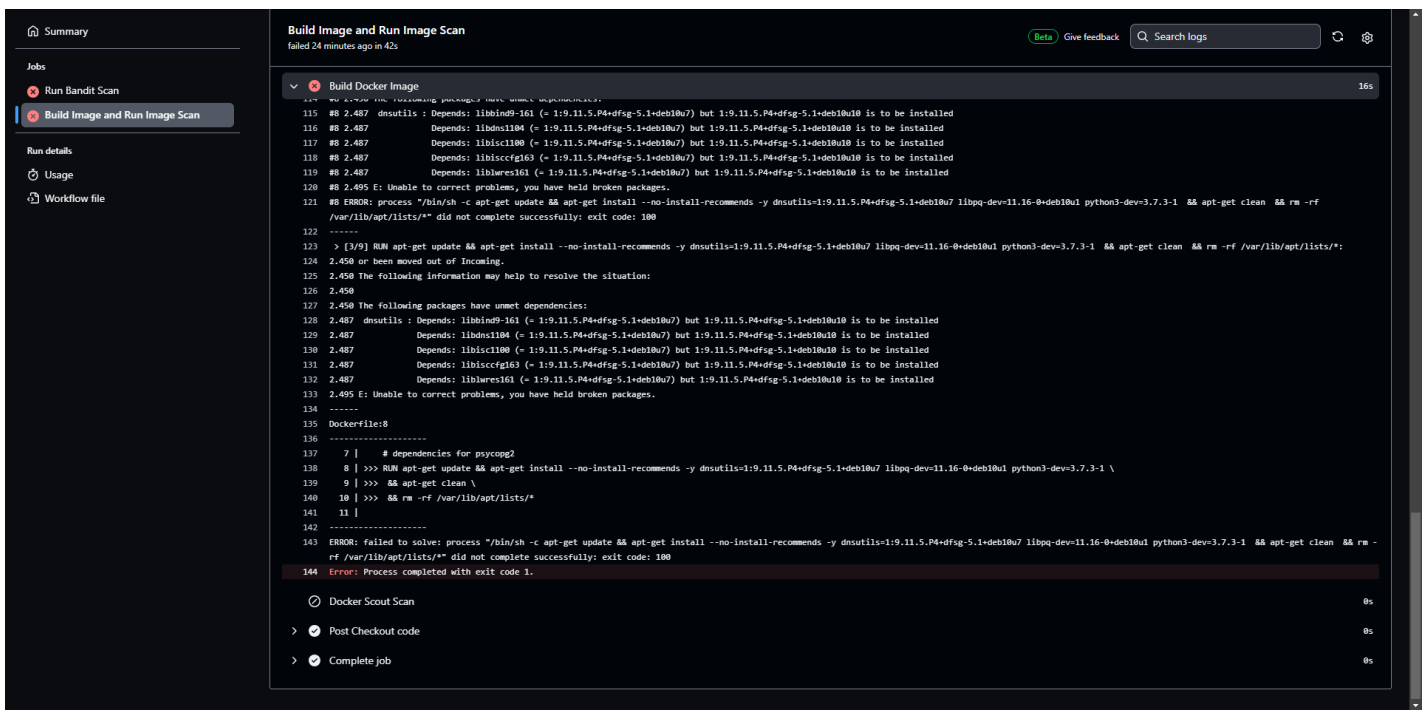


Step 3: Fixing the dependencies

- After the CI pipeline was triggered following errors were encountered as shown

```
#8 [3/9] RUN apt-get update && apt-get install --no-install-recommends -y dnsutils=1:9.11.5.P
4+dfsg-5.1+deb10u7 libpq-dev=11.16-0+deb10u1 python3-dev=3.7.3-1  && apt-get clean  && rm -rf
/var/lib/apt/lists/*
#8 0.245 Get:1 http://security.debian.org/debian-security buster/updates InRelease [34.8 kB]
#8 0.246 Get:2 http://deb.debian.org/debian buster InRelease [122 kB]
#8 0.263 Get:3 http://deb.debian.org/debian buster-updates InRelease [56.6 kB]
#8 0.360 Get:4 http://security.debian.org/debian-security buster/updates/main amd64 Packages
[590 kB]
#8 0.467 Get:5 http://deb.debian.org/debian buster/main amd64 Packages [7909 kB]
#8 0.572 Get:6 http://deb.debian.org/debian buster-updates/main amd64 Packages [8788 B]
#8 1.376 Fetched 8721 kB in 1s (7546 kB/s)
#8 1.376 Reading package lists...
#8 1.818 Reading package lists...
#8 2.267 Building dependency tree...
#8 2.363 Reading state information...
#8 2.394 libpq-dev is already the newest version (11.16-0+deb10u1).
#8 2.394 Some packages could not be installed. This may mean that you have
#8 2.394 requested an impossible situation or if you are using the unstable
#8 2.394 distribution that some required packages have not yet been created
#8 2.394 or been moved out of Incoming.
#8 2.394 The following information may help to resolve the situation:
#8 2.394
#8 2.394 The following packages have unmet dependencies:
#8 2.430  dnsutils : Depends: libbind9-161 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+d
fsg-5.1+deb10u10 is to be installed
#8 2.430             Depends: libdns1104 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+dfs
g-5.1+deb10u10 is to be installed
#8 2.430             Depends: libisc1100 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+dfs
g-5.1+deb10u10 is to be installed
#8 2.430             Depends: libisccfg163 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+d
fsg-5.1+deb10u10 is to be installed
#8 2.430             Depends: liblwres161 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+df
sg-5.1+deb10u10 is to be installed
#8 2.438 E: Unable to correct problems, you have held broken packages.
#8 ERROR: process "/bin/sh -c apt-get update && apt-get install --no-install-recommends -y dn
sutils=1:9.11.5.P4+dfsg-5.1+deb10u7 libpq-dev=11.16-0+deb10u1 python3-dev=3.7.3-1  && apt-get
clean  && rm -rf /var/lib/apt/lists/*" did not complete successfully: exit code: 100
------
 > [3/9] RUN apt-get update && apt-get install --no-install-recommends -y dnsutils=1:9.11.5.P
```

```
4+dfsg-5.1+deb10u7 libpq-dev=11.16-0+deb10u1 python3-dev=3.7.3-1  && apt-get clean  && rm -rf
/var/lib/apt/lists/*:
2.394 or been moved out of Incoming.
2.394 The following information may help to resolve the situation:
2.394
2.394 The following packages have unmet dependencies:
2.430  dnsutils : Depends: libbind9-161 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+dfsg
-5.1+deb10u10 is to be installed
2.430             Depends: libdns1104 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+dfsg-
5.1+deb10u10 is to be installed
2.430             Depends: libisc1100 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+dfsg-
5.1+deb10u10 is to be installed
2.430             Depends: libisccfg163 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+dfsg
-5.1+deb10u10 is to be installed
2.430             Depends: liblwres161 (= 1:9.11.5.P4+dfsg-5.1+deb10u7) but 1:9.11.5.P4+dfsg-
5.1+deb10u10 is to be installed
2.438 E: Unable to correct problems, you have held broken packages.
------
Dockerfile:8
--------------------
   7 |     # dependencies for psycopg2
   8 | >>> RUN apt-get update && apt-get install --no-install-recommends -y dnsutils=1:9.11.
5.P4+dfsg-5.1+deb10u7 libpq-dev=11.16-0+deb10u1 python3-dev=3.7.3-1 \
   9 | >>>  && apt-get clean \
  10 | >>>  && rm -rf /var/lib/apt/lists/*
  11 |
--------------------
ERROR: failed to solve: process "/bin/sh -c apt-get update && apt-get install --no-install-re
commends -y dnsutils=1:9.11.5.P4+dfsg-5.1+deb10u7 libpq-dev=11.16-0+deb10u1 python3-dev=3.7.3
-1  && apt-get clean  && rm -rf /var/lib/apt/lists/*" did not complete successfully: exit cod
e: 100
```



- The Docker build failed due to unmet dependencies in the specified package versions, requiring adjustments to the Dockerfile or a different base image.

> ## Follow commands from the Dockerfile were changed
>
> `RUN apt-get update && apt-get install --no-install-recommends -y dnsutils=1:9.11.5.P4+dfsg-5.1+deb10u7 libpq-dev=11.16-0+deb10u1 python3-dev=3.7.3-1 \`

```
&& apt-get clean \
&& rm -rf /var/lib/apt/lists/*
```

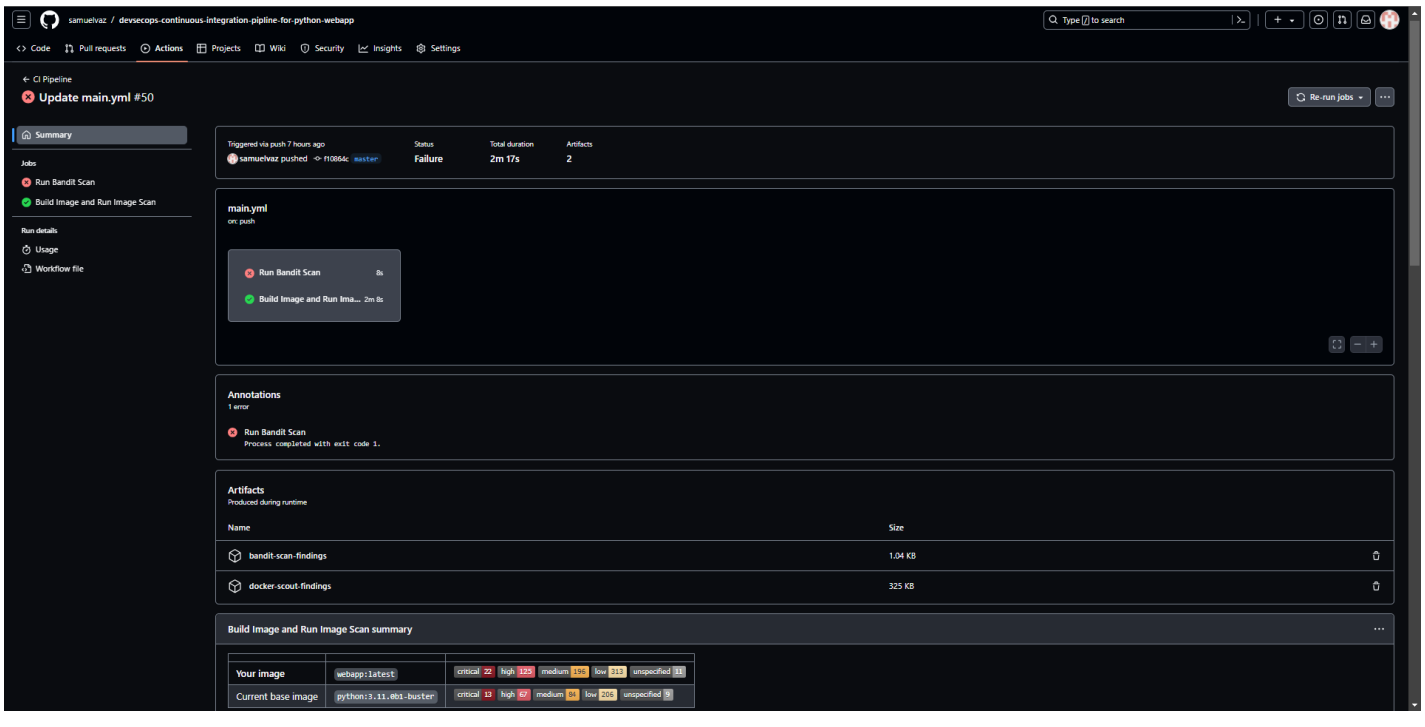
```
FROM python:3.11.0b1-buster

# set work directory
WORKDIR /app

# dependencies for psycopg2

# Original command
# RUN apt-get update && apt-get install --no-install-recommends -y dnsutils=1:9.11.5.P4+dfsg-
5.1+deb10u7 libpq-dev=11.16-0+deb10u1 python3-dev=3.7.3-1 \
#  && apt-get clean \
#  && rm -rf /var/lib/apt/lists/*

# Edited commands - the commands were changed due to dependency issues
RUN apt-get update && apt-get install --no-install-recommends -y dnsutils libpq-dev python3-d
ev \
 && apt-get clean \
 && rm -rf /var/lib/apt/lists/*

# Set environment variables
ENV PYTHONDONTWRITEBYTECODE 1
ENV PYTHONUNBUFFERED 1

# Install dependencies
RUN python -m pip install --no-cache-dir pip==22.0.4
COPY requirements.txt requirements.txt
RUN pip install --no-cache-dir -r requirements.txt

# copy project
COPY . /app/

# install pygoat
EXPOSE 8000

RUN python3 /app/manage.py migrate
WORKDIR /app/pygoat/
CMD ["gunicorn", "--bind", "0.0.0.0:8000", "--workers","6", "pygoat.wsgi"]
```

Step 4: Create Docker Scout Findings Report

- After the Container image scan is complete, it is necessary to generate an output of all the findings that can be used by a vulnerability management tool to further analyze.

```
name: CI Pipeline

on: [push]

jobs:
  sast_scan:
    name: Run Bandit Scan
    runs-on: ubuntu-latest

    steps:
```

```
    - name: Checkout Code
      uses: actions/checkout@v4.1.2

    - name: Set up Python
      uses: actions/setup-python@v5.1.0
      with:
        python-version: 3.11

    - name: Install Bandit
      run: pip install bandit

    - name: Run Bandit Scan
      run: bandit -ll -ii -r . -f csv -o py-webapp-ci-pipeline.csv

    - name:  Upload Artifact
      uses: actions/upload-artifact@v4.3.1
      if: always()
      with:
        name: bandit-scan-findings
        path: py-webapp-ci-pipeline.csv

  image_scan:
   name: Build Image and Run Image Scan
   runs-on: ubuntu-latest

   steps:
   - name: Checkout code
     uses: actions/checkout@v4.1.2

   - name: Set up Docker
     uses: crazy-max/ghaction-setup-docker@v3.1.0

   - name: Build Docker Image
     run: docker build -f Dockerfile -t webapp:latest .

   - name: Docker Scout Filtered Scan
     uses: docker/scout-action@v1.6.4
     if: always()
     with:
       dockerhub-user: ${{ secrets.DOCKER_HUB_USERNAME }}
       dockerhub-password: ${{ secrets.DOCKER_HUB_PASSWORD }}
       command: quickview,cves
       only-severities: critical,high
       sarif-file: dockerScoutReport.json

   - name: Uploading Artifact
     uses: actions/upload-artifact@v4.3.1
     if: always()
     with:
       name: docker-scout-findings
       path: dockerScoutReport.json
```

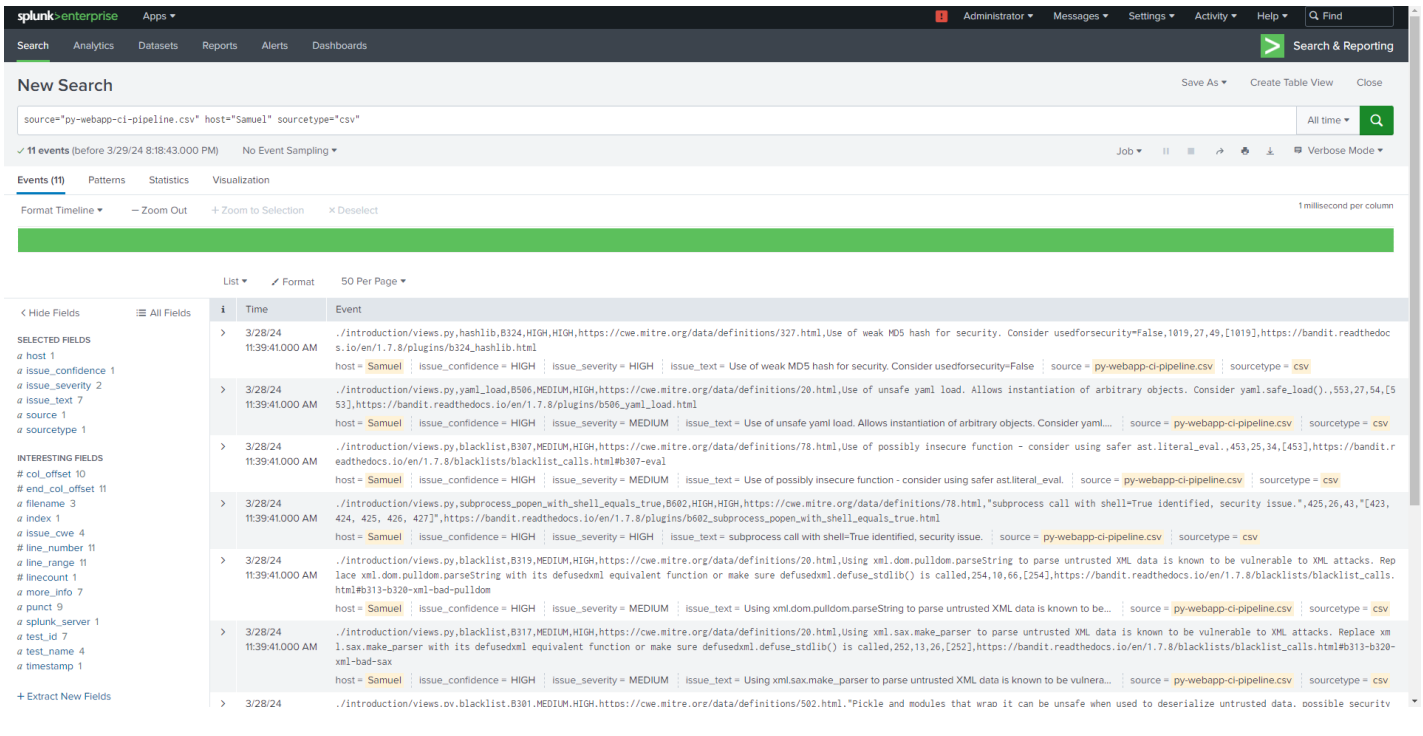


Analyzing Bandit Findings in Splunk

After scanning for the security issues in the application it is necessary to analyze the findings

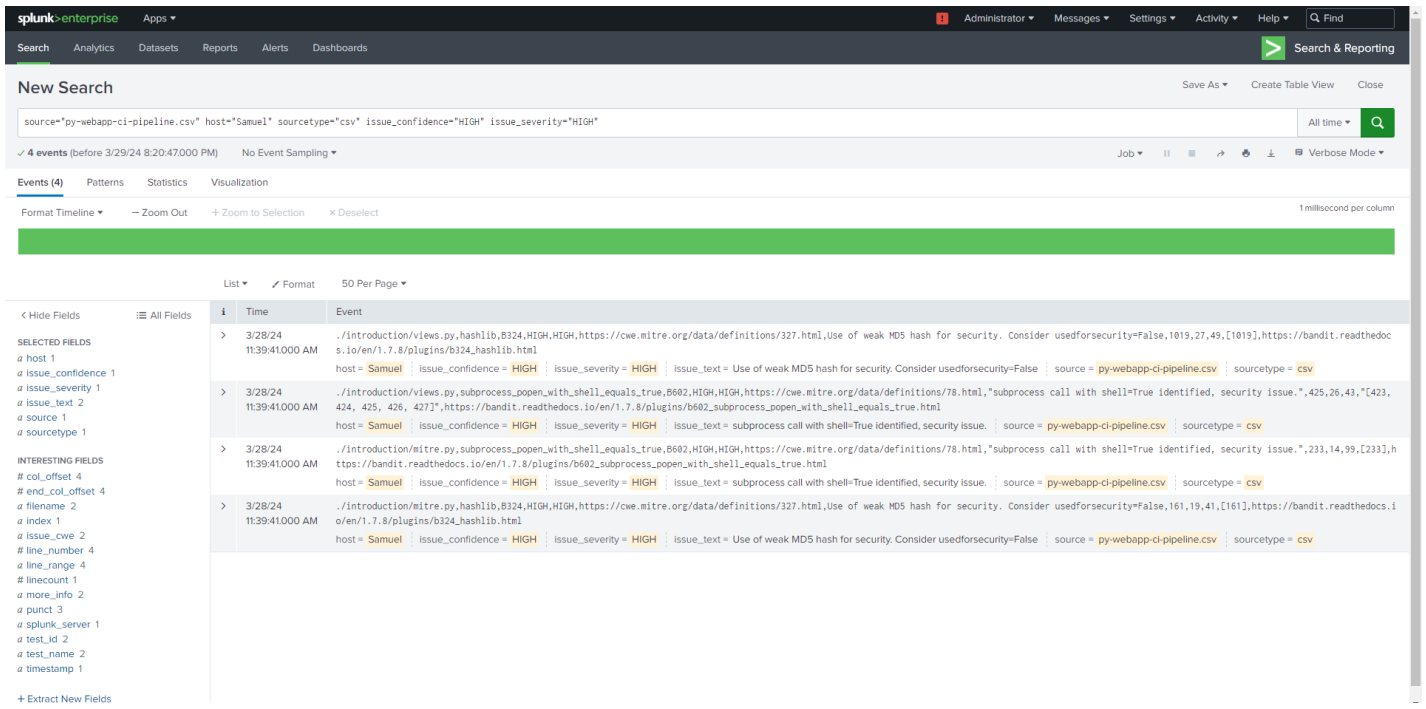
Further, the findings are ingested in Splunk Enterprise to analyze the findings.

Step 1: Analyzing the findings



Step 2: High Confidence and High Severity Findings

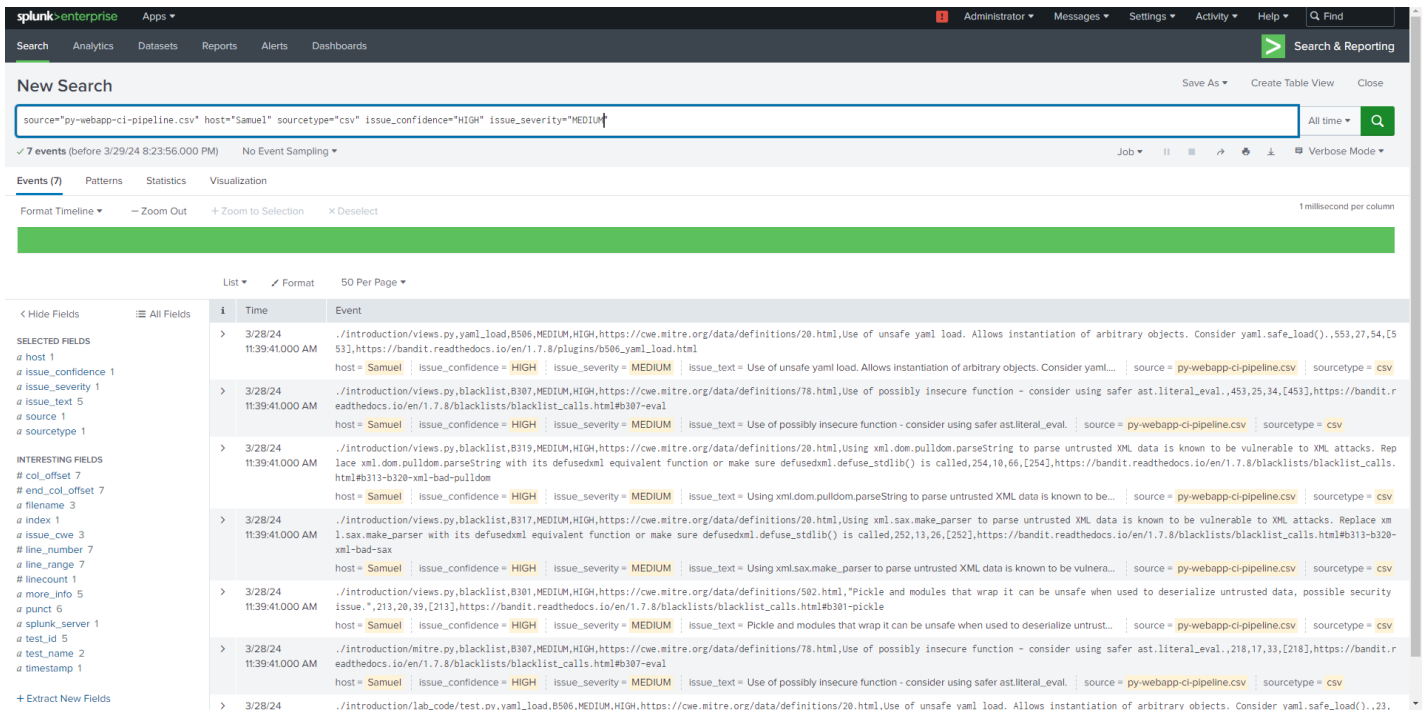
- Search Query: `source="py-webapp-ci-pipeline.csv" host="Samuel" sourcetype="csv" issue_confidence="HIGH" issue_severity="HIGH"`



- The bandit scan found 4 results with High confidence and High severity.

Step 3: High Confident and Medium Severity findings

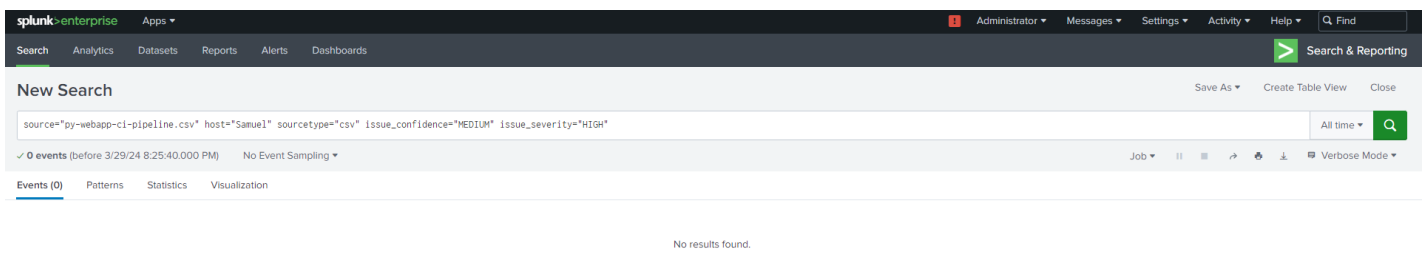
- Search Query: `source="py-webapp-ci-pipeline.csv" host="Samuel" sourcetype="csv" issue_confidence="HIGH" issue_severity="MEDIUM"`



- The bandit scan found 7 results with High confidence and Medium severity.

Step 4: Medium Confidence and High Severity Findings

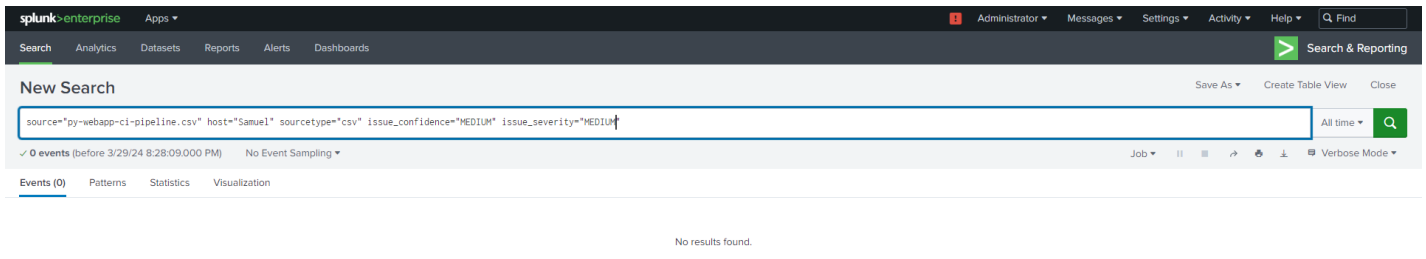
- Search Query: `source="py-webapp-ci-pipeline.csv" host="Samuel" sourcetype="csv" issue_confidence="MEDIUM" issue_severity="HIGH"`



- The bandit scan found zero results with Medium confidence and Medium severity.

Step 5: Medium Confidence and Medium Severity Findings

- Search Query: `source="py-webapp-ci-pipeline.csv" host="Samuel" sourcetype="csv" issue_confidence="MEDIUM" issue_severity="MEDIUM”`



- The bandit scan found zero results with Medium confidence and Medium severity.