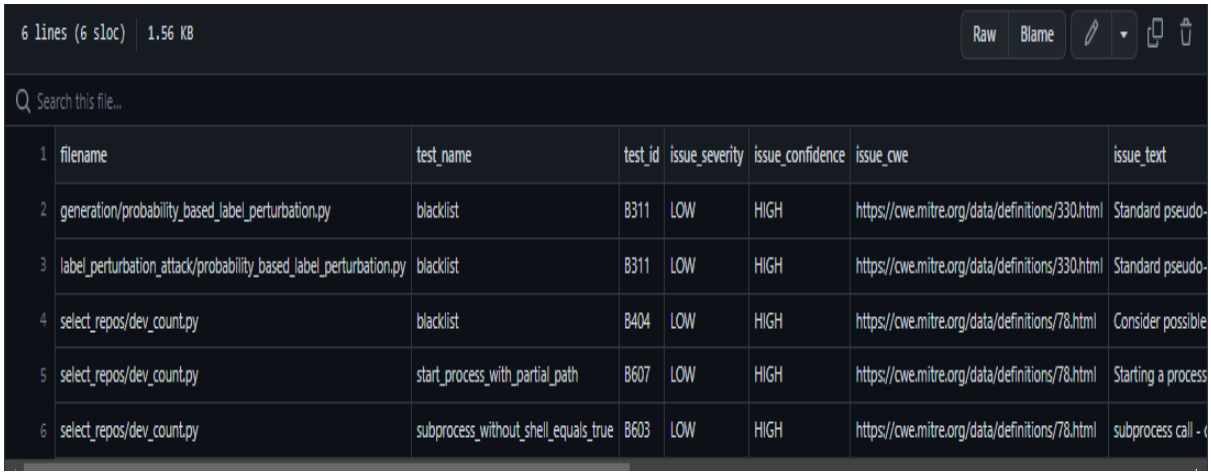
MAS2022-SQA Project Report

Medha Kanakamedala, Anvesh Guduri, Sajith Muralidhar

December 1 2022

1. Security Weakness: -

A git hook is created (pre-commit.sample) is created to run bandit on all python files in the repository to detect security weakness. For the githook to work, copy the pre-commit into .git/hooks which is hidden folder in the repository. The weakness if detected are copied to the bandit_weakness.csv file. Below is the screenshot of the csv file from github repository.



6 lines (6 sloc) | 1.56 KB — Raw | Blame

Search this file...

	filename	test_name	test_id	issue_severity	issue_confidence	issue_cwe	issue_text
2	generation/probability_based_label_perturbation.py	blacklist	B311	LOW	HIGH	https://cwe.mitre.org/data/definitions/330.html	Standard pseudo-
3	label_perturbation_attack/probability_based_label_perturbation.py	blacklist	B311	LOW	HIGH	https://cwe.mitre.org/data/definitions/330.html	Standard pseudo-
4	select_repos/dev_count.py	blacklist	B404	LOW	HIGH	https://cwe.mitre.org/data/definitions/78.html	Consider possible
5	select_repos/dev_count.py	start_process_with_partial_path	B607	LOW	HIGH	https://cwe.mitre.org/data/definitions/78.html	Starting a process
6	select_repos/dev_count.py	subprocess_without_shell_equals_true	B603	LOW	HIGH	https://cwe.mitre.org/data/definitions/78.html	subprocess call -

2. Fuzzing: -

Five methods were implemented for fuzzing in fuzzer.py which is present at the main directory of the repository. The fuzzing type message from each method is as follows.

(a) label_flip_perturbation(): - Missing one argument due to passing of different type which returns a statement to ‘change_unit’ of the passed value.

(b) runDetectionTest(): - Method takes 4 arguments but 6 were passed. A static method does not receive an implicit first argument and can be called on the class or on an instance of the class.

(c) getDevCount(): - Takes from 1 to 3 arguments in a loop but 6 were passed. A static

method does not receive an implicit first argument and can be called on the class or on an instance of the class.

(d) getDevEmailForCommit(): - Takes 2 arguments but 5 were passed. Performs no argument type checking and requires a ‘self’ parameter despite the fact that the method is not defined in a class.

(e) generateAttack(): - Takes 2 positional arguments but 5 were passed. Performs no argument type checking and doesn’t completely fail while passing invalid strings.

Screenshot of Output: -

```
Fuzz: generateAttack Failed
Traceback (most recent call last):
  File "C:\Users\mural\MAS-SQA2022-AUBURN\fuzzer.py", line 15, in fuzzer
    result = method(*fuzzer_args)
TypeError: generateAttack() takes 2 positional arguments but 5 were given
Fuzz: generateAttack Failed
Traceback (most recent call last):
  File "C:\Users\mural\MAS-SQA2022-AUBURN\fuzzer.py", line 15, in fuzzer
    result = method(*fuzzer_args)
TypeError: generateAttack() takes 2 positional arguments but 5 were given
Fuzz: generateAttack Failed
Traceback (most recent call last):
  File "C:\Users\mural\MAS-SQA2022-AUBURN\fuzzer.py", line 15, in fuzzer
    result = method(*fuzzer_args)
TypeError: generateAttack() takes 2 positional arguments but 5 were given
Fuzz: generateAttack Failed
Traceback (most recent call last):
  File "C:\Users\mural\MAS-SQA2022-AUBURN\fuzzer.py", line 15, in fuzzer
    result = method(*fuzzer_args)
TypeError: generateAttack() takes 2 positional arguments but 5 were given
```

3. Forensics

For forensics, logging is done for all five methods that were used in fuzz.py for fuzzed. The logging includes the method called and the argument passed through the method which invokes the error that was used for fuzzing. All the logs are stored in logger.log file. This is how the log looks like.

DEBUG:matplotlib:matplotlib data path: C:\Users\mural\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\mpl-data

DEBUG:matplotlib:CONFIGDIR=C:\Users\mural\.matplotlib

DEBUG:matplotlib:interactive is False

DEBUG:matplotlib:platform is win32

DEBUG:matplotlib:CACHEDIR=C:\Users\mural\.matplotlib

DEBUG:matplotlib.font_manager:Using fontManager instance from C:\Users\mural\.matplotlib\fontlist-v330.json

DEBUG:tensorflow:Falling back to TensorFlow client; we recommended you install the Cloud TPU client directly with pip install cloud-tpu-client.

DEBUG:h5py._conv:Creating converter from 7 to 5

DEBUG:h5py._conv:Creating converter from 5 to 7

DEBUG:h5py._conv:Creating converter from 7 to 5

DEBUG:h5py._conv:Creating converter from 5 to 7

Conclusion: -

Successfully created a githook to scan all python files to detect vulnerability and create a log file to log all methods used for fuzzing in fuzz.py. Developing the tools and to integrate them into GitHub helps in build a resilient project which helps in keeping track of the progress of the project.