SQA Warriors

Jackson Bacon, Tanner Finlay, Paul Summerford

COMP 5710

12/1/22

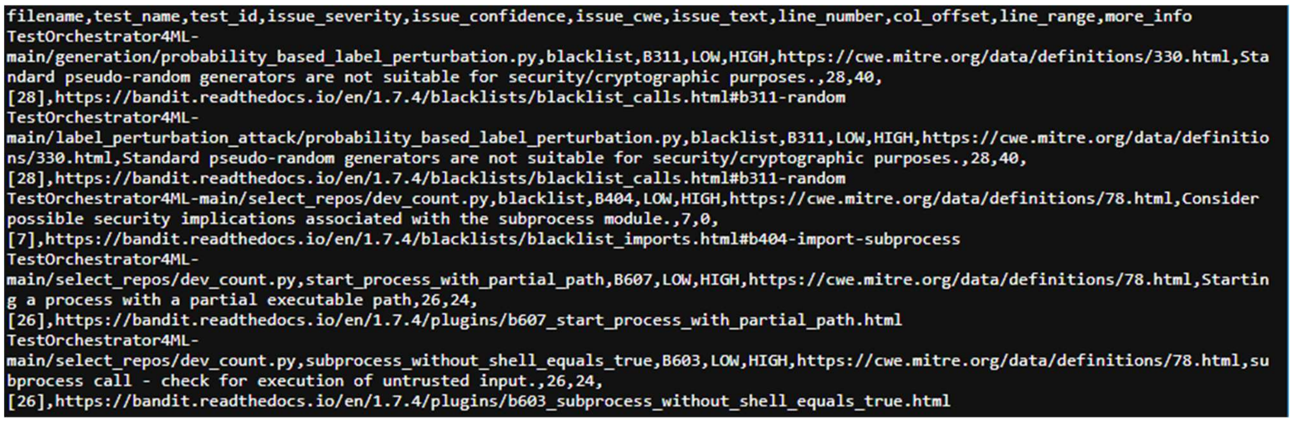
Final Project Report

Security Weaknesses

In order to detect vulnerabilities, a bandit call was added to a pre-commit Git hook. The bandit call is as follows:

```
bandit –r –f csv –a file –o ../../security_weaknesses.csv
TestOrchestrator4ML-main/
```

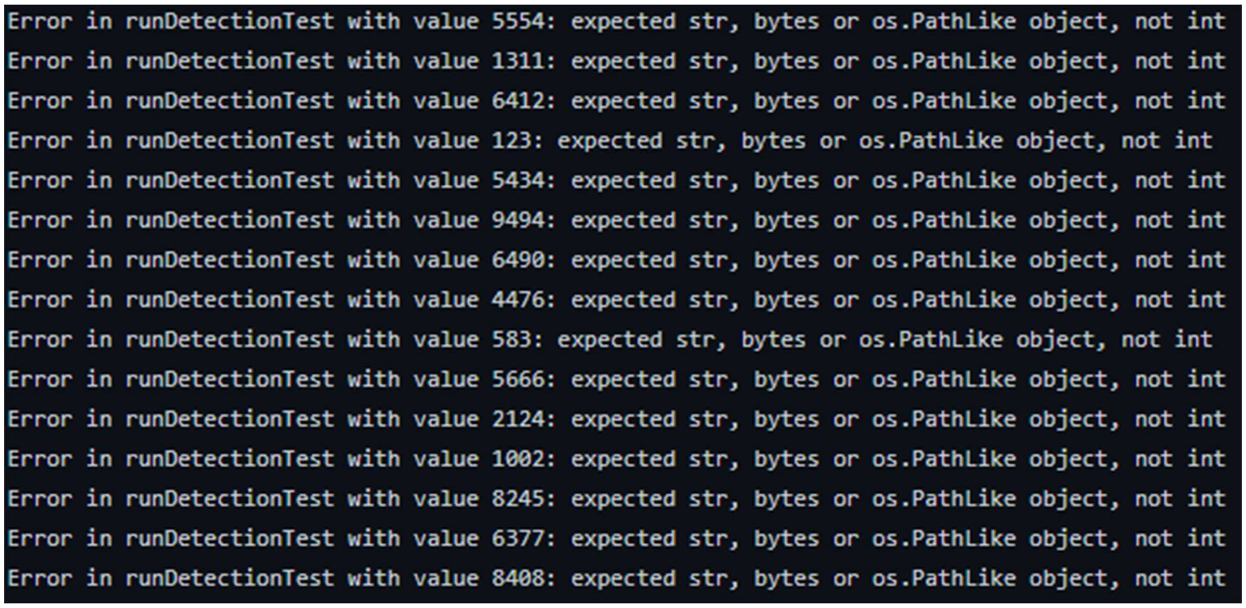
A screenshot of the output file can be seen here:



```
filename,test_name,test_id,issue_severity,issue_confidence,issue_cwe,issue_text,line_number,col_offset,line_range,more_info
TestOrchestrator4ML-
main/generation/probability_based_label_perturbation.py,blacklist,B311,LOW,HIGH,https://cwe.mitre.org/data/definitions/330.html,Sta
ndard pseudo-random generators are not suitable for security/cryptographic purposes.,28,40,
[28],https://bandit.readthedocs.io/en/1.7.4/blacklists/blacklist_calls.html#b311-random
TestOrchestrator4ML-
main/label_perturbation_attack/probability_based_label_perturbation.py,blacklist,B311,LOW,HIGH,https://cwe.mitre.org/data/definitio
ns/330.html,Standard pseudo-random generators are not suitable for security/cryptographic purposes.,28,40,
[28],https://bandit.readthedocs.io/en/1.7.4/blacklists/blacklist_calls.html#b311-random
TestOrchestrator4ML-main/select_repos/dev_count.py,blacklist,B404,LOW,HIGH,https://cwe.mitre.org/data/definitions/78.html,Consider
possible security implications associated with the subprocess module.,7,0,
[7],https://bandit.readthedocs.io/en/1.7.4/blacklists/blacklist_imports.html#b404-import-subprocess
TestOrchestrator4ML-
main/select_repos/dev_count.py,start_process_with_partial_path,B607,LOW,HIGH,https://cwe.mitre.org/data/definitions/78.html,Startin
g a process with a partial executable path,26,24,
[26],https://bandit.readthedocs.io/en/1.7.4/plugins/b607_start_process_with_partial_path.html
TestOrchestrator4ML-
main/select_repos/dev_count.py,subprocess_without_shell_equals_true,B603,LOW,HIGH,https://cwe.mitre.org/data/definitions/78.html,su
bprocess call - check for execution of untrusted input.,26,24,
[26],https://bandit.readthedocs.io/en/1.7.4/plugins/b603_subprocess_without_shell_equals_true.html
```

Fuzzing

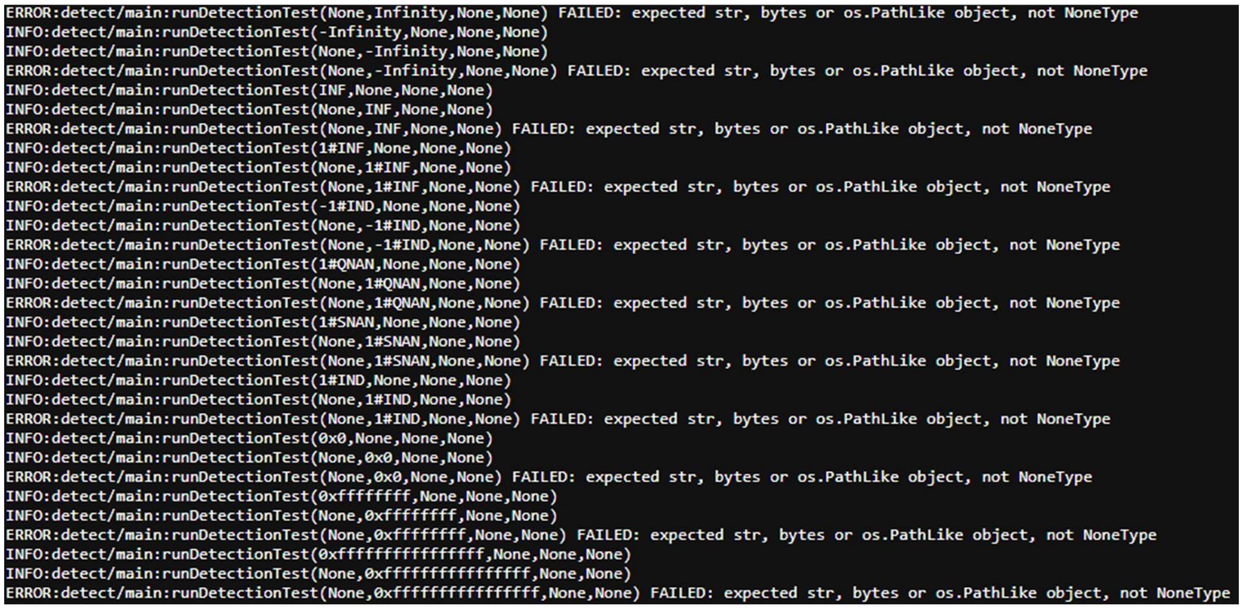
We implemented fuzzing in the file, fuzz.py. Five methods were chosen to be fuzzed from TestOrchestrator4ML-main/detection/main.py: get_test_details, checkClassificationAlgoTest, checkAccuracyTest, chackAttackTest, and runDetectionTest. Each method was fuzzed with a combination of bad strings, random integers and NoneTypes. Each parameter in the function was tested. Analysis of the output stored in fuzz_output.txt indicated that the runDetectionTest method was not graceful at handling parameters that are integers or NoneTypes. An excerpt taken from fuzz_output.txt is shown below.



```
Error in runDetectionTest with value 5554: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 1311: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 6412: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 123: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 5434: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 9494: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 6490: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 4476: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 583: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 5666: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 2124: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 1002: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 8245: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 6377: expected str, bytes or os.PathLike object, not int
Error in runDetectionTest with value 8408: expected str, bytes or os.PathLike object, not int
```

Forensics

We decided to use forensics to track whenever a method is called, the arguments used, and any subsequent errors/exceptions raised due to the input. All recordings are stored in a file titled "logging_output.log," assigned to the respective file path for the method called. We chose to use the same methods that were fuzzed in fuzz.py. An excerpt from the log is seen here:



```
ERROR:detect/main:runDetectionTest(None,Infinity,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(-Infinity,None,None,None)
INFO:detect/main:runDetectionTest(None,-Infinity,None,None)
ERROR:detect/main:runDetectionTest(None,-Infinity,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(INF,None,None,None)
INFO:detect/main:runDetectionTest(None,INF,None,None)
ERROR:detect/main:runDetectionTest(None,INF,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(1#INF,None,None,None)
INFO:detect/main:runDetectionTest(None,1#INF,None,None)
ERROR:detect/main:runDetectionTest(None,1#INF,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(-1#IND,None,None,None)
INFO:detect/main:runDetectionTest(None,-1#IND,None,None)
ERROR:detect/main:runDetectionTest(None,-1#IND,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(1#QNAN,None,None,None)
INFO:detect/main:runDetectionTest(None,1#QNAN,None,None)
ERROR:detect/main:runDetectionTest(None,1#QNAN,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(1#SNAN,None,None,None)
INFO:detect/main:runDetectionTest(None,1#SNAN,None,None)
ERROR:detect/main:runDetectionTest(None,1#SNAN,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(1#IND,None,None,None)
INFO:detect/main:runDetectionTest(None,1#IND,None,None)
ERROR:detect/main:runDetectionTest(None,1#IND,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(0x0,None,None,None)
INFO:detect/main:runDetectionTest(None,0x0,None,None)
ERROR:detect/main:runDetectionTest(None,0x0,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(0xffffffff,None,None,None)
INFO:detect/main:runDetectionTest(None,0xffffffff,None,None)
ERROR:detect/main:runDetectionTest(None,0xffffffff,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
INFO:detect/main:runDetectionTest(0xffffffffffffffff,None,None,None)
INFO:detect/main:runDetectionTest(None,0xffffffffffffffff,None,None)
ERROR:detect/main:runDetectionTest(None,0xffffffffffffffff,None,None) FAILED: expected str, bytes or os.PathLike object, not NoneType
```

During this process, we learned that logging is as complicated as you make it. Doing something as simple as recording a method's call is plenty effective and makes for good security practice. Other methods among the repository were logged as well, but to demonstrate a proper output, those used with fuzzing were implemented in addition.