

COMP 5710 - Project Report

Auburn University, Spring FY23

Team Name: SoftwareQualityAssumption

Team members: George Martin, Matthew Jayroe, Jared Prather.

Github URL:

<https://github.com/jprather7966/SoftwareQualityAssumption-SQA2023-AUBURN>

Objective: “The objective of this project is to integrate software quality assurance activities into an existing Python project” by methods of an integrated Git Hook, Fuzzing, and embedded forensics that output a log-file.

Git Hook:

We created a pre-commit file to run bandit on the repository and format the output in csv and place it in a file called static-analysis.csv. This file is placed in the github repo for the user to place in their .git/hooks folder. This is a sample of the static-analysis.csv file after running the git hook:

```
static-analysis.csv
1 filename,test_name,test_id,issue_severity,issue_confidence,issue_cwe,issue_text,line_number,col_offset,line_range,more_info
2
3 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/helm.v
4
5 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/tango.
6
7 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/charts
8
9 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/skampi
10
11 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/minecr
12
13 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/kubecf
14
15 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/nextcl
16
17 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/keyclo
18
19 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/empty.
20
21 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/kubecf
22
23 ./TEST_CONSTANTS.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'TEST_ARTIFACTS/specia
24
25 ./constants.py,hardcoded_password_string,B105,LOW,MEDIUM,https://cwe.mitre.org/data/definitions/259.html,Possible hardcoded password: 'Secret',81,31,"[81, 82]",h
26
```

COMP 5710 - Project Report

Auburn University, Spring FY23

Fuzzing:

The fuzz.py file was created to generate arguments for 5 different methods found throughout KubeSec, then use said arguments to fuzz each selected method. The fuzz.py file runs automatically when new code is pushed to the repository using fuzz.yml, a simple workflow action file. The fuzz results can easily be found in the Actions section of the repository on GitHub. Screenshots are shown below.

```
1  Set up job 15
2  1 Current runner version: '2.303.0'
3  2 Operating System
4  3 Runner Image
5  6 Runner Image Provisioner
6  11 GitHub_TOKEN Permissions
7  13 Secret source: Actions
8  17 Prepare workflow directory
9  18 Prepare all required actions
10 20 Getting action download info
11 21 Download action repository 'actions/checkout@v2' (SHA:ee0669d1cc54295c223e0bb66b733df41de1c5)
12 22 Download action repository 'actions/setup-python@v4' (SHA:d27e3f3d7c64b4b0f8e4abfb9b63b63e846e0435)
13 23 Complete job name: fuzz

14  checkout repo content 15
15  1 Run actions/checkout@v2
16  2 with:
17  3 repository: jprather7966/SoftwareQualityAssumption-SQA2023-AUBURN
18  4 token: ***
19  5 ssh-strict: true
20  6 persist-credentials: true
21  7 clean: true
22  8 fetch-depth: 1
23  9 lfs: false
24 10 submodule: false
25 11 set-safe-directory: true
26 12 Syncing repository: jprather7966/SoftwareQualityAssumption-SQA2023-AUBURN
27 13 Getting Git version info
28 14 Temporarily overriding HOME='/home/runner/work/_temp/0721d4ef-d25e-437e-bfad-3fa2e8c52127' before making global git config changes
29 15 Adding repository directory to the temporary git global config as a safe directory
30 16 /usr/bin/git config --global --add safe.directory /home/runner/work/SoftwareQualityAssumption-SQA2023-AUBURN/SoftwareQualityAssumption-SQA2023-AUBURN
31 17 Deleting the contents of '/home/runner/work/SoftwareQualityAssumption-SQA2023-AUBURN/SoftwareQualityAssumption-SQA2023-AUBURN'
32 18 Initializing the repository
33 19 Disabling automatic garbage collection
34 20 Setting up auth
35 21 Fetching the repository
36 22 Determining the checkout info
37 23 Checking out the ref
38 100 /usr/bin/git log -1 --format="%h"
39 101 'bbed8df7151fe749e70fa3a5e401427b171989378'

40  setup python 05
41  1 Run actions/setup-python@v4
42  7 Installed versions

43  install python packages 115
44  1 Run python -m pip install --upgrade pip
45 12 Requirement already satisfied: pip in /opt/hostedtoolcache/Python/3.10.11/x64/lib/python3.10/site-packages (23.0.1)
46 13 Collecting pip
47 14 Downloading pip-23.1-py3-none-any.whl (2.1 MB)
48 15 _____ 2.1/2.1 MB 12.0 MB/s eta 0:00:00
49 16 Installing collected packages: pip
50 17 Attempting uninstall: pip
51 18 Found existing installation: pip 23.0.1
52 19 Uninstalling pip-23.0.1:
53 20 Successfully uninstalled pip-23.0.1
54 21 Successfully installed pip-23.1
55 22 Collecting numpy==1.24.2 (from -r requirements.txt (line 1))
56 23 Downloading numpy-1.24.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (17.3 MB)
57 24 _____ 17.3/17.3 MB 73.8 MB/s eta 0:00:00
58 25 Collecting pandas==1.5.3 (from -r requirements.txt (line 2))
59 26 Downloading pandas-1.5.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.1 MB)
60 27 _____ 12.1/12.1 MB 123.6 MB/s eta 0:00:00
61 28 Collecting pbr==5.11.1 (from -r requirements.txt (line 3))
62 29 Downloading pbr-5.11.1-py2.py3-none-any.whl (112 kB)
63 30 _____ 112.7/112.7 kB 24.2 MB/s eta 0:00:00
64 31 Collecting python-dateutil==2.8.2 (from -r requirements.txt (line 4))
65 32 Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
66 33 _____ 247.7/247.7 kB 73.3 MB/s eta 0:00:00
67 34 Collecting pytz==2022.7.1 (from -r requirements.txt (line 5))
68 35 Downloading pytz-2022.7.1-py2.py3-none-any.whl (499 kB)
69 36 _____ 499.4/499.4 kB 74.6 MB/s eta 0:00:00
70 37 Collecting PyYAML==6.0 (from -r requirements.txt (line 6))
71 38 Downloading PyYAML-6.0-cp310-cp310-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux2010_x86_64.whl (682 kB)
72 39 _____ 682.2/682.2 kB 99.2 MB/s eta 0:00:00
73 40 Collecting six==1.16.0 (from -r requirements.txt (line 7))
74 41 Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
75 42 Collecting smmap==5.0.0 (from -r requirements.txt (line 8))
76 43 Downloading smmap-5.0.0-py3-none-any.whl (24 kB)
77 44 Installing collected packages: pytz, smmap, six, PyYAML, pbr, numpy, python-dateutil, pandas
78 45 Successfully installed PyYAML-6.0 numpy-1.24.2 pandas-1.5.3 pbr-5.11.1 python-dateutil-2.8.2 pytz-2022.7.1 six-1.16.0 smmap-5.0.0
```

The above screenshots show the setup of the GitHub action.

COMP 5710 - Project Report

Auburn University, Spring FY23

```
execute fuzz.py
1 ▶ Run python fuzz.py
11 -----
12 Fuzz result: method "getVAMLFiles" passed: []
13 -----
14 Fuzz result: method "getVAMLFiles" failed: expected str, bytes or os.PathLike object, not int
15 -----
16 Fuzz result: method "getVAMLFiles" failed: expected str, bytes or os.PathLike object, not float
17 -----
18 Fuzz result: method "getVAMLFiles" failed: expected str, bytes or os.PathLike object, not list
19 -----
20 Fuzz result: method "getVAMLFiles" failed: expected str, bytes or os.PathLike object, not dict
21 -----
22 Fuzz result: method "getVAMLFiles" failed: expected str, bytes or os.PathLike object, not NoneType
23 -----
24 Fuzz result: method "isvalidUsername" passed: False
25 -----
26 Fuzz result: method "isvalidUsername" passed: False
27 -----
28 Fuzz result: method "isvalidUsername" passed: False
29 -----
30 Fuzz result: method "isvalidUsername" passed: False
31 -----
32 Fuzz result: method "isvalidUsername" passed: False
33 -----
34 Fuzz result: method "isvalidUsername" passed: False
35 -----
36 Fuzz result: method "isvalidPasswordName" passed: False
37 -----
38 Fuzz result: method "isvalidPasswordName" passed: False
39 -----
40 Fuzz result: method "isvalidPasswordName" passed: False
41 -----
42 Fuzz result: method "isvalidPasswordName" passed: False
43 -----
44 Fuzz result: method "isvalidPasswordName" passed: False
45 -----
46 Fuzz result: method "isvalidPasswordName" passed: False
47 -----
48 Fuzz result: method "isvalidkey" passed: False
49 -----
50 Fuzz result: method "isvalidkey" passed: False
```

The above screenshot shows some of the output from running fuzz.py (the full output can be seen in the Actions section of the GitHub repository).

```
execute fuzz.py
54 Fuzz result: method "isvalidkey" passed: False
55 -----
56 Fuzz result: method "isvalidkey" passed: False
57 -----
58 Fuzz result: method "isvalidkey" passed: False
59 -----
60 Fuzz result: method "checkIfValidSecret" passed: False
61 -----
62 Fuzz result: method "checkIfValidSecret" passed: False
63 -----
64 Fuzz result: method "checkIfValidSecret" passed: False
65 -----
66 Fuzz result: method "checkIfValidSecret" passed: False
67 -----
68 Fuzz result: method "checkIfValidSecret" passed: False
69 -----
70 Fuzz result: method "checkIfValidSecret" passed: False

Post setup python
1 Post job cleanup.

Post checkout repo content
1 Post job cleanup.
2 /usr/bin/git version
3 git version 2.40.0
4 Temporarily overriding HOME='/home/runner/work/_temp/2be329ef-3b91-4bdd-8a9e-09750397211f' before making global git config changes
5 Adding repository directory to the temporary git global config as a safe directory
6 /usr/bin/git config --global --add safe.directory /home/runner/work/SoftwareQualityAssumption-SQA2023-AUBURN/SoftwareQualityAssumption-SQA2023-AUBURN
7 /usr/bin/git config --local --name-only --get-regexp core.sshcommand
8 /usr/bin/git submodule foreach --recursive sh -c "git config --local --name-only --get-regexp 'core.sshcommand' && git config --local --unset-all 'core.sshcommand' || :"
9 /usr/bin/git config --local --name-only --get-regexp http.https://github.com/.extraheader
10 http.https://github.com/.extraheader
11 /usr/bin/git config --local --unset-all http.https://github.com/.extraheader
12 /usr/bin/git submodule foreach --recursive sh -c "git config --local --name-only --get-regexp 'http.https://github.com/.extraheader' && git config --local --unset-all 'http.https://github.com/.extraheader' || :"

Complete job
1 Cleaning up orphan processes
```

The above screenshot shows the post job cleanup of the GitHub action.

COMP 5710 - Project Report

Auburn University, Spring FY23

Forensics and Logging:

A simpleLogger.py file was integrated similar to that seen in a previous workshop. This file was imported into scanner.py and a loggerObject instantiated. In front of key methods which would read potentially sensitive (or malicious) data from user files, logs were taken with descriptive outputs for each action. When a new YAML file was read this would appear in the log, when a username, password, key, or secret was scanned, they too would appear in the log. Our team chose to include more than the minimum 5 logging statements, which resulted in a sizeable output file named "SIMPLE-LOGGER.log".

Screen shots are shown below.

```
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/charts.values.yaml COUNT: 1
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/special.secret1.yaml COUNT: 2
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/bakis.rs.yaml COUNT: 3
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/fp.no.reso7.yaml COUNT: 4
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/fp.http.yaml COUNT: 5
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/k8s.doc.network.yaml COUNT: 6
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/present.roll.yaml COUNT: 7
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/ANOTHER.DOCKERSOCK.yaml COUNT: 8
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/tp.seccomp.unconfined.yaml COUNT: 9
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/allow.privilege.yaml COUNT: 10
-----
Analyzing... /SLI-KUBE-WORK/KubeSec-master/TEST_ARTIFACTS/present.default.ksql.yaml COUNT: 11
```

Above: an image of when python3 main.py is running. The program iterates through the YAML files and updates the count. In total there were 56 read yaml files.

```
GNU nano 6.2
2023-04-18 19:05:01,118 Begin Scanner...
2023-04-18 19:05:01,119 Reading in YAML Files...
2023-04-18 19:05:01,207 Scanning Secrets...
2023-04-18 19:05:01,208 Scanning User Name...
2023-04-18 19:05:01,208 Checking for Valid User Name...
2023-04-18 19:05:01,208 Scanning Passwords...
2023-04-18 19:05:01,209 Validating Password...
2023-04-18 19:05:01,209 Scanning Keys...
2023-04-18 19:05:01,209 Validating Key...
2023-04-18 19:05:01,209 Scanning User Name...
2023-04-18 19:05:01,209 Checking for Valid User Name...
2023-04-18 19:05:01,209 Scanning Passwords...
2023-04-18 19:05:01,209 Validating Password...
2023-04-18 19:05:01,210 Scanning Keys...
2023-04-18 19:05:01,210 Validating Key...
2023-04-18 19:05:01,210 Scanning User Name...
2023-04-18 19:05:01,210 Checking for Valid User Name...
2023-04-18 19:05:01,210 Scanning Passwords...
2023-04-18 19:05:01,210 Validating Password...
2023-04-18 19:05:01,210 Scanning Keys...
2023-04-18 19:05:01,211 Validating Key...
```

From the SIMPLE-LOGGER.log file, date and timestamps accompany each logged item from the yaml files. When the scanner.py class is first called it is logged as "Begin Scanner...". From there each event or key methods are logged.

COMP 5710 - Project Report
Auburn University, Spring FY23

```
2023-04-18 19:05:15,374 Scanning for Over Privileges...
2023-04-18 19:05:15,391 Scanning for HTTP...
2023-04-18 19:05:15,407 Scanning for Missing Security...
2023-04-18 19:05:15,459 Scanning for Updates...
2023-04-18 19:05:15,476 Scanning for Network Policy...
2023-04-18 19:05:16,323 Scanning for Over Privileges...
2023-04-18 19:05:16,339 Scanning for HTTP...
2023-04-18 19:05:16,353 Scanning for Missing Security...
2023-04-18 19:05:16,399 Scanning for Updates...
2023-04-18 19:05:16,414 Scanning for Network Policy...
2023-04-18 19:05:16,587 Scanning for Over Privileges...
2023-04-18 19:05:16,598 Scanning for HTTP...
2023-04-18 19:05:16,609 Scanning for Missing Security...
2023-04-18 19:05:16,642 Scanning for Updates...
2023-04-18 19:05:16,653 Scanning for Network Policy...
```

Other methods were logged such as “Scanning for Over Privileges...” (sic).
The logs here indicate that security aspects were executed.

Lessons Learned:

In this project we first learned on how to use git hooks to perform static analysis on our code so we can see potential vulnerabilities before we fully commit to them. Next, we utilized github actions to fuzz 5 methods in our project so we can be able to find any critical problems that can potentially cause if we provide random inputs. Finally we place a ton of logging statements in front of important methods so we can know when sensitive data is utilized and where certain issues can arise.