MONICA BERNARD
CAP6135

FUZZ TESTING – PROGRAMMING ASSIGNMENT 2

Section A: Introduction

In this assignment, I performed fuzz testing on the executable "jpg2bmp" which converts a given image "cross.jpg" to a bmp file. Fuzz testing is a technique of finding bugs in the software by suppling random data inputs in order to check for any malfunctioning in the software. In this assignment, mutation-based fuzz testing was implemented, where several random mutations were generated in the input jpeg file ("cross.jpg") and fed into the "jpg2bmp" executable file. Doing so, I checked to see if the mutations of the input image would trigger any bug in the executable file. I was successfully able to find seven out of the eight numbered bugs in my executable.

For this assignment, I used Python to write my source code. In this report, I discuss the design and implementation of my fuzzer, and I have documented all my findings. Then, a detailed analysis of my findings is provided.

Section B: Design and Implementation

Steps used in my implementation:

```
1) Convert "cross.jpg" into a byte array
2) Choose a set of random locations in the byte array to be mutated
3) Feed random values (in the range 0 to 256) in each of these random
   locations
4) Convert the mutated byte array to a jpg file
5) Run the executable 'jpg2bmp' with the mutated jpg file
6) Check if a bug is triggered in the standard error message
7) If a bug is triggered
         Save mutated image file along with the bug number
   Else
         Discard the mutated image file
8) Loop through the process until required number of bugs are found
```

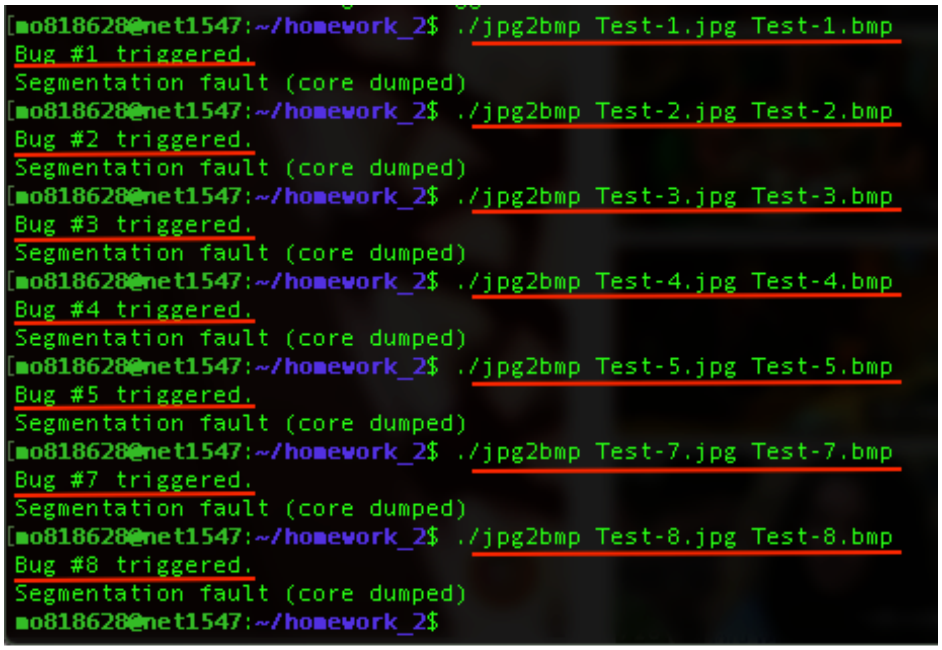
Section C: Output and Output Analysis:

My program was looped through 100 times each time changing one random byte value in the original image, thus producing 'n' number of mutated images (n represents the number of mutation images and was taken to be n = 10000, 1000 and 100) and the average output over 100 execution cycle was recorded for more analysis.

To execute the program on Eustis, change directory to the folder that contains the source code along with the input image and jpg2bmp executable. Then on the console, type: **`python3 assignment2.py`** to get the output. Output will include all the mutated images that trigger a bug and a text file (statsFile.txt) showing all the bugs triggered.

FUZZ TESTING – PROGRAMMING ASSIGNMENT 2

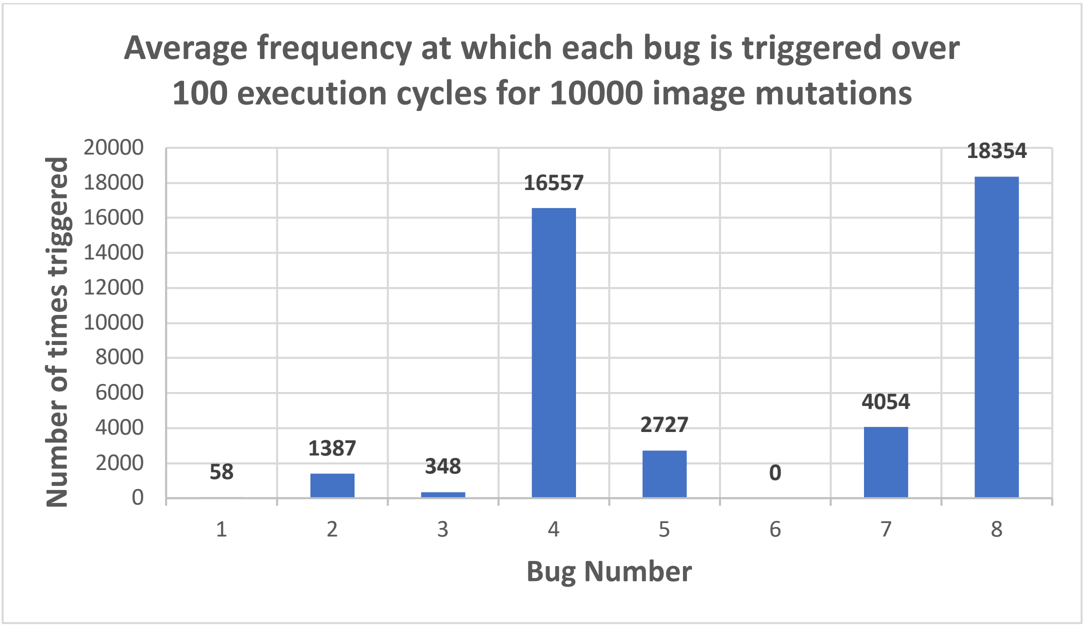
Console output showing 7 out of 8 bugs triggered:



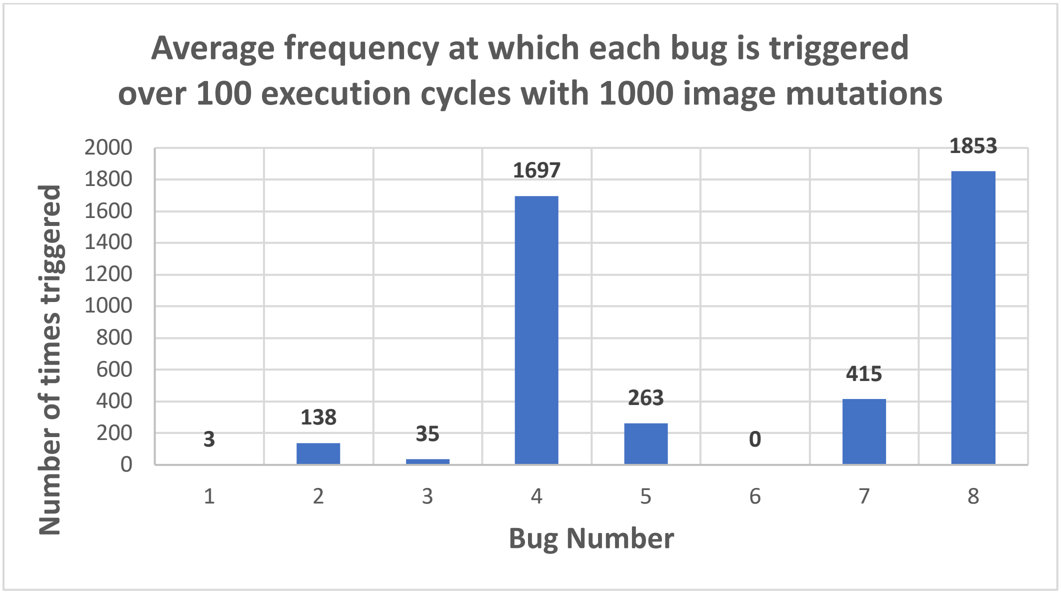
```
[mo818628@net1547:~/homework_2$ ./jpg2bmp Test-1.jpg Test-1.bmp
Bug #1 triggered.
Segmentation fault (core dumped)
[mo818628@net1547:~/homework_2$ ./jpg2bmp Test-2.jpg Test-2.bmp
Bug #2 triggered.
Segmentation fault (core dumped)
[mo818628@net1547:~/homework_2$ ./jpg2bmp Test-3.jpg Test-3.bmp
Bug #3 triggered.
Segmentation fault (core dumped)
[mo818628@net1547:~/homework_2$ ./jpg2bmp Test-4.jpg Test-4.bmp
Bug #4 triggered.
Segmentation fault (core dumped)
[mo818628@net1547:~/homework_2$ ./jpg2bmp Test-5.jpg Test-5.bmp
Bug #5 triggered.
Segmentation fault (core dumped)
[mo818628@net1547:~/homework_2$ ./jpg2bmp Test-7.jpg Test-7.bmp
Bug #7 triggered.
Segmentation fault (core dumped)
[mo818628@net1547:~/homework_2$ ./jpg2bmp Test-8.jpg Test-8.bmp
Bug #8 triggered.
Segmentation fault (core dumped)
mo818628@net1547:~/homework_2$
```

Analysis of the Results:
Detailed analysis was made to check the frequency of triggering each of the bugs. Graphs of my findings are shown below.

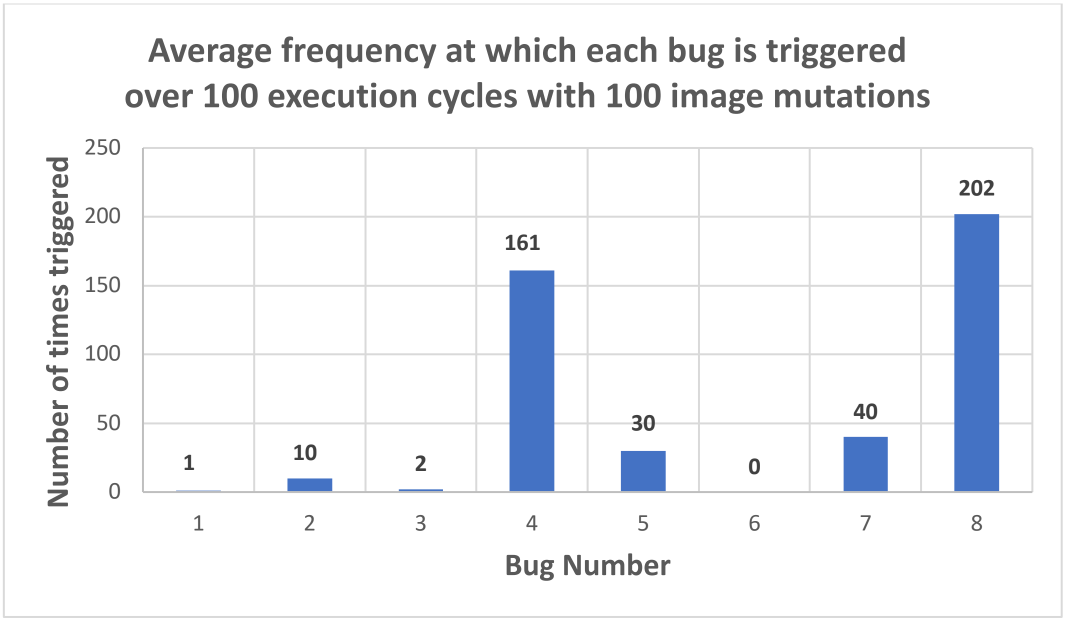
Graph showing the average frequency at which each bug is triggered over 100 execution cycles with 10000 image mutations:



Graph showing the average frequency at which each bug is triggered over 100 execution cycles with 1000 image mutations:



Graph showing the average frequency at which each bug is triggered over 100 execution cycles with 100 image mutations:



Conclusion:

It can be concluded that Bugs 8 and 4 were triggered the greatest number of times followed by bugs 2, 7 and 5. Bug 1 was not triggered frequently. Bug 6 was not found. When the program was executed with 100 mutation images, sometimes no bugs other than bug 8 and 4 were triggered.