Nick Renner
HW1 Report
Application Security

GitHub Repo -

HW Part 2 - Program Details

Explanation

Load Dictionary - The program parses each word line by line using getline, removing the newline character. It also lowercases the word for proper checking.
It uses the hash function to find the correct bucket, then inits or inserts into a linked list based on that hash.

Check Words - Check words uses getline again, and parses the lines using strtok, getting rid of all garbage characters (except ' which is a weird case).
I decided to just discard words that begin with an apostrophe, but otherwise check them. It then checks each word, and if they return misspelled I put them into the misspelled array.

Check Word - I lowercase each word for checking and compare them to the dictionary. If the words isn't in the LL for it's hashed bucket, I return false as it is not in the dictionary.

Valgrind

Output - (For a tale of two cities with the aspell word list)

```
valgrind ./spell_check wordlist.txt tale.txt
==10082== Memcheck, a memory error detector
==10082== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et
al.
==10082== Using Valgrind-3.13.0 and LibVEX; rerun with -h for
copyright info
==10082== Command: ./spell_check wordlist.txt tale.txt
==10082==
Mispelled: 1000
==10082==
==10082== HEAP SUMMARY:
==10082==  in use at exit: 0 bytes in 0 blocks
==10082==   total heap usage: 265,027 allocs, 265,027 frees,
8,166,073 bytes allocated
==10082==
==10082== All heap blocks were freed -- no leaks are possible
==10082==
==10082== For counts of detected and suppressed errors, rerun with:
-v
==10082== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from
0)
```

Possible Bugs

My code will likely incorrectly declare possessives (with apostrophes) as spelled incorrectly. It's unclear from the instructions whether this is or isn't a bug from the spec. I had some memory leakage with regards to lowercasing each string improperly (was calling to_lower on NULLs), but that was fixed. I made sure of properly freeing all allocated memory.
I made sure not to use memory after freeing, or double freeing, and using printf formatting properly.

HW Part 3 - Testing and Fuzzing

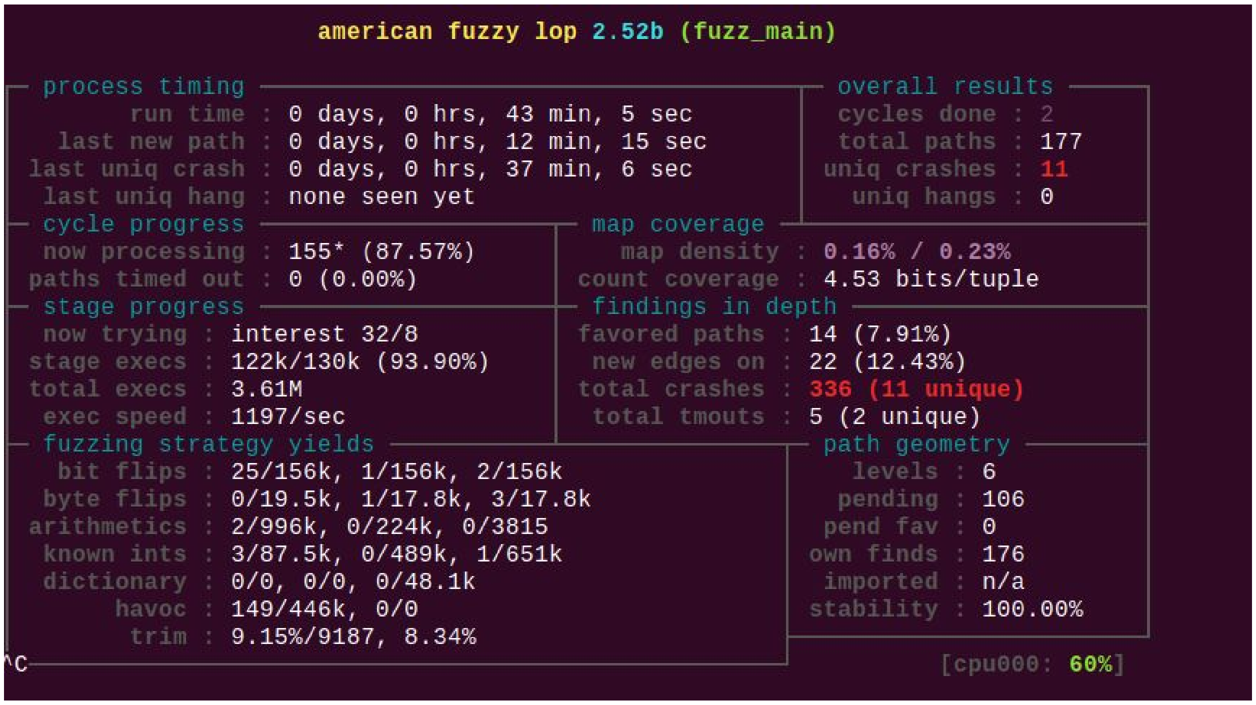
Bugs Found by Tests

1. -Buffer overflow input words - I had words over max length being skipped instead of truncated, which lead to problems with the autograder since they weren't registered as misspelled words.
2. I originally made my strtok delimiter include a vast array of random characters instead of just removing from the beginning and end of the word.
3. I wasn't handling non-english characters because I had limited char values to < 127.

4. I had a bug where I was truncating number strings like '1976' to nothing and then was still trying to use the string.

Bugs Found by Fuzzing

1. I found crashes when I encountered inserted characters which were interpreted as non-unicode values through AFL.



```
                     american fuzzy lop 2.52b (fuzz_main)

┌─ process timing ─────────────────────────────────────┬─ overall results ────┐
│        run time : 0 days, 0 hrs, 43 min, 5 sec       │  cycles done : 2     │
│   last new path : 0 days, 0 hrs, 12 min, 15 sec      │  total paths : 177   │
│ last uniq crash : 0 days, 0 hrs, 37 min, 6 sec       │ uniq crashes : 11    │
│  last uniq hang : none seen yet                      │   uniq hangs : 0     │
├─ cycle progress ────────────────────┬─ map coverage ─┴──────────────────────┤
│  now processing : 155* (87.57%)     │    map density : 0.16% / 0.23%        │
│ paths timed out : 0 (0.00%)         │ count coverage : 4.53 bits/tuple      │
├─ stage progress ────────────────────┼─ findings in depth ───────────────────┤
│  now trying : interest 32/8         │ favored paths : 14 (7.91%)            │
│ stage execs : 122k/130k (93.90%)    │  new edges on : 22 (12.43%)           │
│ total execs : 3.61M                 │ total crashes : 336 (11 unique)       │
│  exec speed : 1197/sec              │  total tmouts : 5 (2 unique)          │
├─ fuzzing strategy yields ───────────┴───────────────┬─ path geometry ───────┤
│   bit flips : 25/156k, 1/156k, 2/156k               │    levels : 6         │
│  byte flips : 0/19.5k, 1/17.8k, 3/17.8k             │   pending : 106       │
│ arithmetics : 2/996k, 0/224k, 0/3815                │  pend fav : 0         │
│  known ints : 3/87.5k, 0/489k, 1/651k               │ own finds : 176       │
│  dictionary : 0/0, 0/0, 0/48.1k                     │  imported : n/a       │
│       havoc : 149/446k, 0/0                         │ stability : 100.00%   │
│        trim : 9.15%/9187, 8.34%                     ├───────────────────────┘
^C────────────────────────────────────────────────────┘          [cpu000: 60%]
```

AFL Screenshot

How Bugs Were Fixed

1. I limited the bug found in fuzzing by checking my characters to make sure they were in the valid unicode range.
2. I changed my parser to remove punctuation from beginning and end instead of using a large delimiting string.

• how similar bugs can be avoided in the future.

- Dealing with Unicode, as opposed to ASCII, is tricky! Would be nice to limit the input.
- It's good to deal with any input size to a buffer immediately.