Part 3 Answer:

```
INPUT="computer %s%s%s%s%s"
…
# Get the 3rd word in the string
third_word=$(echo $TMP | awk '{print $3}')

# Get the substring after the last occurance of %s
substring_after_last_space=${third_word##*%s}

echo "$substring_after_last_space"
```

_____-

The snprintf functions in vulnerable() are vulnerable to accessing information stored outside the buffer. Because the snprintf function doesn't have a "%s" string format parameter as an argument, it will interpret each "%s" in the input string as a reference to a string pointer, so it will try to interpret every %s as a pointer to a string, starting from the location of the buffer on the stack. We know that the secret word is somewhere on the stack because the pointer to the word is an argument to the vulnerable() function. I used trial and error to try to access a hidden word stored somewhere on the stack. I started with a random word with one %s after it: "computer %s". I ran the program and saw that the output was "computer H=" on one line, and "Was Wrong!" on another so we were not able to find the secret word on the stack. The %s interpreted the values at the location after "word" as a string pointer and attempted to dereference and print the pointed-to value as a string. This explains the "H=" part of the output that I saw. Then I used two "%s" and got another output without any secret word in it. I increased the number of "%s" until I had 5 ("computer %s%s%s%s%s") which revealed the secret word to me which was preceded by useless outputs from the other %s and proceeded by "Was Wrong!" on a new line. This strategy worked because the word was hidden somewhere on the stack and by adding %s format specifiers one by one I kept attempting to dereference sequential points on the stack so I would eventually get the hidden word. Then the latter part of my script just processes the program output by breaking the output string into words that were separated by spaces, then I select the correct word (happens to be the 3rd word based on my input string) and then I got the substring after the last occurrence of %s. This leaves the final output as just the hidden word as the output. My postprocessing should work for any hidden word at that location on the stack.