CS 4910 Intro to Computer Security
Assignment 4

Number 1 – 5 are multiple choice questions. Chose the correct answer. 10 points for each. 50 points in total.

1. Which one of the following descriptions about the Intel architecture RET instruction is correct?
    a. The RET instruction pops whatever EBP/RBP points to to EIP/RIP
    b. The RET instruction pops whatever ESP/RSP points to to EIP/RIP
    c. The RET instruction checks if EBP/RBP points to is a valid code address, if yes it pops the value to EIP/RIP
    d. The RET instruction checks if ESP/RSP points to is a valid code address, if yes it pops the value to EIP/RIP

2. Which one of the following descriptions of the Intel architecture CALL instruction is correct?
    a. The CALL instruction pushes the destination address onto the stack
    b. The CALL instruction only moves the destination address into EIP/RIP
    c. The CALL pushes the address of the instruction after call onto the stack, then moves the destination address to EBP/RBP
    d. The CALL pushes the address of the instruction after call onto the stack, then moves the destination address to EIP/RIP

3. The kernel makes the decision whether a process has the privilege by looking at the ____ of the process.
    a. Real UID
    b. Effective UID
    c. Saved UID

4. When a process is created, the environment variables and command line arguments are placed at ____.
    a. Stack
    b. Heap
    c. .data section
    d. .rodata section

5. Which one of the following descriptions about the Data Execution Prevention (DEP) is CORRECT.
    a. With DEP, an attacker cannot overwrite the RET address on stack
    b. With DEP, an attacker cannot overwrite the Saved EBP/RBP on stack

c. With DEP, an attacker cannot execute code on the stack
d. With DEP, an attacker cannot copy shellcode onto the stack

Number 6 -7 are short answer questions. Answer them shortly and concisely.

6. [40 points] Read the following code running on x86 system (32-bit) and answer questions.

```
1: int vulfoo()
2: {
3:   char buf[40];
4:   gets(buf);
5:   return 0;
6: }
7: int main(int argc, char *argv[])
8: {
9:   vulfoo();
10:  printf("I pity the fool!\n");
11:}
```

(i) [10 points] Why this piece of code is vulnerable?
(ii) [10 points] How many input bytes in theory we need to overwrite the return address?
(iii) [10 points] How can we execute arbitrary code by exploiting this vulnerablity? Explain if the stack is executable or non-executable.
(iii) [10 points] How can we fix the problem of this piece of code? List three methods and simply explain them will be fine.

7. [10 points] Compare the traditional shadow stack with the parallel shadow stack, including the strengths and weaknesses of each.