CS4440/7440 Midterm Examination 1

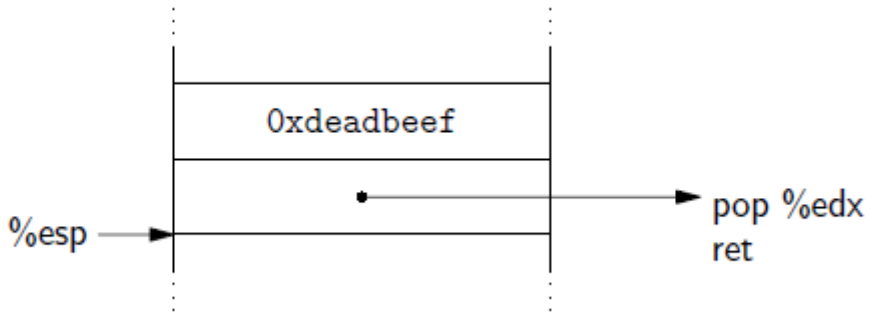
Midterm Examination 3/1/2017

Name:

DIRECTIONS

There are sixteen questions worth a total of 50 points. Answer each question in the space provided. Remember to write your name on your exam.

Question 1. What is the effect of executing **`return`** with the stack in the configuration below? Recall that the stack grows downward. (4 points)



Question 2. In compilers, a basic block (BB) is a portion of the code within an assembly language program with certain desirable properties that make it highly amenable to analysis. Compilers usually decompose programs into their BBs as a first step in the analysis process. BBs form the vertices or nodes in a control flow graph.

Properties of the BB are:

1. The code in a BB has one entry point.
2. No code within a BB (other than the first instruction) is the destination of a jump instruction from anywhere else in the program.
3. A BB has one and only one exit point, meaning that only the last instruction can cause a change in control flow.
4. Whenever the first instruction in a BB is executed, the rest of the instructions are necessarily executed exactly once and in order.

Circle each number above which is also holds of a "gadget" from return-oriented programming. (4 points)

Question 3. In conventional programming, the instruction pointer directs the control flow of a program. In return-oriented programming, another register directs the program control flow. On an x86 architecture, what is this other register? (2 points)

Question 4. Would legitimate software ever engage in interrupt hooking? If you answer “no”, explain why not; if you answer “yes”, give an example. (2 points) .

Question 5. Define “Tricky Jump”. In particular, what makes it tricky and why would a virus writer use it? (2 points)

Question 6. The x86 architecture is more susceptible to return-oriented attacks than RISC architectures. Why? (3 points)

Question 7. Realistic analysis of a possibly infected system begins with quick methods and then proceeds to more thorough analyses. Classify each of the following analyses as *quick* or *thorough*. (Write **quick** or **thorough** next to each.) (3 points)

1. Compare file checksums to a database of known good checksums.
2. Disassemble a suspicious file and examine its code for a new virus.
3. Scan memory for memory-resident viruses.

Question 8. The following code contains a vulnerability that is similar, but not identical, to the stack-based buffer overflows we saw in class. What is the vulnerability? Or, in other words, what could be accomplished by overflowing **`buffer`**? (4 point)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(int argc, char **argv) {
int ch = 0, i = 0;
FILE *f = NULL;
static char buffer[16], *szFileName = "C:\\harmless.txt";
ch = getchar();
while (ch != EOF) {
      buffer[i++] = ch; ch = getchar();
}
f = fopen(szFileName, "w+b");
fputs(buffer, f);
fclose(f);
return 0;
}
```

Question 9. Why does an anti-virus analyst software attempt to determine exactly which virus is infecting a file? (3 points)

Question 10. A new processor design prevents any data region from being executed, whether the data is global or heap or stack data. Will this help defeat the operation of return-oriented programming attacks? Explain your answer. (3 points)

Question 11. The PE file format for binaries often has dead space that can be used by a virus. Explain why the PE file would have dead space. (3 points)

Question 12. A virus that attempts to use the dead space in a PE file is called a _____ virus. (1 point).

Question 13. Entry Point Obscuring viruses do not change the entry point of the application to infect it; neither do they change the code at the entry point. Why? (3 points)

Question 14. W⊕X or DEP (Dynamic Execution Prevention) mark all writeable locations in memory as non-executable. Why? (3 points)

Question 15. Describe, at a very high level, the steps involved in accomplishing a buffer overflow exploit. (4 points)

Question 16. Short definitions: Define the following in 1-3 sentences. (6 points total)

i. **Shell code.**

ii. **Gadget.**

iii. **Compressor virus.**