R·I·T	Rochester Institute of Technology Golisano College of Computing and Information Sciences Department of Computing Security

Professor Jonathan S. Weissman CSEC 466 Data Types

Name: Smayan Daruka
Date: 10/23/18

Requirements:

- You can use a debugger to understand and answer the questions
- When you are asked to provide a value, do so in hexadecimal

Part One:

```
xor eax, eax
mov ecx, 5
mov edi, [esp]
rep stosd
mov al, 0xFF
stosw
stosb
stosb
```

1. How many times does the 'stosd' operation run? Why?
Stosd runs 5 times because the ecx register is 5.

2. If esp was storing address 0x4b0000, what would edi be after the sequence is complete? Why?
EDI would be whatever the value was stored in memory location 0x4b0000. The reason for this is that esp is dereferenced before being moved into edi.

3. How many bytes on the heap are modified by the code above? Why?
Unless I am mistaken, I think 160 bytes on the heap were modified because of the size of the registers being used and the operations being performed.

4. Describe as completely as possible what the code above is doing.
The above code simply stores string xor's eax and then later moves FF into al which is in the lower half and then continues to store words and bytes.

Part Two:

The string "reverseengineer" is placed onto the heap at address 0x4b0000 with a null byte at the end:

004B0000 72 65 76 65 72 73 65 65 reversee
004B0008 6E 67 69 6E 65 65 72 00 ngineer.

Answer the following questions, given the below code sequence, assuming esp is storing address 0x4b0000:

```
mov edx, 0x14
xor eax, eax
mov edi, [esp]
mov ecx, edx
repne scasb
sub edx, ecx
mov ecx, edx
mov edi, [esp]
add edi, 0x20
rep stosb
```

1. What is the value of edx after the subtraction operation? What does it represent?

13
Ecx contained 1. Edx contained 14. After sub edx was 13

2. What is the purpose of the above code segment?
The above code segment simply stores the whole string in registers and moves each byte individually for comparison.

3. Provide a complete example that illustrates what the heap of the address esp+20 looks like. How big is it?
The heap at esp+20 should most likely be 16 bytes big and should contain values of all 0s. The different memory addresses would be 0x4b0020, 0x4b0024, 0x4b0028, and 0x4b002c.