Binary Exploitation

Buffer Overflows – 0x2

Roadmap

- Motivation
- Revisiting Memory Layout
  - Stack and Stack Frames
- Oh no! Assembly
  - Changing the stack
- Protostar Challenges

Motivation

- 372 CVE entries in 2017 (~1.5/day)
- Easy to understand
- Practice binary interaction

Revisiting Memory Layout

- Recall:
  - File sections are interpreted as segments during execution
  - Segments are flagged as read, write and/or execute
  - Segments: .text, .bss, .data, heap, **stack**

Revisiting Memory Layout

- The **Stack** is a FILO data structure (*a stack*)
  - Data can be added by **pushing** onto the (top of the) stack
  - Data can be removed by **popping** from the (top of the) stack
- Starts at the “end” of addressable memory, grows into lower memory addresses
- The **stack** allows context to be stored

Revisiting Memory Layout

- The stack gives context by creating **stack frames**
- Stack frames are created when a function is called and include:
  - Function arguments
  - Function local variables
  - Previous frame
  - Where to return (code execution)

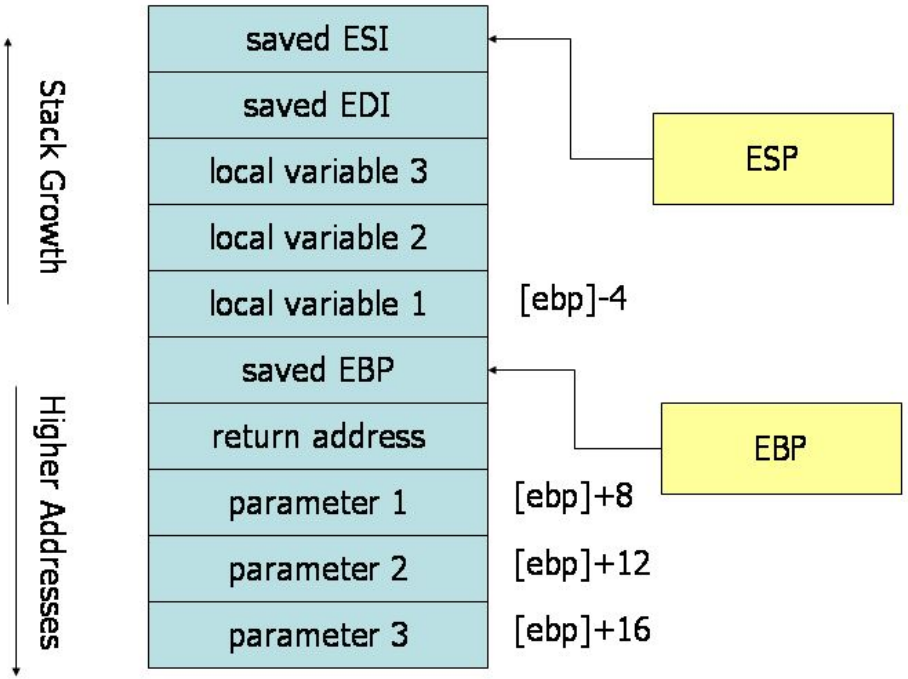
```c
#include <stdio.h>
int main() {
    // This is not available outside main
    int i = 1;
    func();
}

void func() {
    // i is not available here
    i = 2;
}
```

Oh No! Assembly

- **ESP** and **EBP** reference the frame
  - ESP points to the top of the stack
  - EBP points to the base of the stack (for the current frame)
- **PUSH val**
  - **SUB ESP, 0x4**
  - **MOV [ESP], val**
- **POP reg**
  - **MOV reg, [ESP]**
  - **ADD ESP, 0x4**

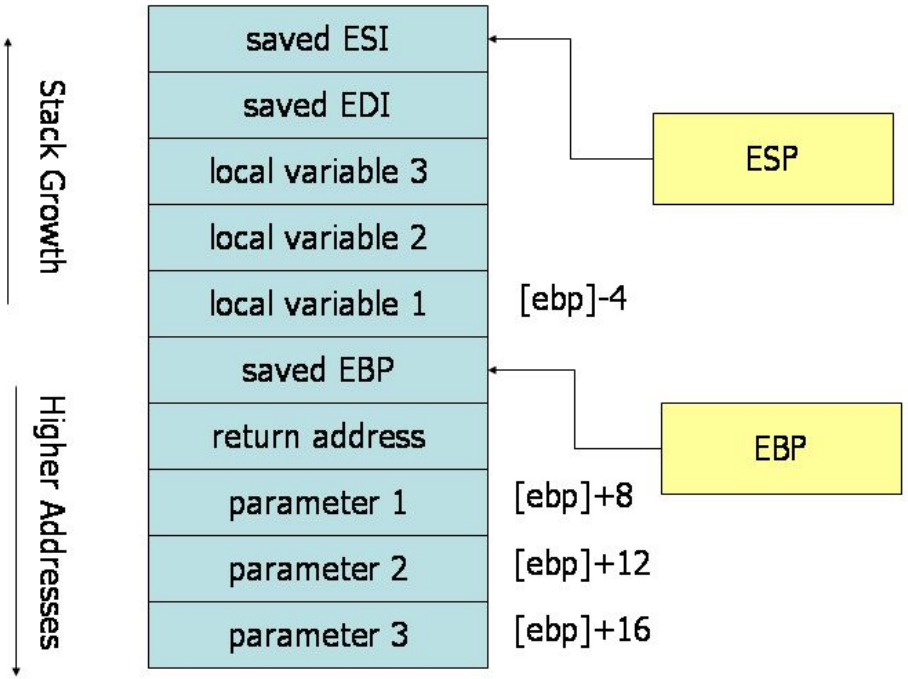
Oh No! Assembly (now with drawings)

- **CALL label**
  - **PUSH EIP**
  - **JMP label**
- <func prologue>
- <user code>
- <func epilogue>
- **RET**
  - **POP EIP**



Oh No! Assembly (now with drawings)

- Function prologue creates a new stack frame
  - **PUSH EBP**
  - **MOV EBP, ESP**
  - **SUB ESP, 0xNN**
- Function epilogue restores a previous stack frame
  - **LEAVE**
    - **MOV ESP, EBP**
    - **POP EBP**



References

Hacking The Art of Exploitation, 2nd Edition - Jon Erickson

Intel IA32 Architecture Manual

Next week...

- More Buffer Overflows
  - With shellcode