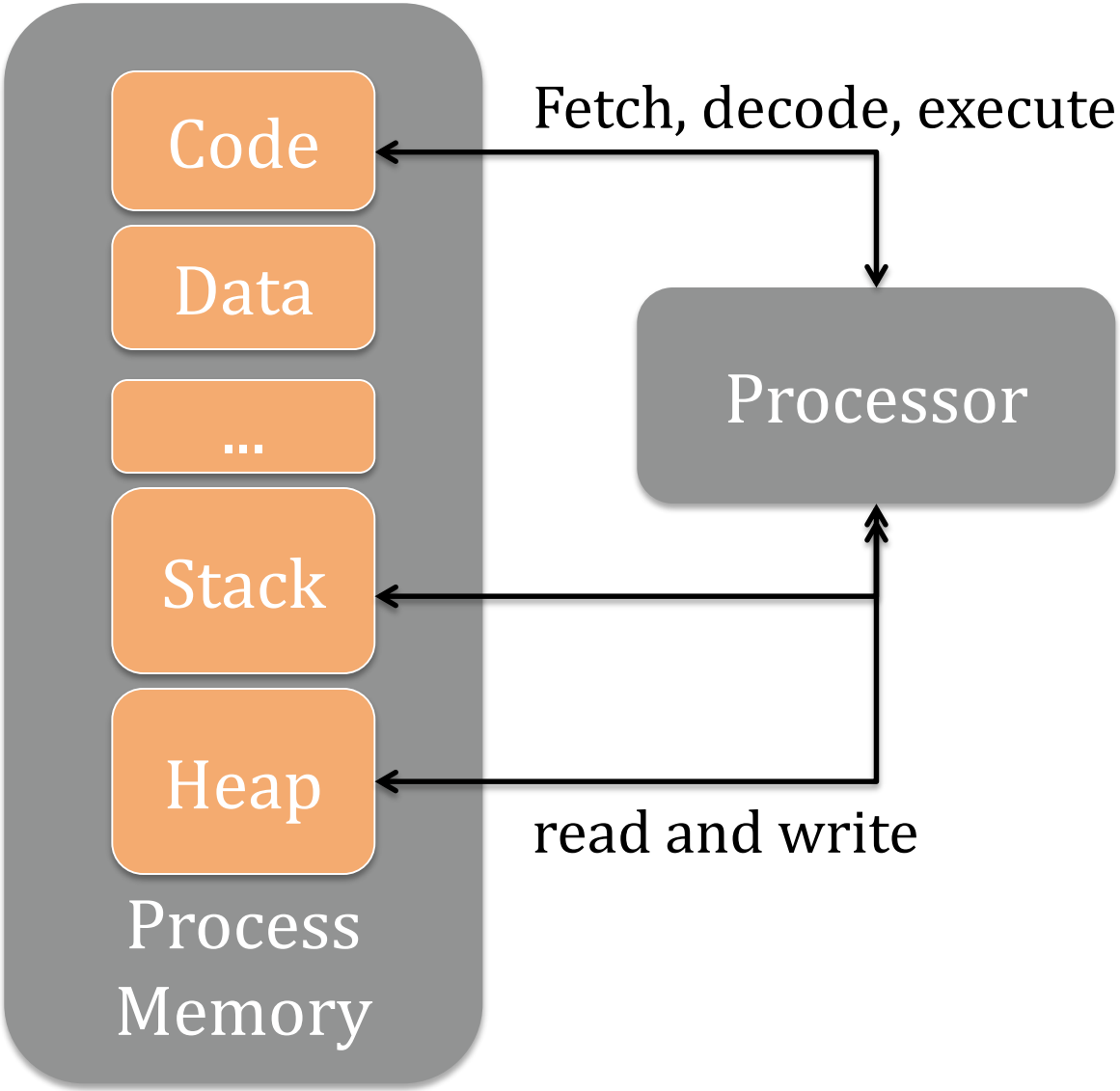
Review: Software Security

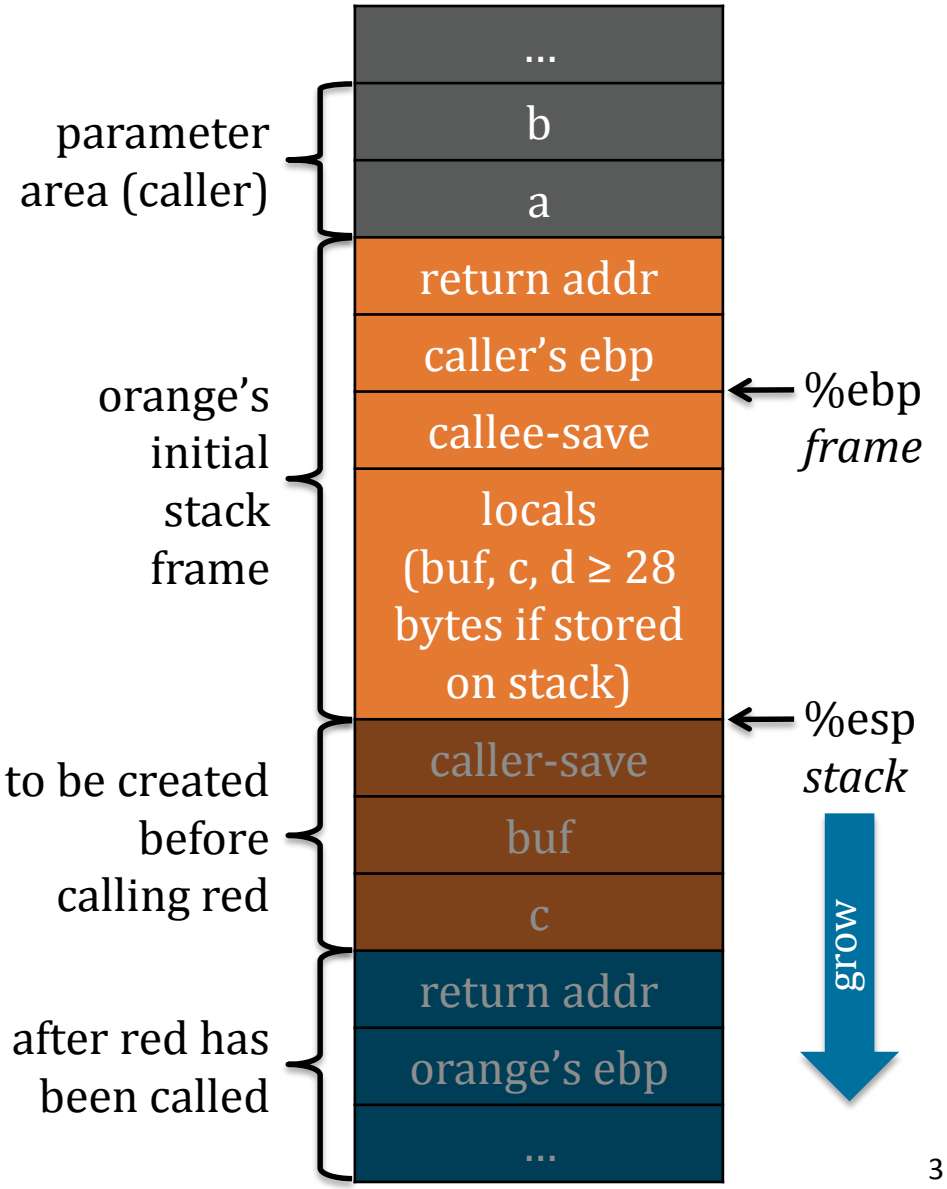
David Brumley
dbrumley@cmu.edu
Carnegie Mellon University

Basic Execution Model



cdecl – the default for Linux & gcc

```
int orange(int a, int b)
{
   char buf[16];
   int c, d;
   if(a > b)
      c = a;
   else
      c = b;
   d = red(c, buf);
   return d;
}
```



Be prepared to draw and analyze stack diagrams

Control Flow Hijack:
Always Computation + Control

computation + ***control***

- code injection
- return-to-libc
- Heap metadata overwrite
- return-oriented programming
- …

Same principle, different mechanism

Channeling Vulnerabilities

... arise when control and data are mixed into one channel.

Situation	Data Channel	Control Channel	Security
Format Strings	Output string	Format parameters	Disclose or write to memory
malloc buffers	malloc data	Heap metadata info	Control hijack/write to memory
Stack	Stack data	Return address	Control hijack
Phreaking	Voice or data	Operator tones	Seize line control

Buffer overflows

- Gaining control through...
  - Overwriting saved return addresses
  - Overwriting function pointers

format strings

- For non-variadic functions, the compiler:
  - knows number and types of arguments
  - emits instructions for caller to push arguments right to left
  - emits instructions for callee to access arguments via frame pointer (or stack pointer [advanced])

- For variadic functions, the compiler emits instructions for the program to

  walk the stack at runtime for arguments

format string exploits

- Occur when the user can control the format string specifier

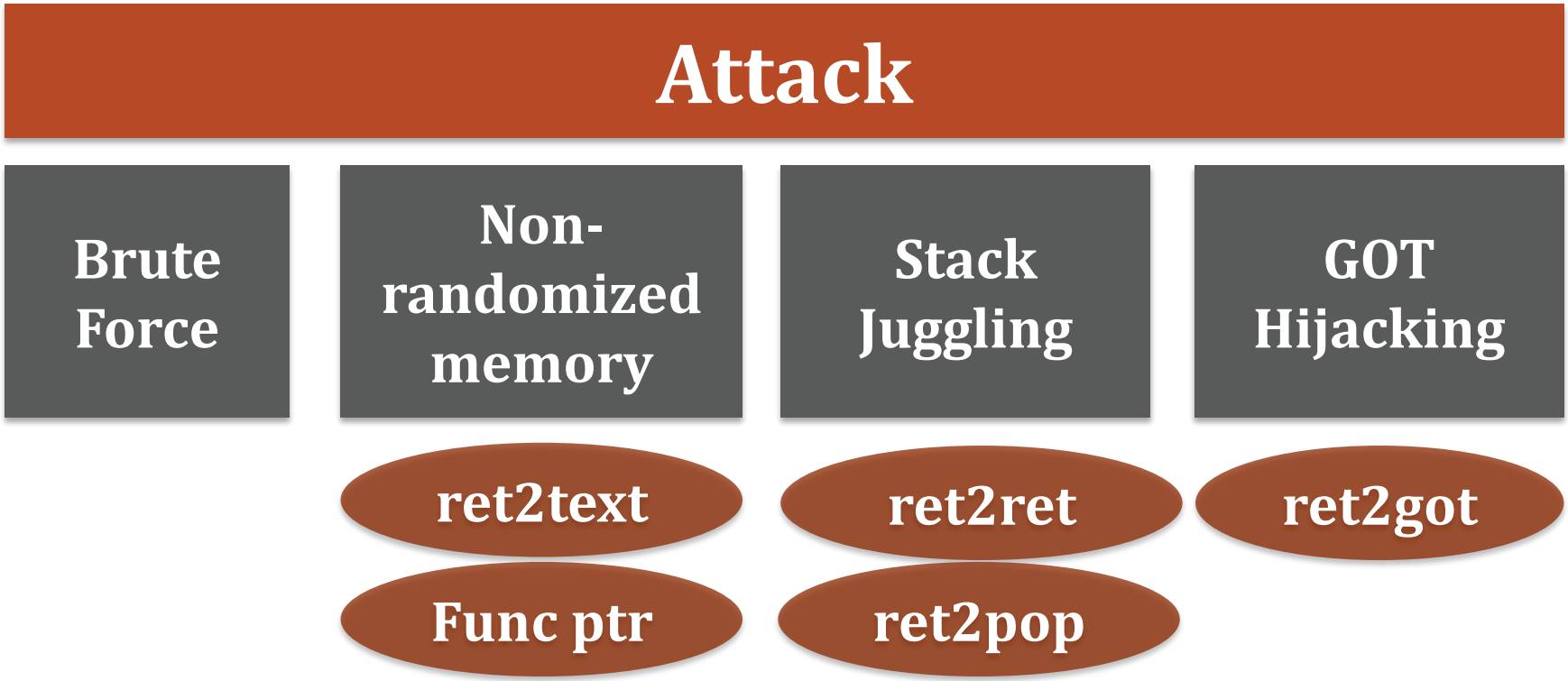
- Can be used to:
  1. View memory (e.g., information disclosure)
  2. Write to specific addresses
  3. sprintf: expand user input to cause a buffer overflow

Defenses

computation + ***control***

Primarily DEP Primarily ASLR

How to attack with ASLR?



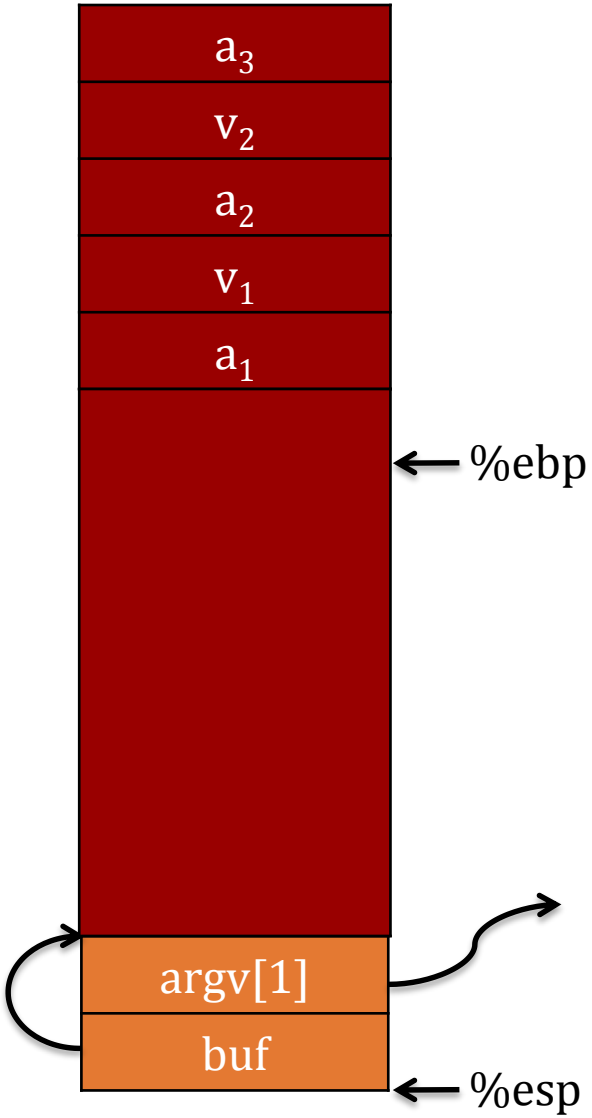
Return-Oriented Programming (ROP) how it works and when it is needed

`Mem[v2] = v1`

Desired *Shellcode*

```
a1: pop eax; ret
a2: pop ebx; ret
a3: mov [ebx], eax
```

Desired store executed!



CFI

- Sound/Complete
- Sensitivity in program analysis
- CFI instrumentation
- CFI assumptions

Test

- In-class
- Timed
- Closed book, closed note, closed computer

Good Luck!


Questions?

END

Thought