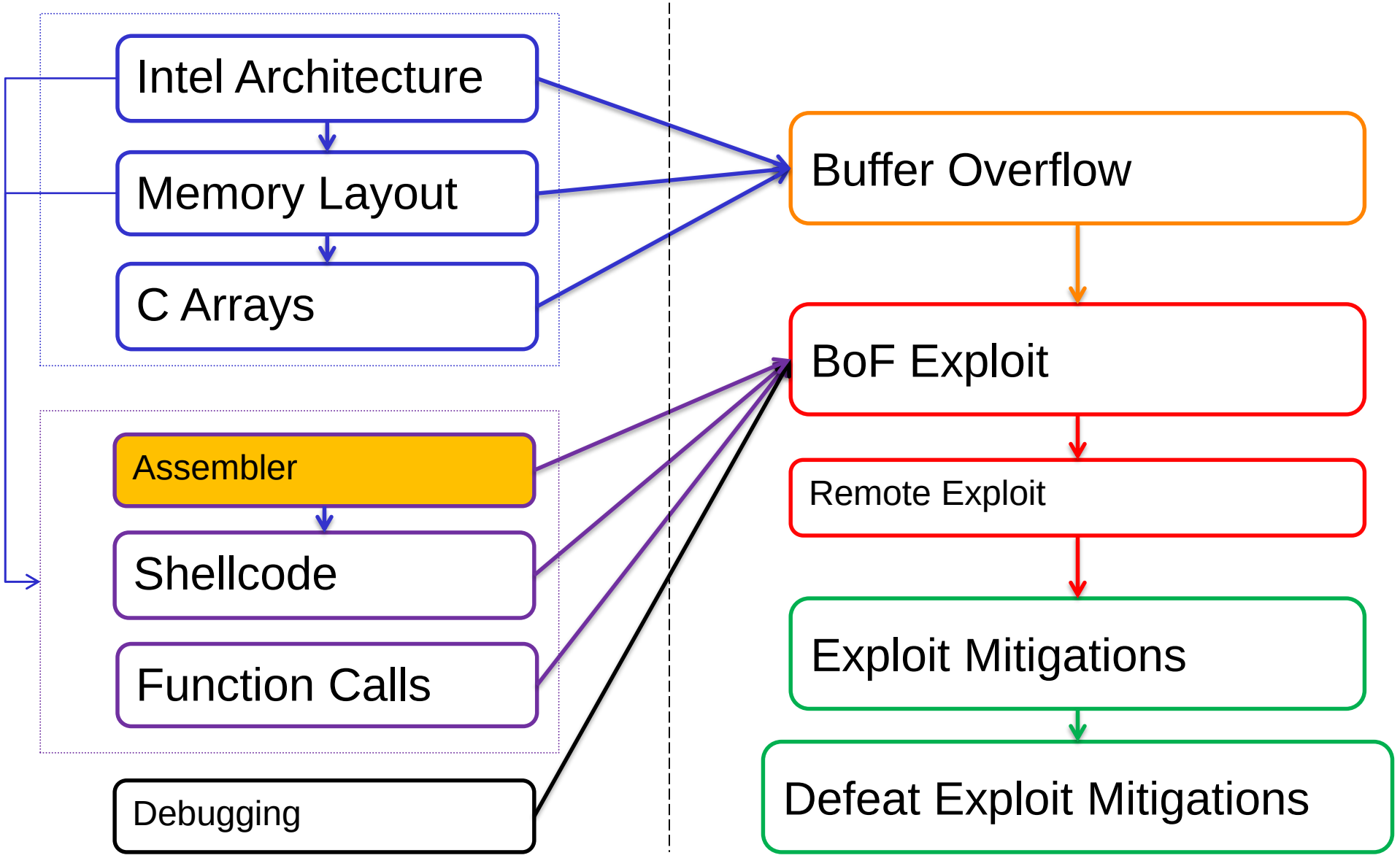
# Assembler 101

# Content


Intel Architecture
Memory Layout
C Arrays
Assembler
Shellcode
Function Calls
Debugging
Buffer Overflow
BoF Exploit
Remote Exploit
Exploit Mitigations
Defeat Exploit Mitigations

# Short Assembler Intro

# Assembler Intro

*Initialize a variable*

```
int number;

number += 1
```

```
number dw 0

mov number, eax

inc eax

mov eax, number
```

# Assembler Intro

*Array on the stack*

```
char test[5];

test[0] = 1;

test[3] = 9;
```

```
sub    $0x10,%esp

movb   $0x1,-0x5(%ebp)

movb   $0x9,-0x2(%ebp)
```

# Assembler Intro

*Array on the heap*

```
char *test = malloc(5);
```

```
sub    $0x28, %esp

movl   $0x5, (%esp)

call   8048300 <malloc@plt>

mov    %eax, -0xc(%ebp)
```

```
test[3] = 9;
```

```
mov    -0xc(%ebp), %eax

add    $0x3, %eax

movb   $0x9, (%eax)
```

# Assembler Intro

*Conditional statement*

```
if (number < 0) {
   <smallerzero>
}
<restofcode>
```

```
number dw 0
mov number, eax
cmp eax, 0
jge label    # jump greater equal
<smallerzero>
label:
<restofcode>
```

# Assembler Intro

*Loop*

```
int n;

for(n=0; n<12; n++)

{

  printf("A");

}
```

```
03    sub    $0x28,%esp

06    movl   $0x0,-0xc(%ebp)

13    jmp    0x8048403 <bla+31>

15    movl   $0x41,(%esp)

22    call   0x8048320 <putchar>

27    addl   $0x1,-0xc(%ebp)

31    cmpl   $0xb,-0xc(%ebp)

35    jle    0x80483f3 <bla+15>
```

# Online Assemblers

Compile Online with NASM

- https://www.jdoodle.com/compile-assembler-nasm-online
- https://www.tutorialspoint.com/compile_assembly_online.php

Decompile C source Code:

- https://godbolt.org/
- https://retdec.com/decompilation-run/