

Lab 3

Naming Convention

- ZIP archive:
 - `FirstName_LastName.zip`
- When unzipped:
 - Code directory: `code/`
 - Report PDF: `FirstName_LastName.pdf`

Main Goals

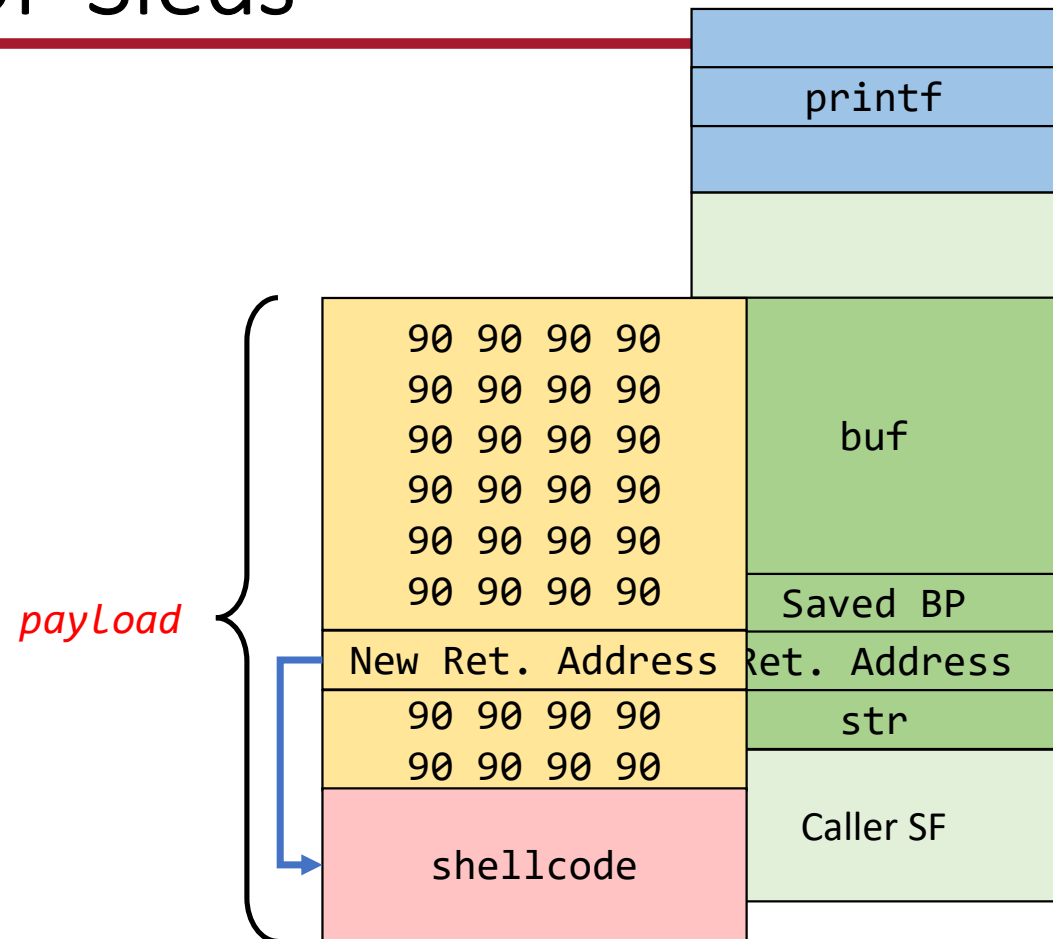
- **Optimize** shellcode to exploit buffer overflow (BOF) vulnerabilities
- **Analyze** potential BOF vulnerabilities in source code
- **Exploit** BOF vulnerabilities using different techniques

Task 1: Optimize Shellcode

- The Assembly code for `labsh.asm` is provided. You need to:
 - Analyze why it cannot be used to exploit BOF in a given C program
 - Optimize the bytecode (shellcode) for `labsh.asm` program
 1. Make it suitable for BOF
 2. Reduce its size (as much as you can)

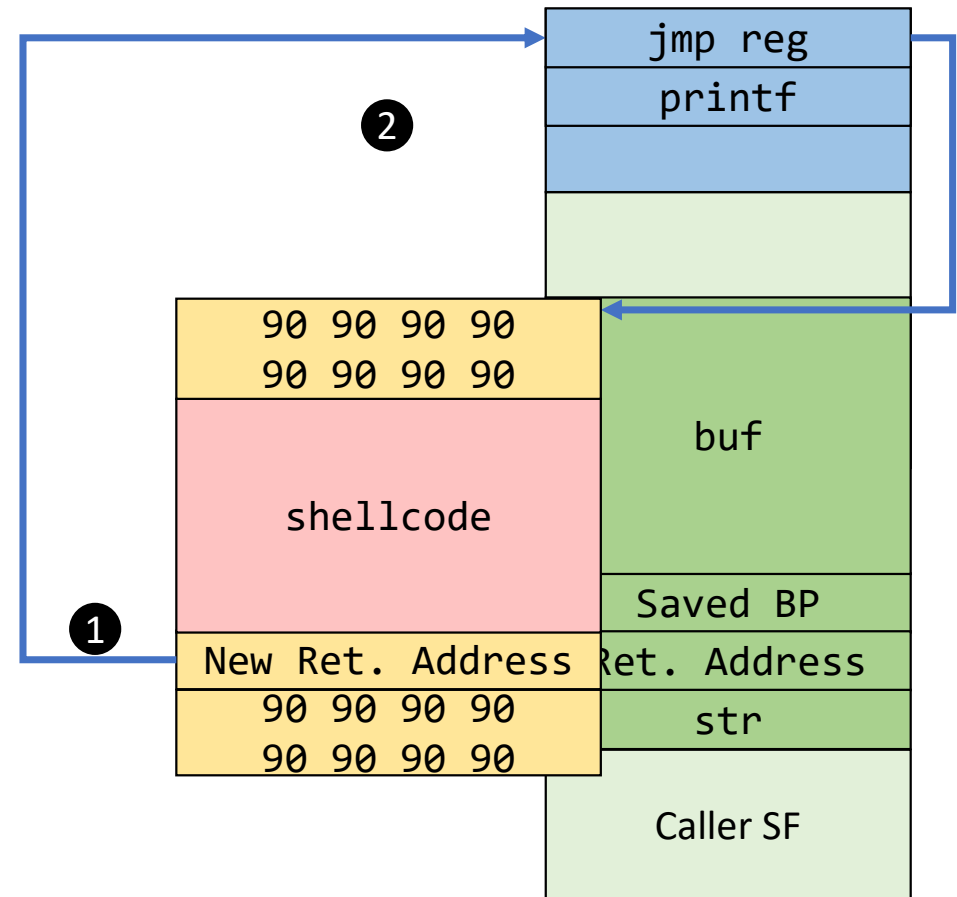
Task 2: Exploit BOF – NOP Sleds

- A vulnerable C code is given.
- You need to generate payload to exploit BOF (using NOP sleds)
- The shellcode is the one you optimized in Task 1
- Few notes:
 - The C program copies a relatively large number of bytes
 - The shellcode needs to be at the end of the payload
 - You need to modify the BUF_SIZE in the C code
 - Some randomness is added



Task 3: Exploit BOF – JMP-to-REG

- A vulnerable C code is given.
- You need to generate payload to exploit BOF (using jmp-to-reg)
- The shellcode is the one you optimized in Task 1



Task 3: Exploit BOF – JMP-to-REG

Two subtasks:

(1) Analysis

- Analyze the C program
- (if needed) Modify the program to make it vulnerable to jmp-to-reg
- Determine what general-purpose register to jump to (not esp)

(2) Exploitation

- Determine the gadget address (using ropper and gdb)
- Generate a suitable payload

Questions?
