

# Lab 6

# Main Goals

---

- **Build** various ROP chains
- **Explore** different ROP gadgets
- **Bypass** NX bit using ROP

# Task 1: Setting rax Value

---

- Create a ROP chain to set rax value to 21
- You shouldn't use `inc rax/inc eax`
- Think of different arithmetic operations!

## Task 2: Open a Shell

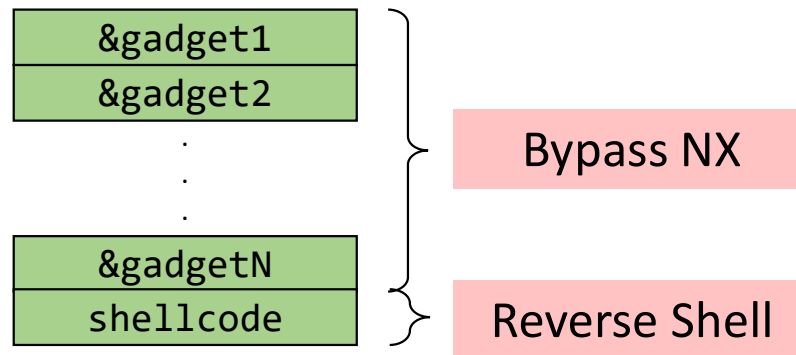
---

- Using the `execve` system call
- Main Steps:
  - `rdi` = address of null-terminated **string**
  - `rsi` = `NULL`
  - `rdx` = `NULL`
  - `rax` = `0x3b`
  - Invoke `int 0x80` or `call gs:[0x10]`
- Where can you insert the string?
  - Can you use the stack?

## Task 3: Open a Reverse Shell

---

- Using ROP + Shellcode. Is this possible?

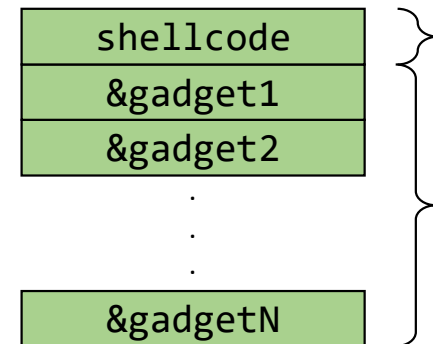
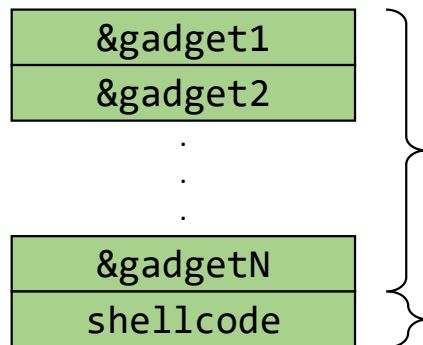


You may place the shellcode in any proper location in your payload.

## Task 3: Open a Reverse Shell

---

- Using ROP + Shellcode. Is this possible?

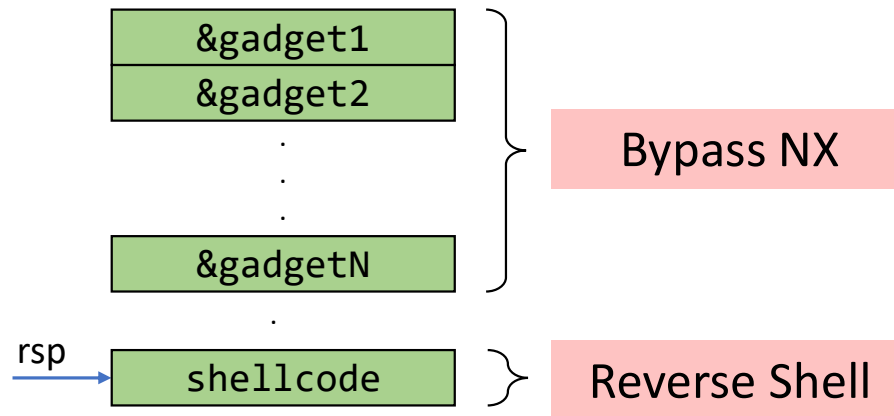


You may place the shellcode in any proper location in your payload.

# Task 3: Open a Reverse Shell

---

- If shellcode is placed **after** the ROP chain?
  - After ROP chain is done: esp would be pointing to shellcode
  - This cannot execute the shellcode

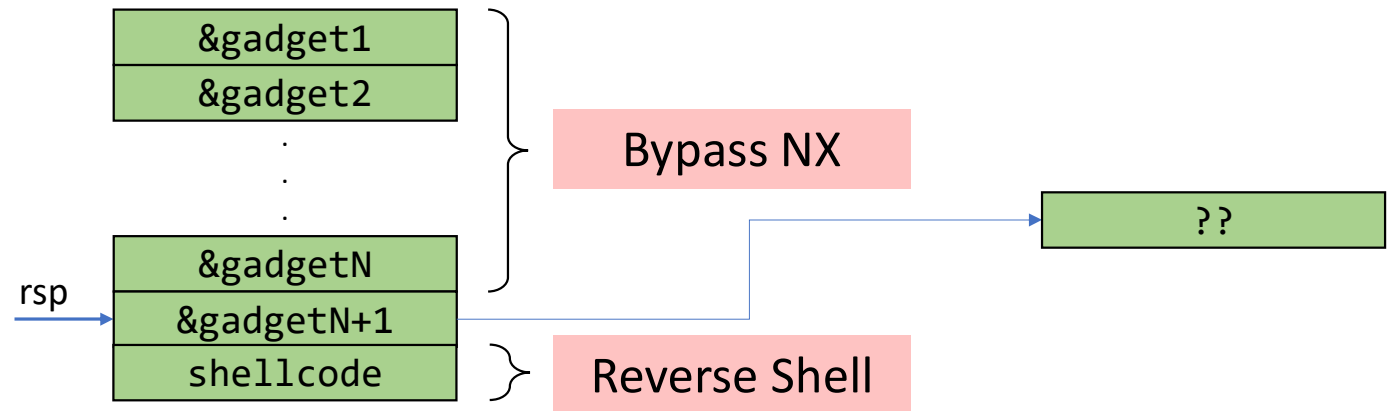


How should your shellcode start?

# Task 3: Open a Reverse Shell

---

- If shellcode is placed **after** the ROP chain?
  - What gadget can control eip?

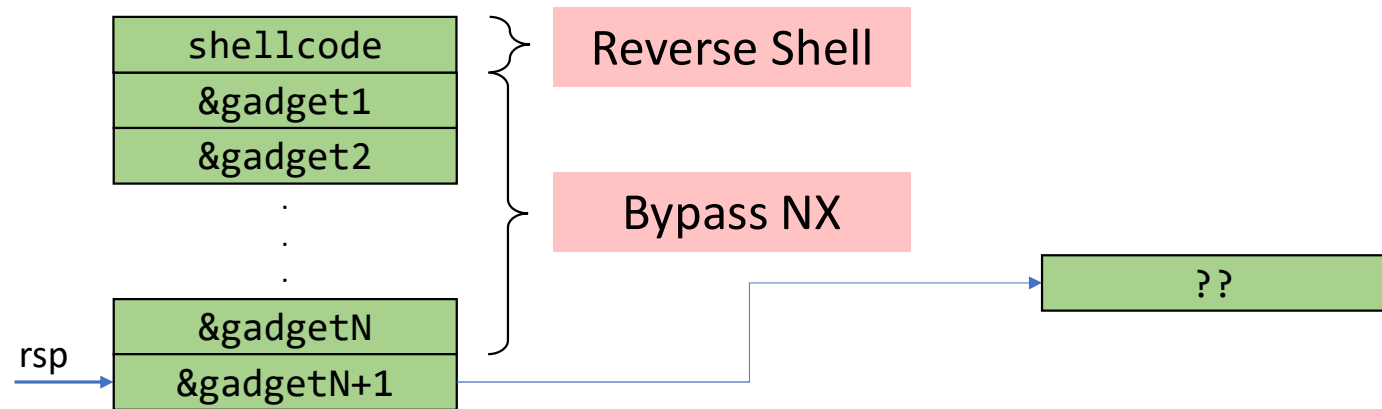




## Task 3: Open a Reverse Shell

---

- If shellcode is placed **before** the ROP chain?



# Helpful Tools and Commands

---

- `gdb > info files`    # files linked to the binary and sections addresses
- ROPgadget
- ropper

# Questions?

---