HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts and Telecommunications Institute of Technology

KIỂM THỬ XÂM NHẬP

KHOA AN TOÀN THÔNG TIN
TS. ĐINH TRƯỜNG DUY

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Posts and Telecommunications Institute of Technology

KIỂM THỬ XÂM NHẬP

Cách hoạt động của hàm trong ngăn xếp

KHOA AN TOÀN THÔNG TIN
TS. ĐINH TRƯỜNG DUY

Biên soạn từ bài giảng: Nguyễn Ngọc Điệp, Bài giảng Kiểm thử xâm nhập, Học viện Công nghệ Bưu chính Viễn thông, 2021.

Mục lục

Khái niệm về hàm

- Là một chương trình con (subroutine) đặc biệt
  - Có thể tái sử dụng khối mã
  - Có thể được gọi từ bất kỳ đâu trong chương trình
  - khi một hàm được gọi, chương trình sẽ thực hiện các lệnh trong hàm, sau đó quay lại lệnh tiếp theo sau lời gọi hàm

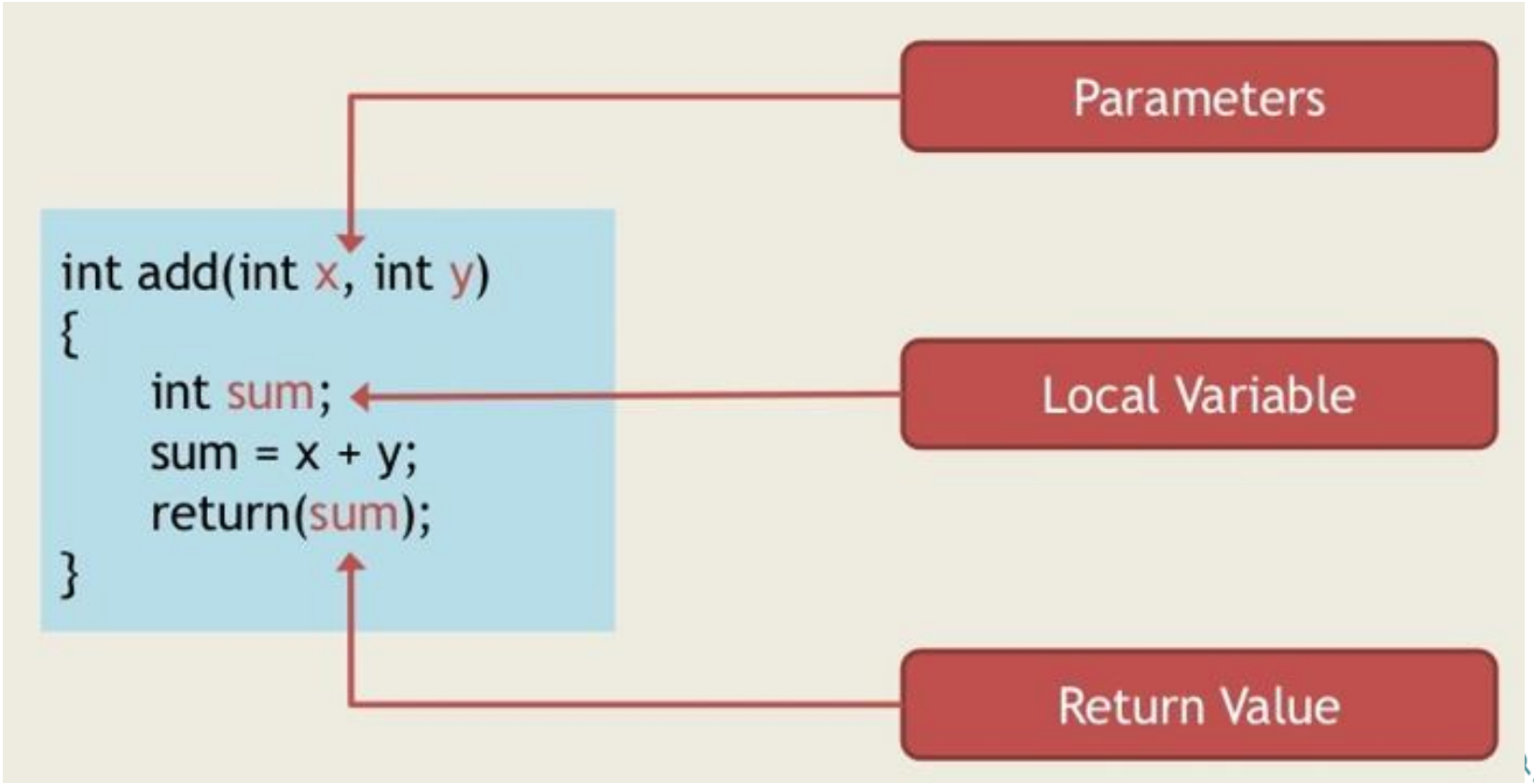
Khái niệm về hàm

- Một hàm có khả năng nhận các tham số đầu vào
- Một hàm trả về một giá trị
- Có thể có các biến cục bộ
  - Được tạo ra khi hàm được gọi trong chương trình
  - Và bị hủy khi hàm thực hiện xong (trả về)
  - Chỉ tồn tại trong phạm vi của hàm

Ví dụ 1: add(x,y)

```
int add (int x, int y)
{
    int sum;
    sum = x + y;
    return (sum);
}
```

Ví dụ 1: add(x,y)


Parameters
int add(int x, int y)
{
int sum;
sum = x + y;
return(sum);
}
Local Variable
Return Value

Vị trí các giá trị được lưu

- Tham số đầu vào được truyền thế nào?
- Các biến cục bộ được lưu ở đâu?

→ Được thực hiện thông qua ngăn xếp (stack)

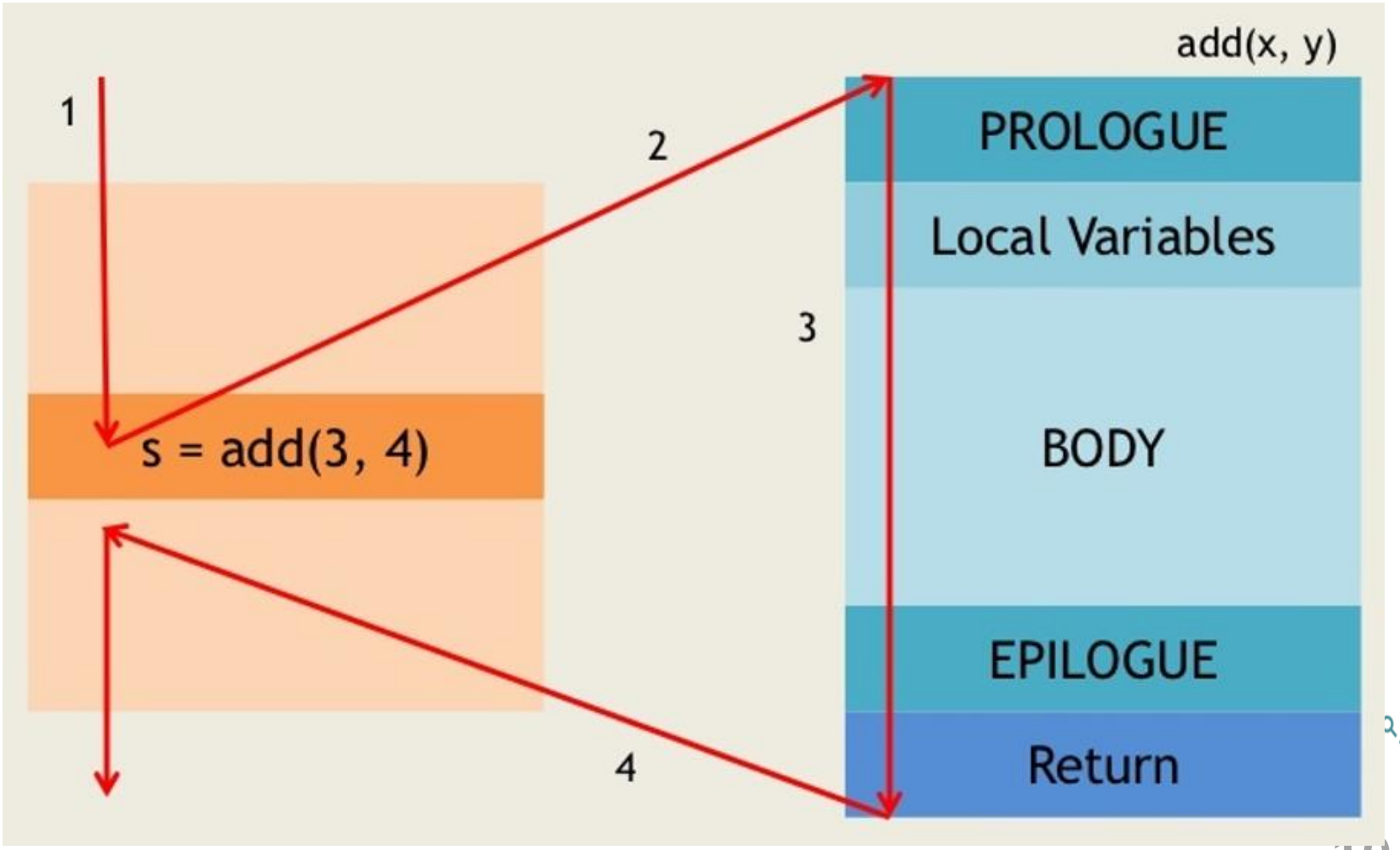
✓Các tham số đầu vào được đẩy vào trong ngăn xếp trước khi gọi hàm

✓Các biến cục bộ được lưu trong bộ nhớ của ngăn xếp

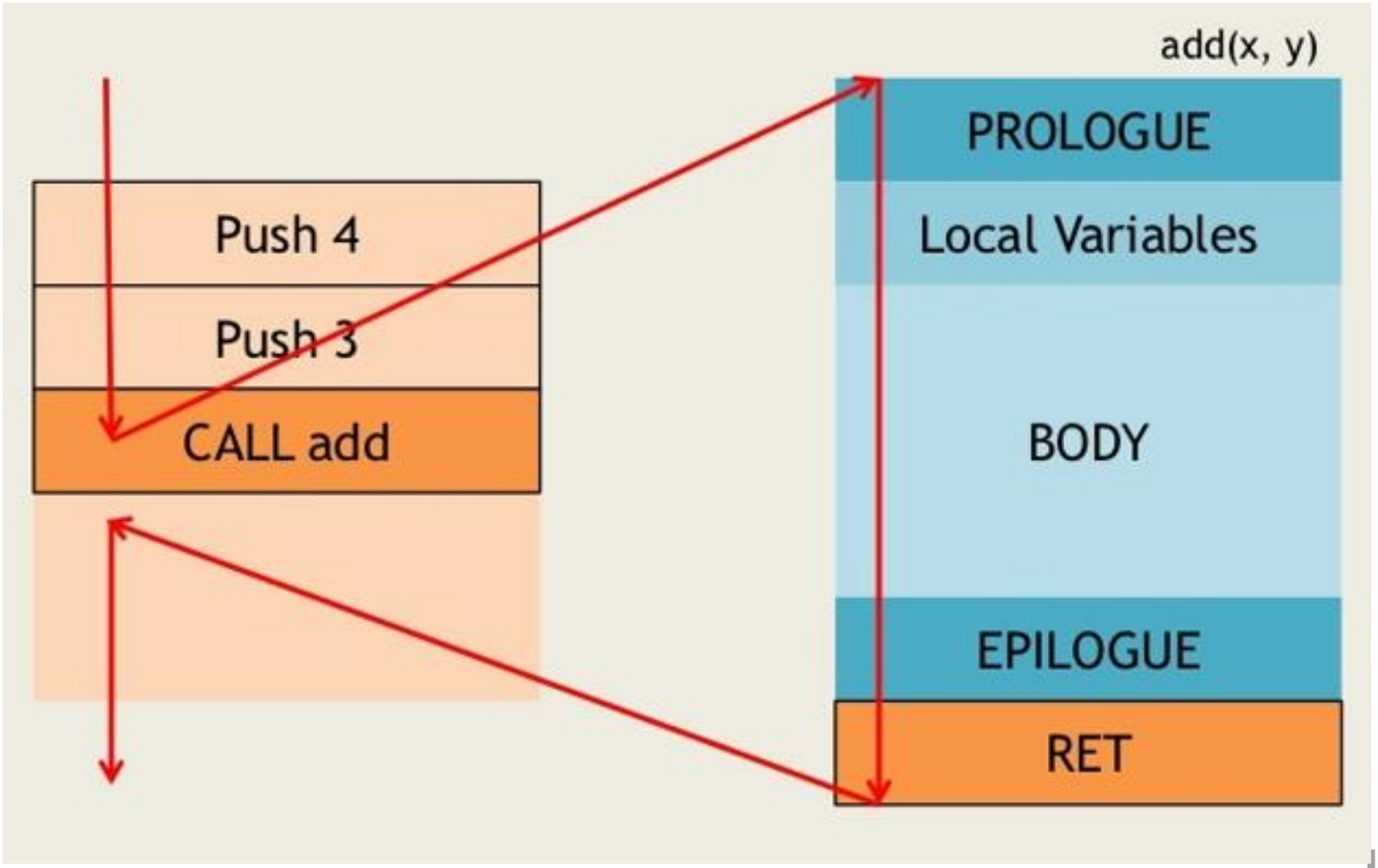
Gọi hàm (calling a function)

- Đẩy các tham số đầu vào vào ngăn xếp
- Nhảy đến địa chỉ bắt đầu của hàm để bắt đầu thực thi nội dung của hàm
- Thực thi mã hàm
- Quay lại câu lệnh tiếp theo sau khi gọi hàm

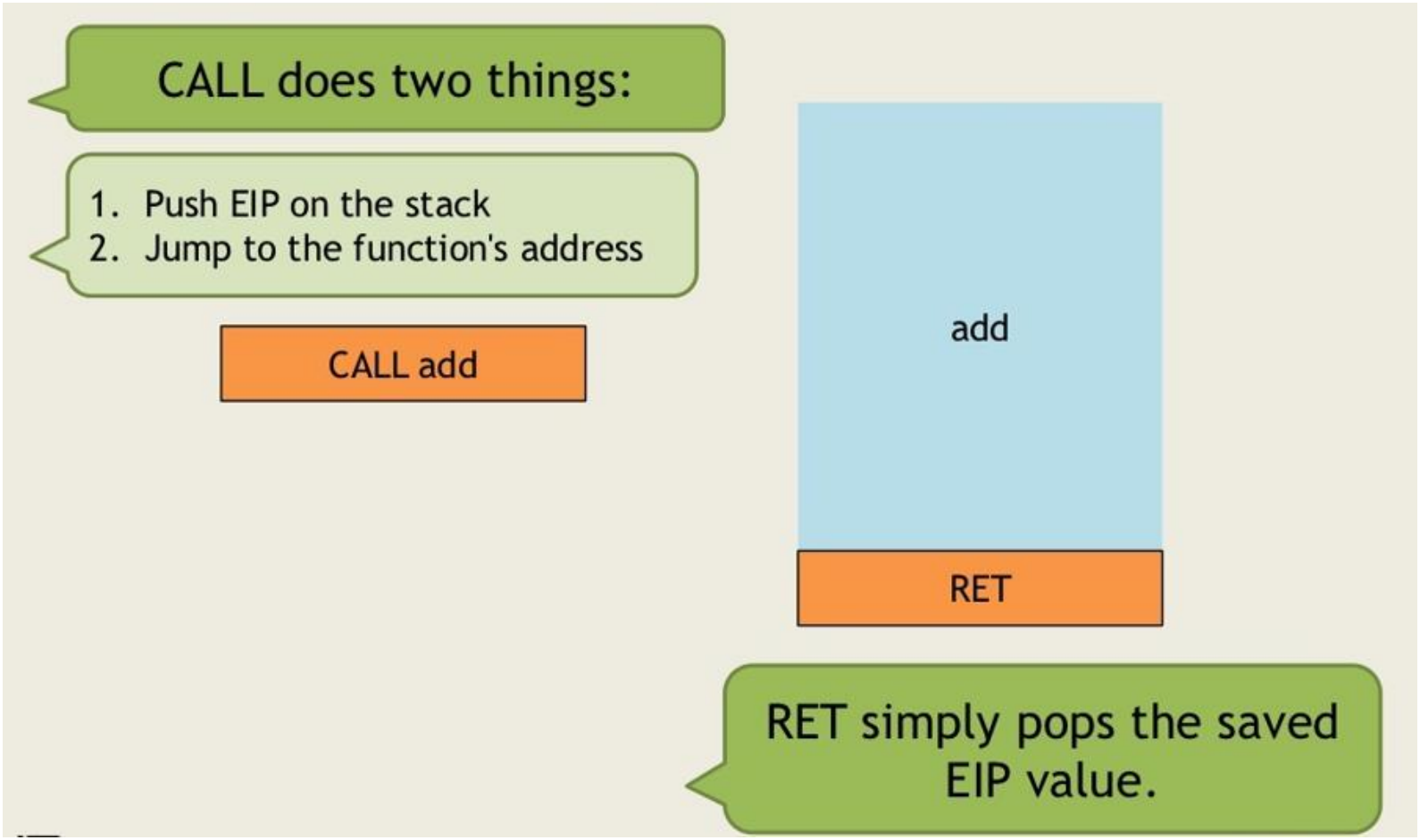
Gọi hàm (calling a function)



Gọi hàm (calling a function)

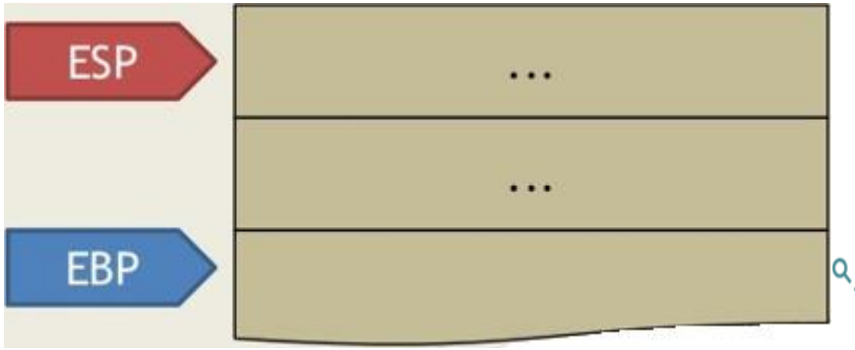


Gọi hàm (calling a function)



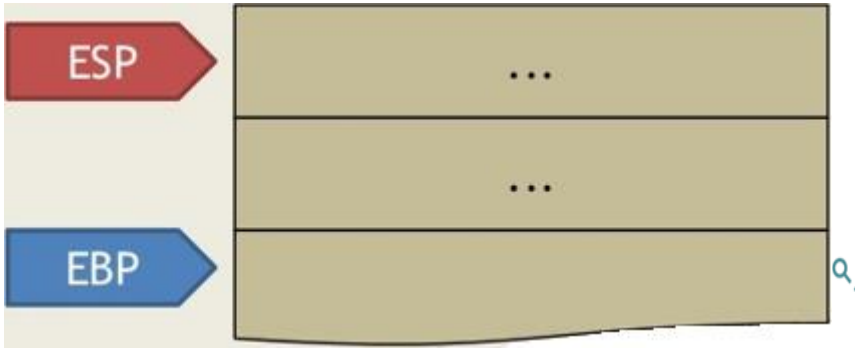
Bắt đầu

- ESP thường trỏ đến đầu ngăn xếp



Bắt đầu

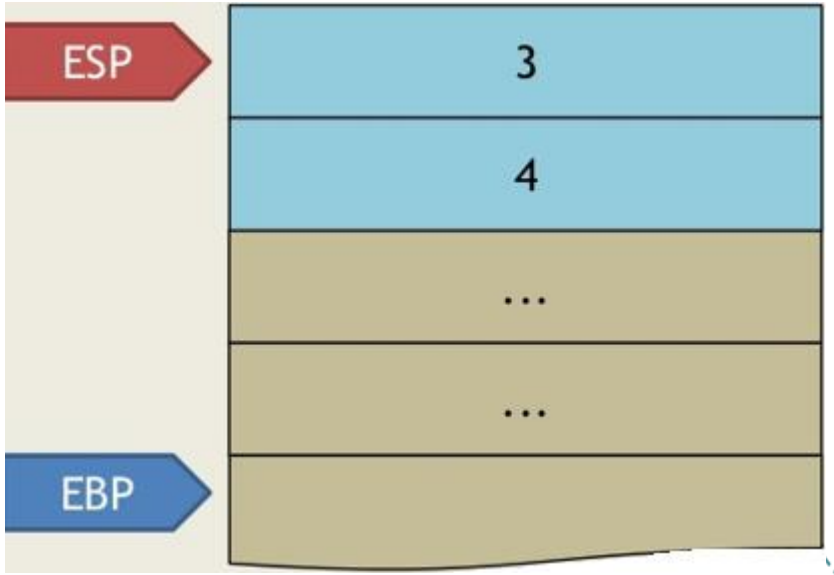
- ESP – con trỏ ngăn xếp, thường trỏ đến đầu ngăn xếp
- EBP – con trỏ cơ sở, chỉ đến vùng nằm trong ngăn xếp



Đưa tham số đầu vào

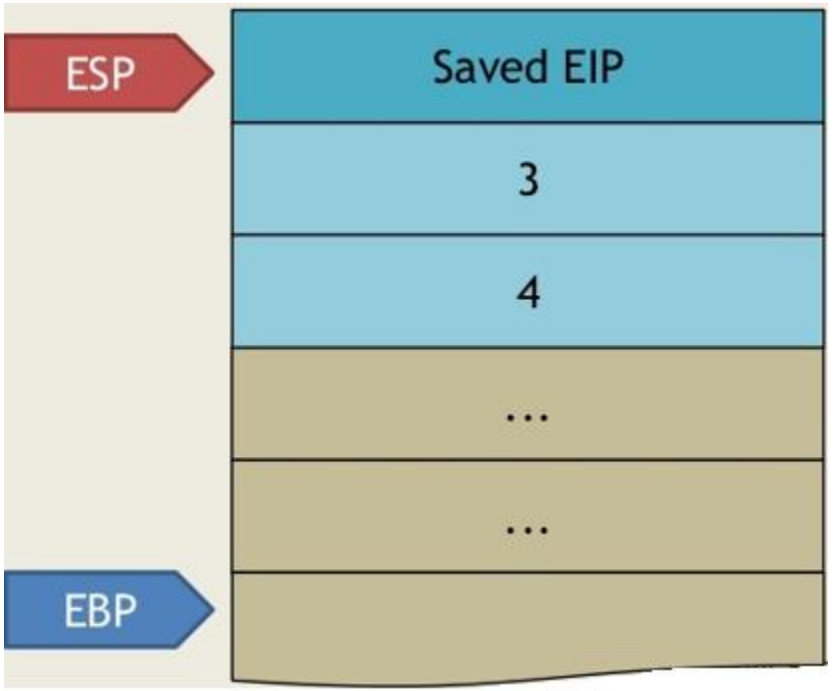
add(3,4) → đẩy 3 và

4 vào trong ngăn xếp



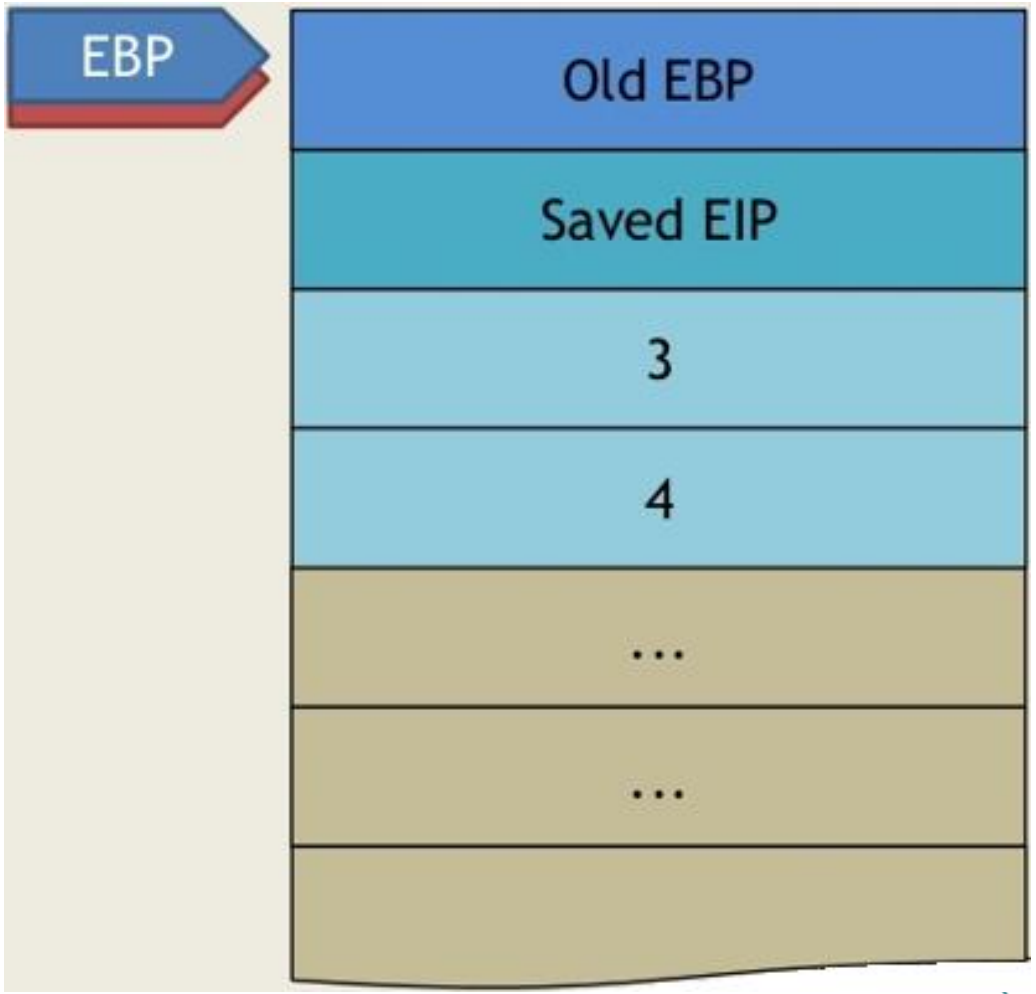
CALL add

- Lệnh CALL đặt EIP hiện tại vào ngăn xếp và nhảy đến lệnh add()



Prologue

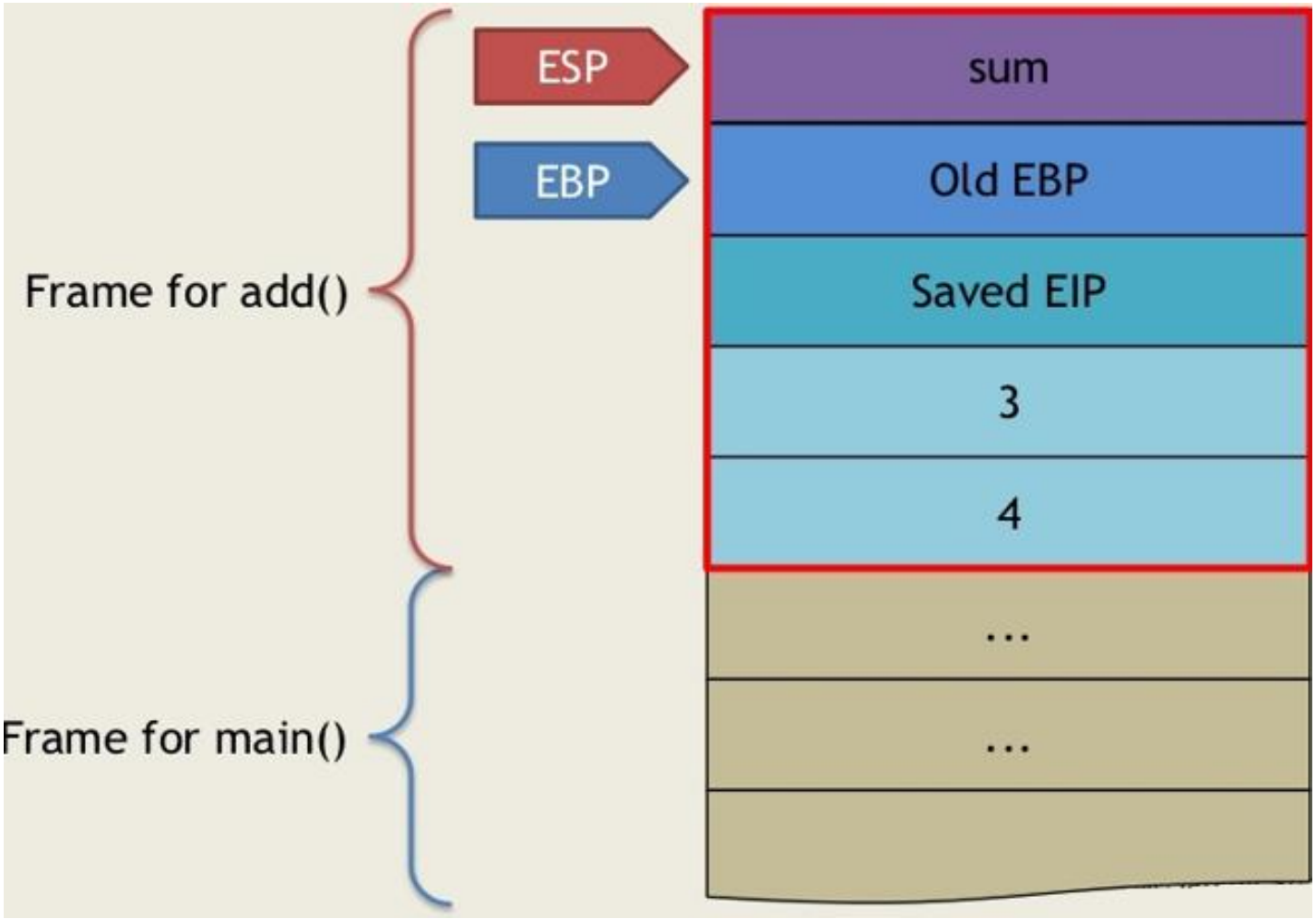
- Prologue lưu vị trí EBP cũ và đặt EBP lên đầu của ngăn xếp



Biến cục bộ

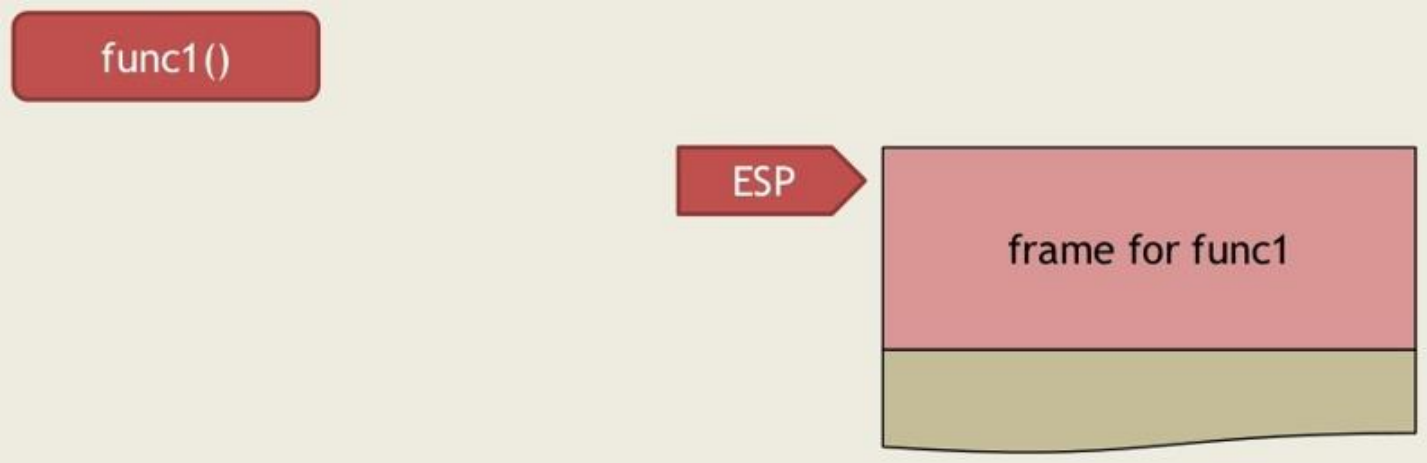
- Các biến cục bộ được tạo trong bộ nhớ của ngăn xếp (stack)

The stack frame (khung ngăn xếp)



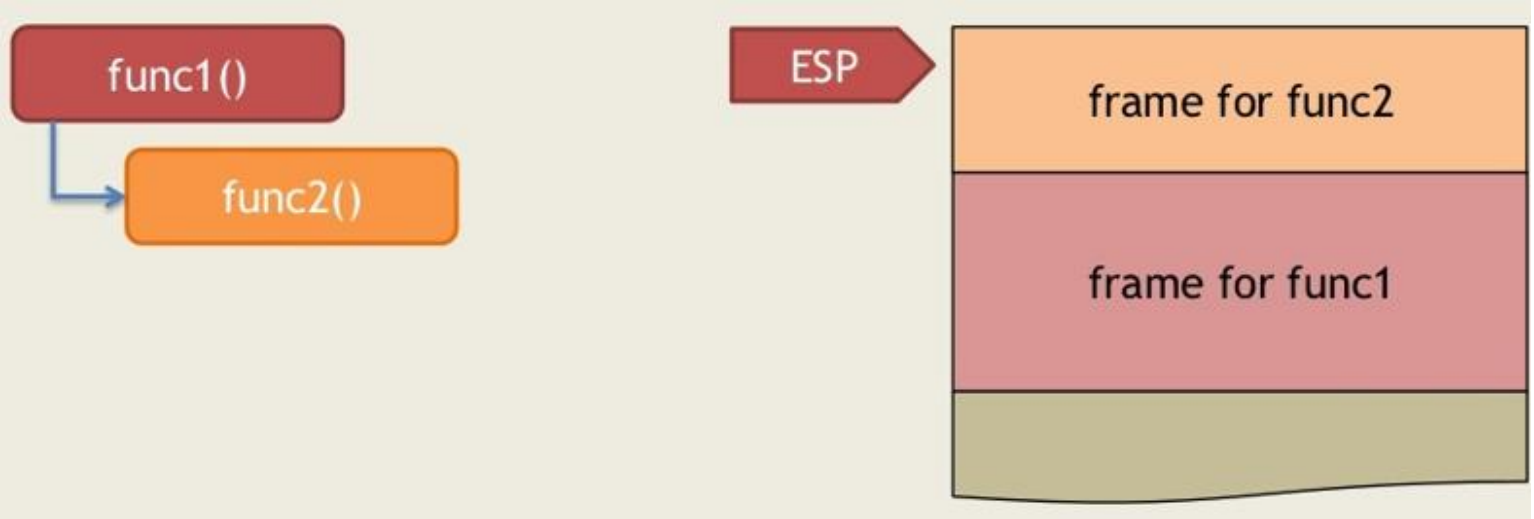
Các hàm và khung

- Mỗi lệnh gọi hàm dẫn đến một khung mới được tạo trên ngăn xếp.



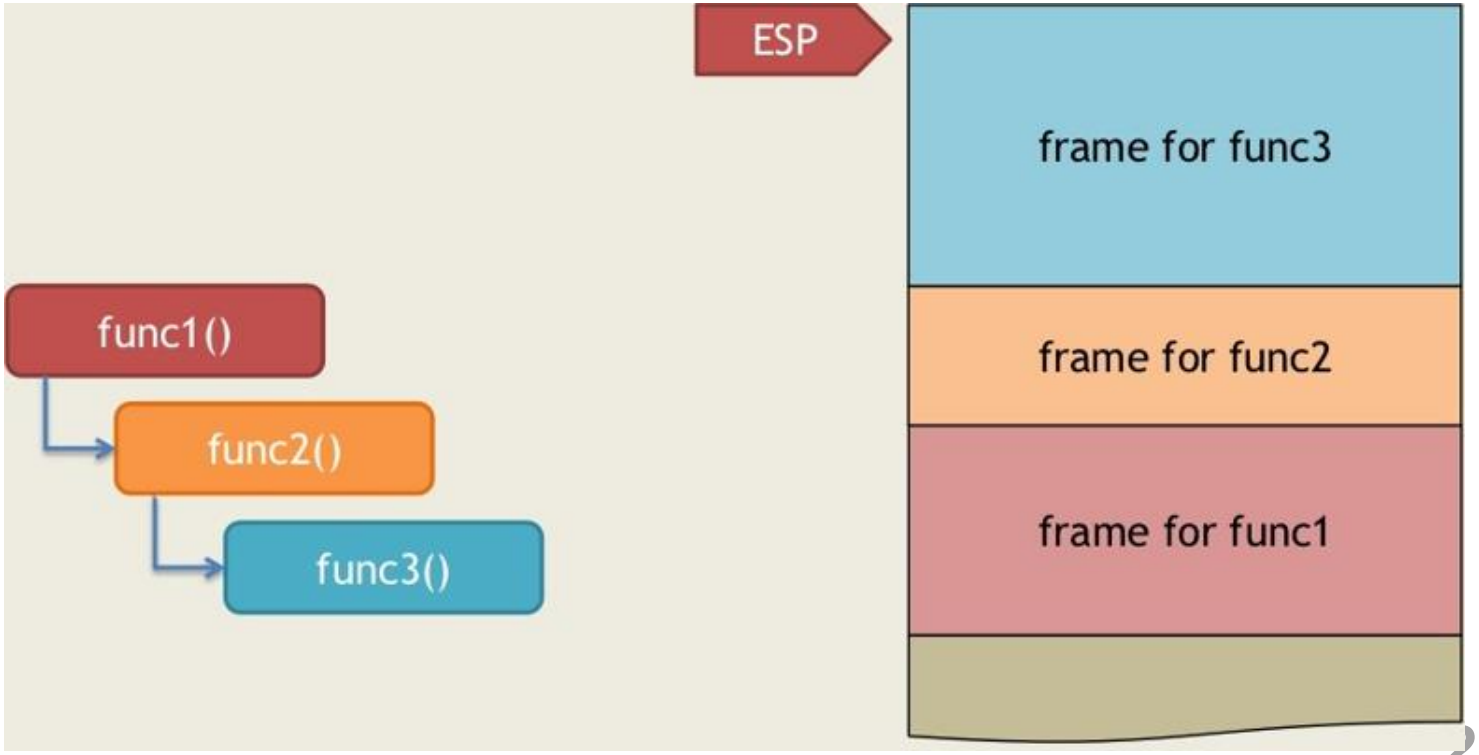
Các hàm và khung

- Mỗi lệnh gọi hàm dẫn đến một khung mới được tạo trên ngăn xếp.



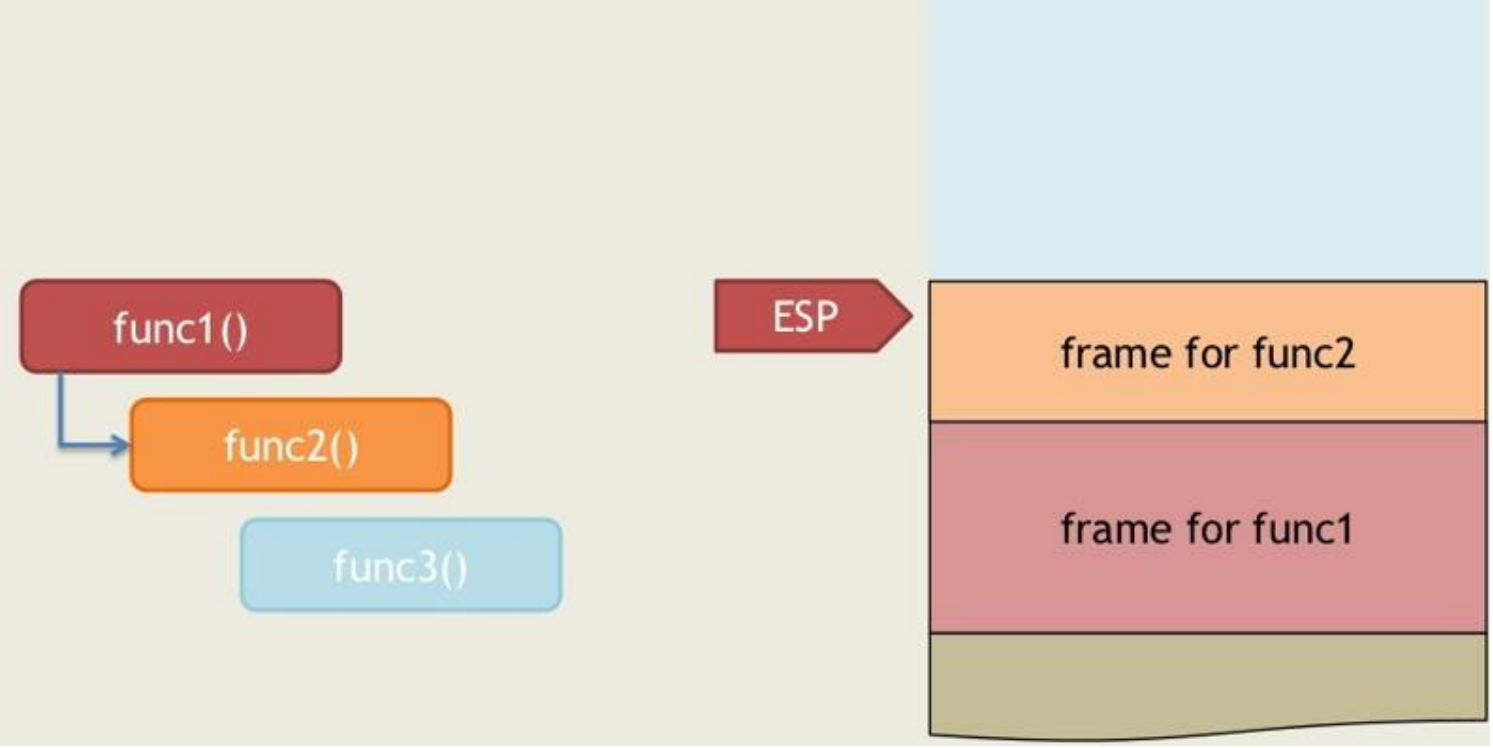
Các hàm và khung

- Mỗi lệnh gọi hàm dẫn đến một khung mới được tạo trên ngăn xếp.



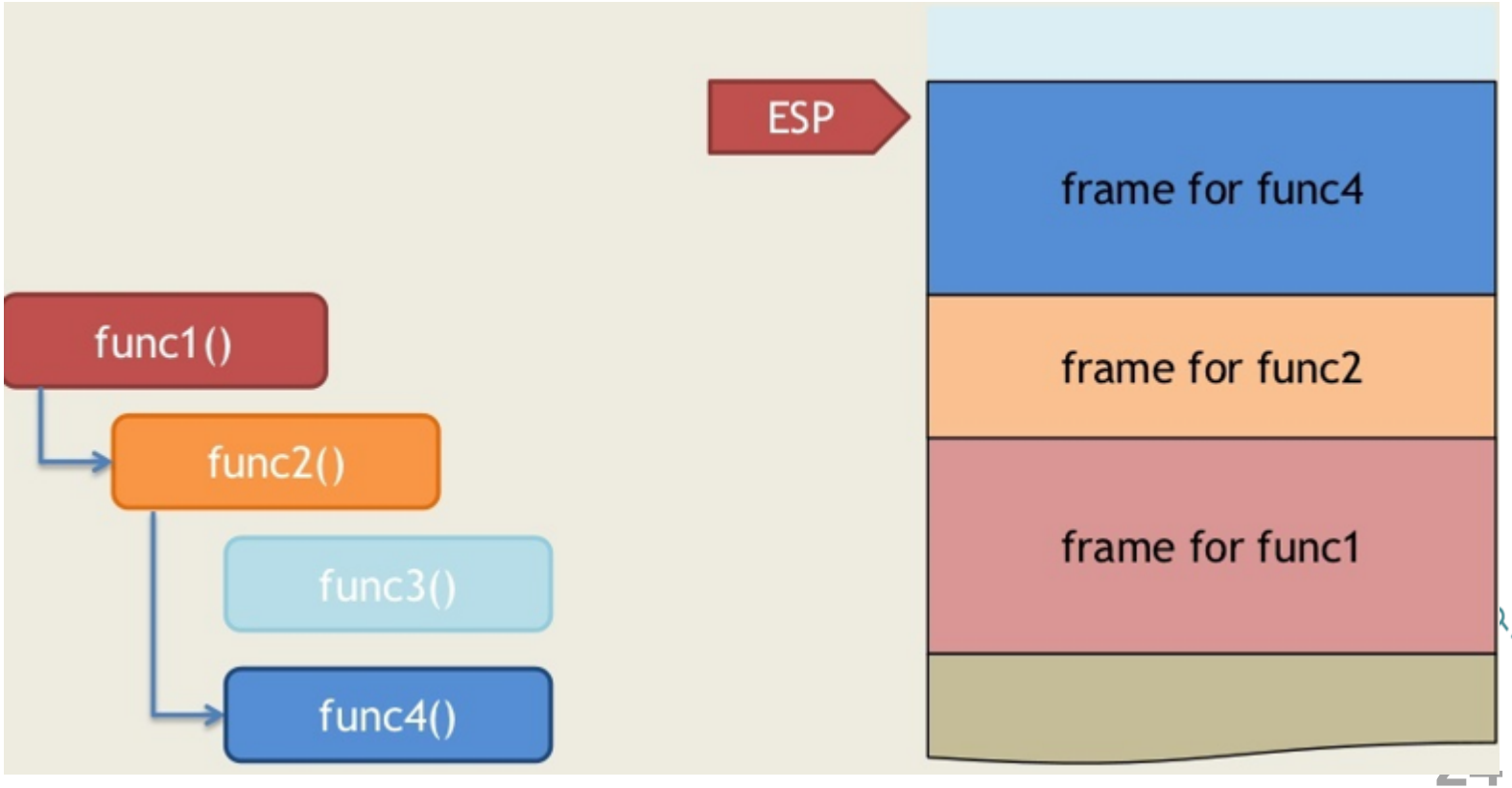
Các hàm và khung

- Khi một hàm trả về, khung sẽ "unwound" (bung ra) hoặc "collapsed" (thu lại).



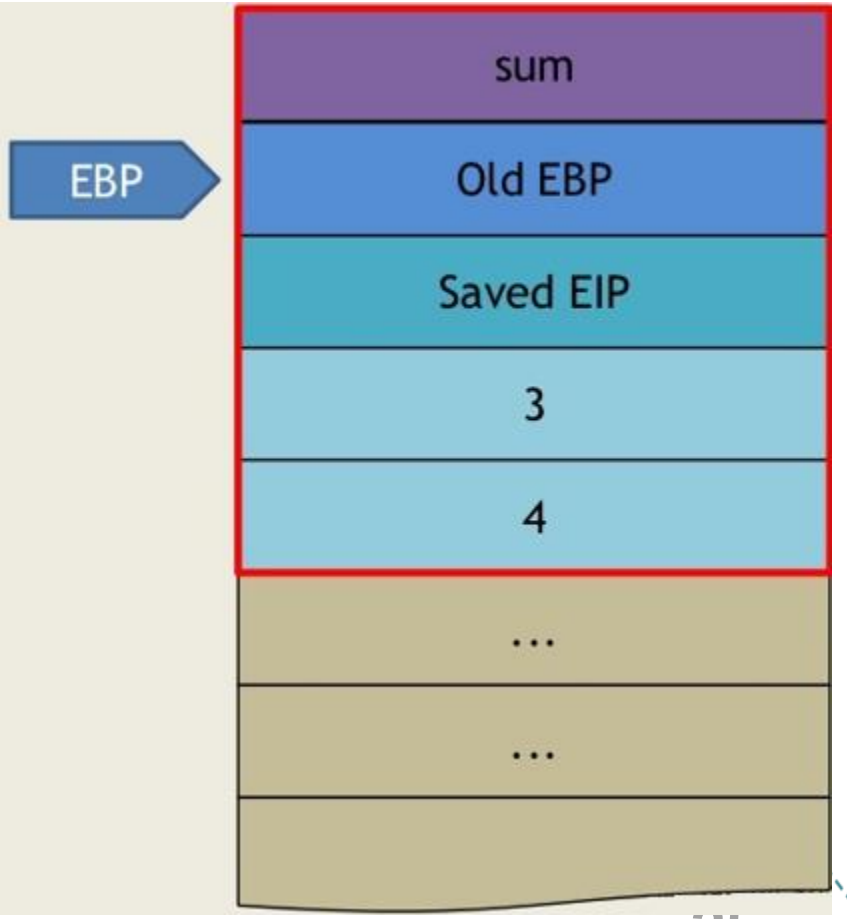
Các hàm và khung

- Khi hàm mới được gọi, các khung mới cũng được tạo ra.



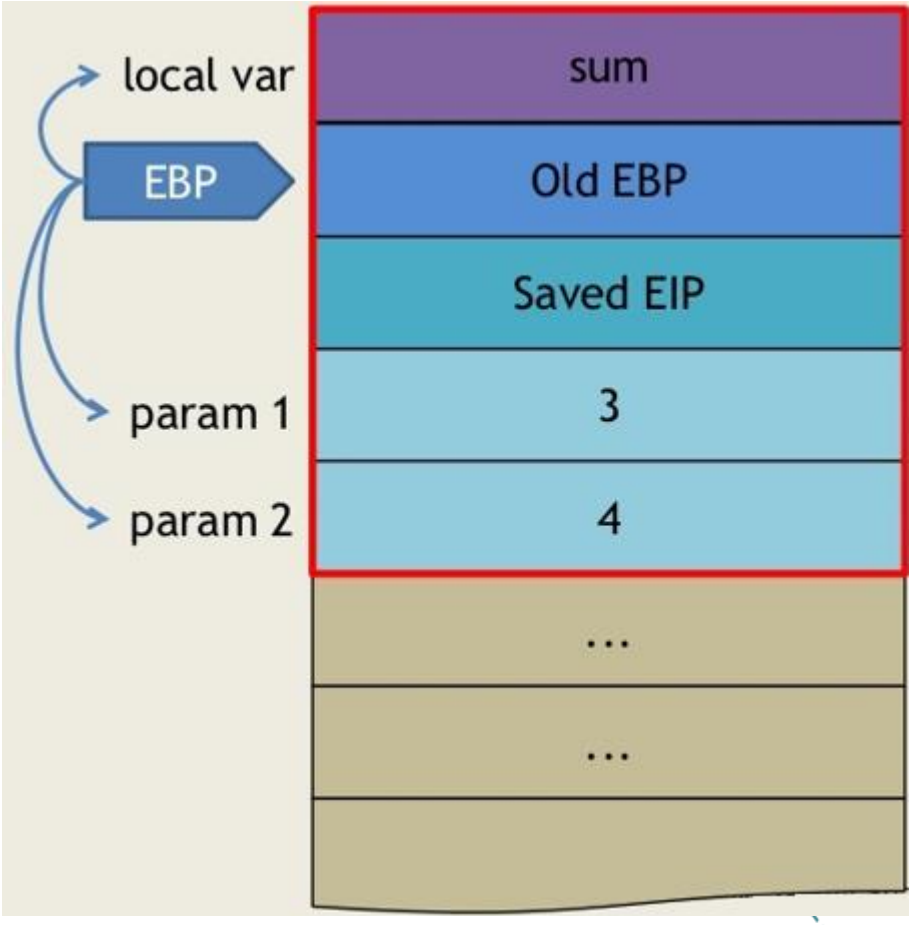
The frame pointer

- EBP- con trỏ cơ sở hay còn gọi là frame pointer



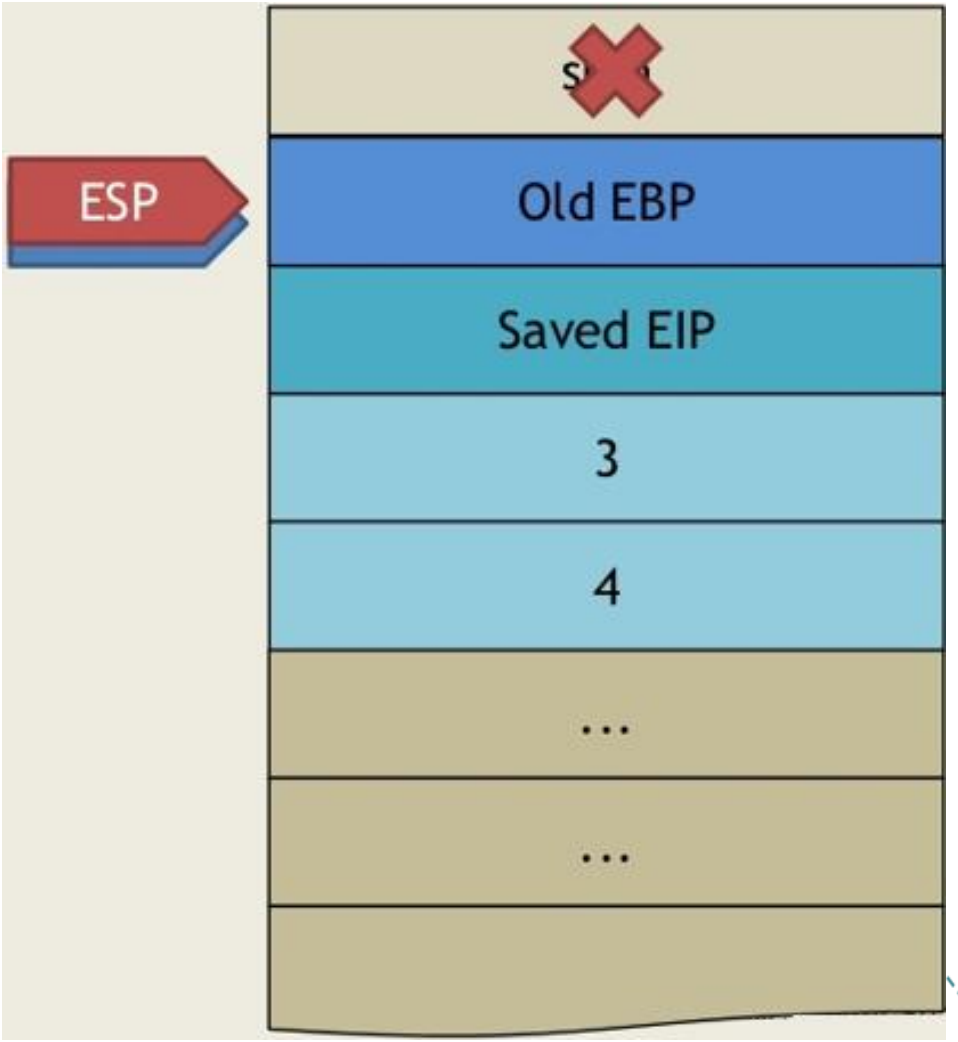
The frame pointer

- EBP- con trỏ cơ sở hay còn gọi là frame pointer
- Các biến cục bộ và tham số đầu vào có quan hệ với EBP



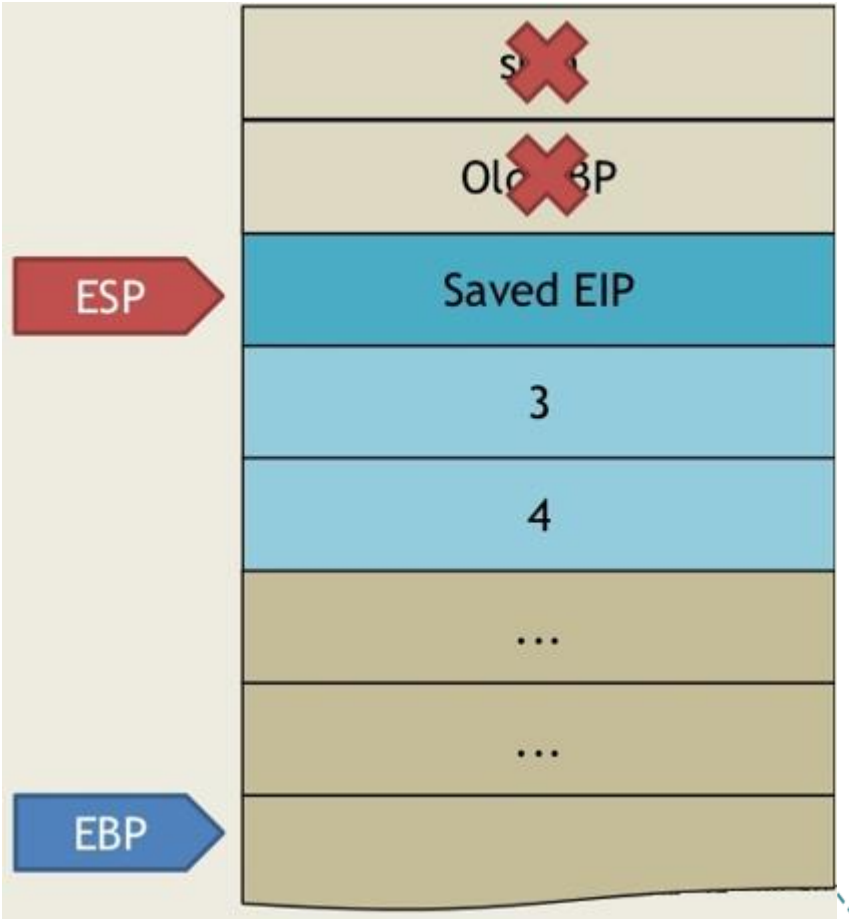
Epilogue

- Epilogue dọn dẹp khung ngăn xếp. Các biến cục bộ được xóa một cách hiệu quả.



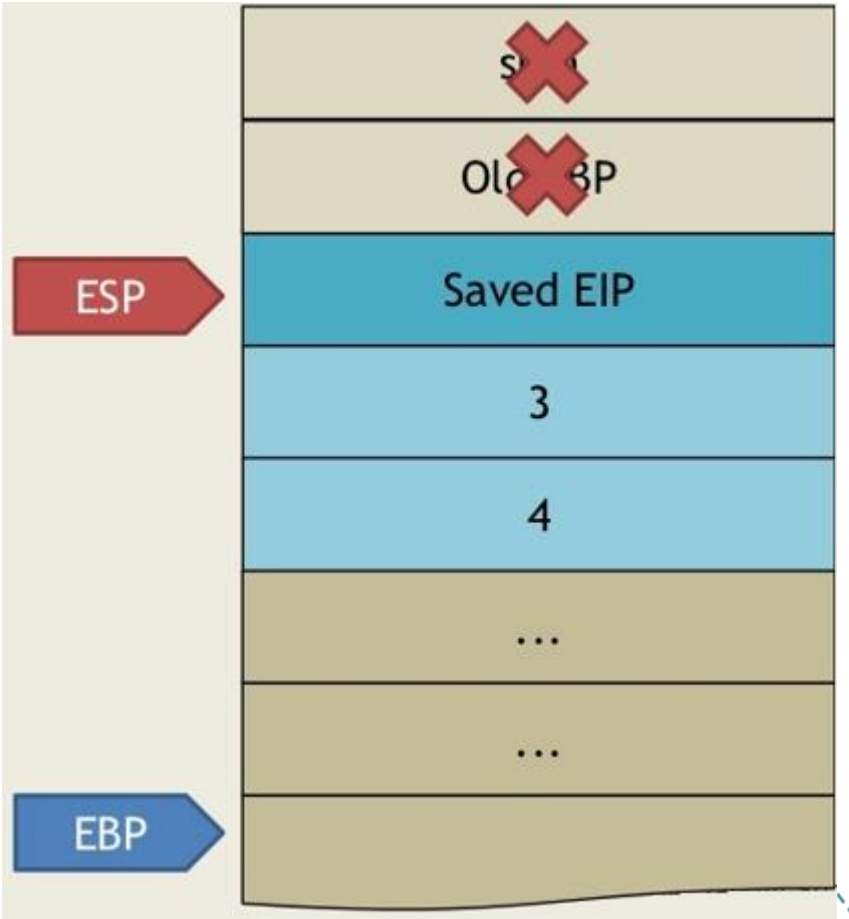
Epilogue

- Epilogue dọn dẹp khung ngăn xếp. Các biến cục bộ được xóa một cách hiệu quả.
- POP EBO. Khôi phục EBP quay lại khung cũ
- Stack pointer bây giờ chỉ đến nơi EIP được lưu trước khi thực hiện CALL add()



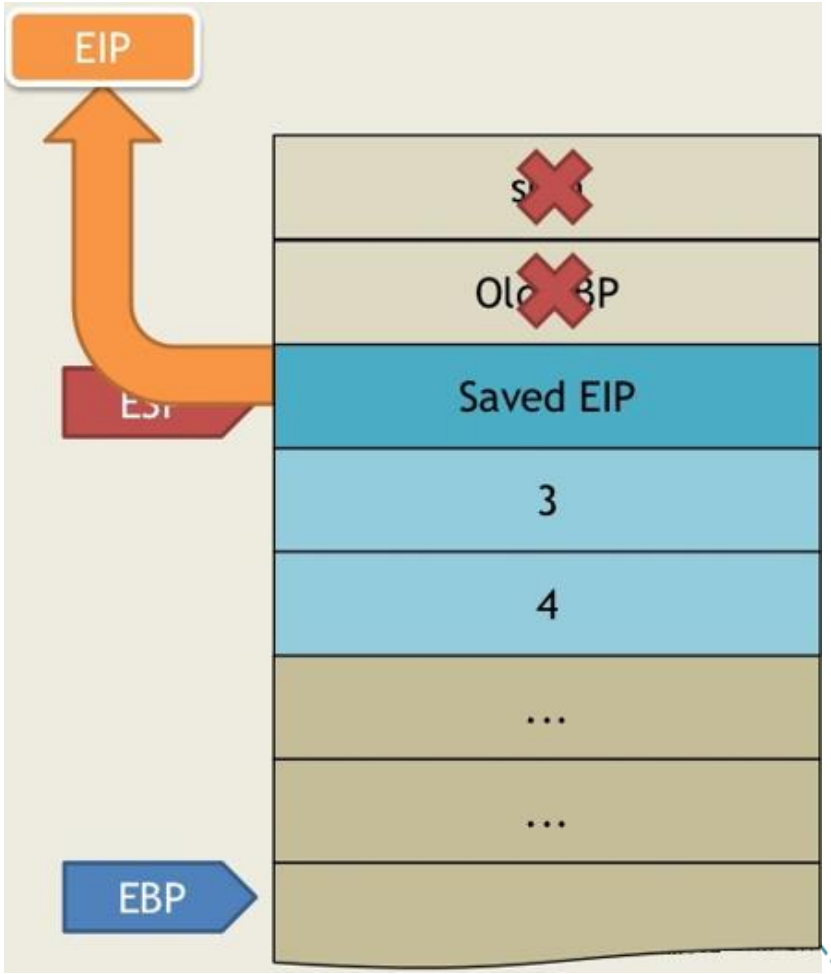
Return!

- Lệnh RET bật giá trị EIP đã lưu trở lại thanh ghi EIP.



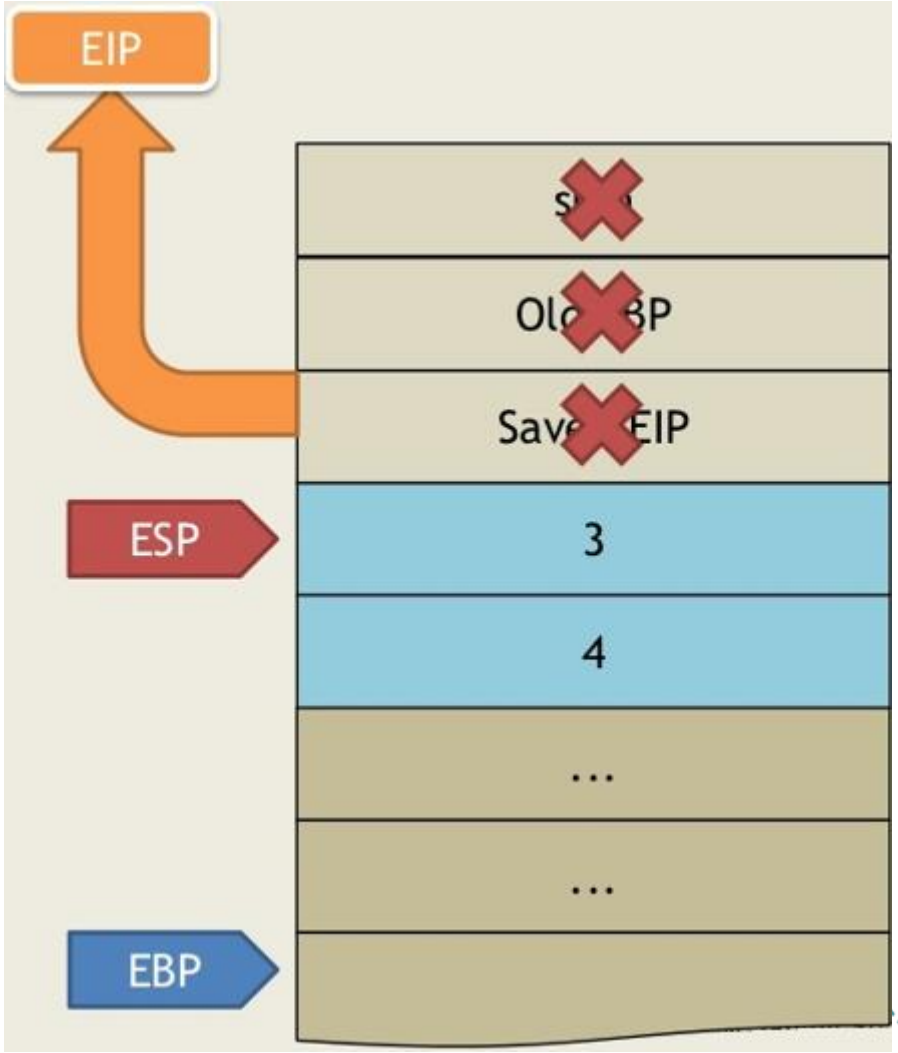
Return!

- Lệnh RET bật giá trị EIP đã lưu trở lại thanh ghi EIP.
- Chương trình trả về câu lệnh tiếp theo sau add()
- ESP dịch chuyển xuống 1 từ (word)



Return!

- Chương trình trả về câu lệnh tiếp theo sau add()



Tổng kết

- Các hàm sử dụng ngăn xếp
- Các tham số đầu vào được đẩy vào trong ngăn xếp
- CALL func() = lưu EIP trên ngăn xếp; nhảy tới thực hiện hàm
- Prologue cài đặt cho khung, đặt Frame Pointer (EBP)
- Các biến cục bộ được tạo trong ngăn xếp
- **FRAME = Params + Saved EIP + Local Vars**
- Epilogue dọn dẹp khung và khôi phục lại Frame Pointer
- RET = POP EIP. Điều khiển trả về lệnh tiếp theo sau CALL