درس ۸

ASM

قرارداد صدا کردن

```hello.asm
; args: rdi rsi rdx rcx r8 r9
; return: rax
call add3
```

یک کورس رایگان از MyWebUniversity

زبان Assembly

درس ۸

تابع‌ها و قرارداد صدا کردن



System V ABI: register های آرگومان، مقدار برگشتی، register های ذخیره‌شده توسط caller و callee، ترا

چرا یک قرارداد صدا کردن؟

یک قرارداد صدا کردن یک توافق است: آرگومان‌ها کجا می‌روند، نتیجه کجا برمی‌گردد، و یک تابع ...

روی Linux و macOS، x86-64 از System V ABI استفاده می‌کند. پیروی از آن اجازه می‌دهد Assembly ...



01

ذخیره توسط caller و callee

یک تابع می‌تواند آزادانه rax، rcx، rdx، rsi، rdi و r8 تا r11 را تغییر دهد. اگر caller بعداً ...

rbx، rbp و r12 تا r15 باید حفظ شوند: اگر یک تابع از آن‌ها استفاده کند، اول آن‌ها را push و ...



02

نوشتن یک تابع

این تابع سه عدد را جمع می‌کند. آرگومان‌ها در rdi، rsi و rdx می‌رسند، و نتیجه در rax می‌رود.

فقط از register های scratch استفاده می‌کند، پس اصلاً به prologue ضرورت ندارد.

ASSEMBLY LANGUAGE · 8

add3.asm

```
; long add3(long a, long b, long c)
global add3
add3:
    lea rax, [rdi + rsi]
    add rax, rdx
    ret

; caller:
    mov rdi, 10
    mov rsi, 20
    mov rdx, 12
    call add3          ; rax = 42
```

03

تراز stack

پیش از یک call، rsp باید مضرب شانزده باشد.
call هشت بایت push می‌کند، پس داخل یک تابع ...

شکستن این قاعده باعث می‌شود تابع‌های کتابخانه مثل printf به روش‌های مرموز crash کنند.

ASSEMBLY LANGUAGE · 8

align.asm

```
my_func:                ; on entry rsp ends in ...8
    push rbx            ; now rsp ends in ...0  ✓ aligned
    ; call other functions safely here
    pop rbx
    ret

other:
    sub rsp, 8          ; align before calling
    call printf
    add rsp, 8
    ret
```

04

بازگشت

تابع‌های بازگشتی طبیعی کار می‌کنند چون هر صدا کردن stack frame خودش را می‌گیرد. آنچه را در ...

اینجا فاکتوریل است: اگر n حداکثر یک باشد یک برگردان، وگرنه n ضرب فاکتوریل n منهای یک.

ASSEMBLY LANGUAGE · 8

fact.asm

```
; long fact(long n)
fact:
    cmp rdi, 1
    jle .base
    push rbx
    mov rbx, rdi          ; keep n
    dec rdi
    call fact             ; rax = fact(n-1)
    imul rax, rbx         ; n * fact(n-1)
    pop rbx
    ret
.base:
    mov eax, 1
    ret
```

05

نقشهٔ درس


1
چرا یک قرارداد صدا کردن؟
2
ذخیره توسط caller و callee
3
نوشتن یک تابع
4
تراز stack
5
بازگشت

منابع و مطالعهٔ بیشتر

ASSEMBLY LANGUAGE · 8

آرگومان	register
1st · 2nd · 3rd	rdi · rsi · rdx
4th · 5th · 6th	rcx · r8 · r9
7th and more	روی stack
مقدار برگشتی	rax
آرگومان float	xmm0 … xmm7

ASSEMBLY LANGUAGE · 8

add3.asm

```
; long add3(long a, long b, long c)
global add3
add3:
    lea rax, [rdi + rsi]
    add rax, rdx
    ret

; caller:
    mov rdi, 10
    mov rsi, 20
    mov rdx, 12
    call add3          ; rax = 42
```

تشکر از تماشای شما

MyWebUniversity.com

آموزش رایگان تکنالوژی از سال ۲۰۰۰ — انگلیسی و دری

@MyWebUniversity-Dari · @MyWebUniversity