EDDI
Electronic Design
Development Institute

에디로봇아카데미
임베디드 마스터 Lv1 과정

제 #기

2022. 01. 21

정성훈

CONTENTS

EDDI
Electronic Design
Development Institute

A. 기계어 분석

2. push rbp

현재 스택의 최상위(rsp)에 rbp의 값 대입

rsp 주소	rsp 메모리 값	rbp 주소	rpb 메모리 값
0x7fffffffe008	0xf7ded0b3		0x0
0x7fffffffe000	0x00000000		

```
(gdb) x $rbp
0x0:    Cannot access memory at address 0x0
```

```
(gdb) x $rsp
0x7fffffffe008: 0xf7ded0b3
```

```
(gdb) x $rsp
0x7fffffffe000: 0x00000000
```

A. 기계어 분석

3. mov rsp rbp

rsp값을 rbp에 대입

rsp 주소	rsp 메모리 값	rbp 주소	rpb 메모리 값
0x7fffffffe008	0xf7ded0b3		0x0
0x7fffffffe000	0x00000000		
		0x7fffffffe000	0x00000000

```
(gdb) x $rbp
0x0:    Cannot access memory at address 0x0
```

```
(gdb) x $rbp
0x7fffffffe000: 0x00000000
```

A. 기계어 분석

4. sub $0x10 %rsp

rsp 값에서 0x10(16byte) 값을 뺌

스택공간을 확보하기 위함

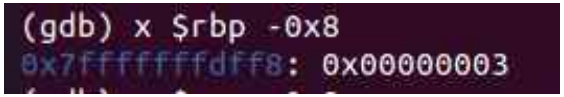
rsp 주소	rsp 메모리 값	rbp 주소	rpb 메모리 값
0x7fffffffe008	0xf7ded0b3		0x0
0x7fffffffe000	0x00000000		
		0x7fffffffe000	0x00000000
0x7fffffffdff0	0xffffe0f0		

A. 기계어 분석

5. movl $0x3, -0x8(%rbp)

0x3(4byte)을 rbp -0x8 에 위치한 메모리에 대입

int형 변수 3을 대입



A. 기계어 분석

6. mov -0x8(%rbp),%eax

cpu 어딘가 위치한 eax에 rbp -0x8의 값을 대입

```
(gdb) x $eax
0x5555515b:     Cannot access memory at address 0x5555515b
```

```
(gdb) x $eax
0x3:    Cannot access memory at address 0x3
```

A. 기계어 분석

7. mov %eax,%edi

edi에 eax값을 대입

```
(gdb) x $edi
0x1:    Cannot access memory at address 0x1
```

```
(gdb) x $edi
0x3:    Cannot access memory at address 0x3
```

A. 기계어 분석

8. callq 0x555555555149 <test_func>

복귀주소(다음에 실행해야하는 주소 5178)가 스택에 저장되는지 확인

rsp 주소	rsp 메모리 값	rbp 주소	rpb 메모리 값
0x7fffffffe008	0xf7ded0b3		0x0
0x7fffffffe000	0x00000000		
		0x7fffffffe000	0x00000000
0x7fffffffdff0	0xffffe0f0		
0x7fffffffdfe8	0x55555178		

```
(gdb) x $rsp
0x7fffffffdff0: 0xffffe0f0
```

```
(gdb) x $rsp
0x7fffffffdfe8: 0x55555178
```

A. 기계어 분석

9. push %rbp

rsp 주소	rsp 메모리 값	rbp 주소	rpb 메모리 값
0x7fffffffe008	0xf7ded0b3		0x0
0x7fffffffe000	0x00000000		
		0x7fffffffe000	0x00000000
0x7fffffffdff0	0xffffe0f0		
0x7fffffffdfe8	0x55555178		
0x7fffffffdfe0	0xffffe000		

```
(gdb) x $rsp
0x7fffffffdfe8: 0x55555178
```

```
(gdb) x $rsp
0x7fffffffdfe0: 0xffffe000
```

A. 기계어 분석

10. mov %rsp,%rbp

rsp 주소	rsp 메모리 값	rbp 주소	rpb 메모리 값
0x7fffffffe008	0xf7ded0b3		0x0
0x7fffffffe000	0x00000000		
		0x7fffffffe000	0x00000000
0x7fffffffdff0	0xffffe0f0		
0x7fffffffdfe8	0x55555178		
0x7fffffffdfe0	0xffffe000		
		0x7fffffffdfe0	0xffffe000

```
(gdb) x $rbp
0x7fffffffe000: 0x00000000
```

```
(gdb) x $rbp
0x7fffffffdfe0: 0xffffe000
```

A. 기계어 분석

11. mov %edi,-0x4(%rbp)

```
(gdb) x $rbp -0x4
0x7fffffffdfdc: 0x00005555
```

```
(gdb) x $rbp -0x4
0x7fffffffdfdc: 0x00000003
```

A. 기계어 분석

13. add %eax, %eax

eax에 리턴값인 6이 저장됨

```
(gdb) x $eax
0x3:    Cannot access memory at address 0x3
```

```
(gdb) x $eax
0x6:    Cannot access memory at address 0x6
```

A. 기계어 분석

14. pop rbp

스택의 최상위 에서 값을 빼서 rbp에 대입(rbp 복원)

rsp 주소	rsp 메모리 값	rbp 주소	rpb 메모리 값
0x7fffffffe008	0xf7ded0b3		0x0
0x7fffffffe000	0x00000000		
		0x7fffffffe000	0x00000000
0x7fffffffdff0	0xffffe0f0		
0x7fffffffdfe8	0x55555178		
0x7fffffffdfe0	0xffffe000		
		0x7fffffffdfe0	0xffffe000
0x7fffffffdfe8	0x55555178	0x7fffffffe000	0x00000000

A. 기계어 분석

15. retq

pop rip의 동의어

현재 스택의 최상위에 저장된 값을 rip 레지스터에 대입

복귀주소로 복귀

rsp 주소	rsp 메모리 값	rbp 주소	rpb 메모리 값
0x7fffffffe008	0xf7ded0b3		0x0
0x7fffffffe000	0x00000000		
		0x7fffffffe000	0x00000000
0x7fffffffdff0	0xffffe0f0		
0x7fffffffdfe8	0x55555178		
0x7fffffffdfe0	0xffffe000		
		0x7fffffffdfe0	0xffffe000
0x7fffffffdfe8	0x55555178	0x7fffffffe000	0x00000000
0x7fffffffdff0	0xffffe0f0