리버스 엔지니어링 바이블

저자: 강병탁

김영철 2015. 10. 13.

Reverse Engineering

- 소스코드를 역추적하는 것
- 빌드된 파일의 원래의 소스코드를 파악하는 것

Register

esp, ebp, edx, eax....

```
#include <stdio.h>
int main()
{
    int a=0, b=1, c=2, d=3, e;
    a = b + c;
    e = c + d;
    a = a + e;

return 0;
}
```



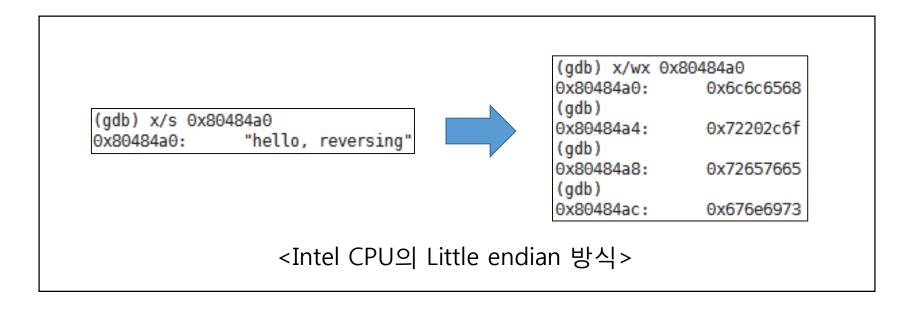
```
Dump of assembler code for function main:
   0x080483ed <+0>:
                        push
                               %ehn
   0x080483ee <+1>:
                        mov
                                %esp,%ebp
   0x080483f0 <+3>:
                        sub
                                $0x20,%esp
   0x080483f3 <+6>:
                        movl
                                $0x0,-0x14(%ebp)
   0x080483fa <+13>:
                        movl
                                $0x1,-0x10(%ebp)
                                $0x2,-0xc(%ebp)
                        movl
   0x08048401 <+20>:
   0x08048408 <+27>:
                        movl
                                $0x3,-0x8(%ebp)
   0x0804840f <+34>:
                                -0xc(%ebp),%eax
                        mov
                                -0x10(%ehn), %edx
   0x08048412 <+37>:
                        mov
   0x08048415 <+40>:
                         add
                                %edx.%eax
   0x08048417 <+42>:
                        mov
                                %eax,-⊍x14(%ebp)
   0x0804841a <+45>:
                                -0x8(%ebp),%eax
                        mov
                                -0xc(%ebp),%edx
   0x0804841d <+48>:
                        mov
                         add
                                %edx.%eax
   0x08048420 <+51>:
   0x08048422 <+53>:
                                %eax, -0x4(%ebp)
                        mov
   0x08048425 <+56>:
                                -0x4(%ebp),%eax
                        mov
   0x08048428 <+59>:
                         add
                                %eax, -0x14(%ebp)
   0x0804842b <+62>:
                                $0x0,%eax
                        mov
                        leave
   0x08048430 <+67>:
   0x08048431 <+68>:
                         ret
End of assembler dump.
```

Register

레지스터 이름	역할	비고
EAX	산술 연산에 사용, 함수의 리턴값 저장	Accumulator
EDX	산술 연산에 사용	Data
ECX	반복문의 반복 횟수 저장	Counter
EBX	여분의 레지스터	-
ESI	문자열 연산에서 사용되는 Source Index	Source Index
EDI	문자열 연산에서 사용되는 Destination Index	Destination Index
EBP	현재 스택의 바닥을 가리키는 포인터	Base Pointer
ESP	현새 스택의 꼭대기를 가리키는 포인터	Stack Pointer

Endian

• Endian - 바이트 저장 순서, big endian & little endian Big endian - 흔히 사용하는 순서(왼쪽부터 시작)
Little endian - Big endian의 반대 방향



Assembly Language

- push 스택에 값을 넣는 명령
- mov 값을 넣는 명령
- sub 뺄셈 연산
- add 덧셈 연산
- leave ebp 복원
- ret 리턴주소로 점프
- Etc.

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                         push
                                %ebp
   0x080483ee <+1>:
                        mov
                                %esp,%ebp
   0x080483f0 <+3>:
                         sub
                                $0x20,%esp
   0x080483f3 <+6>:
                        movl
                                $0x0,-0x14(%ebp)
   0x080483fa <+13>:
                        movl
                                $0x1,-0x10(%ebp)
   0x08048401 <+20>:
                        movl
                                $0x2,-0xc(%ebp)
   0x08048408 <+27>:
                                $0x3,-0x8(%ebp)
                         movl
   0x0804840f <+34>:
                                -0xc(%ebp),%eax
                         mov
   0x08048412 <+37>:
                                -0x10(%ebp),%edx
                         mov
                                %edx,%eax
   0x08048415 <+40>:
                         add
   0x08048417 <+42>:
                                %eax, -0x14(%ebp)
                        mov
   0x0804841a <+45>:
                                -0x8(%ebp),%eax
                         mov
                                -0xc(%ebp),%edx
   0x0804841d <+48>:
                         mov
                                %edx.%eax
   0x08048420 <+51>:
                         add
   0x08048422 <+53>:
                                %eax, -0x4(%ebp)
                         mov
   0x08048425 <+56>:
                                -0x4(%ebp),%eax
                         mov
   0x08048428 <+59>:
                         add
                                %eax, -0x14(%ebp)
   0x0804842b <+62>:
                                $0x0.%eax
                        mov
                         leave
   0x08048430 <+67>:
   0x08048431 <+68>:
                         ret
End of assembler dump.
```

- 함수 호출 시 파라미터가 들어가는 방향
- 리턴 주소
- •지역 변수 사용

```
#include <stdio.h>
int add(int a, int b);
int main()
{
    int a = 1, b = 2;
    add(a, b);
    return 0;
}
int add(int a, int b)
{
    return a+b;
}
```



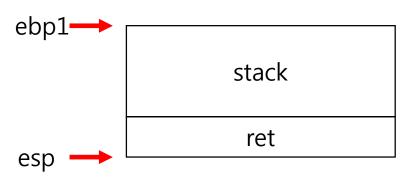
```
Dump of assembler code for function main:
   0x080483ed <+0>:
                               %ebp
                        push
   0x080483ee <+1>:
                               %esp,%ebp
                        mov
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
                                                   Dump of assembler code for function add:
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
                                                     0x08048421 <+0>:
                                                                                 %ebp
                                                                          push
   0x080483f6 <+9>:
                        movl
                               $0x1,0x18(%esp)
                                                      0x08048422 <+1>:
                                                                                 %esp,%ebp
                                                                          mov
   0x080483fe <+17>:
                        movl
                               $0x2,0x1c(%esp)
                                                      0x08048424 <+3>:
                                                                                 0xc(%ebp),%eax
                                                                          mov
   0x08048406 <+25>:
                               0xlc(%esp),%eax
                        mov
                                                      0x08048427 <+6>:
                                                                                 0x8(%ebp),%edx
                                                                          mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
                                                                          add
                                                                                 %edx,%eax
                                                      0x0804842a <+9>:
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
                                                     0x0804842c <+11>:
                                                                                 %ebp
                                                                          pop
   0x08048412 <+37>:
                        mov
                               %eax,(%esp)
                                                     0x0804842d <+12>:
                                                                          ret
                        call
                               0x8048421 <add>
   0x08048415 <+40>:
                                                   End of assembler dump.
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
               <main함수>
                                                                 <add함수>
```

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                        push
                               %ebp
   0x080483ee <+1>:
                               %esp,%ebp
                        mov
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                               $0x1,0x18(%esp)
                        movl
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
   0x08048406 <+25>:
                               0xlc(%esp),%eax
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```



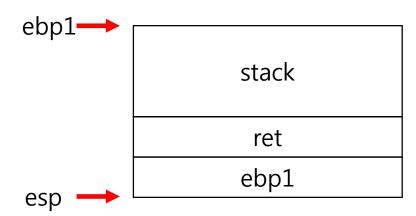
• call main

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                        push
                               %ebp
   0x080483ee <+1>:
                        mov
                               %esp,%ebp
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                               $0x1,0x18(%esp)
                        movl
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
                               0x1c(%esp),%eax
   0x08048406 <+25>:
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```



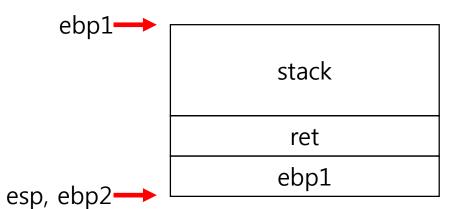
push %ebp

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                        push
                               %ebp
   0x080483ee <+1>:
                               %esp,%ebp
                        mov
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                        movl
                               $0x1,0x18(%esp)
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
                               0x1c(%esp),%eax
   0x08048406 <+25>:
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```



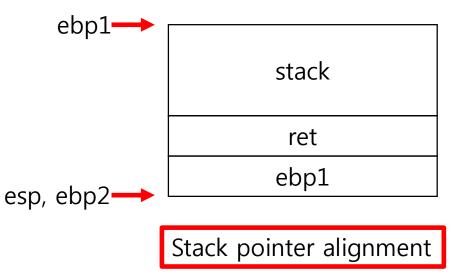
• mov %esp, %ebp

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                               %ebp
                        push
                               %esp,%ebp
   0x080483ee <+1>:
                        mov
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                        movl
                               $0x1,0x18(%esp)
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
   0x08048406 <+25>:
                               0x1c(%esp),%eax
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```



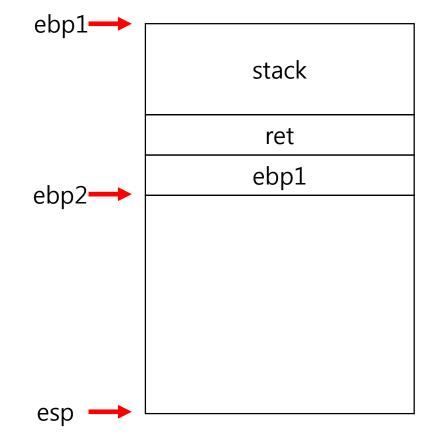
• and \$0xfffffff0, %esp

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                               %ebp
                        push
   0x080483ee <+1>:
                               %esp,%ebp
                        mov
   0x080483f0 <+3>:
                                $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                        movl
                               $0x1,0x18(%esp)
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
   0x08048406 <+25>:
                               0x1c(%esp),%eax
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                                $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```



• sub \$0x20, %esp

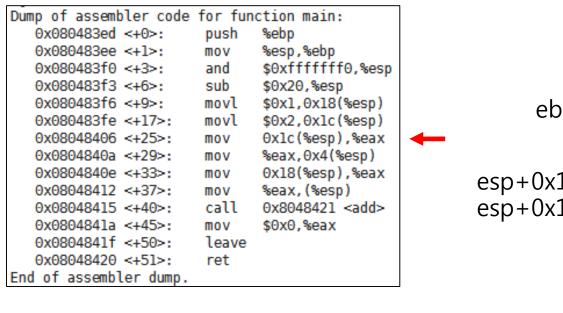
```
Dump of assembler code for function main:
   0x080483ed <+0>:
                               %ebp
                        push
   0x080483ee <+1>:
                               %esp,%ebp
                        mov
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                        movl
                               $0x1,0x18(%esp)
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
   0x08048406 <+25>:
                               0x1c(%esp),%eax
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```

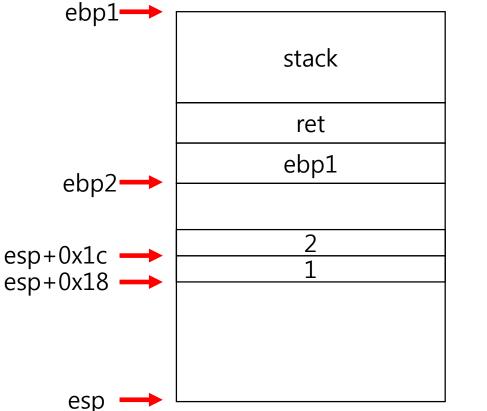


movl \$0x1, 0x18(%esp) ebp1 stack Dump of assembler code for function main: 0x080483ed <+0>: %ebp push ret 0x080483ee <+1>: mov %esp,%ebp 0x080483f0 <+3>: \$0xfffffff0,%esp and 0x080483f3 <+6>: sub \$0x20,%esp ebp1 0x080483f6 <+9>: movl \$0x1,0x18(%esp) ebp2→ 0x080483fe <+17>: \$0x2,0x1c(%esp) movl 0x08048406 <+25>: 0x1c(%esp),%eax mov 0x0804840a <+29>: %eax,0x4(%esp) mov 0x0804840e <+33>: 0x18(%esp),%eax mov 0x08048412 <+37>: %eax, (%esp) mov esp+0x180x08048415 <+40>: call 0x8048421 <add> 0x0804841a <+45>: \$0x0,%eax mov 0x0804841f <+50>: leave 0x08048420 <+51>: ret End of assembler dump. esp

movl \$0x1, 0x18(%esp) ebp1→ stack Dump of assembler code for function main: 0x080483ed <+0>: %ebp push ret 0x080483ee <+1>: mov %esp,%ebp 0x080483f0 <+3>: \$0xfffffff0,%esp and 0x080483f3 <+6>: sub \$0x20,%esp ebp1 0x080483f6 <+9>: movl \$0x1,0x18(%esp) ebp2→ 0x080483fe <+17>: \$0x2,0x1c(%esp) movl 0x1c(%esp),%eax 0x08048406 <+25>: mov 0x0804840a <+29>: %eax,0x4(%esp) mov 0x0804840e <+33>: 0x18(%esp),%eax mov esp+0x1c → 0x08048412 <+37>: %eax, (%esp) mov esp+0x18 → 0x08048415 <+40>: call 0x8048421 <add> 0x0804841a <+45>: \$0x0,%eax mov 0x0804841f <+50>: leave 0x08048420 <+51>: ret End of assembler dump. esp

mov 0x1c(%esp), %eax

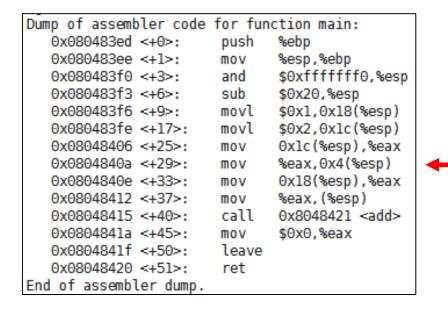


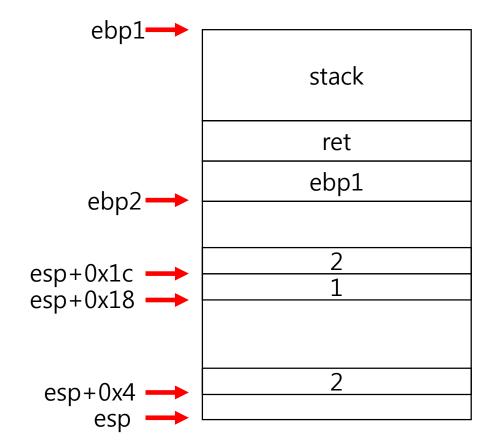


17

eax = 2

mov %eax, 0x4(%esp)

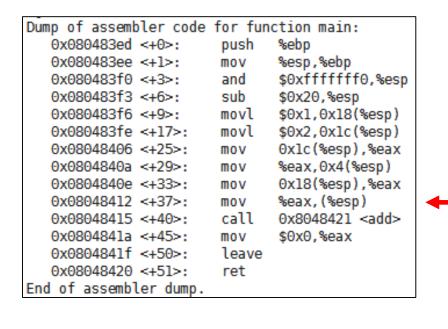


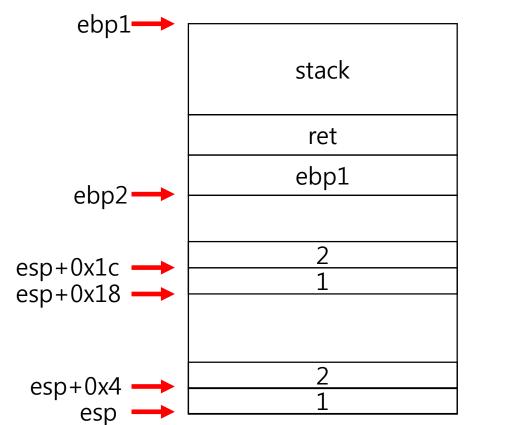


eax = 2

 mov 0x18(%esp), %eax ebp1 stack eax = 1Dump of assembler code for function main: 0x080483ed <+0>: %ebp push ret %esp,%ebp 0x080483ee <+1>: mov 0x080483f0 <+3>: \$0xfffffff0,%esp and 0x080483f3 <+6>: sub \$0x20,%esp ebp1 0x080483f6 <+9>: movl \$0x1,0x18(%esp) ebp2→ 0x080483fe <+17>: \$0x2,0x1c(%esp) movl 0x08048406 <+25>: 0xlc(%esp),%eax mov 0x0804840a <+29>: %eax,0x4(%esp) mov 0x0804840e <+33>: 0x18(%esp),%eax mov esp+0x1c → 0x08048412 <+37>: %eax, (%esp) mov esp+0x18 → 0x08048415 <+40>: call 0x8048421 <add> 0x0804841a <+45>: \$0x0,%eax mov 0x0804841f <+50>: leave 0x08048420 <+51>: ret End of assembler dump. esp+0x4esp

mov %eax, (%esp)

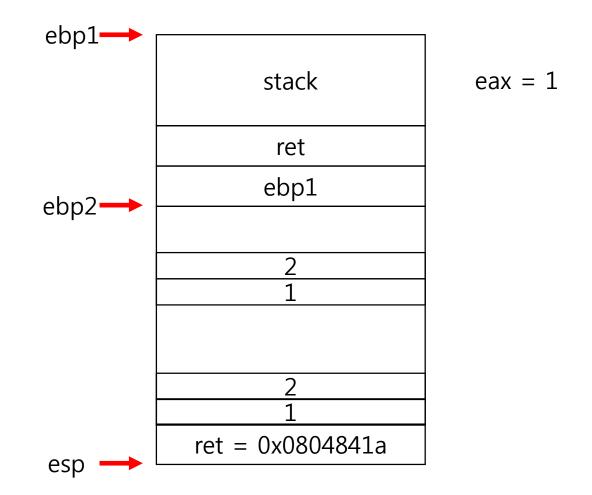




eax = 1

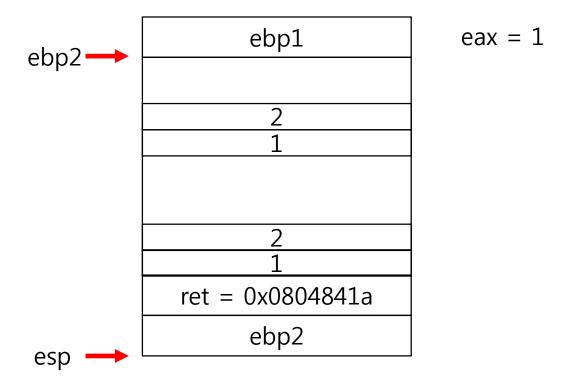
• call add

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                        push
                               %ebp
   0x080483ee <+1>:
                        mov
                               %esp,%ebp
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                        movl
                               $0x1,0x18(%esp)
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
                               0x1c(%esp),%eax
   0x08048406 <+25>:
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```

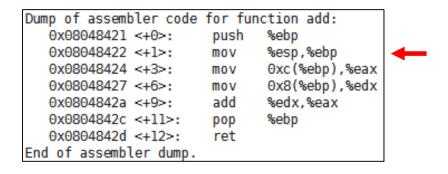


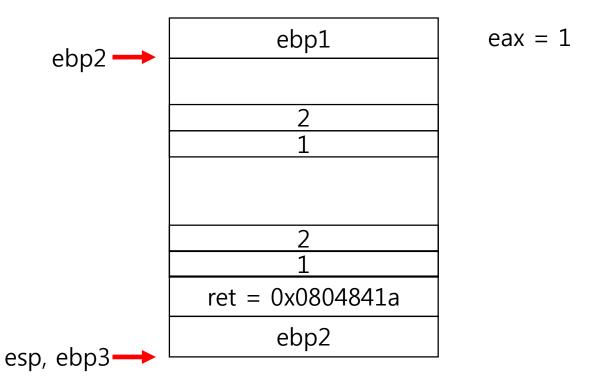
push %ebp

```
Dump of assembler code for function add:
   0x08048421 <+0>:
                        push
                               %ebp
   0x08048422 <+1>:
                               %esp,%ebp
                        mov
   0x08048424 <+3>:
                               0xc(%ebp),%eax
                        mov
                               0x8(%ebp),%edx
   0x08048427 <+6>:
                        mov
   0x0804842a <+9>:
                        add
                               %edx,%eax
   0x0804842c <+11>:
                               %ebp
                        pop
   0x0804842d <+12>:
                        ret
End of assembler dump.
```



mov %esp, %ebp

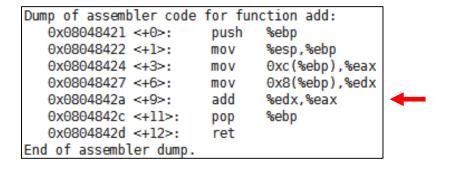


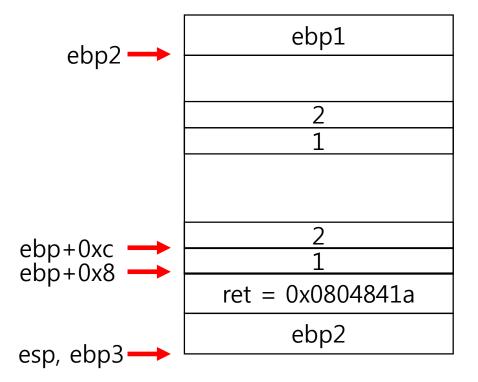


mov 0xc(%ebp), %eax ebp1 eax = 2ebp2 → Dump of assembler code for function add: 0x08048421 <+0>: push %ebp %esp,%ebp 0x08048422 <+1>: mov 0xc(%ebp),%eax 0x08048424 <+3>: mov 0x08048427 <+6>: 0x8(%ebp),%edx ebp+0xc mov 0x0804842a <+9>: add %edx,%eax 0x0804842c <+11>: %ebp pop ret = 0x0804841a0x0804842d <+12>: ret End of assembler dump. ebp2 esp, ebp3→

 mov 0x8(%ebp), %edx ebp1 eax = 2ebp2 → edx = 1Dump of assembler code for function add: 0x08048421 <+0>: push %ebp %esp,%ebp 0x08048422 <+1>: mov 0xc(%ebp),%eax 0x08048424 <+3>: mov ebp+0xc ebp+0x8 → 0x08048427 <+6>: 0x8(%ebp),%edx mov 0x0804842a <+9>: add %edx,%eax 0x0804842c <+11>: %ebp pop ret = 0x0804841a0x0804842d <+12>: ret End of assembler dump. ebp2 esp, ebp3 →

add %edx, %eax

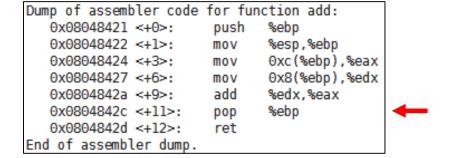


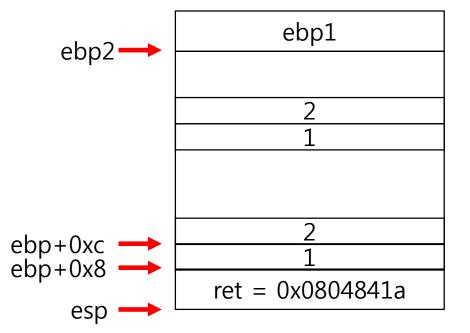


$$eax = 2 + 1 = 3$$

 $edx = 1$

pop %ebp

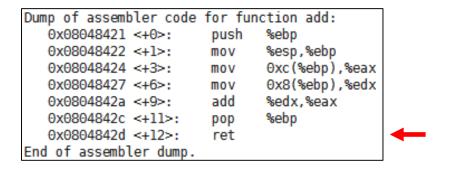


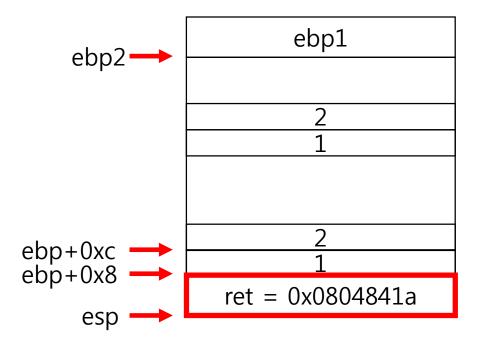


$$eax = 2 + 1 = 3$$

 $edx = 1$

• ret



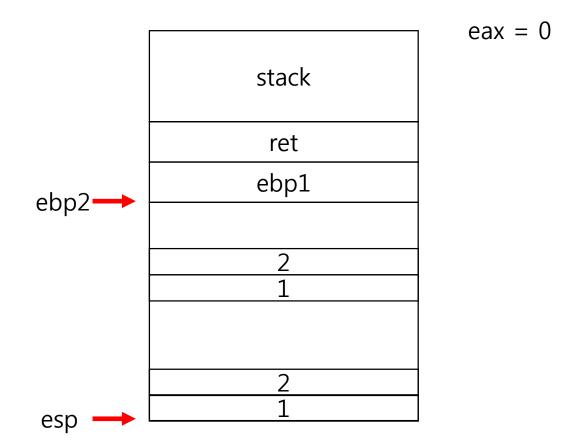


$$eax = 2 + 1 = 3$$

 $edx = 1$

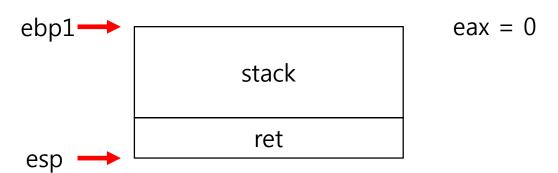
• mov \$0x0, %eax

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                               %ebp
                        push
   0x080483ee <+1>:
                        mov
                               %esp,%ebp
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                        movl
                               $0x1,0x18(%esp)
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
   0x08048406 <+25>:
                               0x1c(%esp),%eax
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```



leave

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                        push
                               %ebp
   0x080483ee <+1>:
                               %esp,%ebp
                        mov
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                               $0x1,0x18(%esp)
                        movl
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
                               0x1c(%esp),%eax
   0x08048406 <+25>:
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```



• ret

```
Dump of assembler code for function main:
   0x080483ed <+0>:
                        push
                               %ebp
   0x080483ee <+1>:
                               %esp,%ebp
                        mov
   0x080483f0 <+3>:
                               $0xfffffff0,%esp
                        and
   0x080483f3 <+6>:
                        sub
                               $0x20,%esp
   0x080483f6 <+9>:
                               $0x1,0x18(%esp)
                        movl
   0x080483fe <+17>:
                               $0x2,0x1c(%esp)
                        movl
                               0x1c(%esp),%eax
   0x08048406 <+25>:
                        mov
   0x0804840a <+29>:
                               %eax,0x4(%esp)
                        mov
   0x0804840e <+33>:
                               0x18(%esp),%eax
                        mov
   0x08048412 <+37>:
                               %eax, (%esp)
                        mov
   0x08048415 <+40>:
                        call
                               0x8048421 <add>
   0x0804841a <+45>:
                               $0x0,%eax
                        mov
   0x0804841f <+50>:
                        leave
   0x08048420 <+51>:
                        ret
End of assembler dump.
```

