新竹人最愛逆向(工程)

terrynini38514

terrynini


WHO AM I ?

ID : Terrynini38514

- 跟我有點熟：
  逆逆
- 跟我麻吉麻：
  國立交通大學SENSE LAB
  資電玄客與安全碩士學位學程陳廷宇
- 稍微能拿來吹的東西：
  2019 年金盾獎冠軍
  2019, 2020 FireEye Flare On Challenge 破台
- CTF Team：
  DoubleSigma（已慘遭 Balsn 併吞化作其血肉）
  Balsn

沒照片可用 救命

Stack Frame

```
080484cc <bossA>:
80484cc:       55                      push   ebp
80484cd:       89 e5                   mov    ebp,esp
80484cf:       83 ec 18                sub    esp,0x18
80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
...
80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
...
80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
...
80484f4:       83 ec 0c                sub    esp,0xc
80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
80484ff:       83 c4 10                add    esp,0x10
8048502:       90                      nop
8048503:       c9                      leave
8048504:       c3                      ret
```

eax: 0x20

Stack Frame

```
1  080484cc <bossA>:
2  80484cc:        55                      push   ebp
3  80484cd:        89 e5                   mov    ebp,esp
4  80484cf:        83 ec 18                sub    esp,0x18
5  80484d2:        8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:        89 45 f4                mov    DWOR
8  ...
9  80484ec:        e8 4f f
10 ...
                                           esp,0xc
                                push   DWORD PTR [ebp-0xc]
                   a7 ff ff ff  call   80484a6 <bossB>
                   83 c4 10     add    esp,0x10
15 8048502:        90                      nop
16 8048503:        c9                      leave
17 8048504:        c3                      ret
eax: 0x20
```

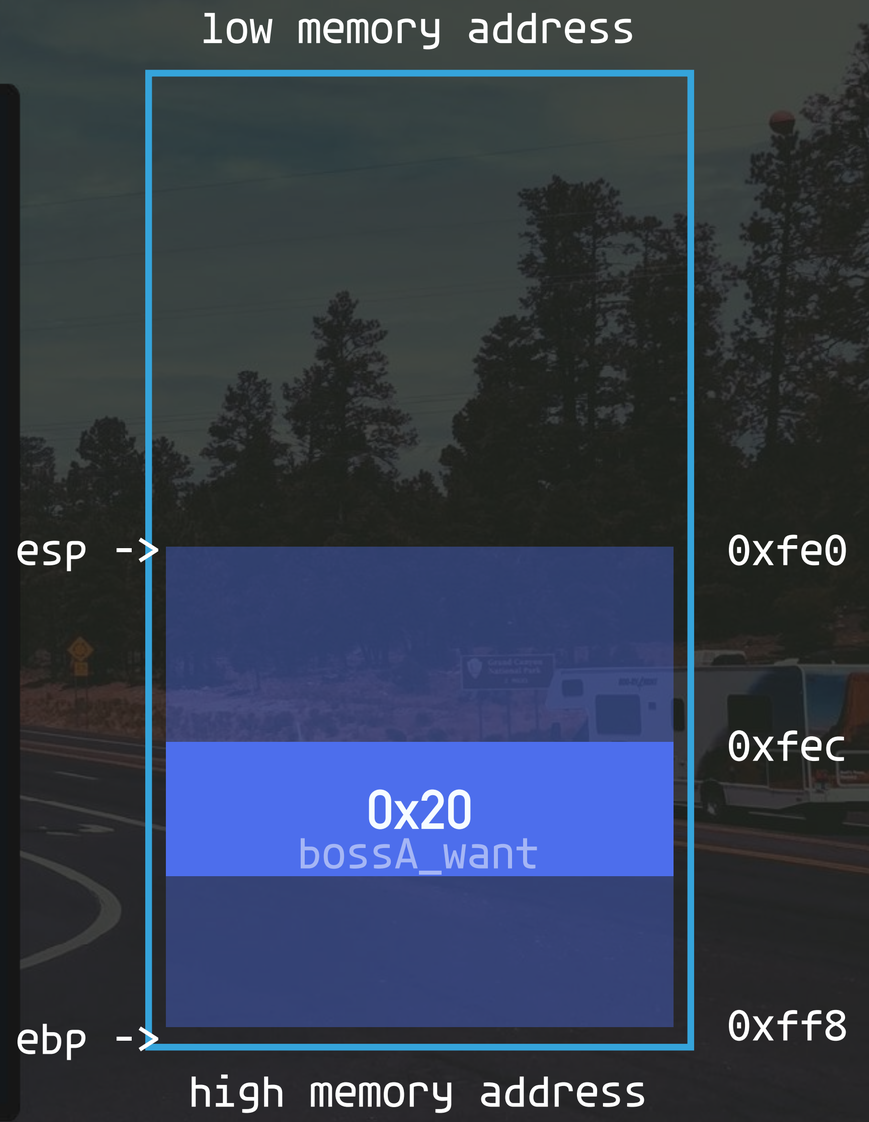
SKIP SOME OPERATIONS !

Stack Frame

```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```

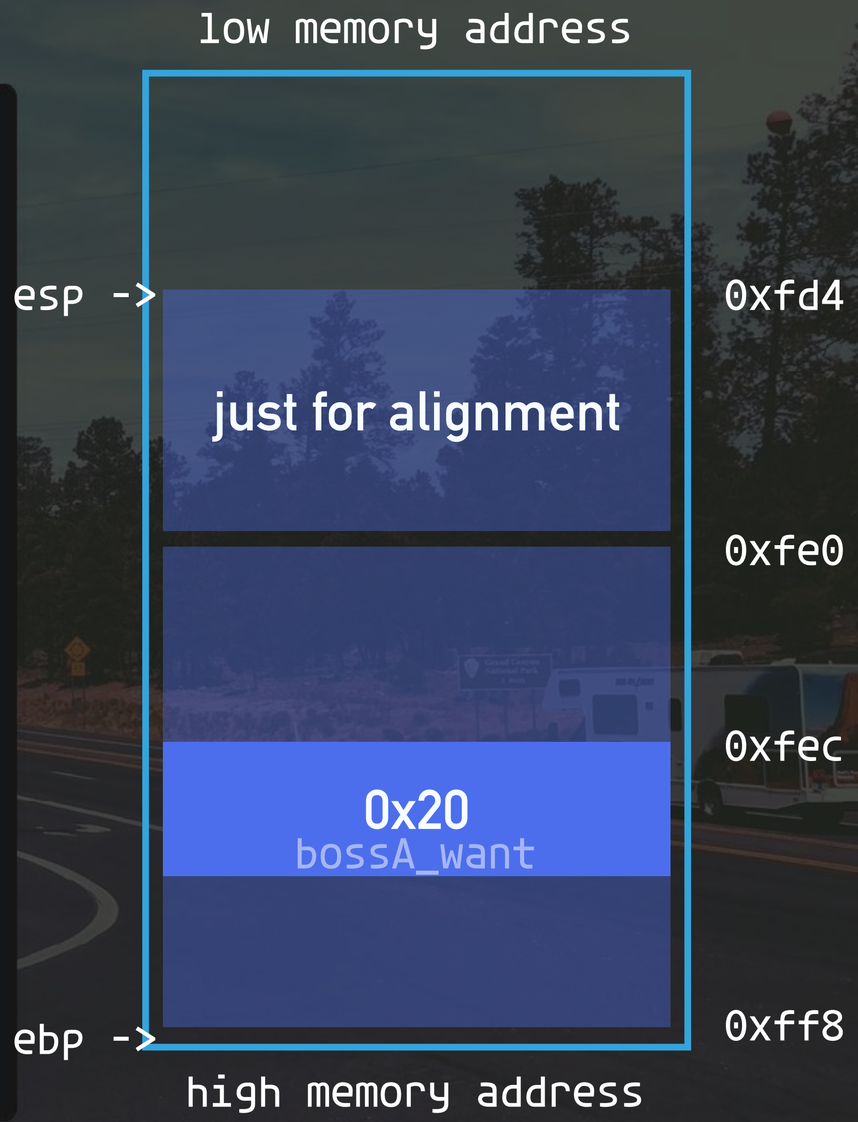
Stack Frame

```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```



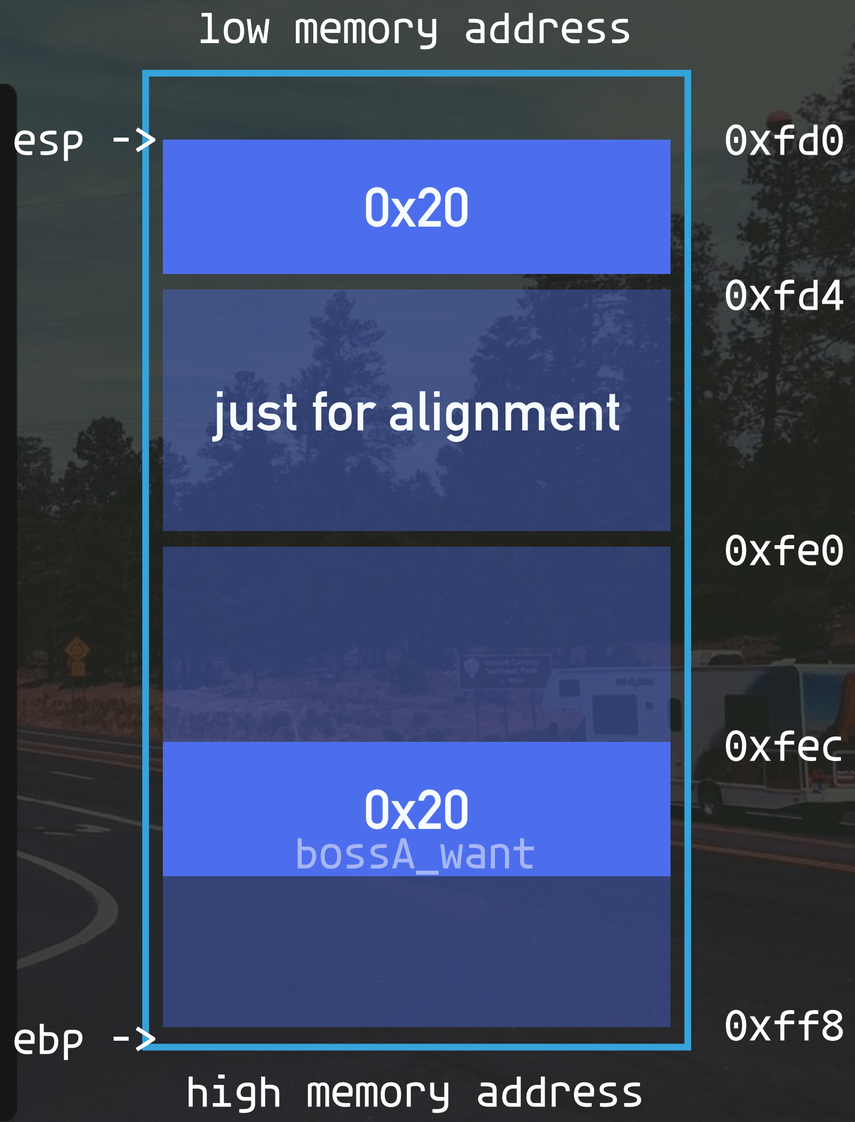
Stack Frame

```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```



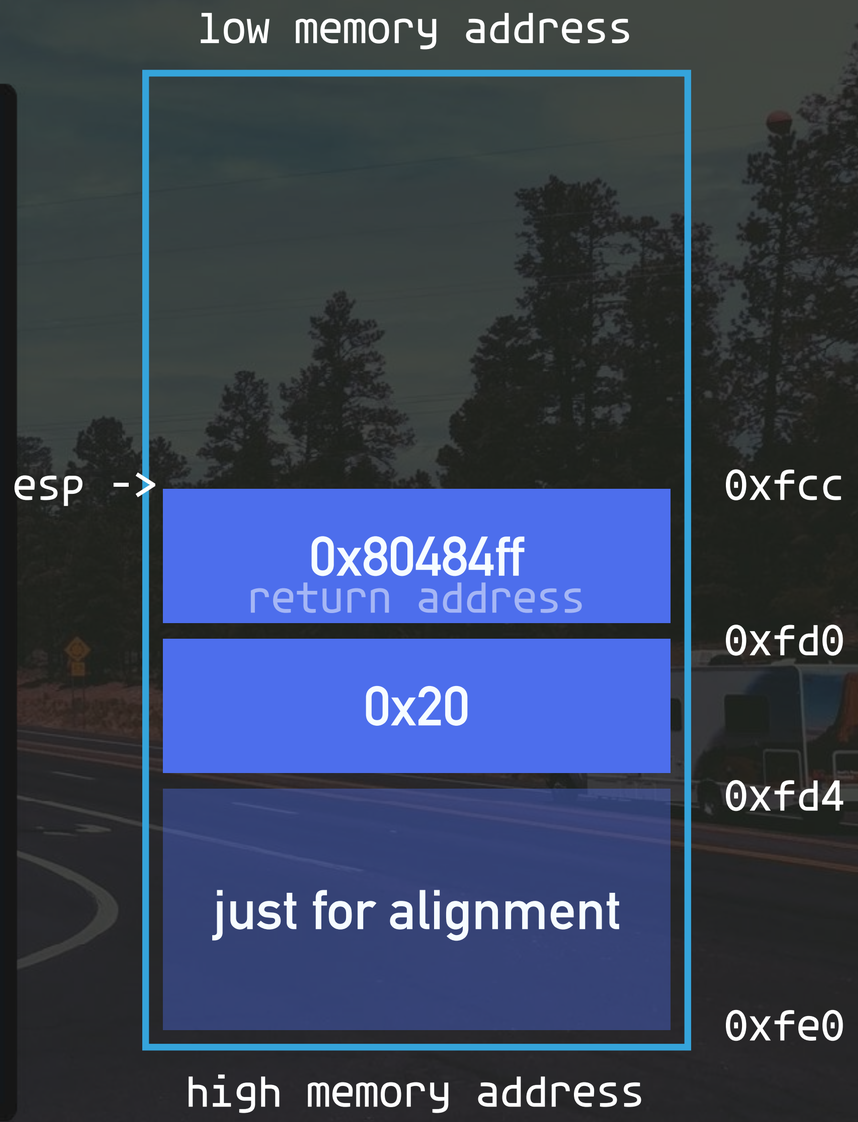
Stack Frame

```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```



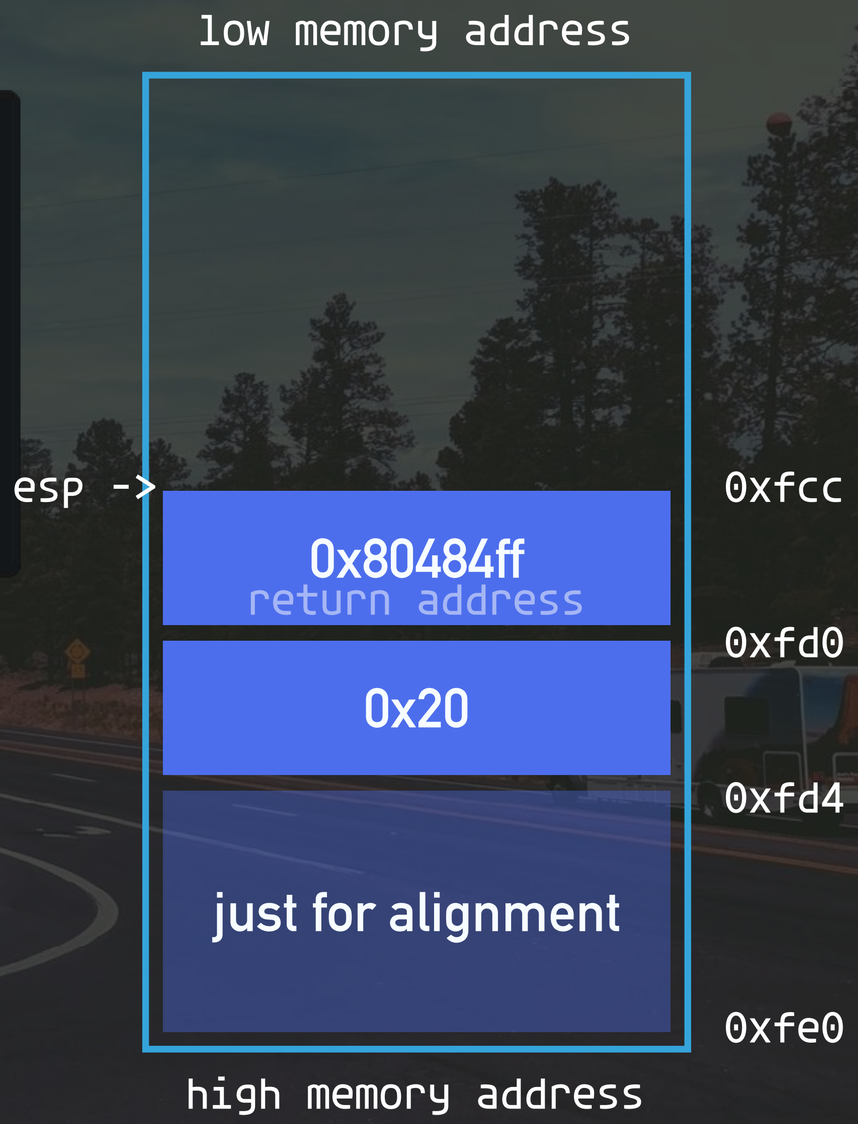
Stack Frame

```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```



Stack Frame

```
1  080484a6 <bossB>:
2  80484a6:       55                   push   ebp
3  80484a7:       89 e5                mov    ebp,esp
4  ...
5  80484ca:       c9                   leave
6  80484cb:       c3                   ret
```



Stack Frame

```
1  080484a6 <bossB>:
2  80484a6:       55                      push   ebp
3  80484a7:       89 e5                   mov    ebp,esp
4  ...
5  80484ca:       c9                      leave
6  80484cb:       c3                      ret
```

Stack Frame

```
1  080484a6 <bossB>:
2  80484a6:       55                   push   ebp
3  80484a7:       89 e5                mov    ebp,esp
4  ...
5  80484ca:       c9                   leave
6  80484cb:       c3                   ret
```

leave = mov esp, ebp
pop ebp

Stack Frame

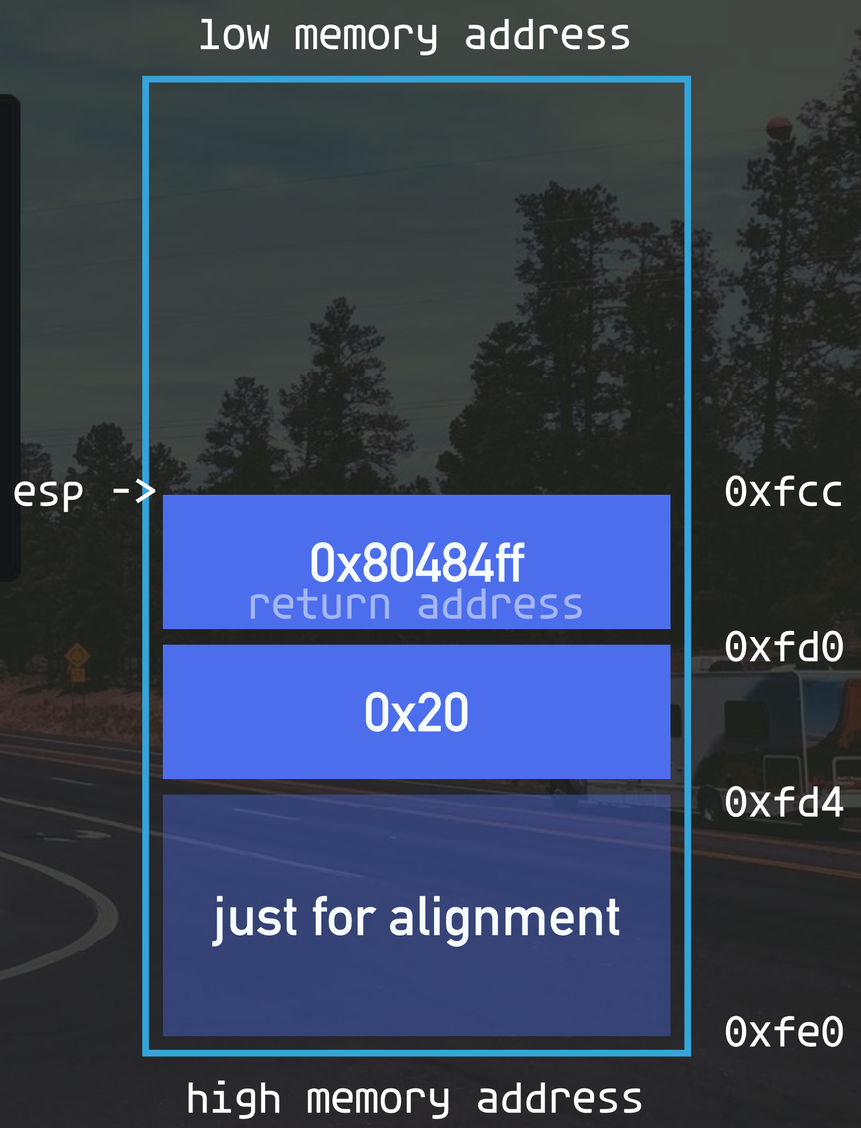
```
1  080484a6 <bossB>:
2  80484a6:       55                   push   ebp
3  80484a7:       89 e5                mov    ebp,esp
4  ...
5  80484ca:       c9                   leave
6  80484cb:       c3                   ret
```

leave = mov esp, ebp
pop ebp

Stack Frame

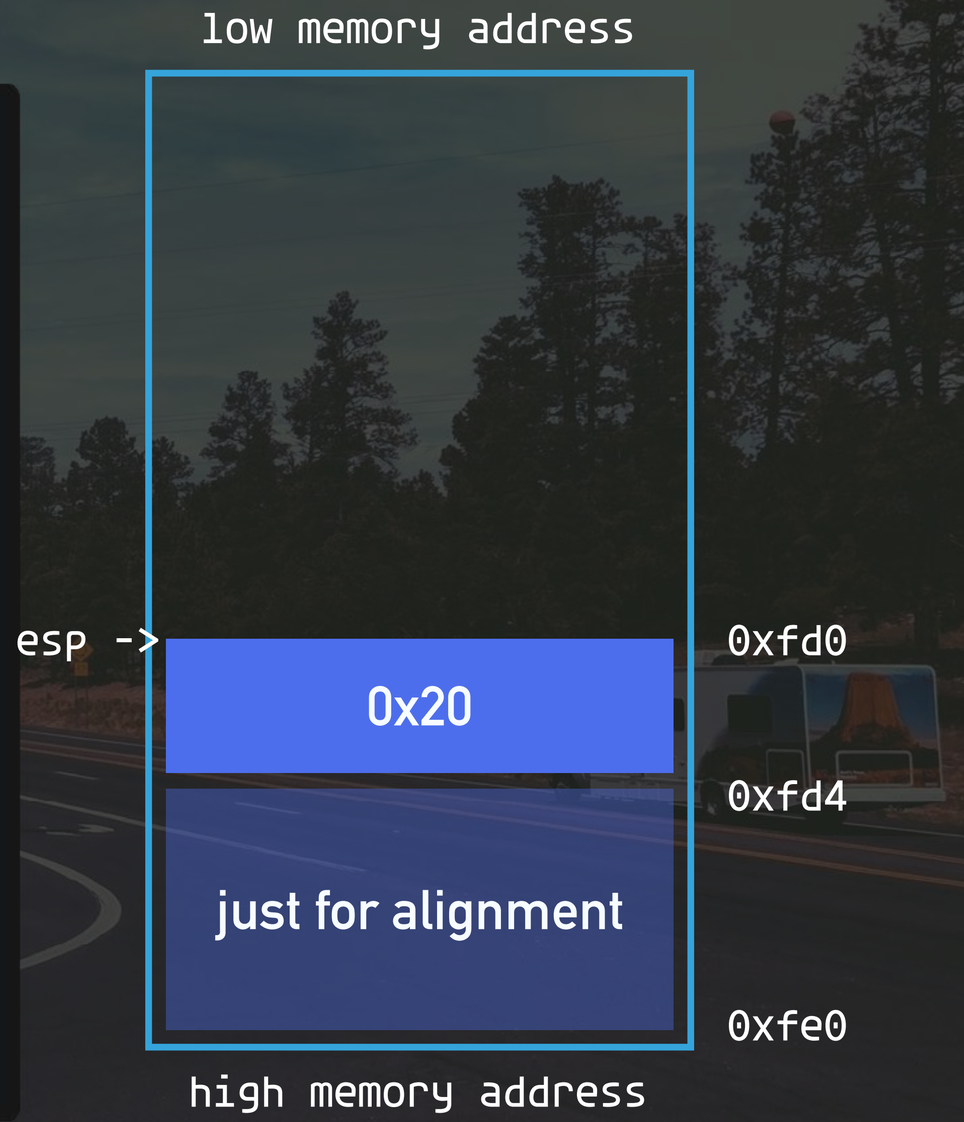
```
1  080484a6 <bossB>:
2  80484a6:       55                      push   ebp
3  80484a7:       89 e5                   mov    ebp,esp
4  ...
5  80484ca:       c9                      leave
6  80484cb:       c3                      ret
```

ret = pop eip



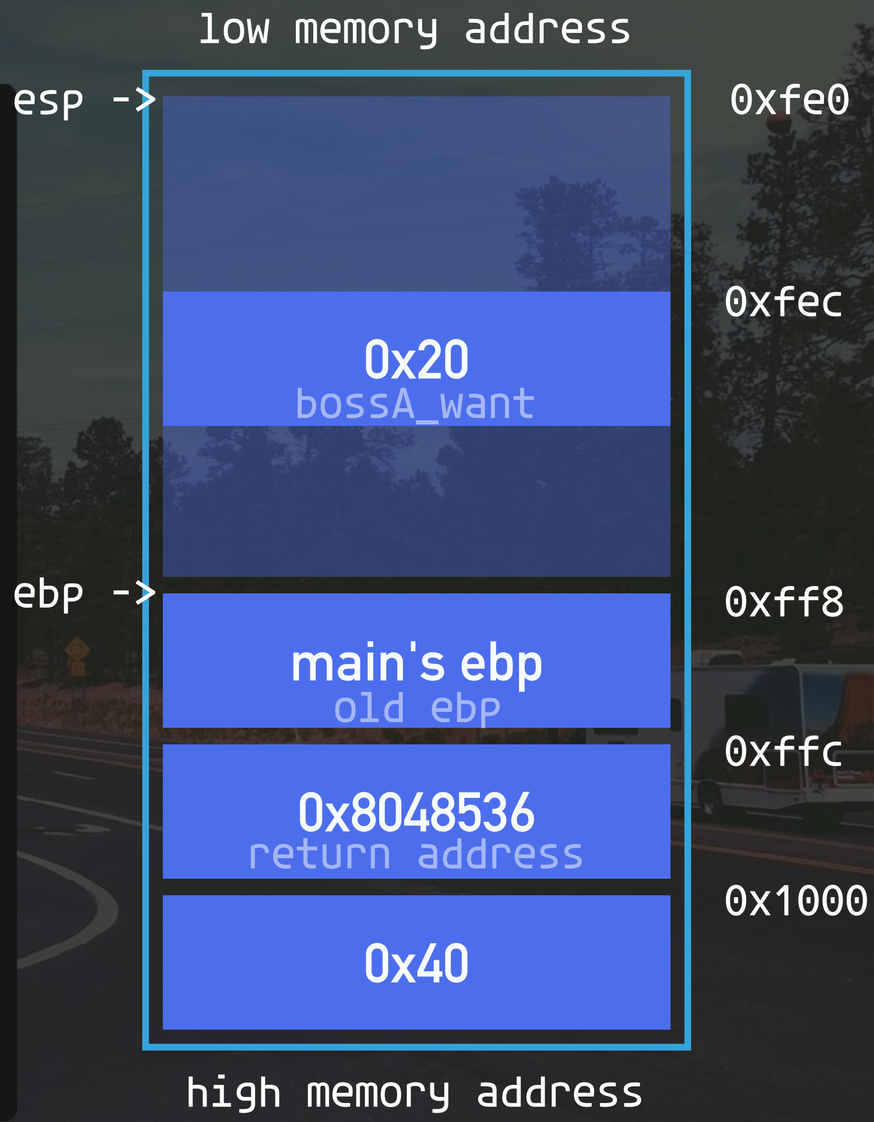
Stack Frame

```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```



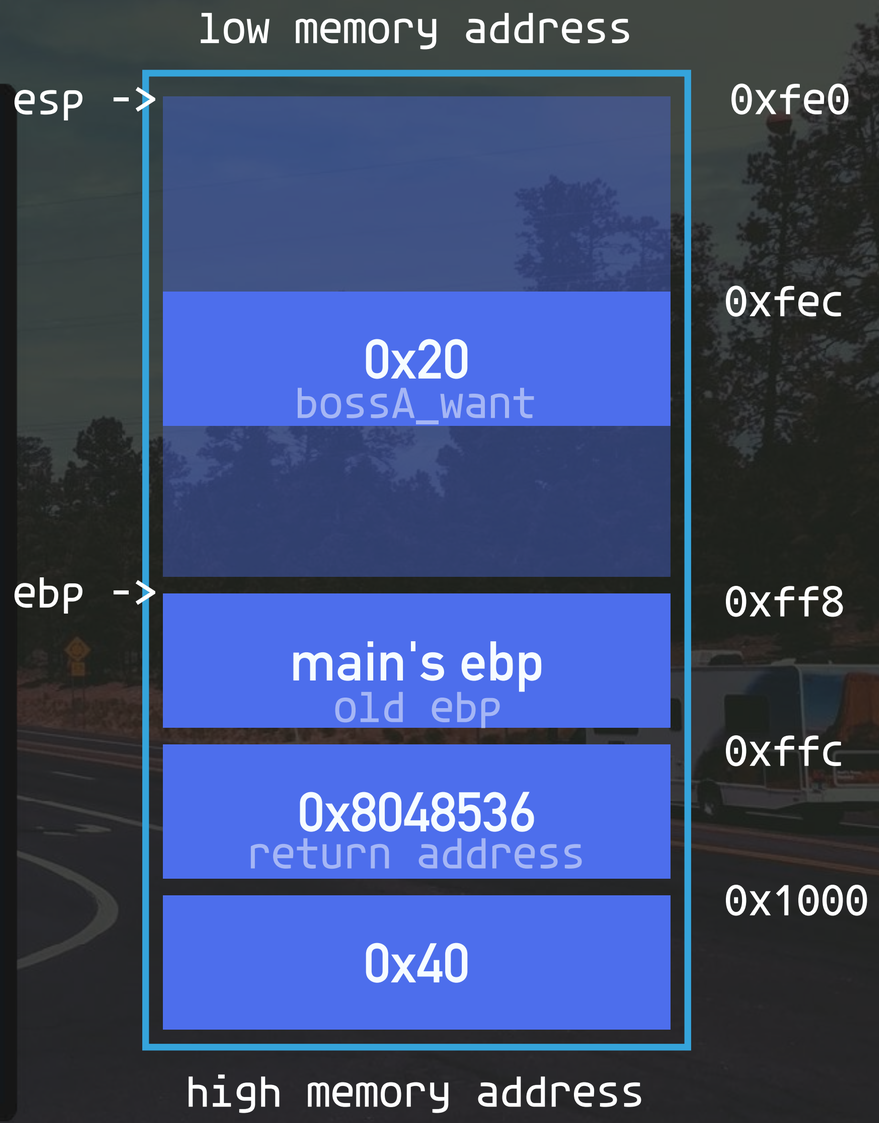
Stack Frame

```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```



Stack Frame

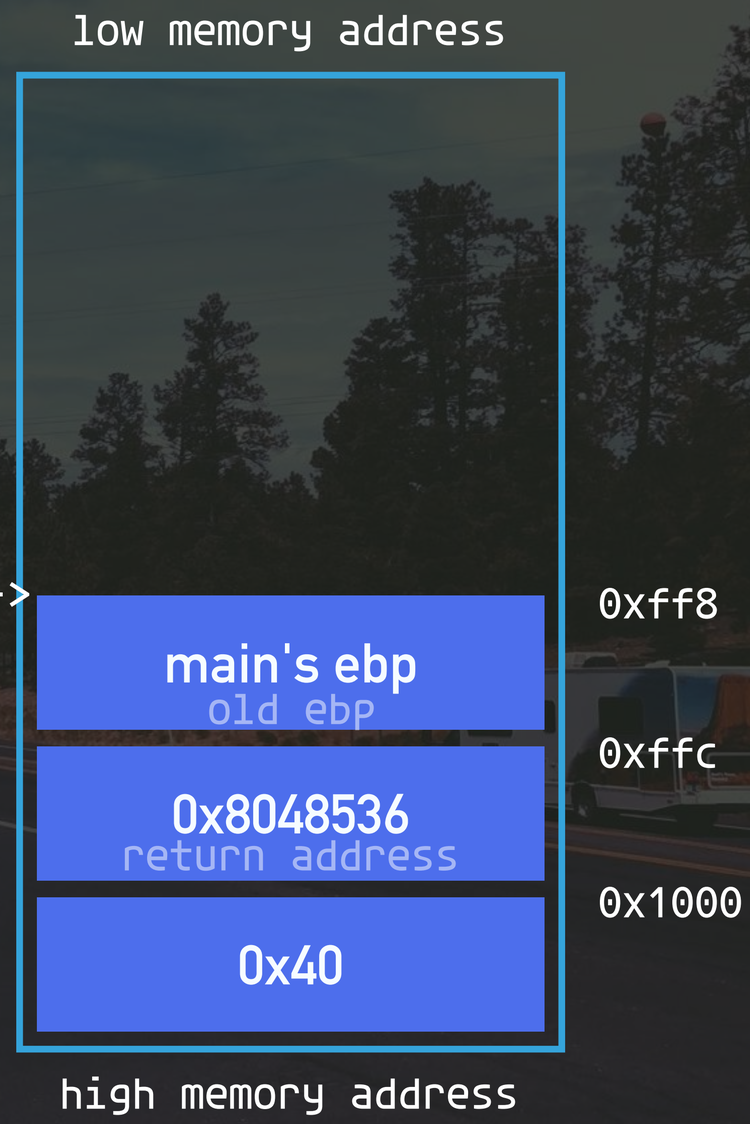
```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave  leave =  mov esp, ebp
17 8048504:       c3                      ret                pop ebp
```



Stack Frame

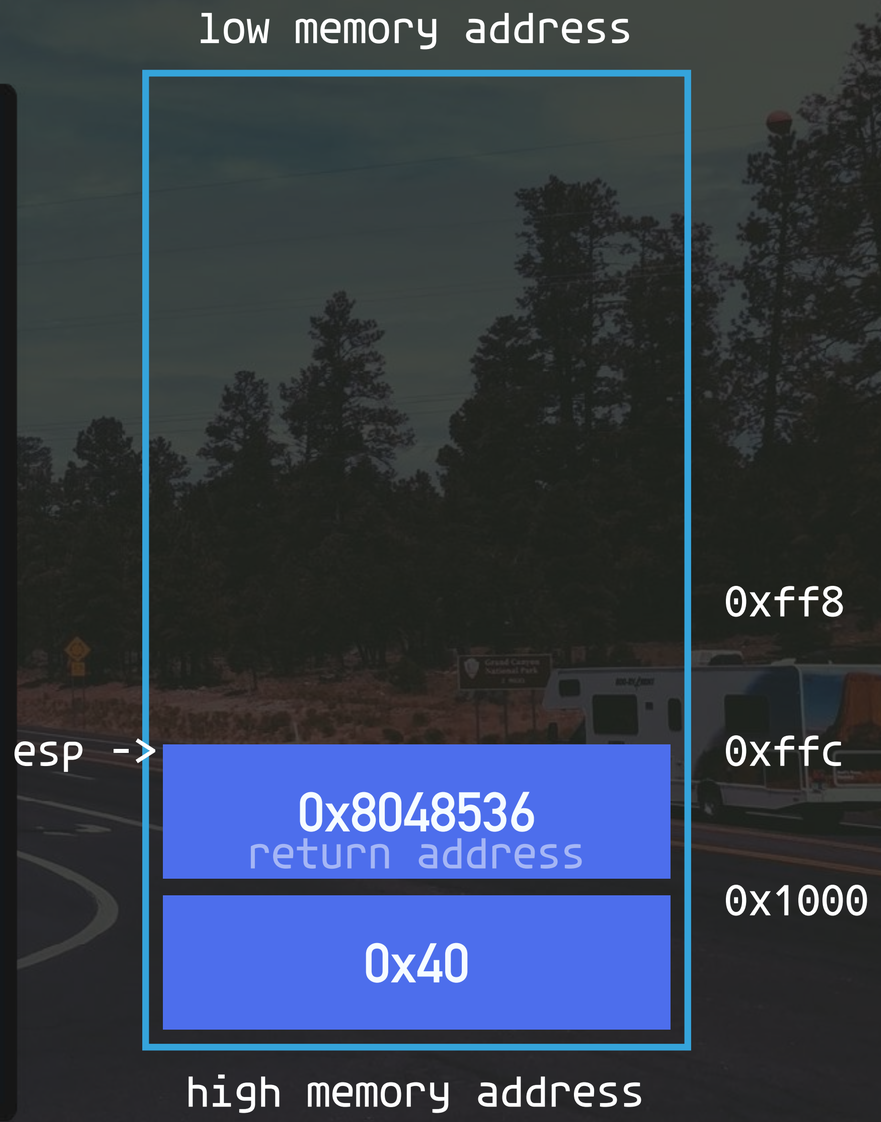
```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```

leave = mov esp, ebp
pop ebp

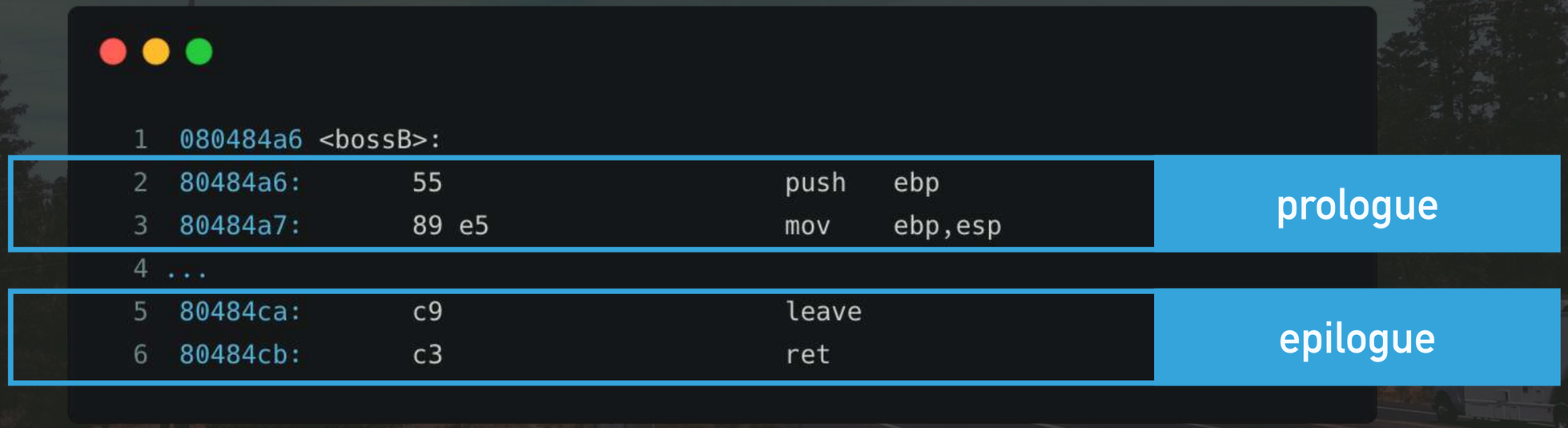


Stack Frame

```
1  080484cc <bossA>:
2  80484cc:       55                      push   ebp
3  80484cd:       89 e5                   mov    ebp,esp
4  80484cf:       83 ec 18                sub    esp,0x18
5  80484d2:       8b 45 08                mov    eax,DWORD PTR [ebp+0x8]
6  ...
7  80484de:       89 45 f4                mov    DWORD PTR [ebp-0xc],eax
8  ...
9  80484ec:       e8 4f fe ff ff          call   8048340 <printf@plt>
10 ...
11 80484f4:       83 ec 0c                sub    esp,0xc
12 80484f7:       ff 75 f4                push   DWORD PTR [ebp-0xc]
13 80484fa:       e8 a7 ff ff ff          call   80484a6 <bossB>
14 80484ff:       83 c4 10                add    esp,0x10
15 8048502:       90                      nop
16 8048503:       c9                      leave
17 8048504:       c3                      ret
```



Stack Frame


1 080484a6 <bossB>:
2 80484a6: 55 push ebp
3 80484a7: 89 e5 mov ebp,esp
4 ...
5 80484ca: c9 leave
6 80484cb: c3 ret
prologue
epilogue

Calling Convention - cdecl

```
1  0804844d <main>:
2  ...
3   804845e:       83 ec 04                sub    esp,0x4
4   8048461:       6a 03                   push   0x3
5   8048463:       6a 02                   push   0x2
6   8048465:       6a 01                   push   0x1
7   8048467:       e8 ba ff ff ff          call   8048426 <iCanAddThreeIntTogether>
8   804846c:       83 c4 10                add    esp,0x10
9   804846f:       b8 00 00 00 00          mov    eax,0x0
10  8048474:       8b 4d fc                mov    ecx,DWORD PTR [ebp-0x4]
11  8048477:       c9                      leave
12  8048478:       8d 61 fc                lea    esp,[ecx-0x4]
13  804847b:       c3                      ret
```

from right to left

caller pop stack

由 caller 清空 stack 比較容易實作 printf 之類的不定長引數函式

Calling Convention - stdcall (win32api use this convention)

```
1  08048426 <iCanAddThreeIntTogether>:
2   8048426:       55                      push   ebp
3   8048427:       89 e5                   mov    ebp,esp
4   8048429:       8b 55 08                mov    edx,DWORD PTR [ebp+0x8]
5   804842c:       8b 45 0c                mov    eax,DWORD PTR [ebp+0xc]
6   804842f:       01 c2                   add    edx,eax
7   8048431:       8b 45 10                mov    eax,DWORD PTR [ebp+0x10]
8   8048434:       01 d0                   add    eax,edx
9   8048436:       50                      push   eax
10  8048437:       68 e0 84 04 08          push   0x80484e0
11  804843c:       e8 9f fe ff ff          call   80482e0 <printf@plt>
12  8048441:       83 c4 08                add    esp,0x8
13  8048444:       90                      nop
14  8048445:       c9                      leave
15  8048446:       c2 0c 00                ret    0xc
```

callee pop stack

由 callee 清空 stack 比較節省空間

Calling Convention - fastcall

```
1 08048452 <main>:
2  8048452:       55                      push   ebp
3  8048453:       89 e5                   mov    ebp,esp
4  8048455:       6a 03                   push   0x3
5  8048457:       ba 02 00 00 00          mov    edx,0x2
6  804845c:       b9 01 00 00 00          mov    ecx,0x1
7  8048461:       e8 c0 ff ff ff          call   8048426 <iCanAddThreeIntTogether>
8  8048466:       b8 00 00 00 00          mov    eax,0x0
9  804846b:       c9                      leave
10 804846c:       c3                      ret
```

put rest args on stack

put first two args in regs

callee pop stack

Calling Convention - thiscall

```
1   4015a6:     8d 55 cc              lea    edx,[ebp-0x34]
2   4015a9:     8b 4d e4              mov    ecx,DWORD PTR [ebp-0x1c]
3   4015ac:     89 0c 24              mov    DWORD PTR [esp],ecx
4   4015af:     c7 45 98 ff ff ff ff  mov    DWORD PTR [ebp-0x68],0xffffffff
5   4015b6:     89 d1                 mov    ecx,edx
6   4015b8:     ff d0                 call   eax
7   4015ba:     83 ec 04              sub    esp,0x4
8   4015bd:     8d 45 cc              lea    eax,[ebp-0x34]
9   4015c0:     89 44 24 04           mov    DWORD PTR [esp+0x4],eax
```

put "this" in ecx

x86_64 Calling Convention

windows

```
function(rcx, rdx, r8, r9)
```

Linux

```
function(rdi, rsi, rdx, rcx, r8, r9)
```

如果參數超過 6 個時，超過的同 x86 calling convention，直接放 stack

(windows 放的位置要注意一下)


OPTIMIZATION

Constant Folding

```
i = 320 * 200 * 32;
```

>>

```
i = 2048000;
```

(example from wiki)

Constant Propagation (+ Folding)

```
1   int x = 14;
2   int y = 7 - x / 2;
3   return y * (28 / x + 2);
```

>>

```
1   int x = 14;
2   int y = 7 - 14 / 2;
3   return y * (28 / 14 + 2);
```

>>

```
1   int x = 14;
2   int y = 0;
3   return 0;
```

(example from wiki)

Tail Call

```
1  foo:
2    mov  reg,[sp+data1]
3    push reg
4    call B
5    pop
6    mov  reg,[sp+data2]
7    push reg
8    call A
9    pop
10   ret
```

>>

```
1  foo:
2    mov  reg,[sp+data1]
3    push reg
4    call B
5    pop
6    mov  reg,[sp+data2]
7    mov  [sp+data1],reg
8    jmp  A
```

(example from wiki)

Some magic trick

```c
#include <stdio.h>

int main(){
    int a = 0 ;
    scanf("%d", &a);
    printf("%d", a/9);
    return 0;
}
```

```
74b:   e8 a0 fe ff ff          call   5f0 <__isoc99_scanf@plt>
750:   8b 4d f4                mov    ecx,DWORD PTR [rbp-0xc]
753:   ba 39 8e e3 38          mov    edx,0x38e38e39
758:   89 c8                   mov    eax,ecx
75a:   f7 ea                   imul   edx
75c:   d1 fa                   sar    edx,1
75e:   89 c8                   mov    eax,ecx
760:   c1 f8 1f                sar    eax,0x1f
763:   29 c2                   sub    edx,eax
765:   89 d0                   mov    eax,edx
767:   89 c6                   mov    esi,eax
769:   48 8d 3d b4 00 00 00    lea    rdi,[rip+0xb4]        # 824 <_IO_stdin_used+0x4>
770:   b8 00 00 00 00          mov    eax,0x0
775:   e8 66 fe ff ff          call   5e0 <printf@plt>
```

Some magic trick

```
edx:eax = input*0x38e38e39

edx = (input*0x38e38e39)>>33

eax = input >> 31

edx = edx - eax
```

```
1  74b:   e8 a0 fe ff ff          call   5f0 <__isoc99_scanf@plt>
2  750:   8b 4d f4                mov    ecx,DWORD PTR [rbp-0xc]
3  753:   ba 39 8e e3 38          mov    edx,0x38e38e39
4  758:   89 c8                   mov    eax,ecx
5  75a:   f7 ea                   imul   edx
6  75c:   d1 fa                   sar    edx,1
7  75e:   89 c8                   mov    eax,ecx
8  760:   c1 f8 1f                sar    eax,0x1f
9  763:   29 c2                   sub    edx,eax
10 765:   89 d0                   mov    eax,edx
11 767:   89 c6                   mov    esi,eax
12 769:   48 8d 3d b4 00 00 00    lea    rdi,[rip+0xb4]        # 824 <_IO_stdin_used+0x4>
13 770:   b8 00 00 00 00          mov    eax,0x0
14 775:   e8 66 fe ff ff          call   5e0 <printf@plt>
```

Some magic trick

$$\frac{x}{d} = \frac{x}{2^n} * \frac{2^n}{d}$$

$$\frac{x}{d} = \frac{x}{2^n} * magic$$

$$\frac{x}{d} = (x * magic) >> n$$

```
1  74b:   e8 a0 fe ff ff          call   5f0 <__isoc99_scanf@plt>
2  750:   8b 4d f4                mov    ecx,DWORD PTR [rbp-0xc]
3  753:   ba 39 8e e3 38          mov    edx,0x38e38e39
4  758:   89 c8                   mov    eax,ecx
5  75a:   f7 ea                   imul   edx
6  75c:   d1 fa                   sar    edx,1
7  75e:   89 c8                   mov    eax,ecx
8  760:   c1 f8 1f                sar    eax,0x1f
9  763:   29 c2                   sub    edx,eax
10 765:   89 d0                   mov    eax,edx
11 767:   89 c6                   mov    esi,eax
12 769:   48 8d 3d b4 00 00 00    lea    rdi,[rip+0xb4]        # 824 <_IO_stdin_used+0x4>
13 770:   b8 00 00 00 00          mov    eax,0x0
14 775:   e8 66 fe ff ff          call   5e0 <printf@plt>
```

CLASS & STRUCT

DEMO

Memo for demo

- name mangling
- 成員間對齊取 min(sizeof(member),pack)
- 結構最後對齊取 min(maxMemberSize, pack)
- 嵌套結構不以整體長度來計算，而是以該結構所使用的對齊值來對齊
- 為了實現多態，class 的第一個 member 會指向 vtable

下週主題

- Anti-Debug & Anti-Analyze
- PE-File format
- PE related trick