

What is ASLR?

A: Address

S: Space

L: Layout

R: Randomization



```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
```

```
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push esi
push esi
push esi
call sub_31486A
test eax, eax
jz short loc_31306D
jnz short loc_313066
```

_313066:

loc_31306D:

```
call sub_31486A
test eax, eax
jg short loc_31306D
call sub_3140F3
jmp short loc_31308C
```

loc_31307D:

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

loc_31308C:

```
mov [ebp+var_4], eax
```

Course Terminology

- **Address Space Layout Randomization**
 - An exploit mitigation technology used to ensure that address ranges for important memory segments are random for every execution
 - Meant to mitigate exploits leveraging hardcoded stack, heap, code, libc addresses
 - Known as **ASLR** for short



Runtime Process Without ASLR



0x00000000 - Start of memory

0x08049290 - 0x0805033c (R-X)

0x08050360 - 0x08051208 (R--)

0x08055000 - 0x08076000 (RW-)

0xb7e25000 - 0xb7fcd000

0xbffdf000 - 0xc0000000 (RW-)

0xFFFFFFFF - End of memory

Run #1 Without ASLR



0x00000000 - Start of memory

0x08049290 - 0x0805033c (R-X)

0x08050360 - 0x08051208 (R--)

0x08055000 - 0x08076000 (RW-)

0xb7e25000 - 0xb7fcd000

0xbffdf000 - 0xc0000000 (RW-)

0xFFFFFFFF - End of memory

Run #2 Without ASLR



← 0x00000000 - Start of memory

← 0x08049290 - 0x0805033c (R-X)

← 0x08050360 - 0x08051208 (R--)

← 0x08055000 - 0x08076000 (RW-)

← 0xb7e25000 - 0xb7fcd000

← 0xbffdf000 - 0xc0000000 (RW-)

← 0xFFFFFFFF - End of memory

```

push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
mov     [ebp+var_70], eax
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
loc_313066:
; CODE XREF: sub_312FD8
; sub_312FD8+56
push    0Dh
call    sub_31411B
loc_31306D:
; CODE XREF: sub_312FD8
; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
jz      short loc_3130F3
jmp     short loc_31308C
; -----
loc_31307D:
; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
loc_31308C:
; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax

```

Run #3 Without ASLR



0x00000000 - Start of memory

0x08049290 - 0x0805033c (R-X)

0x08050360 - 0x08051208 (R--)

0x08055000 - 0x08076000 (RW-)

0xb7e25000 - 0xb7fcd000

0xbffdf000 - 0xc0000000 (RW-)

0xFFFFFFFF - End of memory

```

push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
mov     [ebp+var_70], eax
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
loc_313066:
; CODE XREF: sub_312FD8
; sub_312FD8+56
push    0Dh
call    sub_31411B
loc_31306D:
; CODE XREF: sub_312FD8
; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
jz      short loc_3130F3
jmp     short loc_31308C
; -----
loc_31307D:
; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
loc_31308C:
; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax

```



```

push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

```

```

loc_313066:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+56

```

```

push    0Dh
call    sub_31411B

```

```

loc_31306D:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+49

```

```

call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C

```

```

loc_31307D:                                     ; CODE XREF: sub_312FD8

```

```

call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

```

```

loc_31308C:                                     ; CODE XREF: sub_312FD8

```

```

mov     [ebp+var_4], eax

```

ya so, nothing changes...

Runtime Process Without ASLR



0x00000000 - Start of memory

0x08049290 - 0x0805033c (R-X)

0x08050360 - 0x08051208 (R--)

0x08055000 - 0x08076000 (RW-)

0xb7e25000 - 0xb7fcd000

0xbffdf000 - 0xc0000000 (RW-)

0xFFFFFFFF - End of memory

Run #1 With ASLR



0x00000000 - Start of memory

0x08049290 - 0x0805033c (R-X)

0x08050360 - 0x08051208 (R--)

0x244b9000 - 0x24661000

0x7fa54000 - 0x7fa75000 (RW-)

0x98429000 - 0x9844a000 (RW-)

0xFFFFFFFF - End of memory

Run #2 With ASLR



0x00540000 - 0x006e8000

0x08049290 - 0x0805033c (R-X)

0x08050360 - 0x08051208 (R--)

0x10962000 - 0x10983000 (RW-)

0xa07ee000 - 0xa080f000 (RW-)

0xFFFFFFFF - End of memory

```

push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    esi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

```

```

push    0Dh
call    sub_31411B
loc_31306D:
call    sub_3140F3
test    eax, eax
jz      short loc_31307D
jmp     short loc_31308C
; CODE XREF: sub_312FD8
; sub_312FD8+56
; sub_312FD8+49
; CODE XREF: sub_312FD8
; sub_312FD8+49
loc_31307D:
call    sub_3140F3
test    eax, eax
jz      short loc_31308C
; CODE XREF: sub_312FD8
; sub_312FD8+49
loc_31308C:
mov     [ebp+var_4], eax

```

Run #3 With ASLR



0x00000000 - Start of memory

0x08049290 - 0x0805033c (R-X)

0x08050360 - 0x08051208 (R--)

0x094fb000 - 0x0951c000 (RW-)

0x43db2000 - 0x43dd3000 (RW-)

0xbf8c3000 - 0xbf8e4000

0xFFFFFFFF - End of memory

```

push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
loc_313066:
push    0Dh
call    sub_31411B
loc_31306D:
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
loc_31308C:
loc_31307D:
and     eax, 0FFFFFFh
or      eax, 80070000h
mov     [ebp+var_4], eax

```

ASLR in Action

> Open up a terminal.

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+56
push    0Dh
call    sub_31411B

loc_31306D:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
; -----

loc_31307D:                                     ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                     ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```

ASLR in Action

- > Open up a terminal.
- > Type “**cat /proc/self/maps**”

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+56
push    0Dh
call    sub_31411B

loc_31306D:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
; -----

loc_31307D:                                     ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                     ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```

ASLR in Action

- > Open up a terminal.
- > Type “`cat /proc/self/maps`”
- > Repeat a few times :)

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                ; CODE XREF: sub_312FD8
                                           ; sub_312FD8+56
push    0Dh
call    sub_31411B

loc_31306D:                                ; CODE XREF: sub_312FD8
                                           ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
; -----

loc_31307D:                                ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```


ASLR in Action

- > Open up a terminal.
- > Type “**cat /proc/self/maps**”
- > Repeat a few times :)

You'll see lots of lines like this:

bfe49000-bfe6a000 rw-p 00000000 00:00 0

...

bfa23000-bfa44000 rw-p 00000000 00:00 0

...

bfdab000-bfdcc000 rw-p 00000000 00:00 0

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
```

[stack]

```
push    0Dh
call    sub_31411B
```

[stack]

```
call    sub_31411B
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
```

[stack]

loc_31307D: ; CODE XREF: sub_312FD8

```
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
```

loc_31308C: ; CODE XREF: sub_312FD8

```
mov     [ebp+var_4], eax
```

ASLR in Action

- > Open up a terminal.
- > Type “`cat /proc/self/maps`”
- > Repeat a few times :)
- Stack Address Changes

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
```

```
loc_313066:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+56
```

```
push    0Dh
call    sub_31411B
```

```
loc_31306D:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+49
```

```
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
```

```
; -----
```

```
loc_31307D:                                     ; CODE XREF: sub_312FD8
```

```
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
```

```
loc_31308C:                                     ; CODE XREF: sub_312FD8
```

```
mov     [ebp+var_4], eax
```

ASLR in Action

- > Open up a terminal.
- > Type “`cat /proc/self/maps`”
- > Repeat a few times :)

- Stack Address Changes
- Heap Address Changes

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+56
push    0Dh
call    sub_31411B

loc_31306D:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C

; -----
loc_31307D:                                     ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                     ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```

ASLR in Action

- > Open up a terminal.
- > Type “**cat /proc/self/maps**”
- > Repeat a few times :)

- Stack Address Changes
- Heap Address Changes
- Library Addresses Change

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+56
push    0Dh
call    sub_31411B

loc_31306D:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
; -----
loc_31307D:                                     ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                     ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```

ASLR Basics

- Memory segments are no longer in static address ranges, rather they are unique for every execution

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
call    sub_31466A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                ; CODE XREF: sub_312FD8
                                           ; sub_312FD8+56
push    0Dh
call    sub_31411B

loc_31306D:                                ; CODE XREF: sub_312FD8
                                           ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
; -----

loc_31307D:                                ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```

ASLR Basics

- Memory segments are no longer in static address ranges, rather they are unique for every execution
- A simple stack smash may get you control of EIP, but what does it matter if you have no idea where you can go with it?

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
call    sub_31466A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+var_70]
push    edi
call    sub_314663
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                ; CODE XREF: sub_312FD8
                                           ; sub_312FD8+56
push    0Dh
call    sub_31411B

loc_31306D:                                ; CODE XREF: sub_312FD8
                                           ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C

; -----
loc_31307D:                                ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```


ASLR Basics

- Memory segments are no longer in static address ranges, rather they are unique for every execution
- A simple stack smash may get you control of EIP, but what does it matter if you have no idea where you can go with it?
 - The essence of ASLR
- You must work with no expectation of where anything is in memory anymore

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
```

```
push    esi
push    eax
push    edi
call    sub_31466A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+var_70]
push    edi
call    sub_314663
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
```

```
loc_313066:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+56
```

```
push    esi
call    sub_31411B
```

```
loc_31306D:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+49
```

```
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
```

```
; -----
```

```
loc_31307D:                                     ; CODE XREF: sub_312FD8
```

```
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
```

```
loc_31308C:                                     ; CODE XREF: sub_312FD8
```

History of ASLR

- When was **ASLR** implemented?
 - **May 1st, 2004** - OpenBSD 3.5 (**mmap**)
 - **June 17th, 2005** - Linux Kernel 2.6.12 (**stack, mmap**)
 - **January 30th, 2007** - Windows Vista (**full**)
 - **October 26th, 2007** - Mac OSX 10.5 Leopard (**sys libraries**)
 - **October 21st, 2010** - Windows Phone 7 (**full**)
 - **March 11th, 2011** - iPhone iOS 4.3 (**full**)
 - **July 20th, 2011** - Mac OSX 10.7 Lion (**full**)

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
```

```
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
call sub_314623
```

```
loc_313066:                                     ; CODE XREF: sub_312FD8
; sub_312FD8+56
push 1D0h
call sub_31411B
```

```
loc_31306A:                                     ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D:                                     ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C:                                     ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

History of ASLR

- When was **ASLR** implemented?
 - **May 1st, 2004** - OpenBSD 3.5 (**mmap**)
 - **June 17th, 2005** - Linux Kernel 2.6.12 (**stack, mmap**)
 - **January 30th, 2007** - Windows Vista (**full**)
 - **October 26th, 2007** - Mac OSX 10.5 Leopard (**sys libraries**)
 - **October 21st, 2010** - Windows Phone 7 (**full**)
 - **March 11th, 2011** - iPhone iOS 4.3 (**full**)
 - **July 20th, 2011** - Mac OSX 10.7 Lion (**full**)

perspective: markus is accepted to **RPI**

Reminder:

Security is rapidly evolving

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
jz      [ebp+arg_0], esi
jz      short loc_31306F
loc_313066:                                ; CODE XREF: sub_312FD8
                                           ; sub_312FD8+56
push    0Dh
call    sub_31411B
loc_31306D:                                ; CODE XREF: sub_312FD8
                                           ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
; -----
loc_31307D:                                ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
loc_31308C:                                ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```

Checking for ASLR

```
$ cat /proc/sys/kernel/randomize_va_space
```

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
call    sub_31466A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
```

```
loc_313066:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+56
```

```
push    0Dh
call    sub_31411B
```

```
loc_31306D:                                     ; CODE XREF: sub_312FD8
                                              ; sub_312FD8+49
```

```
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
```

```
; -----
```

```
loc_31307D:                                     ; CODE XREF: sub_312FD8
```

```
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
```

```
loc_31308C:                                     ; CODE XREF: sub_312FD8
```

```
mov     [ebp+var_4], eax
```

Checking for ASLR

\$ cat /proc/sys/kernel/randomize_va_space
2

```

push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
call    sub_31466A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

```

```

loc_313066:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+56

```

```

push    0Dh
call    sub_31411B

```

```

loc_31306D:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+49

```

```

call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C

```

```

; -----

```

```

loc_31307D:                                     ; CODE XREF: sub_312FD8

```

```

call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

```

```

loc_31308C:                                     ; CODE XREF: sub_312FD8

```

```

mov     [ebp+var_4], eax

```


Checking for ASLR

```
$ cat /proc/sys/kernel/randomize_va_space  
2
```

0: No ASLR

1: Conservative Randomization

(Stack, Heap, Shared Libs, PIE, mmap(), VDR0)

2: Full Randomization

(Conservative Randomization + memory managed via brk())

```
push edi  
call sub_314623  
test eax, eax  
jz short loc_31306D  
cmp [ebp+arg_0], ebx  
jnz short loc_313066  
mov eax, [ebp+var_70]  
cmp eax, [ebp+var_84]  
jb short loc_313066  
sub eax, [ebp+var_84]  
push esi
```

```
push esi  
push eax  
push edi  
call sub_31466A  
test eax, eax  
jz short loc_31306D  
push esi  
lea eax, [ebp+arg_0]  
push eax
```

```
push esi  
push [ebp+arg_4]  
push edi  
call sub_314623  
test eax, eax  
jz short loc_31306D  
cmp [ebp+arg_0], esi  
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8  
; sub_312FD8+56
```

```
push 0Dh  
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8  
; sub_312FD8+49
```

```
test eax, eax  
jg short loc_31307D  
call sub_3140F3  
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3  
and eax, 0FFFFFFh  
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```