

Tools and Basic Reverse Engineering

Modern Binary Exploitation

CSCI 4968 – Spring 2015

Jeremy Blackthorne

```
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+54
```

```
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
```

Lecture Overview

1. Introduction to Reverse Engineering
2. Tools!
3. Resources

```
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+54
```

```
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, 0FFFFh
or      eax, 80070000h
```

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

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



“What you must learn is that these rules are no different than the rules of a computer system. Some of them can be bent, others can be broken.”

EF: sub_312FD8
FD8+55

EF: sub_312FD8
FD8+49

EF: sub_312FD8

```
-----  
and    eax, 0FFFFh  
or     eax, 80070000h
```

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

Compiling

Source Code

```
int main()  
{  
    printf("Hello world!");  
}
```

Compile



Assembly

```
_main PROC  
    push ebp  
    mov  ebp, esp  
    push OFFSET $SG2985  
    call DWORD PTR __imp_printf  
    add  esp, 4  
    xor  eax, eax  
    pop  ebp  
    ret  0  
_main ENDP  
_TEXT ENDS  
END
```

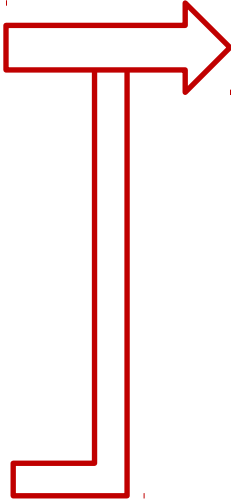
Assemble



Object File

```
68 00 30 40 00 FF 15 90  
5D C3 88 4D 5A 00 00 66  
33 C0 EB 34 8B 0D 3C 00  
50 45 00 00 75 EA 88 08  
40 00 75 DC 33 C0 83 89  
81 E8 00 40 00 0F 95 C0  
15 7C 20 40 00 59 6A FF  
34 20 40 00 A3 88 33 40  
30 40 00 89 01 8B 00 38  
89 01 E8 27 05 00 00 E8  
40 00 00 75 0C 68 E4 12  
59 E8 48 05 00 00 83 3D  
FF FF 15 44 20 40 00 59  
E8 D4 04 00 00 A1 58 30  
00 FF 35 54 30 40 00 03  
-- -- -- -- -- -- -- --
```

Link



Binary File

```
68 00 30 40 00 FF 15 90  
5D C3 88 4D 5A 00 00 66  
33 C0 EB 34 8B 0D 3C 00  
50 45 00 00 75 EA 88 08  
40 00 75 DC 33 C0 83 89  
81 E8 00 40 00 0F 95 C0  
15 7C 20 40 00 59 6A FF  
34 20 40 00 A3 88 33 40  
30 40 00 89 01 8B 00 38  
89 01 E8 27 05 00 00 E8  
40 00 00 75 0C 68 E4 12  
00 68 B4 20 40 00 68 A4  
59 59 85 C0 74 17 C7 45  
00 00 E9 DE 00 00 00 89  
33 40 00 75 1B 68 A0 26  
77 04 00 00 59 59 C7 05
```

Libraries

```
00 68 B4 20 40 00 68 A4  
59 59 85 C0 74 17 C7 45  
00 00 E9 DE 00 00 00 89  
33 40 00 75 1B 68 A0 26  
77 04 00 00 59 59 C7 05
```

Loading

Source Code

```
int main()
{
    printf("Hello world!");
}
```

Compile

Assembly

```
_main PROC
    push ebp
    mov  ebp, esp
    push OFFSET $SG2985
    call DWORD PTR __imp_printf
    add  esp, 4
    xor  eax, eax
    pop  ebp
    ret  0
_main ENDP
_TEXT ENDS
END
```

Assemble

Object File

```
68 00 30 40 00 FF 15 90
5D C3 B8 40 50 00 00 66
33 C0 EB 34 8B 00 3C 00
50 45 00 00 75 EA 8B 00
40 00 75 DC 33 C0 83 B9
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 A3 88 33 40
30 40 00 89 01 8B 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 59 85 C0 74 17 C7 45
00 00 E9 DE 00 00 00 89
33 40 00 75 1B 68 A0 26
E8 D4 00 00 00 A1 58 30
00 FF 35 54 30 40 00 A3
.. .. .. .. .. .. ..
```

Link

Binary File

```
68 00 30 40 00 FF 15 90
5D C3 B8 40 50 00 00 66
33 C0 EB 34 8B 00 3C 00
50 45 00 00 75 EA 8B 00
40 00 75 DC 33 C0 83 B9
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 A3 88 33 40
30 40 00 89 01 8B 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
00 68 B4 20 40 00 68 A4
59 59 85 C0 74 17 C7 45
00 00 E9 DE 00 00 00 89
33 40 00 75 1B 68 A0 26
77 04 00 00 59 59 C7 05
```

Load

Process

```
68 00 30 40 00 FF 15 90
5D C3 B8 40 50 00 00 66
33 C0 EB 34 8B 00 3C 00
50 45 00 00 75 EA 8B 00
40 00 75 DC 33 C0 83 B9
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 A3 88 33 40
30 40 00 89 01 8B 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
00 68 B4 20 40 00 68 A4
59 59 85 C0 74 17 C7 45
00 00 E9 DE 00 00 00 89
33 40 00 75 1B 68 A0 26
77 04 00 00 59 59 C7 05
```

Libraries

```
00 68 B4 20 40 00 68 A4
59 59 85 C0 74 17 C7 45
00 00 E9 DE 00 00 00 89
33 40 00 75 1B 68 A0 26
77 04 00 00 59 59 C7 05
```

Running

Process, t=0

40	00	30	40	00	FF	15	20
50	C0	00	40	50	00	00	6A
30	C0	E0	3A	00	00	3C	00
50	45	00	00	75	E0	00	00
40	00	75	0C	33	C0	03	09
01	E0	00	40	00	0F	95	C0
15	7C	20	40	00	59	6A	FF
3A	20	40	00	03	00	33	40
30	40	00	09	01	00	00	30
09	01	E0	27	05	00	00	E0
40	00	00	75	0C	60	E4	12
59	E0	40	05	00	00	03	30
FF	FF	15	4A	20	40	00	59
E0	0A	00	00	00	01	50	30
00	FF	25	50	30	40	00	03



Process, t=1

60	00	30	40	00	FF	15	20
50	C0	00	40	50	00	00	6A
30	C0	E0	3A	00	00	3C	00
50	45	00	00	75	E0	00	00
40	00	75	0C	33	C0	03	09
01	E0	00	40	00	0F	95	C0
15	7C	20	40	00	59	6A	FF
3A	20	40	00	03	00	33	40
30	40	00	09	01	00	00	30
09	01	E0	27	05	00	00	E0
40	00	00	75	0C	60	E4	12
59	E0	40	05	00	00	03	30
FF	FF	15	4A	20	40	00	59
E0	0A	00	00	00	01	50	30
00	FF	25	50	30	40	00	03



Process, t=i

60	00	30	40	00	FF	15	20
50	C0	00	40	50	00	00	6A
30	C0	E0	3A	00	00	3C	00
50	45	00	00	75	E0	00	00
40	00	75	0C	33	C0	03	09
01	E0	00	40	00	0F	95	C0
15	7C	20	40	00	59	6A	FF
3A	20	40	00	03	00	33	40
30	40	00	09	01	00	00	30
09	01	E0	27	05	00	00	E0
40	00	00	75	0C	60	E4	12
59	E0	40	05	00	00	03	30
FF	FF	15	4A	20	40	00	59
E0	0A	00	00	00	01	50	30
00	FF	25	50	30	40	00	03



Process, t=n

40	00	30	40	00	FF	15	20
50	C0	00	40	50	00	00	6A
30	C0	E0	3A	00	00	3C	00
50	45	00	00	75	E0	00	00
40	00	75	0C	33	C0	03	09
01	E0	00	40	00	0F	95	C0
15	7C	20	40	00	59	6A	FF
3A	20	40	00	03	00	33	40
30	40	00	09	01	00	00	30
09	01	E0	27	05	00	00	E0
40	00	00	75	0C	60	E4	12
59	E0	40	05	00	00	03	30
FF	FF	15	4A	20	40	00	59
E0	0A	00	00	00	01	50	30
00	FF	25	50	30	40	00	03

Step

Step

Step

RE Domain

Binary File

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 EB 34 8B 0D 3C 00
50 45 00 00 75 EA 08 0B
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 A3 88 33 40
30 40 00 89 01 8B 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
00 68 B4 20 40 00 68 A4
59 59 85 C0 74 17 C7 45
00 00 E9 DE 00 00 00 89
33 40 00 75 1B 68 A0 26
77 04 00 00 59 59 C7 05
```



Load

Process, t=0

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 EB 34 8B 0D 3C 00
50 45 00 00 75 EA 08 0B
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 A3 88 33 40
30 40 00 89 01 8B 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 83 30
FF FF 15 A4 20 40 00 59
1B 04 00 00 00 01 58 38
00 FF 35 34 30 A8 00 03
```



Step

Process, t=i

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 EB 34 8B 0D 3C 00
50 45 00 00 75 EA 08 0B
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 A3 88 33 40
30 40 00 89 01 8B 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 83 30
FF FF 15 A4 20 40 00 59
1B 04 00 00 00 01 58 38
00 FF 35 34 30 A8 00 03
```



Step

Process, t=n

```
00 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 EB 34 8B 0D 3C 00
50 45 00 00 75 EA 08 0B
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 A3 88 33 40
30 40 00 89 01 8B 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 83 30
FF FF 15 A4 20 40 00 59
1B 04 00 00 00 01 58 38
00 FF 35 34 30 A8 00 03
```

RE Domain

Binary File

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 88 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 03 88 33 40
30 40 00 89 01 88 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
00 68 B4 20 40 00 68 84
59 59 85 C0 74 17 C7 45
00 00 E9 DE 00 00 00 89
03 40 00 75 18 68 00 20
77 04 00 00 59 59 C7 00
```



Process, t=0

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 88 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 03 88 33 40
30 40 00 89 01 88 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 03 30
FF FF 15 4A 20 40 00 59
18 04 00 00 00 01 58 38
00 FF 35 34 30 48 00 03
```



Process, t=i

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 88 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 03 88 33 40
30 40 00 89 01 88 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 03 30
FF FF 15 4A 20 40 00 59
18 04 00 00 00 01 58 38
00 FF 35 34 30 48 00 03
```



Process, t=n

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 88 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 03 88 33 40
30 40 00 89 01 88 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 03 30
FF FF 15 4A 20 40 00 59
18 04 00 00 00 01 58 38
00 FF 35 34 30 48 00 03
```

Load

Step

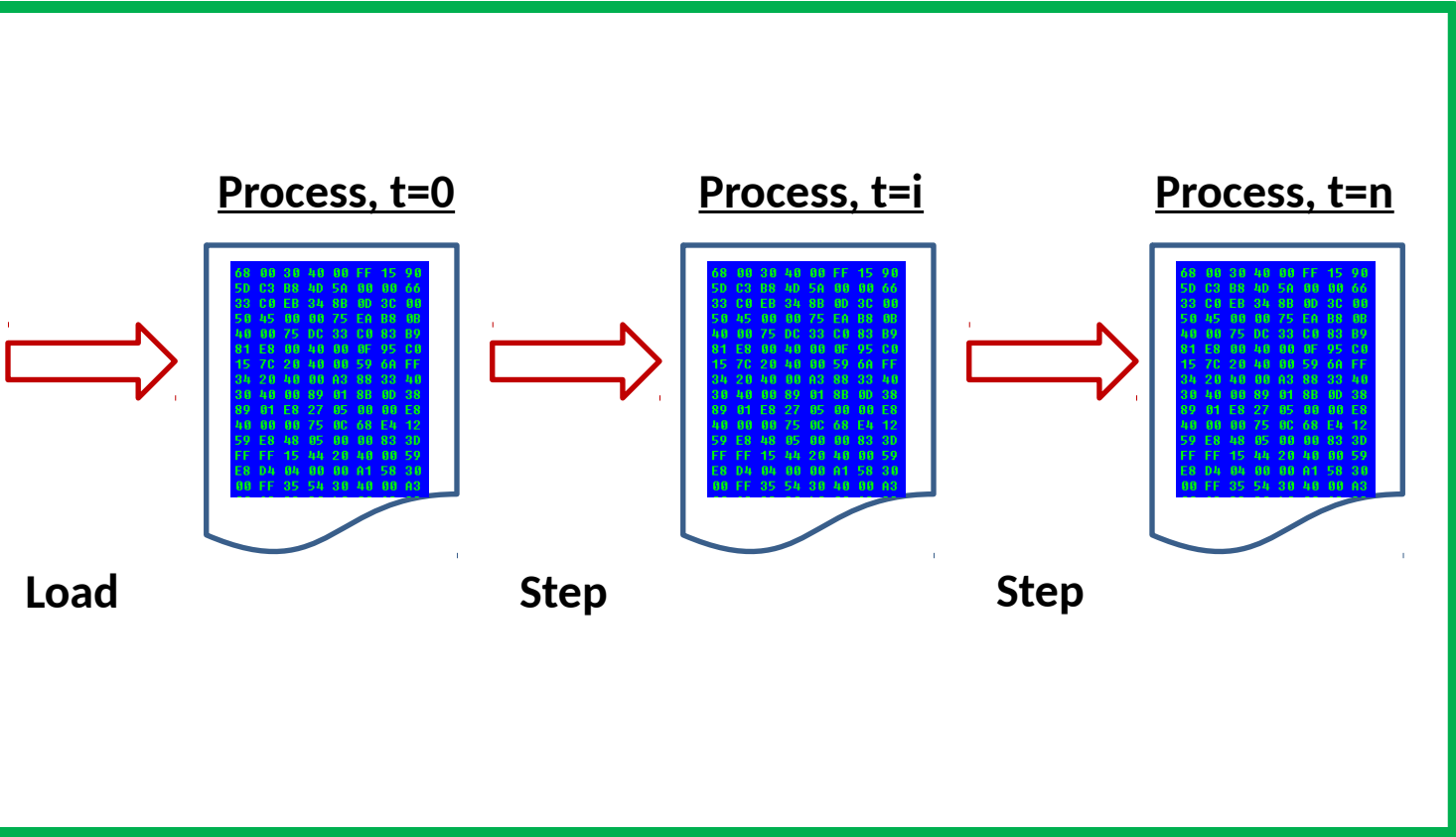
Step

Static

RE Domain

Binary File

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 80 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 A3 88 33 40
34 20 40 00 03 88 33 40
30 40 00 89 01 80 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
00 68 B4 20 40 00 68 A4
59 59 85 C0 74 17 C7 45
00 00 E9 DE 00 00 00 89
03 40 00 75 18 68 A0 20
77 04 00 00 59 59 C7 00
```



Static

Dynamic

RE Domain

Binary File

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 88 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 03 88 33 40
30 40 00 89 01 88 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
00 68 B4 20 40 00 68 84
59 59 85 C0 74 17 C7 45
00 00 E9 DE 00 00 00 89
03 40 00 75 18 68 00 20
77 04 00 00 59 59 C7 00
```



Process, t=0

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 88 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 03 88 33 40
30 40 00 89 01 88 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 03 30
FF FF 15 4A 20 40 00 59
18 04 00 00 00 01 58 38
00 FF 35 34 30 48 00 03
```



Process, t=i

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 88 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 03 88 33 40
30 40 00 89 01 88 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 03 30
FF FF 15 4A 20 40 00 59
18 04 00 00 00 01 58 38
00 FF 35 34 30 48 00 03
```



Process, t=n

```
68 00 30 40 00 FF 15 90
50 C3 08 40 50 00 00 66
33 C0 E8 34 88 00 3C 00
50 45 00 00 75 EA 88 00
40 00 75 DC 33 C0 83 89
81 E8 00 40 00 0F 95 C0
15 7C 20 40 00 59 6A FF
34 20 40 00 03 88 33 40
30 40 00 89 01 88 00 38
89 01 E8 27 05 00 00 E8
40 00 00 75 0C 68 E4 12
59 E8 48 05 00 00 03 30
FF FF 15 4A 20 40 00 59
18 04 00 00 00 01 58 38
00 FF 35 34 30 48 00 03
```

Load

Step

Step

Static

Lecture Overview

1. Introduction to Reverse Engineering
2. Tools!
3. Resources

```
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+54
```

```
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, 0FFFFh
or     eax, 80070000h
```

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

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

Tool Color Coding

- Linux Tool
 - Command
- Windows Tool
 - ToolName.exe
- Associated Challenges:
 - ChallengeName

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

Hex Editor / Viewers

- Hex Editors / Viewers
 - wxHexEditor (GUI)
 - xxd
 - “-i” option is C include style
- Challenge:
 - crackme0x00a

```
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+55
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

ASCII Readable Hex

- strings

- Displays ASCII strings > 4 characters long

- Challenge:

- crackme0x00a

- crackme0x00b

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

ASCII Readable Hex

- strings

- Displays ASCII strings > 4 characters long

- Challenge:

- crackme0x00a

- crackme0x00b

- strings -e ? crackme0x00b

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

File Formats on Disk

- Linux:
 - ELF-Walkthrough.png
 - readelf

```
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+54
```

```
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, 0FFFFh
or     eax, 80070000h
```

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

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


File Formats on Disk

- Linux:
 - ELF-Walkthrough.png
 - **readelf**
- Windows:
 - PE-Layout.jpg
 - Peview.exe

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

File Formats on Disk

- Linux:
 - ELF-Walkthrough.png
 - **readelf**
- Windows:
 - PE-Layout.jpg
 - Peview.exe
- For unknown files / binaries
 - **file**

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

Hashing

- Do we have the same file?
 - md5sum
- Upload hash to virustotal.com
- Google search hash

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

Hashing

- Do we have the same file?
 - md5sum
- Upload hash to virustotal.com
- Google search hash
- Fuzzy hashing:
 - ssdeep -b original.elf >hash.txt
 - ssdeep -bm hash.txt modified.elf

```
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+54

```
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, 0FFFFh
or eax, 80070000h
```

loc_31308C:

; CODE XREF: sub_312FD8

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

Command Line Disassembly

- crackme0x01

```
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+54
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, 0FFFFh
or      eax, 80070000h

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

Command Line Disassembly

- crackme0x01
- objdump -d

```
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+54
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, 0FFFFh
or eax, 80070000h

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

Command Line Disassembly

- crackme0x01
- objdump -d
- Convert hex to decimal
 - echo \$((0xDEADBEEF))

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

Patching Binaries

- It's your binary, you can patch it if you want to
- `objdump -d crackme0x00a | grep -A 30 '<main>'`
- `wxHexEditor-->Edit-->Find`

```
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_4], eax
call sub_31486A
test eax, eax
jz short loc_31306D
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+55
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, 0FFFFh
or eax, 80070000h

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


External Diffing

- Original + modified = HUGE advantage
- **wxHexEditor**-->Tools-->compare files

```
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+54
```

```
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, 0FFFFh
or      eax, 80070000h
```

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

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

Disassembly

- `objdump -d`
- `IDA Pro.exe`
- Challenges:
 - `crackme0x01`

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

Disassembly

- `objdump -d`
- `IDA Pro.exe`
- Challenges:
 - `crackme0x01`
 - `crackme0x02`

```
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+54
```

```
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, 0FFFFh
or eax, 80070000h
```

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

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

IDA Pro

- IDA Pro.exe
- crackme0x04

```
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+54
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, 0FFFFh
or eax, 80070000h

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

IDA Basics

- Change between basic and graphic mode (space bar)
- Rename variables: (n)
- Comment
 - Side: (:), (;)
 - Above/below: (ins)
- Convert const formats: (right-click)
- Cross-reference: (x)
- Change to array: (a)
- IDA->Windows->Reset desktop
- IDA->Options->General->auto comment
- IDA->Options->General->opcode bytes 8

```
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+54
```

```
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
```

The Stack

```
int foo(int a, int b, int c)
{
    int x;
    int y;
    int z;

    x=y=z=0;
    z=x+y+a+b+c;
    return z;
}
int main(int argc, char **argv) {

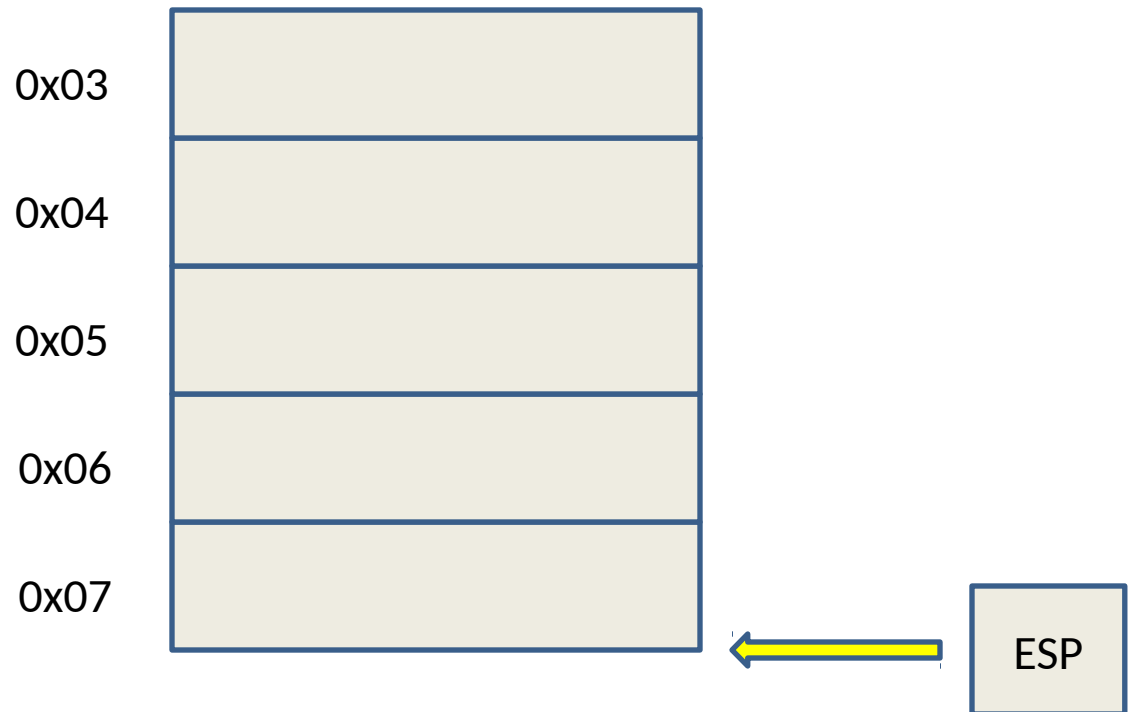
    foo(1,2,3);
}
```

EIP

Foo (a, b, c);

EBP

The animations on this slide will only work in the .pptx of this lecture



Stack

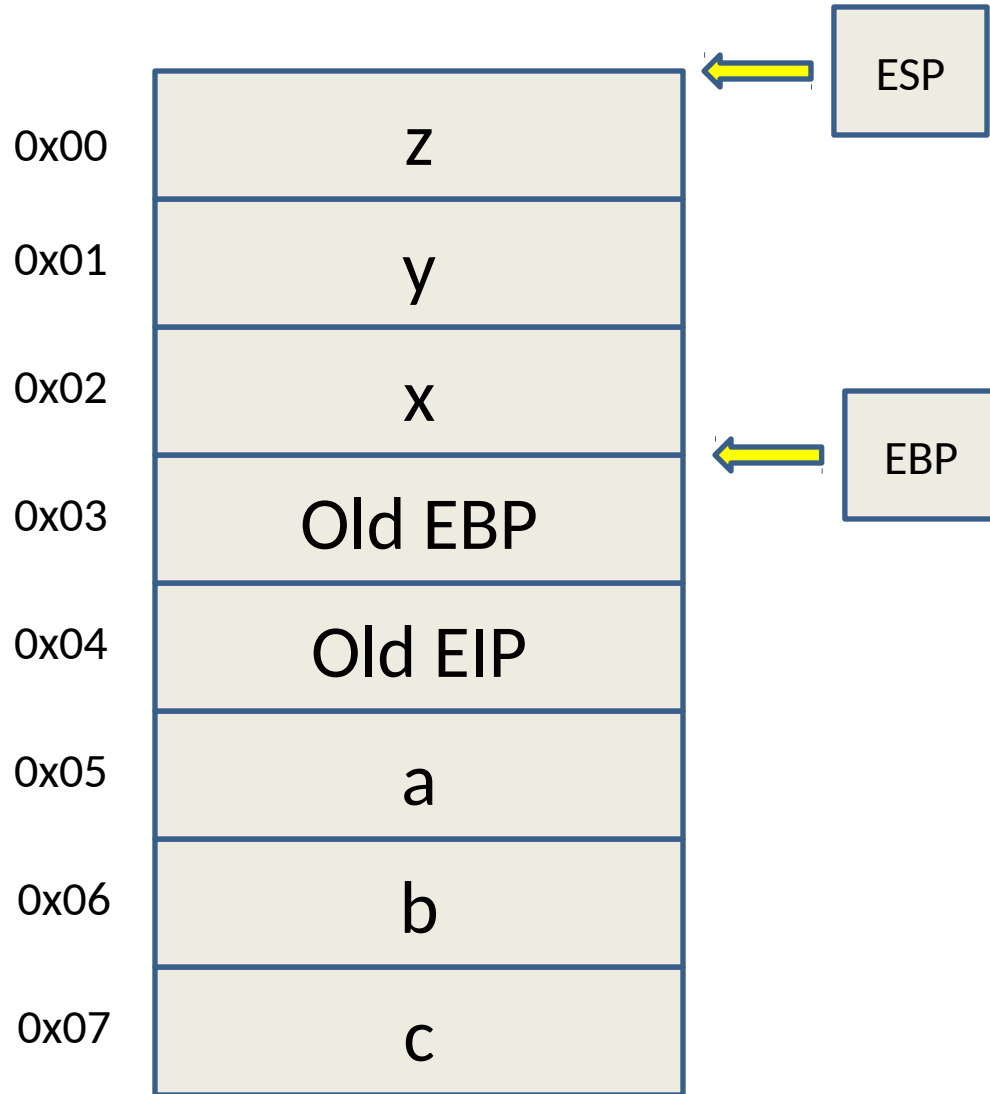
```
int foo(int a, int b, int c)
{
    int x;
    int y;
    int z;

    x=y=z=0;
    z=x+y+a+b+c;
    return z;
}

int main(int argc, char **argv) {

    foo(1,2,3);

}
```



Lecture Overview

1. Introduction to Reverse Engineering
2. Tools!
3. Resources

```
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+54
```

```
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, 0FFFFh
or      eax, 80070000h
```

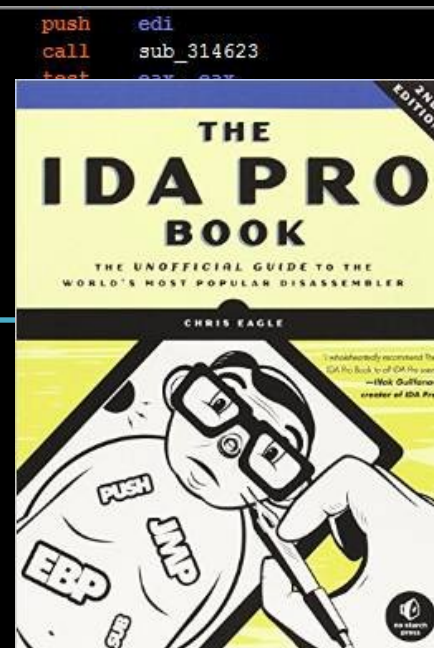
```
loc_31308C:                                     ; CODE XREF: sub_312FD8
```

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

IDA Pro

- IDA_Pro_Shortcuts.pdf
- The book on IDA
- IDA Syntax Highlighting:

– <http://practicalmalwareanalysis.com/2012/03/25/decorating-your-disassembly/>



```
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_313060
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+54
```

```
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
```

Additional Resources

- Corkami.com – diagrams of file structures and other interesting trivia
- Crackmes.de – “Reverser’s Playground”
- Subreddits
 - reddit.com/r/reverseengineering
 - reddit.com/r/netsec
 - reddit.com/r/uic
- <http://www.bottomupcs.com> - Systems background

```
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, 0Dh
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_313056: ; CODE XREF: sub_312FD8
; sub_312FD8+54
```

```
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
push [ebp+var_10]
```

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

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

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

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