המרכז
לחינוך סייבר
CYBER EDUCATION CENTER

Assembly for Reverse Engineering

Conditions, Jumps, Loops

Barak Gonen

Label

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Provide a “name” to a memory address containing code
- During assembling process, name will be replaced by actual address
- Think: why is it useful?
- Let’s try it. Assemble and OllyDbg:

```
        xor     eax, eax
here:
        mov     ebx, here
        mov     eax, ebx
        call    print_eax
```

Control Instructions- Requirements

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Capability to compare values
  - CMP instruction
- Change EIP according to the comparison result
  - Unconditional jump – JMP instruction
  - Conditional jump-
    - JE, JNE, JA, JL…
    - JZ, JNZ, JC, JO…
- Repeat as long as condition valid
  - LOOP instruction

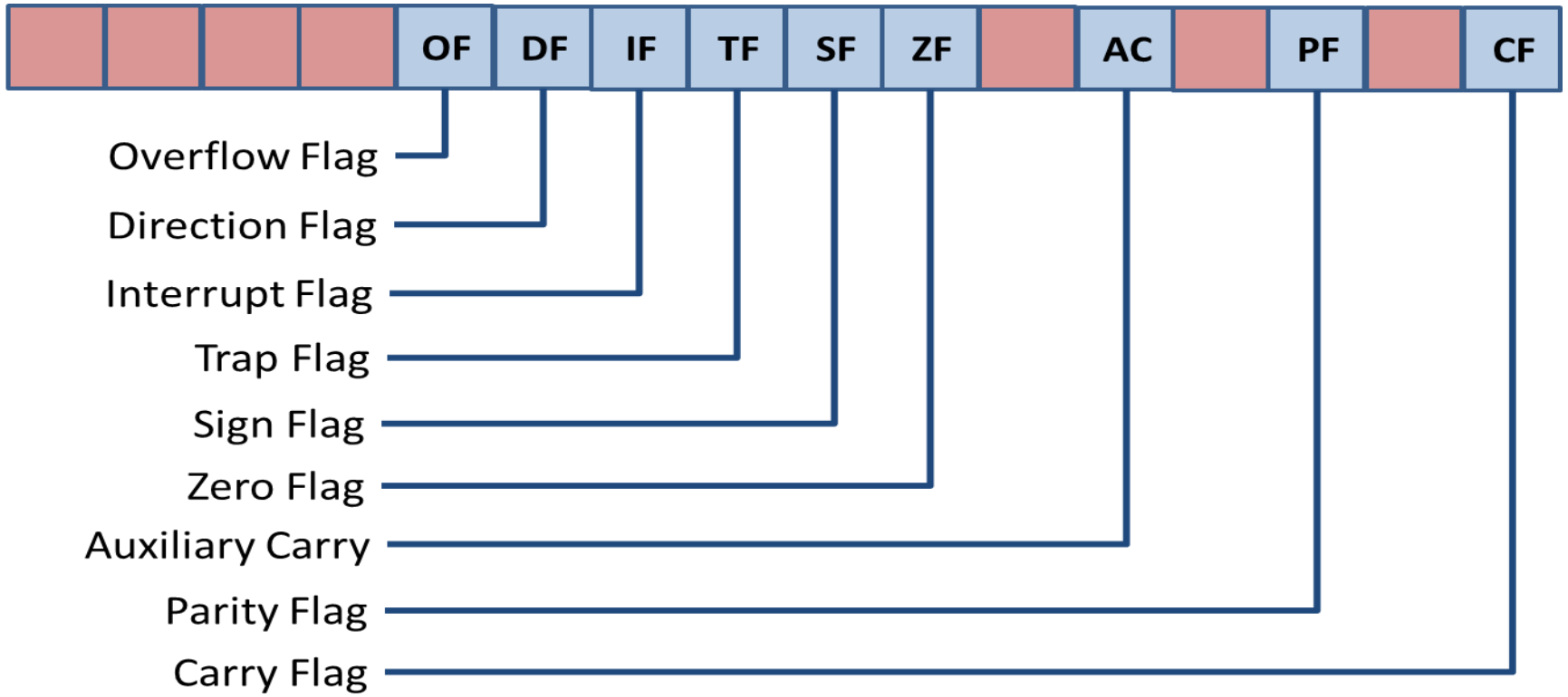
CMP Instruction

- CMP operand1, operand2
- How does the CPU performs CMP?
  - Like SUB, but without storing the value
  - FLAGS changed

FLAGS

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Each bit is a different signal



Zero Flag

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Zero Flag (ZF) is raised (==1) if the result of the last operation is zero

```
; raise zero flag
mov     eax, 0x12345678
mov     ebx, 0x12345678
sub     eax, ebx
```

- Think of an ADD operation which raises ZF
- Now think of an ADD operation with 2 *positive* numbers which raises ZF

Carry Flag

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Carry Flag (CF) is raised (==1) if the result of the last operation is out of unsigned range
  - Note- all operands are considered unsigned

```
; raise carry flag
mov     eax, 0xFFFFFFFF
mov     ebx, 0x00000002
add     eax, ebx
```

Overflow Flag

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Overflow Flag (OF) is raised (==1) if the result of the last operation is out of signed range
  - Note- all operands are considered signed

```
; raise overflow flag
mov     eax, 0x7FFFFFFF
mov     ebx, 0x00000001
add     eax, ebx
```

CMP cont.

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Assume all FLAGS are initially 0

Instruction	ZF	CF	OF
mov al, 3			
cmp al, 3			
cmp al, 2			
cmp al, 4			
add al, 125			

Unconditional Jumps

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- JMP
  - Memory location – JMP 0x00401000
  - Label – much more easy!
- What will be the value of eax?

```
        mov     eax, 1
        jmp     do_something
        xor     eax, eax
do_something:
        inc     eax
```

JMP – cont.

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Using only JMP instructions, make the following code print ‘7’

```
xor     eax, eax
add     eax, 4
inc     eax
add     eax, 3
inc     eax
call    print_eax
```

Conditional Jumps

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- Jump only if a certain condition is true
- Usually post CMP instruction
- Which code will be executed for:
  - EAX == 5 ?
  - EAX != 5 ?

```
        cmp     eax, 5
        je      good
        ; code A
        ; ...
good:
        ; code B
        ; ...
```

TEST

- Consider this code:

```
test    eax, eax
jz      do_something
```

- TEST – do AND between operands
  - Zero Flag will be raised only if EAX == 0
- Equivalent code:

```
cmp     eax, 0
je      do_something
```

Practice

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- https://data.cyber.org.il/reversing/convert2asm.py
- Convert “if” statement to asm

Riddle

- A programmer translated the following to assembly:
  - “Compare operand A and operand B. If operand A is bigger, perform jump”
- Operand A has 0000 0001b (binary)
- Operand B has 1000 0001b
- Will the code jump?

10000001b 00000001b

Answer

- 1000 0001 may represent:
  - 129 (unsigned)
  - -127 (signed)
- We must instruct the CPU which comparison to perform
- Bottom line – it depends :-)

Signed, Unsigned Jumps

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- All jumps start with ‘J’
- Unsigned - include ‘A’ or ‘B’ (above/below)
- Signed – ‘G’ or ‘L’ (greater/lower)
- On top of that:
  - ‘E’ – Equal
  - ‘N’ – Not
- http://unixwiz.net/techtips/x86-jumps.html

Summary – Unsigned Jumps

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

Instruction	Description	Flags tested
JE/JZ	Jump Equal or Jump Zero	ZF
JNE/JNZ	Jump not Equal or Jump Not Zero	ZF
JA/JNBE	Jump Above or Jump Not Below/Equal	CF, ZF
JAE/JNB	Jump Above/Equal or Jump Not Below	CF
JB/JNAE	Jump Below or Jump Not Above/Equal	CF
JBE/JNA	Jump Below/Equal or Jump Not Above	AF, CF

Summary – Signed Jumps

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

Instruction	Description	Flags tested
JE/JZ	Jump Equal or Jump Zero	ZF
JNE/JNZ	Jump not Equal or Jump Not Zero	ZF
JG/JNLE	Jump Greater or Jump Not Less/Equal	OF, SF, ZF
JGE/JNL	Jump Greater/Equal or Jump Not Less	OF, SF
JL/JNGE	Jump Less or Jump Not Greater/Equal	OF, SF
JLE/JNG	Jump Less/Equal or Jump Not Greater	OF, SF, ZF

Summary of Signed / Unsigned

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- So how come high level languages have signed / unsigned types?
  - …The compiler translates to assembly code

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	2 or 4 bytes	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
unsigned int	2 or 4 bytes	0 to 65,535 or 0 to 4,294,967,295
short	2 bytes	-32,768 to 32,767
unsigned short	2 bytes	0 to 65,535
long	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned long	4 bytes	0 to 4,294,967,295

Hands On

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

```c
#include <stdio.h>

int x = -4;
unsigned int y = 4;

int main()
{
    if (x >= 3) {
        printf("x is good\n");
    };
    if (y >= 3) {
        printf("y is good\n");
    };
    return 0;
}
```

- Use Visual Studio
  - Breakpoint
  - Disassembly window
  - Registers
    - Debug->Windows->Regs
  - Memory
- What is the difference between the two “if” statements?

Summary of Signed / Unsigned

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- We have seen that memory stores only 1’s and 0’s
- The CPU does not know if a value is signed or not
  - 255 will be stored same as -1 (0xFF)
- The programmer provides the interpretation
  - JA / JG
  - JB / JL

Another Look At Pointers

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

```
int x, z;
int *y;

int main()
{
    x = 5;
    y = &x;
    z = *y;
    printf("%d %d %d\n", y, *y, z);
    return 0;
}
```

- Use Visual Studio
  - Breakpoint
  - Disassembly window
- What is the assembly code for “z = *y” ?

LOOP

המרכז לחינוך סייבר
CYBER EDUCATION CENTER

- LOOP will:
  - Decrement ECX
  - Compare ECX to zero
  - If not zero – jump to a label
- How many ‘X’ will be printed?

```
        xor     ecx, ecx
printx:
        call    print_x
        loop    printx
```

Homework

המרכז
לחינוך סייבר
CYBER EDUCATION CENTER

- Convert:
  - If
  - If-else
  - For
  - While
  - Nested loop
  - Challenge - Switch case (Not mandatory, but you can show how good you are ☺)


QUESTIONS